FACULTY OF ENGINEERING AND ARCHITECTURE

Augmenting Bayesian Networks with Textual Evidence for Expert-Based, Uncertainty-Aware Clinical Decision Support

Paloma Rabaey

Doctoral dissertation submitted to obtain the academic degree of
Doctor of Computer Science Engineering

Supervisors

Prof. Thomas Demeester, PhD* - Prof. Stefan Heytens, PhD**

* Department of Information Technology
Faculty of Engineering and Architecture, Ghent University

** Department of Public Health and Primary Care
Faculty of Medicine and Health Sciences, Ghent University

February 2026

GHENT UNIVERSITY

ISBN 978-94-93513-16-7
NUR 984, 981
Wettelijk depot: D/2026/10.500/24

Members of the Examination Board

Chair

Prof. Hennie De Schepper, PhD, Ghent University

Other members entitled to vote

Prof. Antoon Bronselaer, PhD, Ghent University
Prof. Edward Choi, PhD, Korea Advanced Institute of Science and Technology, South Korea
Prof. Johan Kwisthout, PhD, Radboud Universiteit, the Netherlands
Lucas Sterckx, PhD, LynxCare
Prof. Wim Van Biesen, PhD, Ghent University

Supervisors

Prof. Thomas Demeester, PhD, Ghent University
Prof. Stefan Heytens, PhD, Ghent University

Acknowledgments

This dissertation is the reward to a long and challenging PhD journey. When starting the PhD, the end seemed a thousant lightyears away, but by finishing this work I have proven to myself that I can do the most difficult things, as long as I put in the time and effort. However, difficult things are never achieved solo, and so in this final step of the journey I would like to express my gratitude to the people who helped along the way.

First of all, thank you Thomas for being such an excellent supervisor. You have shown me an example of positive leadership that I will not forget and — unfortunately for any future managers — that I will continue to use as a standard throughout my career. It will be very hard to find a supervisor who is as enthusiastic, positive, supportive and understanding as you. Thank you for being so involved in my research, while at the same time leaving me the space to explore whatever I want, in research and beyond.

Stefan, thank you for always being willing to volunteer your expert knowledge; myself and many other students in the group were able to reap the rewards from the Bayesian network you designed. I trust that I did not traumatize you with my persistence for answers to questions that had no easy solution. Your idealism, creativity, enthusiasm and positive view on the future of AI in medicine was a source of refreshment throughout my PhD. I hope my research brings you one step closer to the clinically reasoning AI assistant that you dream of.

I would like to extend my gratitude to all jury members of my PhD committee for their time, efforts and valuable feedback on my work. Thank you Prof. Hennie De Schepper, Prof. Wim Van Biesen, Prof. Antoon Bronselaer, Prof. Edward Choi, Prof Johan Kwisthout and Dr. Ir. Lucas Sterckx for making the defense a pleasant experience.

One of the best parts of my PhD was traveling to conferences, which taught me independence and inspired me to start traveling solo. From exchanging ideas and presenting posters, to going on small and big adventures with my colleagues-turned-friends, I will never forget the great times I had in New Orleans, Utah, Philadelphia, Barcelona, and more. I hope that upon reading this dissertation, my friends can finally forgive me for spending their hard-earned tax money!

The absolute cherry on top of my PhD was traveling to South-Korea for a research visit at KAIST university, in the group of Prof. Edward Choi. Thank you Ed for welcoming me into your group and all your valuable

insights, and thank you Jong Hak, Jiho and Yeonsu for trusting me to do my part in your ongoing projects. I am eternally grateful to Jiho and Yeonsu for taking me under their wing, I will never forget our trips around Seoul and their endless efforts in showcasing me the deliciousness of Korean food.

Of course, doing this PhD would have been very lonely without the company of the current and previous members of the T2K team. Thank you to Adrick, Alexandre, Cédric, Chris, Félix, Florian, Gianni, Henri, Janica, Joachim, Johannes, Pranay, Rembert, Ruben, Seza, Valentin, Warre, Amir, Cedric, François, JJ, Karel, Klim, Maarten, Semere, Sofie, Tom, and Yiwei. Shoutout to the energy team (old and new!), as well as all other friends on the 9th floor of iGent. I am also thankful to my thesis students Eva, Max, Géraldine, and Yaqiao, for teaching me the joy of mentoring and collaborating. Last but not least, a very big thank you to Heidelinde and Alexander for providing me with a second home at UZ, where I could always come to enjoy your warm company. Who knew statisticians could be so fun that they even become your friends?

Finally, good work cannot exist without a good work-life balance. For this reason, I send all my love to my amazing friends: exies, vosjes, scouts, and zonnestraaltjes, you make my life lighter and give me something to look forward to every day. Thank you especially to my dear friends Marija and Severine, who have been my computer science buddies for many years. No matter where we are in the world or in the workplace, we will always be a dreamteam!

Of course, the final thanks of this dissertation goes out to my family. Thank you mama, papa, Karline and Django for supporting me. I will always feel close to you, even when we are far away.

Ghent, February 2026
Paloma

Table of Contents

List of Figures

List of Tables

List of Acronyms

AI Artificial Intelligence

BN Bayesian Network

CDSS Clinical Decision Support System

CIE Clinical Information Extraction

CPT Conditional Probability Table

CXR Chest X-Ray

DAG Directed Acyclic Graph

EHR Electronic Health Record

GP General Practitioner

LLM Large Language Model

MAE Mean Absolute Error

NeSy Neuro-Symbolic

NN Neural Network

Samenvatting (Dutch Summary)

Net als op veel andere gebieden heeft artificiële intelligentie (AI) het potentieel om de gezondheidszorg te transformeren door middel van hulpmiddelen die clinici ondersteunen bij het nemen van complexe beslissingen. Voorbeelden hiervan zijn chatbots die differentiële diagnoses voorstellen, mobiele applicaties die mogelijk kankerverwekkende huidletsels kunnen detecteren en monitoringsystemen die het personeel op de intensieve zorg waarschuwen wanneer patiënten het risico lopen sepsis te ontwikkelen. Deze technologieën staan bekend als klinische beslissingsondersteunende systemen (of clinical decision support systems, CDSS) en zijn ontworpen om het klinisch oordeel te verbeteren in plaats van te vervangen.

Elektronische medische dossiers (EMD) vormen een waardevolle trainingsbron voor moderne, op AI gebaseerde CDSS's, omdat ze enorme hoeveelheden informatie bevatten, zoals medische voorgeschiedenis, vitale functies zoals hartslag en bloeddruk, aantekeningen van artsen, operatieverslagen en medische beelden. Wanneer deze rijke datasets effectief worden benut, hebben ze het potentieel om gepersonaliseerde zorg waar te maken, waarbij behandelingen zouden kunnen worden afgestemd op de unieke behoeften van elke patiënt.

Hoewel het potentieel van AI in de gezondheidszorg algemeen wordt erkend, verloopt de invoering ervan trager dan in veel andere domeinen. Deze voorzichtigheid is begrijpelijk, aangezien de gezondheidszorg een risicovol domein is waar beslissingen rechtstreeks invloed hebben op het welzijn van patiënten. In hoofdstuk 1 stellen we vast dat de ontwikkeling van CDSS's die klaar zijn voor de klinische praktijk afhankelijk is van drie fundamentele technische uitdagingen.

Ten eerste zijn zorgprofessionals begrijpelijkerwijs terughoudend om te vertrouwen op 'black-box'-AI-systemen waarvan de interne werking niet transparant is. Om vertrouwen te wekken, moeten CDSS's interpreteerbaar zijn en aanbevelingen doen die door clinici kunnen worden begrepen en geverifieerd. Deze aanbevelingen moeten gebaseerd zijn op gevestigde medische expertise, die rechtstreeks in het ontwerp van het systeem moet worden geïntegreerd, zodat de werking van het systeem aansluit bij de klinische praktijk.

Ten tweede speelt onzekerheid een centrale rol bij besluitvorming in de gezondheidszorg. Artsen redeneren voortdurend in termen van waarschijnli-

jkheid, verzamelen bewijs om de onzekerheid in een diagnose te verminderen of wegen opties af om de behandeling te vinden die de kans op herstel van de patiënt het meest waarschijnlijk maakt. CDSS's moeten dit proces van klinisch redeneren onder onzekerheid kunnen modelleren om betrouwbare aanbevelingen te genereren. Bovendien moeten ze hun betrouwbaarheidsniveaus duidelijk communiceren aan artsen, zodat zij de onzekerheden op passende wijze kunnen meenemen in hun eigen oordeel.

Ten derde omvatten EMD's zeer uiteenlopende soorten gegevens, variërend van gestructureerde tabelinformatie zoals vitale functies en diagnosecodes tot ongestructureerde tekst zoals ontslagverslagen en radiologische rapporten. Hoewel het juist deze diversiteit is die EMD-gebaseerde CDSS's zo krachtig maakt, zorgt het er ook voor dat transparante en betrouwbare gegevensintegratie technisch uitdagend wordt. Het succesvol combineren van deze verschillende gegevensbronnen op een duidelijke en interpreteerbare manier is essentieel voor het bouwen van robuuste, betrouwbare systemen.

In hoofdstuk 2 leggen we uit waarom bepaalde AI-technologieën beter geschikt zijn om sommige van deze uitdagingen aan te pakken dan andere. Bayesiaanse netwerken (BN's) zijn probabilistische grafische modellen die expliciet weergeven hoe variabelen elkaar beïnvloeden. Ze zijn bijzonder geschikt voor de eerste uitdaging omdat ze een interpreteerbare, door experten geïnformeerde structuur combineren met een transparant Bayesiaans inferentieproces. Bovendien bieden ze een ideaal kader voor de tweede uitdaging, omdat ze het proces van klinisch redeneren onder onzekerheid op natuurlijke wijze kunnen formaliseren en in staat zijn om de probabilistische vragen te beantwoorden die bepalend zijn voor diagnostische en behandelingsbeslissingen.

Ondanks deze sterke punten vormt het beperkte vermogen van BN's om de derde uitdaging aan te gaan en realistische, heterogene medische gegevens te verwerken, een belemmering voor een bredere toepassing ervan in de klinische praktijk. BN's kunnen geen ongestructureerde tekst verwerken, wat betekent dat ze geen gebruik kunnen maken van de rijke klinische informatie in vrije tekstnotities in EMD's.

Daarentegen blinken diepe neurale netwerken en grote taalmodellen (large language models, LLM's) uit in het verwerken van natuurlijke taal en het extraheren van informatie uit tekst. Ze zijn echter niet goed geschikt om de eerste twee uitdagingen aan te pakken, aangezien ze berucht zijn als 'black boxes' en geen betrouwbare, principiële manier hebben om onzekerheid weer te geven en te communiceren.

Om dit probleem aan te pakken, onderzoeken we in dit werk de volgende centrale onderzoeksvraag: *Hoe kunnen we Bayesiaanse netwerken uitbreiden met tekstuele gegevens om interpreteerbare, door experts geïnformeerde en onzekerheidsbewuste AI-systemen voor de gezondheidszorg te creëren die alle drie de uitdagingen met succes kunnen aangaan?* Deze centrale onderzoeksvraag, samen met gerelateerde subvragen, wordt geformuleerd in

hoofdstuk 3.

Onze zoektocht naar het antwoord op deze vraag begint in hoofdstuk 4, waar we onderzoeken of een BN echt nodig is om de eerste twee uitdagingen aan te pakken, of dat er een neuraal netwerk kan worden ontworpen dat de belangrijkste eigenschappen van een BN nabootst. Om dit te onderzoeken, ontwikkelen we een neurale BN 'understudy' die goed presteert bij het repliceren van het probabilistische en causale gedrag van een BN. Dit neural netwerk slaagt er echter niet in om de interpreteerbaarheid en expliciete integratie van expertenkennis te behouden, beide eigenschappen die BN's juist zo waardevol maken.

In hoofdstuk 5 verleggen we daarom de focus terug naar BN's en onderzoeken we hoe deze het proces van klinisch redeneren onder onzekerheid kunnen automatiseren. We breiden het BN uit door er neurale tekstrepresentaties in op te nemen, waardoor het BN zowel gestructureerde tabelgegevens als ongestructureerde tekstuele invoer kan verwerken. Deze neurale representaties worden rechtstreeks in het Bayesiaanse inferentieproces geïntegreerd, waardoor ze een natuurlijke uitbreiding van BN's vormen die hen in staat stelt multimodale gegevens te verwerken. In dit hoofdstuk komen we voor het eerst het gebrek aan klinische benchmarkdatasets tegen die gestructureerde tabelgegevens koppelen aan ongestructureerde tekst.

Om dit tekort aan te pakken, ontwerpen we in hoofdstuk 6 SimSUM, een op zichzelf staande gesimuleerde dataset die zowel gestructureerde als ongestructureerde medische data omvat en het domein van respiratoire aandoeningen in de eerstelijnszorg modelleert. Door de tabelgegevens te genereren met behulp van een BN en de tekstgegevens met behulp van een 'prompt'-gestuurde LLM, behouden we de volledige controle over het generatieproces van de data en zorgen we ervoor dat dit proces transparant blijft voor de gebruiker.

In hoofdstuk 7 demonstreren we een alternatieve aanpak voor het aanpassen van BN's met tekst. In tegenstelling tot de methode die in hoofdstuk 5 wordt gepresenteerd, waarbij neurale representaties rechtstreeks in het BN worden geïntegreerd, onderzoeken we in dit hoofdstuk een meer modulaire strategie. Hierbij wordt informatie die door een neuraal netwerk uit tekst wordt gehaald, samengevoegd met voorspellingen op basis van tabulaire data die door het BN worden geproduceerd. We passen deze methode toe om informatie-extractie op patiëntniveau mogelijk te maken en evalueren deze met behulp van de SimSUM-dataset. Waar conventionele methoden voor het extraheren van klinische informatie zich uitsluitend richten op het extraheren van klinische concepten uit ongestructureerde tekst, maakt onze benadering ook gebruik van de gestructureerde, tabulaire kenmerken die al aanwezig zijn in het EMD van een patiënt. Dankzij dit ontwerp kan het BN de output van de neurale classificator moduleren, waardoor het ontbrekende informatie in de tekst kan verwerken en tegenstrijdigheden tussen tekstueel bewijs en bestaande tabelgegevens kan oplossen.

Omdat het definiëren van een BN voor een bepaald domein vaak een tijd-

srovende uitdaging is, wordt in hoofdstuk 8 onderzocht hoe onzekerheid kan worden gemodelleerd zonder gebruik te maken van een BN. Hiervoor versoepelen we de oorspronkelijke vereisten voor een BN en vragen we experten om in plaats daarvan diagnostische trajecten te definiëren. Dit zijn deterministische, top-down relaties van differentiële diagnoses naar hun meest belangrijke klinische bevindingen. Aangezien onzekerheid niet expliciet in deze trajecten wordt gemodelleerd, halen we deze rechtstreeks uit de taal die in klinische rapporten wordt gebruikt. Deze communiceren vaak twijfel of juist zekerheid over de aanwezigheid of afwezigheid van specifieke bevindingen. Samen maken deze twee methoden het mogelijk om gestructureerde, onzekerheidsbewuste informatie uit ongestructureerde klinische rapporten te halen.

In hoofdstuk 9 sluiten we af met antwoorden op de centrale onderzoeksvraag van *hoe we Bayesiaanse netwerken kunnen aanvullen met tekstuele gegevens om interpreteerbare, door experts geïnformeerde en onzekerheidsbewuste AI-systemen voor de gezondheidszorg te creëren*, samen met de bijbehorende subvragen. We reflecteren ook op richtingen voor toekomstig werk, waaronder het bredere nut van de SimSUM-dataset, de uitdagingen bij het valideren van onze methoden op basis van echte data, het gebruik van diagnostische trajecten wanneer Bayesiaanse netwerken te restrictief zijn voor praktische gebruikssituaties, en toekomstige onderzoeksmogelijkheden met betrekking tot het kwantificeren van waarschijnlijkheid op basis van natuurlijke taal.

Summary

As in many fields, Artificial Intelligence (AI) has the potential to transform healthcare through tools that assist clinicians in making complex decisions. Examples include chatbots that suggest differential diagnoses, mobile applications that can detect potentially cancerous skin lesions, and monitoring systems that alert staff at the intensive care unit when patients are at risk of developing sepsis. These technologies are known as Clinical Decision Support Systems (CDSSs) and are designed to enhance clinical judgment rather than replace it.

Electronic Health Records (EHRs) form a valuable training source for modern AI-based CDSSs, as they contain vast amounts of information such as medical histories, vital signs like heart rate and blood pressure, clinician notes, surgical records, and medical images. When leveraged effectively, these rich datasets can unlock the potential for personalized care, enabling treatments to be tailored to each patient's unique needs.

Although the potential of AI in healthcare is widely recognized, its adoption has been slower than in many other domains. This caution is understandable, as healthcare is a high-stakes field where decisions directly affect patient well-being. In Chapter 1, we argue that developing CDSSs that are ready for real-world clinical use depends on addressing three fundamental technical challenges.

First, healthcare professionals are understandably hesitant to rely on "black-box" AI systems whose internal workings are opaque. To foster trust, CDSSs must be interpretable and produce recommendations that clinicians can verify and understand. These recommendations should be grounded in established medical expert knowledge, which needs to be integrated directly into the system's design so that its reasoning aligns with clinical practice.

Second, uncertainty is central to healthcare decision-making. Clinicians constantly reason in probabilistic terms, gathering evidence to reduce the uncertainty in a diagnosis or weighing off options to find the treatment that will most likely increase the patient's chance of recovery. CDSSs must be able to model this process of clinical reasoning under uncertainty to generate trustworthy recommendations. In addition, they should clearly communicate their confidence levels to clinicians, enabling them to appropriately factor these uncertainties into their own judgments.

Third, EHRs encompass highly diverse data types, ranging from structured tabular information like vital signs and diagnosis codes to unstruc-

tured text such as discharge summaries and radiology reports. While this diversity is what makes EHR-based CDSSs so powerful, it also makes reliable and transparent data integration technically challenging. Successfully combining these different data sources in a clear and interpretable way is essential for building robust, trustworthy systems.

In Chapter 2, we explain why certain AI technologies are better suited to addressing some of these challenges than others. Bayesian Networks (BNs) are probabilistic graphical models that explicitly represent how variables depend on one another. They are particularly well suited to the first challenge because they combine an interpretable, expert-informed structure with a transparent Bayesian inference process. In addition, they provide an ideal framework for the second challenge, as they can naturally formalize the process of clinical reasoning under uncertainty, and are able to answer the probabilistic questions that drive diagnostic and treatment decisions.

Despite these strengths, the limited ability of BNs to handle the third challenge, and incorporate realistic, heterogeneous medical data, has formed a barrier to their wider adoption in clinical practice. In particular, traditional BNs cannot process unstructured text, which means they are unable to leverage the rich clinical information contained in free-text notes within EHRs.

By contrast, deep neural networks and Large Language Models (LLMs) excel at processing natural language and extracting information from text. However, they are poorly suited to address the first two challenges, as they function infamously as black boxes and lack a reliable, principled way to represent and communicate uncertainty.

To address this gap, this work explores the following central research question: *How can we augment Bayesian networks with textual data to create interpretable, expert-informed, and uncertainty-aware AI systems for healthcare that can successfully meet all three challenges?* This central research question, along with related sub-questions, is formulated in Chapter 3.

Our journey toward answering this question begins in Chapter 4, where we investigate whether a BN is truly necessary to address the first two challenges, or whether a neural network can be designed to mimic the key properties of a BN. To explore this, we develop a neural BN "understudy" that performs well at replicating the probabilistic and causal behavior of a BN. However, it fails to preserve the interpretability and explicit integration of expert knowledge that make BNs so valuable.

In Chapter 5, we therefore shift the focus back to BNs and explore how they can automate the process of clinical reasoning under uncertainty. We extend the BN by incorporating neural text representations, enabling it to process both structured tabular data and unstructured textual input. These neural representations are integrated directly into the Bayesian inference process, making them a natural extension of BNs which allows them to handle multi-modal data. In this chapter, we first encounter the lack

of clinical benchmark datasets that link structured, tabular data with unstructured text.

To address this gap, we design SimSUM in Chapter 6, a self-contained simulated dataset that comprises both structured and unstructured medical records, modeling the domain of respiratory diseases in primary care. By generating the tabular data using a BN and the textual data using a prompt-steered LLM, we maintain full control over the data generation process and ensure that this process remains transparent to the user.

In Chapter 7, we demonstrate an alternative approach to incorporating text into BNs. Unlike the method presented in Chapter 5, which directly integrates neural representations into the BN, we explore a more modular strategy in which information extracted from text by a neural classifier is fused with tabular feature predictions produced by the BN. We apply this approach to support patient-level information extraction and evaluate it using the SimSUM dataset. Where conventional clinical information extraction methods focus solely on extracting clinical concepts from unstructured text, our approach also leverages the structured, tabular features already present in a patient's EHR. This design allows the BN to modulate the neural classifier's outputs, enabling it to handle missing information in the text and resolve contradictions between textual evidence and existing tabular data.

Because defining a BN for a given use case can be challenging and time-consuming, Chapter 8 explores how uncertainty can be modeled without relying on a BN. To do this, we relax the original requirements and instead ask experts to define diagnostic pathways: deterministic, top-down relationships from differential diagnoses to key clinical findings. Since uncertainty is not explicitly modeled within these pathways, we extract it directly from the language used in clinical reports, which often conveys doubt or confidence about the presence or absence of specific findings. Together, these two methods enable the extraction of structured, uncertainty-aware information from unstructured clinical reports.

In Chapter 9, we conclude by providing answers to the central research question of *how we can augment Bayesian networks with textual data to create interpretable, expert-informed, and uncertainty-aware AI systems for healthcare*, along with its sub-questions. We also reflect on directions for future work, which includes the broader utility of the SimSUM dataset, the challenges of validating our methods on real-world data, the use of diagnostic pathways when Bayesian networks are too restrictive for practical use cases, and future research opportunities related to the quantification of probability from natural language.

1

Artificial Intelligence in Healthcare

It is almost impossible to describe how Artificial Intelligence (AI) has taken the world by storm without slipping into familiar clichés. Domains such as finance, transportation, education, and the creative industries are not only becoming acquainted with AI but are rapidly embedding it into their workflows. For the general public, the rapid rise of Large Language Models (LLMs) – most notably ChatGPT – has been the defining moment of this technological shift, offering an accessible demonstration of what contemporary AI systems can do.

Yet not all sectors have embraced AI at the same pace. In healthcare, adoption has been noticeably slower. This hesitation is understandable: the domain is inherently critical, with decisions that directly affect patient safety and well-being. Healthcare professionals are reluctant to rely on black-box systems whose inner workings remain opaque, especially when those systems may influence diagnoses, treatments, or clinical workflows. Moreover, the breadth of data types (ranging from clinical notes and laboratory results to past diagnoses, medical imaging, and physiological signals) poses challenges for developing AI models that reliably and transparently integrate these data. As a result, while the potential is vast, realizing the benefits of AI in healthcare requires careful navigation of trust, transparency and data complexity.

In this first chapter, we will address the potential of and the challenges related to AI in healthcare. In Section 1.1, we first explain the concept of clinical decision support systems, and the various forms of data in the electronic health record that serve as their foundation. Then, Section 1.2 discusses the potential of AI in healthcare, while Section 1.3 dives into four

major challenges of AI in healthcare which must be addressed. This latter section also provides a concise case study of a prominent AI application in healthcare and the circumstances in which it failed.

1.1 Clinical decision support systems

A **Clinical Decision Support System (CDSS)** is defined by Berner and La Lande [2016] as a computer system designed to impact clinical decision making about individual patients at the point in time that these decisions are made. While this definition does not at all require a CDSS to be based on AI, the use of AI enables newer and more powerful CDSSs [Sutton et al., 2020]. These systems are built on a myriad of AI technologies, which range from traditional rule-based engines, to decision trees, to neural networks, to large (vision-)language models, and beyond [Montani and Striani, 2019, Sadeghi et al., 2024, Sutton et al., 2020, Hartsock and Rasool, 2024]. We will address some of the technologies which are relevant to our work in Chapter 2.

AI-based CDSSs are trained on health-related data. The source of this data is found in the **Electronic Health Record (EHR)**, a longitudinal collection of electronic health information about individual patients and populations, which is mainly intended for documentation of the healthcare process [Tayefi et al., 2021]. EHRs contain a myriad of multi-modal health-related data, which can be categorized into two major types: structured data and unstructured data.

Structured EHR data encompasses patient information that is recorded in a (partially) standardized, organized format. Examples of structured EHR data include, but are not limited to, the following:

- Patient demographics, such as age, or gender.
- Medical history, including past diagnoses, chronic diseases, and family genetics.
- Longitudinal monitoring, which includes routine observations such as height, weight, or blood pressure, but also more intensive monitoring such as heart rate or respiratory rate. The latter may be significantly fine-grained if the patient was monitored in an in-hospital setting, such as the intensive care unit.
- Laboratory testing results, such as hematology or microbiology results.
- Prescription medication information.
- Procedures and interventions, including surgeries and other treatments.

In this work, we will often talk about structured **tabular** data, as structured data effectively forms a table, where the rows are samples (i.e., patients), and the columns are features (i.e., time stamps, symptoms, lab results, pre-

scription codes, etc.).

Unstructured EHR data is all other data which is not naturally structured in a machine-readable format. Examples of unstructured EHR data include, but are not limited to, the following:

- Clinical caregiver notes, including primary care physician notes, nursing notes, and hospital discharge summaries.
- Surgical records containing narrative operative details.
- Medical images, such as X-rays, CT scans, or MRI scans.
- Imaging study reports, where radiologists describe findings and impressions based on medical images.
- Raw physiological signals, including ECG and EEG recordings.

It has been suggested that a single person may generate up to 500GB of health data over a lifetime [Senbekov et al., 2020], although this estimate can vary widely depending on demographics and nationality. As we continue to live in an increasingly digital and data-driven society, this figure is likely to grow, highlighting the immense scope and potential of personal health data.

1.2 Potential of AI in healthcare

EHRs form a gold-mine of data. If we are able to successfully leverage this data to train AI-based CDSSs, we would be able to transform healthcare [Sterckx and Van Biesen, 2024, Hinton, 2018]. Specifically, AI has the potential to fulfill the promise of truly personalized care, where every patient receives treatment tailored to their specific needs [Johnson et al., 2021]. Furthermore, AI might be able to accelerate science by generating medical hypotheses, suggesting novel treatment regimes, or uncovering clinical relationships [Rotmensch et al., 2017].

To make the potential of AI in healthcare more tangible, we (non-exhaustively) list some general types of CDSSs, each with one or more specific examples.

- **Feature-based diagnostic or triage systems**: One of the earliest CDSSs are rule-based systems (i.e. systems where rules are provided by experts, rather than learned from data) that take in a patient's history and symptoms and automatically suggest a diagnosis. Examples include MYCIN [Shortliffe et al., 1975] or Isabel [Vardell and Moore, 2011], the former for diagnosis of bacterial infections, and the latter for differential diagnosis in primary care. A more modern CDSS in this category is Bingli [Bingli, 2025], an application for automated medical history questioning and diagnostic landscaping in primary care.
- **Image-based diagnostic systems**: Smartphone applications like SkinVision [Sangers et al., 2022] scan images of the skin to predict the

risk of cancerous lesions. Similarly, Google's DermAssist [Liu et al., 2020, Jain et al., 2021a] provides differential diagnoses across a range of skin disorders.

- **Warning systems**: Real-time patient monitoring systems such as Sepsis Watch [Sendak et al., 2020] or the EPIC Sepsis Model [Wong et al., 2021] analyze continuous observations from hospitalized patients, alerting clinicians when sepsis risk becomes elevated. Another example is FibriCheck, a smartphone app which can monitor and detect abnormal heart rhythm [Proesmans et al., 2019].
- **Treatment prediction**: By integrating diverse clinical patient information, including tumor characteristics and other relevant factors, IBM Watson Oncology [Somashekhar et al., 2018] generates personalized, ranked lists of cancer treatment recommendations.
- **Automated report generation**: CareConnect AI [Corilus, 2025] is an integrated assistant within an electronic patient file system widely used by primary care physicians in Belgium. It automatically transcribes patient encounters and summarizes them into structured reports, thereby reducing administrative burden. Beyond language-based systems, other tools also leverage computer vision capabilities. For instance, Microsoft's MAIRA [Bannur et al., 2024] has been trained on large volumes of radiology data (including both imaging studies and corresponding reports) and can automatically generate radiology reports when provided with new images, such as chest X-ray studies.

1.3 Challenges for AI in healthcare

While the potential of AI to transform healthcare is widely recognized and considerable research is devoted to developing CDSSs, their real-world adoption and integration into clinical workflows lag behind what one might expect given the scale of ongoing research [Hill et al., 2024]. Moreover, several of the systems discussed in the previous section have been the subject of controversy, and some have even been discontinued.

- Dermatologists have questioned the efficacy of SkinVision, as diagnostic evaluation studies have shown that its specificity and sensitivity were likely overestimated [Deeks et al., 2020, Kips et al., 2025]. Still, SkinVision continues to be partnered with Belgian insurers CM and Helan, who pay back the use of this skin cancer screening tool to their customers [SkinVision, 2025].
- DermAssist, a similar tool developed by Google, was discontinued after concerns emerged regarding its accuracy for darker skin tones [Vice, 2025]. In addition to this racial bias, the tool's sensitivity was poorly calibrated, potentially overwhelming dermatologists with patients seeking follow-up care for false-positive diagnoses [The Guardian,

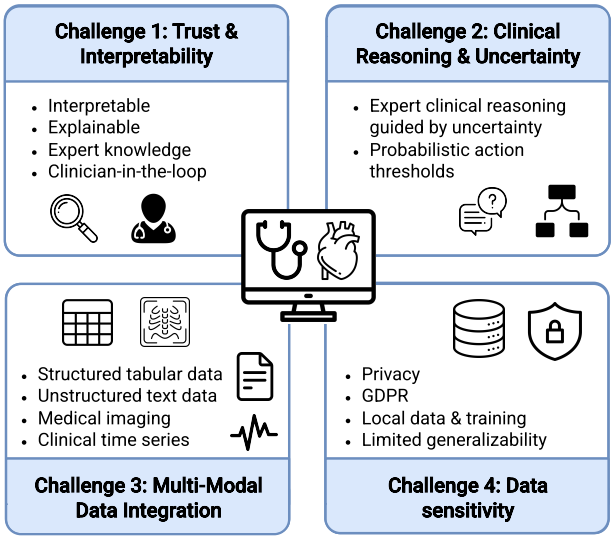


Figure 1.1: Four major technical challenges for AI in healthcare.

2025].

- The EPIC Sepsis Model, despite its deployment in the world's most widely used electronic patient record system, performed poorly in an independent validation study. According to Wong et al. [2021], "...its widespread adoption despite poor performance raises fundamental concerns about sepsis management on a national level."
- IBM Watson Oncology has been the subject of considerable controversy. After failing to deliver on its promises, the system was ultimately discontinued [Strickland, 2019]. We examine this case in greater detail in Section 1.3.5, analyzing the factors that contributed to its failure.

These examples illustrate that the success of CDSSs depends on striking a careful balance between technological innovation and clinical applicability [Sterckx and Van Biesen, 2024]. The gap between the promise and reality of CDSSs can be attributed, at least in part, to the difficulty of addressing **four fundamental technical challenges**, which are shown in Figure 1.1 [Hinton, 2018, Tayefi et al., 2021, Char et al., 2018, Aung et al., 2021]:

- Challenge 1: Building trust through interpretable and expert-guided decisions
- Challenge 2: Modeling clinical reasoning under uncertainty
- Challenge 3: Integrating multi-modal health data
- Challenge 4: Navigating the sensitive nature of health data

We now address each of these challenges in detail.

1.3.1 Challenge 1: Building trust through interpretable and expert-guided decisions

Given the critical nature of healthcare decisions, clinicians are understandably hesitant to adopt CDSSs into their workflows. While many recognize the potential for improved efficiency and reduced medical errors, actual adoption hinges on the fact that they can trust the CDSS's decisions and suggestions [Henzler et al., 2025]. There are two concepts in AI research that aim to enable trust in these systems: explainability and interpretability [Quinn et al., 2022, Rudin, 2019].

Explainability Many of the most powerful AI systems today (including neural models, such as large language models) operate as black boxes, achieving high performance by processing data in ways that are not readily scrutinizable by humans. To enable trust, these black boxes must be opened, at least partially. Explainability methods aim to describe a model's decisions in human-understandable terms after the fact [Sadeghi et al., 2024]; for instance, by highlighting the most influential input features in neural networks, or by prompting language models to articulate their reasoning process.

Interpretability Interpretability goes one step further, referring to how directly understandable a model's inner workings are from the outset. This can only be achieved when all steps taken by the model can be inspected and logically understood by humans, meaning black boxes are inherently out of the question [Rudin, 2019]. Examples of interpretable methods include traditional, non-neural AI models such as decision trees, logistic regression, and Bayesian networks.

Some, like Rudin [2019], argue that only interpretable AI models are acceptable for high-stakes decisions. In their view, explainability is insufficient in such domains: if one could fully explain a model's decisions in human-understandable terms after the fact, then one should be able to replace the model with that explanatory mechanism to achieve full interpretability. Most would agree that at least some level of interpretability – particularly in critical components of the system – is necessary to enable trust in CDSSs [Henzler et al., 2025]. At minimum, decisions or suggestions put forward by the CDSS should be verifiable by a clinician who remains in the loop [Char et al., 2018]. In Europe, this is even enforced by the AI Act, which requires that users should understand the factors influencing AI-based decisions [European Union, 2024].

Beyond interpretable decisions, another way to enable trust in a CDSS is by **integrating medical expert knowledge** into its design [Sirocchi et al., 2024]. Here, we distinguish between data and knowledge: data refers to patient-specific observations captured in EHR records, while knowledge refers to generalized medical understanding that transcends individual cases. While CDSSs trained on large volumes of health data can learn to make accurate predictions through pattern recognition alone, incorporating expert knowledge enhances trust. This integration can be achieved implicitly, for instance, by including biomedical publications in a large language model's training set. More explicit approaches leverage established knowledge graphs that encode relationships between clinical concepts, such as SNOMED-CT, UMLS, or ICD-10 coding hierarchies [Remy et al., 2024, Sirocchi et al., 2024]. Bayesian networks represent an even more explicit approach, directly embedding expert knowledge about causal relationships between variables into the model architecture itself. We discuss this approach in detail in Section 2.2 in the next chapter.

1.3.2 Challenge 2: Modeling clinical reasoning under uncertainty

Underlying many decisions made in patient care is a process called **clinical reasoning**. This refers to how a clinician integrates their own knowledge with patient information (like symptoms, objective measurements, background, medical history, etc.), to arrive at a diagnosis and subsequent therapeutic options [Gruppen, 2017]. A practical example to illustrate the concept of clinical reasoning is provided below.

Example: Clinical reasoning in primary care

A patient visits their primary care physician with an ongoing cough and shortness of breath. Based on these symptoms, the physician considers several possible diagnoses, including pneumonia. To narrow the possibilities, the physician measures the patient's temperature. When it turns out to be elevated, pneumonia becomes the most likely cause. The physician therefore prescribes antibiotics. In this example, the steps taken by the physician – gathering additional evidence (measuring temperature), arriving at a diagnosis (pneumonia), and selecting a treatment (antibiotics) – all stem from their internal clinical reasoning process.

Uncertainty plays an important part in the clinical reasoning process, and therefore in clinical practice more broadly [Meyer et al., 2021]. Whether consciously or not, clinicians are continuously working under conditions of uncertainty. This uncertainty drives their decisions as they seek additional evidence to lower the uncertainty related to a specific diagnosis or treatment option [Strauss et al., 2018, Yazdani et al., 2017]. The example below

illustrates the role of uncertainty in the clinical reasoning process.

Example: Clinical reasoning under uncertainty

In the previous example, the physician measures the patient's temperature, because they know that the outcome of that action will lower their uncertainty about a potential diagnosis of pneumonia. At the same time, the degree of uncertainty based on the evidence collected so far drives the decision to gather additional evidence. When a patient presents with high fever, persistent cough, and shortness of breath – symptoms that together are highly indicative of pneumonia – there is little reason to pursue further testing, such as an expensive chest X-ray, simply to increase diagnostic certainty in pneumonia.

Incorporating uncertainty into CDSSs is essential for ensuring that these systems align with the realities of medical practice [Simpkin and Schwartzstein, 2016, Kong et al., 2008]. As illustrated in the example above, diagnostic and therapeutic uncertainty is a routine aspect of clinical care. Because of this, CDSSs should provide not only their recommended decision but also an estimate of the confidence behind it, allowing clinicians to factor this information into their judgment. The level of uncertainty that a clinician deems acceptable in any given situation depends on many factors, including disease severity, the consequences of missing a diagnosis or prescribing an inappropriate treatment, patient demographics, the patient's ability to access timely care if symptoms worsen, the cost of additional testing, and more. These factors underscore the continued need for an expert in the loop: someone who can weigh all relevant information, including the CDSS's suggestions, before making the final decision.

Example: Clinical reasoning with uncertainty thresholds

Every physician has a personal threshold for how much uncertainty they are willing to tolerate before making a clinical decision, whether diagnosing or initiating treatment. In the example above, the physician might judge that the constellation of symptoms provides sufficient certainty to diagnose pneumonia. Although ordering a chest X-ray could further reduce uncertainty, the physician may view this as unnecessary given the costs, time, and radiation exposure involved. The point at which a clinician feels confident enough to act is called the *action threshold*, and can vary depending on the patient and the condition. For instance, a physician may set a higher threshold when treating young children, where missing a serious diagnosis carries greater risk, than when treating an adult. Similarly, they might be comfortable sending a patient home without treatment when they are 50% confident the symptoms represent a common cold, but would require far greater certainty before ruling out more serious conditions, such as appendicitis.

Building CDSSs that reflect the clinical reasoning process not only brings these systems closer to real-world practice but also enhances their interpretability and trustworthiness. Incorporating uncertainty strengthens this further: by showing users how confident the system is in a proposed diagnosis or treatment plan based on the available evidence, clinicians are better equipped to make informed decisions. In doing so, addressing challenge 2 also helps the system handle challenge 1 more effectively.

Finally, we note that clinical uncertainty has diverse sources. The type of uncertainty discussed above can be characterized as diagnostic uncertainty: the clinician's subjective sense that they cannot fully predict a patient's health condition [Meyer et al., 2021]. The overall diagnostic uncertainty a clinician experiences accumulates from various sources of uncertainty that infiltrate clinical reasoning: missing or incomplete evidence, unreliable evidence stemming from imprecise testing, the clinician's insufficient knowledge or experience, or the inherent complexity of diagnosis itself. Certain sources of uncertainty cannot be eliminated within today's medical framework, such as limited scientific understanding of particular rare diseases. Other problems are subject to radical uncertainty: no matter how much data we gather, we will never be able to predict the outcome of a dice roll [Sterckx and Van Biesen, 2024, King and Kay, 2020]. We emphasize that our goal is not to eliminate all sources of uncertainty, but rather to accept uncertainty as an essential part of the clinical process, and to build CDSSs that can successfully communicate their level of uncertainty to the user.

1.3.3 Challenge 3: Integrating multi-modal health data

In Section 1.1, we discussed the different types of health data, ranging from structured information such as encoded diagnoses to unstructured data such as free text [Tayefi et al., 2021]. This multi-modality is precisely what makes CDSSs so promising: the more diverse the available data, the more complete the representation of the patient, and the more accurate the system's predictions can become. However, effectively harnessing this multi-modal data remains a major challenge which must be overcome for CDSSs to realize their full potential.

Integrating all these data types into a single model is far from trivial, because the optimal AI technologies differ across data modalities. For **static structured tabular** data, tree ensemble models have consistently demonstrated strong performance [Shwartz-Ziv and Armon, 2022]. When data are structured, interpretable approaches such as Bayesian networks are an attractive technology [Koller and Friedman, 2009] (see Section 2.2). For **unstructured text**, neural networks, transformer-based methods, and more recently large language models, offer far greater modeling power [Singhal et al., 2023, Tayefi et al., 2021, Tian et al., 2024] (see Section 2.4). For **images**, convolutional neural networks remain the dominant technology [Sarvamangala and Kulkarni, 2022], while **time series** data are best handled

by transformer-based architectures [Li et al., 2020].

Though several approaches exist for integrating these modalities, such as ensemble methods [Tayefi et al., 2021, Xu et al., 2019], representation learning methods [Zhang et al., 2020, Nguyen et al., 2024, Wang et al., 2023b], or medical LLMs [Li et al., 2020, Rasmy et al., 2021], these solutions often trade interpretability for performance. It therefore remains a big challenge to build multi-modal systems which can robustly handle diverse data sources while remaining interpretable and reliable for use in critical applications.

1.3.4 Challenge 4: Navigating the sensitive nature of health data

Since health data are extremely sensitive, strong privacy-preserving measures are required to ensure that no patient characteristics are disclosed [Murdoch, 2021]. Although privacy regulations differ worldwide, the European General Data Protection Regulation (GDPR) imposes strict requirements on the sharing and secondary use of health data, particularly when data are transferred across institutions or national borders [European Union, 2016]. As a result, CDSSs are often trained solely on locally available hospital data [Aung et al., 2021]. This limits their generalizability in two important ways: (i) a system trained on one patient population is not guaranteed to work well on different populations, and (ii) only a limited number of records can be used for training, which restricts the system's robustness and accuracy, since more data enables better performance.

While the sensitive nature of health data poses a significant challenge for AI in healthcare, this dissertation does not address this issue in depth. For research contributions related to this challenge, we refer to our work within the Syndara project[1], where we investigated the use of non-sensitive synthetic data as a substitute for real data and examined how such data affect the validity of statistical inference [Decruyenaere et al., 2024a,b].

1.3.5 Case study: IBM Watson Oncology

In this section, we take a closer look at IBM Watson Oncology [Somashekhar et al., 2018], a CDSS that promised to be transformative for cancer treatment but was ultimately discontinued after failing to meet those expectations [Strickland, 2019]. Its shortcomings were tied to all four challenges outlined above, making it an instructive case study for illustrating why these challenges matter, and how failing to address them can lead to the failure of a CDSS.

[1] `https://syndara.ai/`

Challenge 1: Building trust through interpretable and expert-guided decisions IBM Watson Oncology provided clinicians with a ranked list of treatment recommendations, leaving it to the clinician to determine how to use this information. Although this setup supports expert-guided decision-making, it remains unclear whether clinicians' decisions were meaningfully altered or improved by the system. In an effort to ensure that its recommendations were evidence-based, the system was restricted to proposing treatments already contained in established clinical guidelines. As a result, clinicians often found the recommendations unsurprising at best and, at worst, not useful [Strickland, 2019]. Conversely, if the system were to recommend something novel (perhaps by analyzing past patient records or drawing from niche medical literature) clinicians would be unlikely to trust it, as they had no insight into Watson's opaque reasoning process.

Challenge 2: Modeling clinical reasoning under uncertainty As far as we can tell, Watson did not explicitly incorporate uncertainty into its reasoning process and was therefore unable to account for uncertain evidence. It provided only a ranked list of recommended treatment options, without quantifying the uncertainty behind these predictions or indicating any recommendation-level confidence.

Challenge 3: Integrating multi-modal health data Watson struggled to process realistic patient records, which were far less clean than the curated data it had been trained on: information was missing, out of order, or expressed ambiguously. It was particularly challenged by the complexity of unstructured data, often failing to extract crucial details such as treatment timelines from textual medical records [Strickland, 2019]. It is worth noting that IBM Watson Oncology was discontinued around mid-2022, just before the emergence of today's most powerful LLMs, which are far more capable of extracting information from large volumes of clinical text than the NLP technologies Watson likely relied on [Xu et al., 2024b].

Challenge 4: Navigating the sensitive nature of health data Watson was trained on a vast corpus of medical literature but reportedly processed real patient records from only a single center [Strickland, 2019]. This limited its generalizability and undermined trust in any treatment guidelines it attempted to infer from historical clinical data.

2

Traditional and Modern Artificial Intelligence

We now provide an overview of the most important AI technologies that we will use in this dissertation. First, Section 2.1 contextualizes the difference between traditional knowledge-driven AI and modern data-driven AI. We then introduce three specific technologies: Bayesian networks (Section 2.2), neural networks (Section 2.3) and large language models (Section 2.4), and explain how each relates to challenges 1, 2, and 3. Finally, we introduce two additional concepts that are relevant to the dissertation: clinical information extraction (Section 2.5) and neuro-symbolic AI (Section 2.6).

2.1 Knowledge-driven versus data-driven AI

Since the inception of the field of AI in the 1940s and 1950s [Russell and Norvig, 2009], the technologies that define it have evolved substantially: from rule-based symbolic reasoners, to probabilistic and statistical learning methods, across deep neural networks, to large-scale foundation models and generative AI. While there is no clear distinction between "traditional" and "modern" AI – after all, the perceptron, a foundational building block of today's neural networks, was introduced as early as 1957 – there is a clear overarching trend: the gradual shift from knowledge-driven AI to data-driven AI.

Knowledge-driven AI Knowledge-driven, or symbolic, AI is a type of AI where human knowledge is formalized and explicitly encoded into the model in a top-down fashion [Montani and Striani, 2019]. This is typically seen as the more traditional form of AI. Examples include:

- Expert systems, which mimic human decision-making by operating on a set of human-defined rules [Russell and Norvig, 2009].
- Knowledge graphs, which structure domain-specific information by formalizing key concepts within a domain and the relationships between them [Russell and Norvig, 2009].
- Expert-defined Bayesian networks, which are probabilistic graphical models that represent variables and the conditional independencies between them, allowing probabilistic reasoning [Koller and Friedman, 2009].

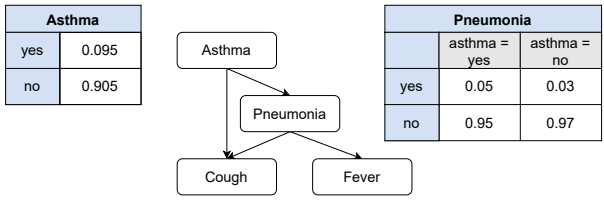
Data-driven AI Data-driven AI refers to methods in which patterns are automatically learned from (large amounts of) data in a bottom-up manner [Montani and Striani, 2019]. This paradigm is typically associated with more modern forms of AI. Examples include:

- Statistical machine learning methods, like logistic regression, decision trees, or random forests [Russell and Norvig, 2009].
- (Artificial) neural networks, which are computational models inspired by the brain and learn weights that map input signals to outputs [Russell and Norvig, 2009].
- Modern deep learning methods [LeCun et al., 2015], which stack many neural network layers into specialized architectures such as convolutional neural networks or transformers, and are trained on large-scale datasets.

As shown in Table 2.1, the distinction between knowledge-driven and data-driven AI approaches is useful for highlighting general trends in their respective strengths and limitations. Note however that this distinction is not black and white. There are knowledge-driven systems which can learn part of their logic from data. For example, Bayesian networks allow the conditional independence structure between variables and the probabilistic relationships among them to be defined entirely by an expert, learned entirely from data, or specified in a hybrid approach [Koller and Friedman, 2009]. On the other hand, data-driven approaches are not devoid of human knowledge: LLMs can implicitly acquire human knowledge from large text corpora [Singhal et al., 2023, Shah et al., 2023] (including Wikipedia, scientific literature, or even broader web sources), while retrieval-augmented language models do not store all knowledge internally but instead learn to query an external knowledge database to find the relevant information needed to solve a task [Lewis et al., 2020].

Table 2.1: Comparison between general trends in strengths and limitations of knowledge-driven AI versus data-driven AI.

Knowledge-driven AI	**Data-driven AI**
Knowledge-driven models are generally interpretable and reliable for high-stakes decisions. A user can easily verify that the model's predictions make sense and are based on an appropriate reasoning path.	Data-driven models are usually less interpretable. Some statistical machine learning methods are partially interpretable [Lundberg et al., 2020]: for example, the coefficients of a logistic regression model indicate feature importance, and a decision tree has a human-readable form. However, modern deep learning methods are typically opaque due to their large number of parameters [Rudin, 2019]. Consequently, they either remain a black box or require substantial effort to explain their predictions [Zhao et al., 2024, Sadeghi et al., 2024].
Knowledge-driven AI models are typically feasible only for small-scale, specific use cases, as they require experts to manually define rules and formalize their knowledge. The benefit of this manual effort is that these models provide strong guarantees of robustness within well-defined domains, making them suitable for high-stakes decisions.	Data-driven models can be scaled according to the amount of available data. By learning arbitrary patterns from the data, they can become more general, provided the training data adequately represents the deployment domain. However, there are no formal guarantees that these models will generalize beyond the distribution of their training data, leaving them vulnerable to unexpected errors, bias, or failures when confronted with out-of-distribution inputs [Aung et al., 2021, Sanchez et al., 2022].
Knowledge-driven models can automate human decision processes, but cannot discover new knowledge patterns without human intervention.	Data-driven models can automatically process large volumes of data and infer new patterns or conclusions that may extend beyond human intuition.



Asthma	
yes	0.095
no	0.905

Pneumonia		
	asthma = yes	asthma = no
yes	0.05	0.03
no	0.95	0.97

Cough				
	Asthma = yes		Asthma = no	
	Pneumonia = yes	Pneumonia = no	Pneumonia = yes	Pneumonia = no
yes	0.80	0.40	0.75	0.10
no	0.20	0.60	0.25	0.90

Fever		
	Pneumonia = yes	Pneumonia = no
yes	0.70	0.15
no	0.30	0.85

Figure 2.1: Example BN for diagnosis of pneumonia.

We now provide an introduction to Bayesian networks, a knowledge-driven technology that is pivotal to the work done in this dissertation. Then, we provide a short introduction to neural networks and large language models, two related data-driven technologies that are used in this dissertation.

2.2 Bayesian networks

In this section, we assume that the reader has some basic notion of graphs, as well as probability theory, including the concept of conditional independencies and the use of Bayes' theorem. For more information and formal definitions, we refer to Koller and Friedman [2009].

2.2.1 Components of a BN

A Bayesian Network (BN) is a probabilistic graphical model that represents how different variables probabilistically depend on one another. Figure 2.1 shows an example BN for the diagnosis of *pneumonia*, which relates this diagnosis to two symptoms (*cough* and *fever*), as well as a background condition (*asthma*). A BN is made up of two ingredients: a graph structure, and a set of conditional probability distributions, one for each variable.

A **Directed Acyclic Graph (DAG)** encodes conditional independencies between variables, by representing them as nodes in a graph. For example, from the DAG in Figure 2.1, we can infer that *fever* is conditionally independent from *cough* given *pneumonia*: $fever \perp\!\!\!\perp cough \mid pneumonia$. This means that once we know a patient has pneumonia, learning whether they have a cough provides no additional information about whether they might also have a fever, since both symptoms are explained by the same underlying cause. In contrast, if we do not know whether a patient has pneumonia,

then *fever* becomes conditionally dependent on *cough*: $fever \not\perp cough$. In this case, observing a cough makes pneumonia more likely, which in turn increases the likelihood of fever. The DAG contains additional conditional independence relations (eight in total), but deriving these is beyond the scope of this introduction. Finally, note that the BN here is a simplified abstraction: it omits many relevant clinical factors, meaning these conditional independence relationships would not necessarily hold in real diagnostic settings.

Having discussed the conditional independencies encoded in the DAG, we can now turn to what this structure allows us to represent: the full **joint probability distribution** over all variables in the network. A BN factorizes this joint distribution into a product of local conditional probability distributions, one for each node given its parents. The joint probability distribution for the BN in Figure 2.1 is defined as follows (here, we abbreviate *pneumonia* by *pneu*):

$$\mathcal{P}(asthma, pneu, cough, fever) = \mathcal{P}(asthma)\mathcal{P}(pneu \mid asthma) \\ \mathcal{P}(cough \mid asthma, pneu)\mathcal{P}(fever \mid pneu) \quad (2.1)$$

To define this joint probability distribution, one needs only to define the probability distribution for each variable, conditional on its set of parents. For discrete variables, this can be done through a **Conditional Probability Table (CPT)**. These CPTs are shown in Figure 2.1 for each of the four variables in our BN. Here, all variables are binary, taking on two possible values ("yes" or "no"). For example, to define $\mathcal{P}(fever \mid pneu)$, we simply need to define the probabilities $\mathcal{P}(fever = yes \mid pneu = yes)$ and $(fever = yes \mid pneu = no)$. The complementary probabilities $\mathcal{P}(fever = no \mid pneu = yes)$ and $\mathcal{P}(fever = no \mid pneu = no)$ are automatically determined by the law of total probability. These conditional probabilities together form the CPT for *fever*, storing a probability for each possible combination of parent and child values.

2.2.2 Defining or learning the BN

Now that we know that a BN is fully defined by its DAG and conditional probability distributions, the question remains how to obtain these. Usually, the DAG is defined by an expert or a panel of experts for a specific use case, based on general clinical knowledge or their own experience [Kyrimi et al., 2020]. Alternatively, the DAG can be learned from data using structure learning algorithms [Koller and Friedman, 2009], which determine which variables are connected and the directions of their dependencies. However, using this data-driven approach comes at a cost: we lose one of the key strengths of BNs, namely their ability to naturally incorporate expert knowledge.

Similarly, conditional probability distributions can either be specified by an expert or learned from data using parameter estimation techniques

[Koller and Friedman, 2009], with the latter being more common. The simplest approach is maximum likelihood estimation, which requires complete data, i.e., every variable in the BN must be observed in each sample. In this method, each conditional probability distribution is learned independently from one another, based on the observed samples. For instance, to learn the CPT for *fever*, we can analyze a dataset containing samples such as $\{fever = yes, pneu = no\}$, $\{fever = yes, pneu = yes\}$, $\{fever = no, pneu = no\}$, and so on. By examining the co-occurrence of the child variable *fever* with its parent *pneumonia* across all possible value combinations, we can populate the corresponding CPT.

We can therefore conclude that a BN can be fully defined by an expert, fully learned from data, or any combination of the two. For example, experts can contribute partially by specifying certain probability distributions, while others are estimated from data. In our example BN, if the prevalence of *asthma* in the general population is well-documented in the literature, it can be set to a fixed value. Meanwhile, probabilities that are less certain, such as the likelihood of *fever* given *pneumonia*, can be learned directly from observed data.

2.2.3 Sampling and Bayesian inference

Once a BN is fully specified, it can be used to perform two key operations: sampling and Bayesian inference.

Sampling The joint probability distribution encoded by a BN can be treated as a ground-truth data distribution, from which new data points can be generated. Sampling is typically performed in a top-down manner, starting from the root nodes (variables without parents, e.g., *asthma*) and proceeding to the leaf nodes (variables without children, e.g., *cough* and *fever*). At each step, the value of a child node is sampled conditionally on the values of its parent nodes. In our example, we would start by sampling *asthma* according to its marginal distribution, $\mathcal{P}(asthma)$; for instance, we might obtain $\{asthma = yes\}$. Next, we sample *pneumonia* conditioned on the sampled value of *asthma*, using $\mathcal{P}(pneu \mid asthma = yes)$. We then proceed similarly for *cough* and *fever*, sampling each node based on the values already sampled for its parent nodes. By repeating this process, we can generate synthetic samples that are consistent with the probabilistic dependencies encoded in the network.

Bayesian inference Crucially, we can also use the BN to infer the probability for any *target* variable, provided the values of a set of *evidence* variables. For example, we can calculate $\mathcal{P}(pneu = yes \mid asthma = no, fever = yes)$, where *pneumonia* is the target, and *fever* and *asthma* are the evidence variables. The most straightforward way to perform exact Bayesian inference is through *variable elimination*. We now illustrate this process on an

example, to provide the reader with an intuitive understanding. For formal definitions, we refer to [Koller and Friedman, 2009].

To calculate $\mathcal{P}(pneu = yes \mid asthma = no, fever = yes)$, we apply Bayes' rule to transform the conditional probability into a fraction of joint probabilities:

$$\mathcal{P}(pneu = yes \mid asthma = no, fever = yes) = \frac{\mathcal{P}(pneu = yes, asthma = no, fever = yes)}{\mathcal{P}(asthma = no, fever = yes)} \tag{2.2}$$

We can obtain these joint probabilities by marginalizing over the full joint probability distribution in Equation 2.1. Here, marginalizing means summing over all possible values for any variables whose values are not observed in the joint probability distribution. In the numerator of Equation 2.2, this entails summing out all variables that are neither targets nor evidence (here, *cough*), while in the denominator this entails summing out all variables that are not evidence (this includes both *cough* and the target *pneumonia*):

$$(2.2) = \frac{\sum_{v_c} \mathcal{P}(asthma = no,\, pneu = yes,\, cough = v_c,\, fever = yes)}{\sum_{v_c}\sum_{v_p} \mathcal{P}(asthma = no,\, pneu = v_p,\, cough = v_c,\, fever = yes)}$$

$$= \frac{\sum_{v_c} \mathcal{P}(asthma = no)\,\mathcal{P}(pneu = yes \mid asthma = no)\,\mathcal{P}(cough = v_c \mid asthma = no,\, pneu = yes)\,\mathcal{P}(fever = yes \mid pneu = yes)}{\sum_{v_c}\sum_{v_p} \mathcal{P}(asthma = no)\,\mathcal{P}(pneu = v_p \mid asthma = no)\,\mathcal{P}(cough = v_c \mid asthma = no,\, pneu = v_p)\,\mathcal{P}(fever = yes \mid pneu = v_p)} \tag{2.3}$$

Here, we sum over all possible values for *cough*, with $v_c \in \{yes, no\}$, and *pneumonia*, with $v_p \in \{yes, no\}$. We can now simplify Equation 2.3 by realizing that $\mathcal{P}(asthma = no)$ appears in both the numerator and denominator, so we can remove it. Furthermore, we can move the factors $\mathcal{P}(pneu = yes \mid asthma = no)$ and $\mathcal{P}(fever = yes \mid pneu = yes)$ outside of the summation over v_c, as they do not contain the iterator v_c.

$$(2.3) = \frac{\mathcal{P}(pneu = yes \mid asthma = no)\mathcal{P}(fever = yes \mid pneu = yes) \sum_{v_c} \mathcal{P}(cough = v_c \mid asthma = no,\, pneu = yes)}{\sum_{v_p} \mathcal{P}(pneu = v_p \mid asthma = no)\mathcal{P}(fever = yes \mid pneu = v_p) \sum_{v_c} \mathcal{P}(cough = v_c \mid asthma = no,\, pneu = v_p)} \tag{2.4}$$

In Equation 2.4, we notice that the sum over *cough* is isolated in both the numerator and denominator. Due to the law of total probability, we know that these sums calculate to 1, and we can remove them. This leaves us with the following expression:

$$(2.4) = \frac{\mathcal{P}(pneu = yes \mid asthma = no)\mathcal{P}(fever = yes \mid pneu = yes)}{\sum_{v_p} \mathcal{P}(pneu = v_p \mid asthma = no)\mathcal{P}(fever = yes \mid pneu = v_p)} \tag{2.5}$$

Finally, we can fill in all values from the CPTs in Figure 2.1 to calculate the target probability.

$$
\begin{aligned}
&\mathcal{P}(pneu = yes \mid asthma = no, fever = yes) \\
&= \frac{\mathcal{P}(pneu = yes \mid asthma = no)\mathcal{P}(fever = yes \mid pneu = yes)}{\mathcal{P}(pneu = yes \mid asthma = no)\mathcal{P}(fever = yes \mid pneu = yes) \\
&\quad + \mathcal{P}(pneu = no \mid asthma = no)\mathcal{P}(fever = yes \mid pneu = no)} \\
&= \frac{0.03 \times 0.70}{0.03 \times 0.70 + 0.97 \times 0.15} = 0.126
\end{aligned}
\tag{2.6}
$$

Clinical reasoning through Bayesian inference The Bayesian inference process helps us address the key questions that arise during clinical reasoning, as introduced in Section 1.3.2. Below, we present examples of questions a clinician might consider when diagnosing pneumonia, and show how each can be translated into conditional probabilities that can be evaluated using the BN in Figure 2.1. Note that all probabilities mentioned below were obtained using variable elimination, although we do not write out the full procedure here.

First, a primary care physician might ask themself what the *prior probability* is for any patient to have pneumonia, without knowing anything else about this patient. In our example, the prior is $\mathcal{P}(pneu = yes) = 0.0319$, which is quite low. When an asthmatic patient walks in who also has a cough, the *posterior probability* can be calculated as $\mathcal{P}(pneu = yes \mid asthma = yes, cough = yes) = 0.0952$, which is higher. However, this probability still does not exceed the physician's *action threshold*, meaning it remains unlikely that the patient has pneumonia and therefore would need treatment. For this reason, it would be sensible to gather additional evidence, for example, whether the patient has a fever.

Before gathering this evidence, it is useful to consider what information it is likely to provide. Suppose the physician decides to check whether the patient has a fever. If the patient does *not* have a fever, the posterior probability $\mathcal{P}(pneu = yes \mid asthma = yes, cough = yes, fever = no)$ decreases to 0.0358. Conversely, if the patient *does* have a fever, the posterior probability $\mathcal{P}(pneu = yes \mid asthma = yes, cough = yes, fever = yes)$ increases to 0.329.

In this case, the physician can conclude that assessing the patient's fever status is informative, as it can substantially increase or decrease their confidence in a pneumonia diagnosis. If these probabilities were much closer together (e.g., if $\mathcal{P}(pneu = yes \mid asthma = yes, cough = yes, fever = no) = 0.08$ and $\mathcal{P}(pneu = yes \mid asthma = yes, cough = yes, fever = yes) = 0.12$), gathering information on fever would provide little added value, and the physician's effort would be better spent obtaining other evidence. In our example, checking for fever is a low-effort task, but evaluating the usefulness of additional evidence becomes particularly important when that evidence is costly or difficult to obtain, such as an X-ray or blood test.

2.2.4 BNs as clinical decision support systems

BNs have been used in research settings to model a wide range of medical conditions [McLachlan et al., 2020b, Arora et al., 2019b]. They are most commonly employed as diagnostic tools, and less frequently as interventional tools, such as for providing treatment recommendations [Kyrimi et al., 2021e, McLachlan et al., 2020b]. To illustrate both the potential and the limitations of BNs, we now discuss how they either address or fall short in addressing the three challenges for AI in healthcare introduced in Section 1.3.

Challenge 1: Building trust through interpretable and expert-guided decisions BNs are well suited to this challenge, as they combine an interpretable causal DAG structure with a transparent Bayesian inference process. As illustrated in the discussion above, a BN can be queried to help the user understand its input-output behavior. Allowing the user to modify the input (evidence) and observe how the output (the probability of one or more target variables) changes, fulfills a key requirement for user-centric explainable models [Sadeghi et al., 2024, Sterckx and Van Biesen, 2024]. The process of variable elimination is also technically interpretable, since it makes explicit how the final probability is derived from the conditional probabilities defined within the BN. However, as more variables are added, it becomes increasingly difficult to trace the exact path from evidence to predicted probability. Methods such as the one proposed by Kwisthout [2023] can support users in understanding and trusting the decisions produced by the BN.

Furthermore, expert knowledge is explicitly embedded in the DAG at the core of the BN. If desired, experts can also contribute to defining parts of the conditional probability distributions, or to validating distributions that are learned from data. In this way, BNs integrate data and expert knowledge effectively and in an interpretable manner.

Challenge 2: Modeling clinical reasoning under uncertainty As illustrated in the discussion above, BNs are an ideal tool for formalizing the process of clinical reasoning, in which a clinician iteratively poses probabilistic questions that can be answered by the BN. BNs explicitly model uncertainty and therefore provide probabilistic predictions that are well-calibrated by design, provided that the network's structure and parameters accurately reflect the true data-generating process. Additionally, the BN's ability to handle missing evidence – a form of uncertainty in itself – is a significant advantage for diagnostic decision-making. Even when information is missing for some variables, the BN can still estimate the probability of the target disease based on the available evidence.

Challenge 3: Integrating multi-modal health data BNs's limited ability to handle realistic medical data has been cited as a major barrier to their wider adoption in clinical practice [Kyrimi et al., 2021c]. BNs perform well primarily in discrete settings with a relatively small number of variables, as in our pneumonia example. In such cases, the variable elimination procedure (which sums over all possible values of variables not included in the evidence or target set) is feasible. However, this exact inference procedure becomes computationally intensive and possibly infeasible when the BN contains many variables or when variables have many possible states. In these situations, approximate inference methods, such as Monte Carlo sampling or variational inference, can be employed [Koller and Friedman, 2009]. These methods, however, typically trade off transparency and simplicity for computational efficiency.

Inference becomes even more challenging when continuous variables are present in the network. These are variables that do not have a fixed number of discrete classes, but instead take values across a continuous range. For example, instead of representing fever as a binary variable, we could use temperature, expressed in degrees Celsius. When continuous variables are involved, exact inference becomes intractable, as it requires integrating over all possible values of the continuous variables rather than summing over a finite set of discrete states. In such cases, Bayesian inference must rely on sampling methods, variational approximations, or discretization strategies [Koller and Friedman, 2009].

A further obvious limitation in the context of medical data is that BNs cannot directly handle unstructured data, since all variables must be explicitly defined as either discrete states or continuous values. Text, images, or other unstructured data must first be transformed into structured features before they can be incorporated into the BN.

2.3 Neural networks

(Artificial) Neural Networks (NNs) are loosely inspired by how neurons in the human brain process information [Russell and Norvig, 2009]. They are trained to map a set of input signals to an output, effectively learning an arbitrary transformation from input to output. An NN consists of **layers of interconnected neurons**, where each neuron applies a simple transformation[1]:

$$output = activation\left(\sum_i w_i \times x_i + b\right) \tag{2.7}$$

In this expression, the input signals x_i are transformed into an *output* signal via a set of learnable parameters: the weights w_i and the bias b.

[1] Here, we aim only to provide intuition for how NNs operate, rather than a formal mathematical description.

The *activation* function, typically chosen to be non-linear, enables the NN to represent complex patterns that cannot be captured by a purely linear transformation.

By placing many neurons side by side within a layer, an NN can transform a set of input signals into a set of outputs through a matrix of learnable weights and biases. We can then stack these layers, by making the output of the previous layer the input to the next layer, forming a deeper NN. The network's parameters (i.e., all weights and biases across all neurons and layers) are learned through an iterative procedure known as **backpropagation**. Here, the NN is presented with a set of training inputs together with their corresponding desired outputs. The network produces its predictions, compares them to the true outputs, and then updates its parameters in the direction that reduces the error, based on the gradient of that error. Given enough neurons and enough training data, an NN can approximate any continuous function from inputs to outputs with arbitrary accuracy, according to the *universal approximation theorem* [Russell and Norvig, 2009].

Deep neural networks This principle forms the foundation of deep learning: by stacking many layers in an NN, the model can learn increasingly complex patterns from large amounts of data. Deep networks become even more powerful when their architecture incorporates an appropriate *inductive bias*: a set of assumptions that guide how the model generalizes beyond its training samples [Mitchell, 1980]. These biases can be built directly into the network's structure and are often tailored to a specific type of input data.

For example, convolutional NNs are highly effective for image processing because they embed two key inductive biases [Sarvamangala and Kulkarni, 2022]: *locality*, stating that nearby pixels in an image are more closely related than distant ones, and *translation equivariance*, stating that a meaningful visual feature (such as an edge or a corner) should be detectable regardless of where it appears in the image. Convolutional NNs realize these biases by connecting each neuron only to a small local region of the input (the receptive field) and by sharing weights across spatial positions.

Another example is transformers, which are designed for processing sequences such as natural language. Their central inductive bias is the ability to model *long-range dependencies* [Vaswani et al., 2017]. Through the attention mechanism, any element of an input sequence can directly relate to any other, allowing the model to capture context over long distances.

Integrating such inductive biases into NN architectures has led to the development of powerful building blocks for modern AI, including but not limited to convolutional NNs and transformers. These building blocks fuel today's dominant models, including large language models, which we discuss in the next section.

2.4 Large language models

NNs form the foundation of the Large Language Models (LLMs) widely used today. We begin by outlining, in broad terms, how NNs can be adapted and extended to obtain an LLM. We then discuss two specific forms of LLMs: foundation models and clinical embedding models.

2.4.1 From NNs to LLMs

There are three paradigm shifts which underlie the development of LLMs from NNs: increasing NN width and depth, introducing specialized neural modules, and training on large language corpora.

Increasing NN width and depth Deep NNs can be constructed by increasing both the depth and width of their layers. Greater width refers to having more neurons per layer, while greater depth refers to stacking more layers. As width and depth grow, so does the number of learnable parameters (i.e., weights). In this way, the number of parameters in LLMs can range from tens of millions to hundreds of billions or more [Wang et al., 2023a]. For example, it has been speculated that the number of parameters in GPT-4 (the primary backbone LLM used in ChatGPT until August 2025) is more than a trillion (10^{12}), though the exact number has not been disclosed.

Specialized neural modules The neural architecture behind LLMs is not simply a single large deep NN. Instead, it is composed of multiple neural modules, each serving a specific purpose, from tokenization (converting text into discrete tokens, such as words or subwords, for the model to process) to embedding (mapping text to vectors that capture semantic meaning) [Wang et al., 2023a]. A central module is the transformer, a type of NN that processes sequential data, such as text, in parallel. Unlike standard NNs, the transformer uses a self-attention mechanism to learn contextual relationships between tokens within a sequence [Vaswani et al., 2017]. LLMs stack many transformer blocks on top of one another to build increasingly rich representations of text.

Training on large language corpora A model with such a large number of parameters also requires an enormous amount of data to learn those parameters effectively and to accurately represent the phenomena it models. LLMs are trained on free text, which does not need to be structured or labeled, opening the door to vast textual data sources. Scientific literature, books, Wikipedia, social media, news websites, and many other types of publicly available text can all form part of an LLM's training corpus [Wang et al., 2023a].

2.4.2 Foundation models

A foundation model is a type of LLM that is pre-trained on large-scale, heterogeneous data and can be adapted to a wide range of downstream tasks. The most influential foundation models are sufficiently large and general to operate across multiple domains. For example, GPT-4 has been trained on data from many different fields and can be applied to tasks within those domains, including the medical domain [Wang et al., 2023a].

Medical foundation models Rather than building general-purpose models, it can be more effective to develop models that are specialized for a single domain. This is the philosophy behind medical foundation models [Wang et al., 2023a, Shah et al., 2023]. These models are trained on clinical corpora, which may include biomedical research publications, clinical notes, and other forms of EHR data. Examples of medical foundation models include:

- BioBERT, which is trained on PubMed abstracts and PMC full-text articles [Lee et al., 2020].
- ClinicalBERT, which is trained on clinical text from the MIMIC-III database, containing real-world EHR records from over 40,000 patients [Huang et al., 2019].
- Med-PaLM, which is adapted from a general-purpose LLM (PaLM) by finetuning on in-context medical examples and optimizing it for medical question answering [Singhal et al., 2023].
- MedGemma, which is trained on a mixture of text-only datasets (such as medical question-answer pairs) and multimodal datasets containing both medical images and reports from domains such as radiology and histopathology [Sellergren et al., 2025].

Fine-tuning vs. prompting There are two main approaches to adapting foundation models for specific tasks.

The first approach is *fine-tuning*, where a model is initially trained on general-purpose data (a process that requires substantial computational resources) and then further trained on a smaller set of task-specific examples. In this second step, the model's weights are adjusted to optimize performance for the target task. For instance, BioBERT was first trained on a large corpus of biomedical literature, giving it a general understanding of biomedical concepts and semantics [Lee et al., 2020]. This foundation allows it to be fine-tuned for specific tasks in the domain, such as named entity recognition in biomedical text.

The second approach is *prompting*, in which a large model that is already adept at following instructions is provided with a detailed description of a specific task. Unlike fine-tuning, prompting does not modify the model's parameters; instead, the model leverages its existing knowledge at inference time to perform the task. For example, Med-PaLM has been instruction

fine-tuned with in-context medical examples [Singhal et al., 2023], after which it can be prompted to perform specific tasks by including instructions and a few (typically one to ten) in-context examples. This paradigm is familiar to users of commercial LLMs, such as ChatGPT.

Recently, we have been seeing a shift from fine-tuning toward prompting. Prompting is more intuitive, requires only a handful of in-context examples, and avoids the extensive computational cost associated with fine-tuning. Fine-tuning an LLM often requires not only hundreds or thousands of task-specific examples but also substantial hardware resources, including large amounts of memory and access to high-performance GPUs.

Clinical applications In the medical domain, foundation models and LLMs have been applied to a wide variety of tasks [Wang et al., 2023a]. These include text classification, such as the automated assignment of disease codes like ICD-10 [Rasmy et al., 2021]; automated radiology report generation [Bannur et al., 2024], where a vision-language foundation model can produce a report directly from a radiology image; and clinical information extraction [Wang et al., 2018, Xu et al., 2024a], which will be discussed in further detail in Section 2.5. Other applications include automated summarization of patient records, support for clinical decision-making through question answering, and medical knowledge retrieval from large corpora of scientific literature [Wang et al., 2023a, Xu et al., 2024c].

2.4.3 Clinical embedding models

In addition to task-specific adaptation, some foundation models are designed to primarily produce high-quality clinical embeddings [Wang et al., 2023a]. **Clinical embeddings** are numerical representations of medical concepts or text that capture their meaning in a way that is easily reusable for downstream tasks. By converting sentences or entire clinical reports into embeddings, texts with similar meaning are represented by vectors that are close to each other in the embedding space, allowing algorithms to identify and compare semantically similar content.

It is important to note that while all foundation models learn internal embeddings of text, not all are optimized to produce embeddings that are directly useful for downstream tasks. General-purpose LLMs, for instance, are primarily optimized to generate plausible and helpful responses to instructions, rather than embeddings tailored for tasks like classification. To address this need, specialized clinical embedding models have been developed, which are either smaller or adapted foundation models explicitly trained to produce embeddings that capture meaningful clinical semantics.

Examples of such models include BioBERT [Lee et al., 2020] and ClinicalBERT [Huang et al., 2019], which are pre-trained on large biomedical and clinical corpora, as well as BioLORD, a model specifically designed to generate high-quality clinical embeddings. BioLORD achieves richer semantic

representations of clinical sentences and biomedical concepts by grounding them in definitions derived from established clinical vocabularies and knowledge graphs, including SNOMED CT and UMLS [Remy et al., 2024].

Clinical embeddings can be leveraged for a variety of downstream tasks, including sentence or text classification. For instance, text embedding models provide a cost-effective approach to text classification. In this workflow, a clinical embedding model is used to generate a static text embedding, which then serves as input to a lightweight model, such as a small feed-forward NN, trained for the specific classification task. In Chapters 5, 6 and 7, we will employ this approach to build simple text classifiers, as it is much more accessible and resource-efficient than fine-tuning a full foundation model for classification.

2.4.4 LLMs as clinical decision support systems

To illustrate the potential role of LLMs in clinical decision support, we discuss how LLMs (and in a broader sense, NNs) are equipped or underequipped to address the challenges raised in Section 1.3.

Challenge 1: Building trust through interpretable and expert-guided decisions Due to their immense size and vast parameter space, LLMs are quintessential black-box AI models and are therefore not inherently interpretable. Considerable research has focused on enhancing their explainability [Zhao et al., 2024], with chain-of-thought prompting emerging as a notable approach. *Chain-of-thought prompting* instructs the LLM to produce a series of intermediate reasoning steps when generating an answer [Wei et al., 2022]. This technique has been shown to improve performance on tasks requiring complex reasoning and to make the LLM's outputs more comprehensible to humans. However, while it encourages users to scrutinize the model's "thinking" process, it does not render the LLM truly interpretable. The chain-of-thought is part of the model's generated output and does not reveal the actual inner workings of the LLM, which remain inherently opaque.

We can also consider the role of expert knowledge in LLMs. Expert sources, such as biomedical articles or clinical guidelines, are often included in their training corpora, allowing the LLM to implicitly encode expert knowledge. Some approaches go further by incorporating structured expert knowledge during training. For example, BioLORD uses expert-defined clinical ontologies to ground medical concepts in their definitions [Remy et al., 2024]. However, expert knowledge can never be strictly enforced or explicitly encoded in an LLM's large architecture, because it is distributed across millions of parameters rather than stored as discrete, enforceable rules. Consequently, there are no formal guarantees that an LLM will retain the expert knowledge it has seen during training or apply it correctly to unseen scenarios.

Challenge 2: Modeling clinical reasoning under uncertainty While it has been suggested that chain-of-thought prompting enables LLMs to mimic the (clinical) reasoning process of clinicians [Savage et al., 2024], an LLM is essentially reproducing patterns it has seen during training by performing large-scale text prediction. There are therefore no guarantees that it is truly engaging in clinical reasoning. Indeed, a recent study comparing the diagnostic decision-making capabilities of various LLMs concluded that they performed significantly worse than physicians [Hager et al., 2024]. Furthermore, LLMs lack a formal representation of probability or uncertainty – both statistical concepts that are notoriously difficult to express in natural language [Reiner, 2018, Turner et al., 2021]. Accordingly, research indicates that LLMs can be misled by non-decisive clinical cues in patient presentations, disproportionately favoring rare diagnoses over more prevalent conditions that provide a better overall fit [Omar et al.].

Stepping down from LLMs, one could argue that simple feed-forward NNs for classification tasks are better equipped to handle uncertainty, as they typically output class probabilities. However, these probabilities are often poorly calibrated [Guo et al., 2017], meaning they do not reliably reflect the true likelihood of an event. For example, a NN trained to predict sepsis may assign a patient a 90% probability of developing the condition. If, in reality, only 50% of patients with similar scores actually develop sepsis, the model is poorly calibrated. Correct calibration of probabilities is crucial for reliable clinical decision-making.

Challenge 3: Integrating multi-modal health data LLMs are particularly well-suited for handling unstructured text data, as this is precisely the type of input they are trained to process. It goes without saying that the current generation of powerful foundation models surpasses previous AI models in their natural language processing capabilities. Moreover, multimodal foundation models, such as MedGemma [Sellergren et al., 2025], are designed to integrate information from both medical text and medical images, enabling them to learn unified representations across multiple modalities.

However, less focus has gone to integrating the structured, tabular portion of the EHR into multi-modal foundation models. For instance, while the textual component of EHRs was included in MedGemma's training set, the tabular data were not [Sellergren et al., 2025]. Although smaller embedding models have successfully incorporated structured EHR data – such as BEHRT [Li et al., 2020], MedM-PLM [Liu et al., 2022b], MedBERT [Rasmy et al., 2021], and GatorTron [Yang et al., 2022] – larger foundation models seem to have neglected this resource, favoring training on unstructured text and image data instead.

2.5 Clinical information extraction

So far, we have discussed Bayesian networks, which handle structured data in an interpretable, expert-informed way, and large language models, which excel at processing unstructured text data. We now turn to another perspective: transforming unstructured data into structured data through clinical information extraction. This approach allows us to leverage the rich information in unstructured text while enabling the use of more interpretable downstream models.

Clinical Information Extraction (CIE) is the process of extracting and encoding information from clinical text to convert it into a structured, machine-learnable format that can be used for downstream tasks [Wang et al., 2018]. This includes applications such as clinical decision support systems that operate on structured inputs, as well as clinical research purposes, including epidemiology, disease studies, and drug safety analysis. Typically, CIE involves multiple sub-tasks [Wang et al., 2018], which we illustrate with an example.

Example: Clinical information extraction

Clinical note: *"A 68-year-old patient with a history of COPD presents with a persistent cough. These symptoms have been gradually worsening over the past two weeks, raising concerns about a possible exacerbation of chronic obstructive pulmonary disease."*

First, *named entity recognition* identifies concept mentions or entity names in the text. This includes disease names like "COPD" and "chronic obstructive pulmonary disease", as well as symptom mentions like "cough". Then, *coreference resolution* links mentions that refer to the same semantic entity. For example, it can determine that "COPD" and "chronic obstructive pulmonary disease" refer to the same disease, and that "cough" and "these symptoms" refer to the same symptom. Finally, *relation extraction* identifies relationships between concepts, entities, and their attributes. For example, it might infer that "exacerbation" is an attribute of "chronic obstructive pulmonary disease" indicating disease progression.

Evolution of CIE tools The tools to perform (clinical) information extraction have evolved throughout the years.

Originally, CIE was addressed using a modular pipeline, where each subtask was handled by a dedicated module [Wang et al., 2018]. For example, named entity recognition could be performed using rule-based methods or statistical approaches such as conditional random fields. These early methods required substantial human intervention, including the design of rules and manual feature engineering for statistical models (e.g., prefix/suffix features, word type features, etc.). Relation extraction, in turn, might rely on

classifiers such as logistic regression or support vector machines, which predicted the presence of a relation between two extracted entities based on manually engineered features.

With the emergence of deep NNs, manual feature engineering was largely replaced by pretrained feature representations, though task-specific modules initially remained. As foundation models became more widespread, end-to-end neural approaches emerged [Hahn and Oleynik, 2020]. Models like BioBERT were pre-trained to capture internal semantic representations of (bio)medical concepts and can be fine-tuned for specific information extraction sub-tasks [Lee et al., 2020]. Consequently, variations of the same underlying architecture could be applied to multiple components of the CIE pipeline, reducing the need for separate hand-crafted modules.

The rise of LLMs has further shifted the paradigm toward generative information extraction [Xu et al., 2024b]. In this approach, LLMs are prompted to perform CIE according to a predefined schema specifying which entities and relations to extract, often including in-context examples. This allows all sub-tasks (including named entity recognition, coreference resolution, and relation extraction) to be performed in a single generative pass, eliminating the need for separate pipeline components.

(Dis)advantages of structuring text with CIE The primary advantage of CIE is that it enables the use of information from unstructured text in interpretable models, such as Bayesian networks or decision trees, which require structured, tabular inputs. Additionally, experts can be involved in defining a schema that specifies the types of information to extract, including relevant concepts and their attributes. However, structuring text with CIE inevitably entails some loss of information, as it is impossible to capture all the nuances of natural language within a predefined schema. If CIE is performed poorly, any downstream analyses or models built on the structured data may rest on shaky foundations.

2.6 Neuro-symbolic AI

In Section 2.1, we introduced two broad categories of AI approaches: knowledge-based and data-driven. In an effort to combine the strengths of traditional knowledge-based (or *symbolic*) AI with modern data-driven (or *neural*) AI, the field of Neuro-Symbolic (NeSy) AI was established [Marra et al., 2024]. NeSy AI serves as an umbrella term for any method that integrates symbolic reasoning into neural models, and our contributions in Chapters 4 through 7 can also be considered part of this paradigm.

Examples A notable contribution to the field of NeSy AI is DeepProbLog [Manhaeve et al., 2018c]. ProbLog is a well-established probabilistic logic programming language within symbolic AI, allowing expert-defined logical

rules (predicates) to be expressed formally and extended with probabilities to support reasoning under uncertainty. DeepProbLog augments this framework with *neural predicates*, whose truth values are provided by an NN. In this way, the probabilistic logical predicates encode expert or domain knowledge, while the neural components enable the system to process unstructured data. Importantly, the parameters of both the probabilistic predicates and the NN can be learned jointly from data, allowing end-to-end differentiable training.

Another notable example are concept bottleneck models [Koh et al., 2020, Barbiero et al., 2024]. These models introduce an interpretable set of human-understandable concepts as a bottleneck layer within a neural architecture. They typically consist of two components: an encoder, in which an NN maps the input into higher-level concepts, and a predictor, which generates the final output from these internal concepts. Because the bottleneck concepts must be meaningful to humans, the training data must include labels for them. The parameters of both the encoder and the predictor can be learned jointly, enabling end-to-end training while maintaining interpretability.

(Dis)advantages of NeSy AI Integrating symbolic reasoning into neural models makes it possible to inject expert-defined rules or concepts directly into the system, while still retaining some of the flexibility of neural approaches for handling unstructured data. Because key rules are provided explicitly rather than learned implicitly, NeSy AI methods can also be more data-efficient than purely neural models. However, this increased interpretability and expert guidance comes at a cost. Enforcing symbolic rules constrains the neural components, which can limit their predictive accuracy and representational power. Moreover, the requirement for expert involvement in defining rules or concepts means that many promising NeSy AI approaches have only been demonstrated on small-scale or synthetic problems. Finally, the integration of symbolic reasoning engines often makes these methods less computationally efficient than end-to-end neural architectures.

3 Contributions

In Chapter 1, we introduced three major challenges for AI in healthcare. Chapter 2 explained why BNs are well equipped to address the first challenge, which emphasizes the need for interpretable and expert-informed clinical decision support. They are also naturally suited to meet the second challenge, as they provide a principled framework for clinical reasoning under uncertainty. However, BNs currently fall short on the third challenge: they cannot handle multi-modal EHR data, particularly unstructured text, because they require structured input features. Neural models, like LLMs, are much better suited to deal with this challenge.

To summarize, BNs already have the capabilities to address challenge 1 and 2, but they are currently hampered by their inability to deal with challenge 3. This dissertation therefore addresses the following central research question: **How can we augment Bayesian networks with textual data for interpretable, expert-based, and uncertainty-aware AI in the medical domain?**

Before breaking down this overarching question into more specific research questions addressed in the subsequent chapters, we first clarify the scope of our research.

3.1 Data scope

As mentioned in Section 1.1, EHR data spans multiple modalities, which can be broadly categorized into structured and unstructured data. In this work, we focus specifically on structured *tabular* data and unstructured *text*

data.

Structured tabular data refers to information that can be arranged in a table, where each row represents a patient and each column corresponds to a feature such as a past diagnosis, symptom, or clinical parameter. For the purposes of this work, we treat this data as static. This is a simplification of reality, where a patient's clinical state evolves continuously over time.

Unstructured text data consists of free-form clinical narratives written by caregivers to document patient encounters. These notes often contain rich contextual information, including the reasoning behind clinical decisions, details about symptom severity or onset, or justification for diagnoses and referrals. Clinical notes may also partially overlap with structured data: for example, discharge summaries may record clinical parameters that also appear in tabular form, though often with varying levels of consistency between modalities [Kwon et al., 2024].

Because BNs readily accommodate structured tabular inputs, we concentrate on incorporating unstructured text so that clinical decision support can leverage both tabular and textual evidence. We prioritize text over other unstructured modalities (which are also part of challenge 3) because it is the dominant unstructured data type in healthcare and captures essential information for clinical decision making [Tayefi et al., 2021].

3.2 Research questions

In this dissertation, each chapter will tackle a specific research question, as shown in Figure 3.1.

RQ1: Can neural networks mimic the causal and probabilistic properties of a Bayesian network? (Chapter 4)

In Chapter 4, we investigate whether a BN is truly necessary to address challenge 1 and challenge 2, or whether an NN can mimic the key properties of a BN. To explore this, we construct a *neural Bayesian network understudy* – an NN trained to output conditional probabilities and thereby approximate the functionality of a given BN. We further incorporate the conditional independence relations encoded by the BN directly into the NN's training process.

Our results show that the NN can indeed serve as an understudy to the BN, both approximating the probabilistic behavior and being informed by the causal properties of its BN counterpart. However, we foresee that training of the neural understudy will become increasingly inefficient as the

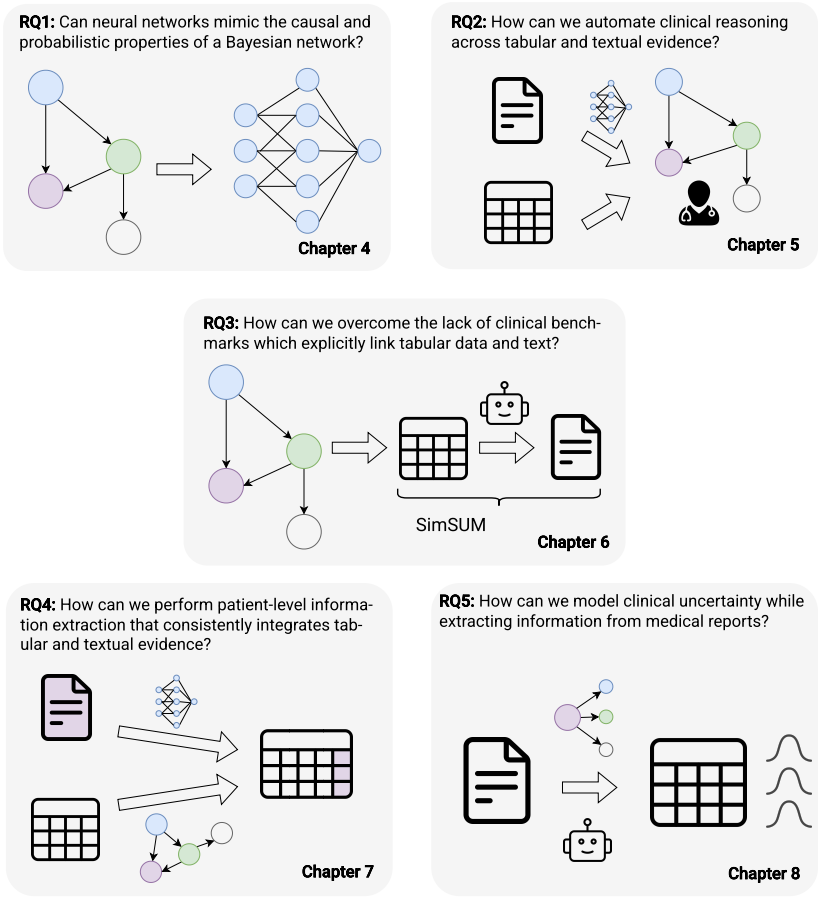


Figure 3.1: Overview of the research questions explored in each chapter of this dissertation.

size of the BN grows, due to the combinatorial listing of all possible conditional probabilities at its input. Furthermore, the neural understudy lacks the interpretability of a BN, as it embeds expert knowledge only implicitly, rather than representing it explicitly within its architecture.

RQ2: How can we automate clinical reasoning across tabular and textual evidence? (Chapter 5)

In Chapter 5, we show how a BN can be used to automate the clinical reasoning process in a use-case emulating the diagnosis of respiratory diseases in primary care. We further augment the BN with neural text representations, enabling it to incorporate not only structured tabular evidence but

also textual evidence.

We explore two strategies for integrating neural text representations into the BN: a generative approach, in which we model a Gaussian distribution over pre-trained text embeddings, and a discriminative approach, in which a neural classifier predicts symptom classes from pre-trained text embeddings. In both cases, the outputs of the neural text models are incorporated directly into the Bayesian inference procedure, yielding a natural and principled extension of the BN. Our results demonstrate the strengths of BNs for formalizing the clinical reasoning process, as well as the clear added value of incorporating unstructured text compared to relying solely on the available structured features.

In this chapter, we first encounter the need for a benchmark dataset linking structured tabular data and unstructured text, which we design for the use case of respiratory diseases. We further extend this benchmark in the next chapter, which brings us to the following research question.

RQ3: How can we overcome the lack of clinical benchmarks which explicitly link tabular data and text? (Chapter 6)

In Chapter 6, we build a benchmark dataset that links structured tabular data with unstructured text for the domain of respiratory diseases. While a limited number of real-world benchmark datasets exist that contain both modalities (most notably MIMIC-III and MIMIC-IV [Johnson et al., 2016, 2023]) they are too complex to provide the controlled research environment needed for developing and prototyping novel methods for integrating tabular and textual information. To address this gap, we created SimSUM, a self-contained simulated dataset comprising both structured and unstructured medical records.

By design, the clinical concepts expressed in the unstructured text and encoded in the structured tabular portion of SimSUM are linked through a BN that captures relevant domain knowledge for respiratory diseases. Generating the tabular data via a BN and the textual data via a prompt-steered LLM provides full control over the data-generation process and ensures that this process remains transparent to the user. We conduct an extensive evaluation of the dataset's utility, including an expert assessment of the generated clinical notes, a descriptive analysis of their content, and baseline experiments using tabular-only and text-only models for a basic symptom-prediction task.

RQ4: How can we perform patient-level information extraction that consistently integrates tabular and textual evidence? (Chapter 7)

As explained in the previous two chapters, interpretable AI models usually require structured tabular data. Through clinical information extraction (CIE, as introduced in Section 2.5), we can transform unstructured text data into structured tabular data, enabling its use in interpretable downstream models.

In Chapter 7, we demonstrate how to integrate the predictions of both a BN and a neural text classifier to perform *patient-level information extraction* on the SimSUM dataset. While conventional CIE methods focus on extracting clinical concepts solely from unstructured text, our approach additionally leverages the structured tabular features already present in a patient's EHR. In our framework, these tabular features are modeled using a BN, whereas symptom information is extracted from the unstructured text using a neural classifier built on pre-trained text embeddings. To fuse the predictions from both models in an interpretable and probabilistic manner, we introduce the use of *virtual evidence* within the BN, augmented with a *consistency node*. This design allows the BN to modulate the neural classifier's outputs, enabling it to handle missing information in the text and to resolve contradictions between the textual and tabular evidence.

RQ5: How can we model clinical uncertainty while extracting information from medical reports? (Chapter 8)

In research questions **RQ2**, **RQ3**, and **RQ4**, we relied on expert knowledge to define a BN (or at least its structure) to model uncertainty across both tabular data and text. In practice, asking an expert to define a full BN for a given use case is challenging, as it requires consideration of many relevant differential diagnoses and background factors that shape the diagnostic landscape for a set of diseases.

In Chapter 8, we relax the requirements of a BN and instead ask experts to define *diagnostic pathways*: sets of deterministic, top-down relations from differential diagnoses to significant findings. Because these pathways are not complete, they cannot reason bottom-up from findings to diagnoses in the same way a BN would. Instead, we use the diagnostic pathways to fill in gaps in clinical reporting, addressing the *implicit uncertainty* that arises when clinicians omit portions of their reasoning.

Furthermore, since we cannot use a BN to predict probabilities in this context, we adopt a complementary approach to quantify *explicit uncertainty* in clinical text. By analyzing the language used in clinical reports to convey doubt about the presence or absence of a finding, we can quantify the confidence associated with each reported finding.

By combining these methods for modeling implicit and explicit uncertainty, we can extract structured, uncertainty-aware information from unstructured text. We specifically apply our methods to radiology reports, where both implicit and explicit uncertainty are commonplace and convey important information about the clinical reasoning process of the reporting radiologist.

3.3 Publications

Below, we provide a full list of the research output produced during this PhD. Not all of these works are directly related to the research questions addressed in this dissertation, and are therefore not discussed further.

Publications in International Journals

1. Géraldine Deberdt, **Paloma Rabaey**, Sam Van Damme, Laure-Ann Quenon, Peter Decat, Thomas Demeester (2026). *"Using Causal Effect Estimation To Evaluate Occupational Stress Factors During A Weekend On-Call Shift Among General Practitioners in Training"*. In: *BMC Primary Care* 27(22). `https://link.springer.com/article/10.1186/s12875-025-03140-1`.

2. **Paloma Rabaey**, Stefan Heytens, Thomas Demeester (2025). *"SimSUM – Simulated Benchmark with Structured and Unstructured Medical Records"*. In: *Journal of Biomedical Semantics* 16(20). Also presented at: *Workshop on Large Language Models and Generative AI for Health (GenAI4Health) at AAAI 2025*. `https://link.springer.com/article/10.1186/s13326-025-00341-6`.

3. Heidelinde Dehaene, Alexander Decruyenaere, Christiaan Polet, Johan Decruyenaere, **Paloma Rabaey**, Thomas Demeester, Stijn Vansteelandt (2024). *"Why synthetic discoveries are not only a problem of differentially private synthetic data"*. In: *Methods of Information in Medicine* 63(05/06), p.203-204. `http://hdl.handle.net/1854/LU-01JSYF40Z4CPDST339VVEPW108`.

4. **Paloma Rabaey**, Peter Decat, Stefan Heytens, Dirk Vogelaers, An Mariman, Thomas Demeester (2024). *"Time-dependent complexity characterisation of activity patterns in patients with Chronic Fatigue Syndrome"*. In: *Biopsychosocial Medicine* 18(1). `https://doi.org/10.1186/s13030-024-00305-9`.

Publications in International Conferences

* denotes equal contribution

1. Max Hallemeesch*, Marija Pizurica*, **Paloma Rabaey**, Olivier Gevaert, Thomas Demeester, Kathleen Marchal (2025). *"Prior knowledge injection into deep learning models predicting gene expression from whole slide images"*. In: *Workshop on Imageomics at AAAI 2025*. https://arxiv.org/abs/2501.14056

2. Alexander Decruyenaere*, Heidelinde Dehaene*, **Paloma Rabaey**, Johan Decruyenaere, Christiaan Polet, Thomas Demeester, Stijn Vansteelandt (2024). *"Debiasing synthetic data generated by deep generative models"*. In: *Advances in Neural Information Processing Systems 37 (NeurIPS 2024)*. https://papers.nips.cc/paper_files/paper/2024/hash/4902603fe8cb095b9ada707a19bd151c-Abstract-Conference.html

3. Henri Arno, **Paloma Rabaey**, Thomas Demeester (2024). *"From text to treatment effects: a meta-learning approach to handling text-based confounding"*. In: *Workshop on Causal Representation Learning at NeurIPS 2024*. https://arxiv.org/abs/2409.15503

4. Alexander Decruyenaere*, Heidelinde Dehaene*, **Paloma Rabaey**, Christiaan Polet, Johan Decruyenaere, Stijn Vansteelandt, Thomas Demeester (2024). *"The real deal behind the artificial appeal: Inferential utility of tabular synthetic data"*. In: *Uncertainty in Artificial Intelligence (UAI 2024)*. Proceedings of Machine Learning Research (PMLR) 244, p. 966-996. https://dl.acm.org/doi/10.5555/3702676.3702722

5. **Paloma Rabaey**, Johannes Deleu, Stefan Heytens, Thomas Demeester (2024). *"Clinical reasoning over tabular data and text with Bayesian networks"*. In: *Artificial Intelligence in Medicine (AIME 2024)*. Lecture Notes in Artificial Intelligence 14844, p.229-250. https://arxiv.org/abs/2403.09481

6. Alexander Decruyenaere*, Heidelinde Dehaene*, **Paloma Rabaey**, Christiaan Polet, Johan Decruyenaere, Stijn Vansteelandt, Thomas Demeester (2023). *"Synthetic data: can we trust statistical estimators?"*. In: *Workshop on Deep Generative Models for Health at NeurIPS 2023*. https://openreview.net/pdf/f9e325408e3dcdce50e66442dc99e238cd133cdd.pdf

7. **Paloma Rabaey**, Cedric De Boom, Thomas Demeester (2022). *"Neural Bayesian network understudy"*. In: *Workshop on Causal Machine Learning for Real-World Impact (CML4Impact) at NeurIPS 2022*. https://arxiv.org/abs/2211.08243

8. Mathieu De Coster, Karel D'Oosterlinck, Marija Pizurica, **Paloma Rabaey**, Severine Verlinden, Mieke Van Herreweghe, Joni Dambre

(2021). *"Frozen Pretrained Transformers for Neural Sign Language Translation"*. In: *18th Biennial Machine Translation Summit*, pp. 88–97. https://aclanthology.org/2021.mtsummit-at4ssl.10

Preprints

* denotes equal contribution

1. **Paloma Rabaey***, Adrick Tench*, Stefan Heytens, Thomas Demeester (2025). *"Patient-level Information Extraction by Consistent Integration of Textual and Tabular Evidence with Bayesian Networks"*. In: *arXiv preprint arXiv:2511.17056*. https://arxiv.org/abs/2511.17056

2. **Paloma Rabaey***, Jong Hak Moon*, Jung-Oh Lee, Min Gwan Kim, Hangyul Yoon, Thomas Demeester, Edward Choi (2025). *"Modeling Clinical Uncertainty in Radiology Reports: from Explicit Uncertainty Markers to Implicit Reasoning Pathways"*. In: *arXiv preprint arXiv:2511.04506*. https://arxiv.org/abs/2511.04506

3. Jong Hak Moon, Geon Choi, **Paloma Rabaey**, Min Gwan Kim, Hyuk Gi Hong, Jung-Oh Lee, Hangyul Yoon, Eun Woo Doe, Jiyoun Kim, Harshita Sharma, Daniel C. Castro, Javier Alvarez-Valle, Edward Choi (2025). *"Lunguage: A Benchmark for Structured and Sequential Chest X-ray Interpretation"*. In: *arXiv preprint arXiv:2505.21190*. https://arxiv.org/abs/2505.21190

Science Communication

Paloma Rabaey (2023). *"Dokter chatbot"*. In: *Eos blogs*. https://www.eoswetenschap.eu/technologie/dokter-chatbot

4

Neural Bayesian Network Understudy

This chapter investigates whether a neural network can effectively replace a Bayesian network. We introduce a neural Bayesian network understudy: a neural model trained to output conditional probabilities while respecting the Bayesian network's conditional independence structure. This allows us to assess how well such a network can reproduce the probabilistic and causal behavior of its Bayesian network counterpart. While the understudy performs well in mimicking these properties, it remains less interpretable than Bayesian networks, as expert knowledge is encoded only implicitly.

* * *

Paloma Rabaey, Cedric De Boom, Thomas Demeester

Proceedings of the Workshop on Causal Machine Learning for Real-World Impact (CML4Impact), NeurIPS 2022

Abstract

Bayesian Networks may be appealing for clinical decision-making due to their inclusion of causal knowledge, but their practical adoption remains limited as a result of their inability to deal with unstructured data. While neural networks do not have this limitation, they are not interpretable and are inherently unable to deal with causal structure in the input space. Our goal is to build neural networks that combine the advantages of both approaches. Motivated by the perspective to inject causal knowledge while

training such neural networks, this work presents initial steps in that direction. We demonstrate how a neural network can be trained to output conditional probabilities, providing approximately the same functionality as a Bayesian Network. Additionally, we propose two training strategies that allow encoding the independence relations inferred from a given causal structure into the neural network. We present initial results in a proof-of-concept setting, showing that the neural model acts as an understudy to its Bayesian Network counterpart, approximating its probabilistic and causal properties.

4.1 Introduction

Bayesian Networks (BNs) have been the subject of a large body of research, and are considered an important technology for high-stakes decision making, in particular in the healthcare domain. Kyrimi et al. [2021e] provide a clear overview of the key properties that make BNs highly appealing in that context: (i) their ability to model complex problems involving causal dependencies as well as uncertainty, (ii) their ability to combine data and expert knowledge, (iii) their interpretable graphical structure, and (iv) their ability to model interventions and reason diagnostically as well as prognostically. Despite efforts from the research community to support healthcare practitioners in using BNs [Arora et al., 2019b, Kyrimi et al., 2020], their practical adoption remains very limited. The analysis of Kyrimi et al. [2021c] indicates data inadequacies as one of several key limiting factors: BNs have a limited capacity to address continuous data, and cannot directly deal with unstructured data such as images or free text. According to Tayefi et al. [2021], the value of information entailed by unstructured text fields in electronic medical records for clinical decision support cannot be underestimated.

In contrast to BNs, neural networks are well-suited for dealing with unstructured data. However, they are inherently not interpretable, which stands in the way of clinical adoption [Montani and Striani, 2019, Peiffer-Smadja et al., 2020b]. Whereas Hinton [2018] argues strongly in favour of the adoption of neural networks in healthcare, he also recognises interpretability as a property that is desired by practitioners.

Our envisioned research goal is to build models that combine the advantages of both approaches: a fully neural model that can be trained on observations including unstructured data, but that also adopts the desirable properties of BNs as listed above. Realistically speaking, we can at best expect only approximate probabilistic reasoning capabilities from a generic neural network, and it will lack the direct interpretability of a BN. However, if the model displays the causal properties of a BN, the ideas of counterfactual explanation [Wachter et al., 2017] can be applied to explain its predictions.

This paper forms the first step along that outlined research track. We focus on the problem of training a neural network on a given set of discrete data samples as well as knowledge of a Directed Acyclic Graph (DAG) [Russell and Norvig, 2009] reflecting causal relationships between the variables. We do not yet aim to augment such models with unstructured data, but for now require that it behaves as a BN with the given DAG, if we were to train it on the same samples. In this sense, our model can be considered a *neural understudy* to the BN. We want to emphasise that our current preliminary work does not aim to build a neural network which outperforms its BN counterpart, but one that approximates its capabilities. After this necessary first step, we expect to reap the benefits of the neural network understudy by incorporating unstructured data. While the current work presents results only in a proof-of-concept setting, the possibility to reason under uncertainty, while combining causal structure and unstructured data, has great potential in real-world (clinical) applications.

In short, this paper makes the following contributions: (1) we propose a neural network model able to infer the probability of a set of target variables conditioned on observed evidence, (2) we introduce two approaches for injecting DAG knowledge while training the neural network, and (3) we present empirical results (including robustness analysis) in a small proof-of-concept setting. The necessary code to reproduce these results can be found in our Github repository.[1]

4.2 Related work

This section focuses on recent works most related to the proposed ideas; we do not aim to provide a broad overview in terms of related research fields. A basic premise for the presented work is the existence of causal knowledge in addition to observational data, to be injected into a neural network model. Our goal is therefore not *causal discovery*, intended to recover the causal mechanisms underlying the distribution from which the observed data was generated [Ke et al., 2019, Brouillard et al., 2020].

Geiger et al. [2022] present a method to align different parts of a neural network with nodes in a causal graph, resulting in a model with improved interpretability. Harradon et al. [2018] pursue a similar goal, summarising a neural model into a Bayesian causal model, to provide counterfactual explanations for the neural model. Instead of starting from a neural network and using causal models to understand or guide its inner workings, we work the other way round.

In Rohekar et al. [2018], conditional independencies in the input distribution are encoded hierarchically in the network structure, as a way of performing unsupervised neural network structure learning. Pawlowski et al. [2020] and Parafita and Vitrià [2020] present each parent-child relation in a given structural causal model as a deep neural network unit. While we

[1] `https://github.com/prabaey/NBN-understudy`

don't explicitly align parts of the neural network with a causal model, we do include structural information in the training process as a regulariser. This is related to the work from Kyono et al. [2020], who propose a prediction task with a regularisation objective to detect causal relationships between features. It is not based on prior knowledge of causal relationships, as in our work.

Attempts have been made to build neural networks that implement belief propagation [Pearl, 1988b] to improve on accuracy and efficiency [Garcia Satorras and Welling, 2021, Kuck et al., 2020]. Our proposed neural networks also aim at answering probabilistic queries, but can freely learn the inference mechanics from the data.

Another key related contribution is DeepProbLog [Manhaeve et al., 2018c]. This neuro-symbolic model is able to implement our envisioned goal: its probabilistic logic programming core can represent any BN exactly, and its so-called neural predicates are able to encode unstructured data. However, whereas a fully neural model can at best reason only approximately with probabilities, it may have some benefits over the exact probabilistic reasoning component in DeepProbLog. Indeed, in future research we aim to model complex realistic observational data. Some of its properties a suitable model should be able to capture, may be hard to express in a probabilistic program.

Finally, our work is not to be confused with the so-called *Bayesian neural networks* MacKay [1992], aiming to obtain a probability distribution over neural networks. Instead, our goal is training a neural understudy for a Bayesian network, i.e., a neural model that behaves similarly.

4.3 Training a neural understudy of a Bayesian network

The following paragraphs describe our proposed neural architecture with full flexibility in the choice of input and output variables (Section 4.3.1), followed by our proposed training strategies to inject causal structure into the model (Section 4.3.2).

4.3.1 Neural architecture and training

A fully specified BN can be queried by providing inputs to any selection of variables (henceforth called the *evidence*), after which the probabilities for each of the remaining variables (the *targets*) can be inferred, conditioned on the evidence. We will require the same ability from our neural counterpart, and show that this can be achieved by using complementary masks at its inputs and outputs. This only induces restrictions on the input and output layers but not on the internal neural architecture.

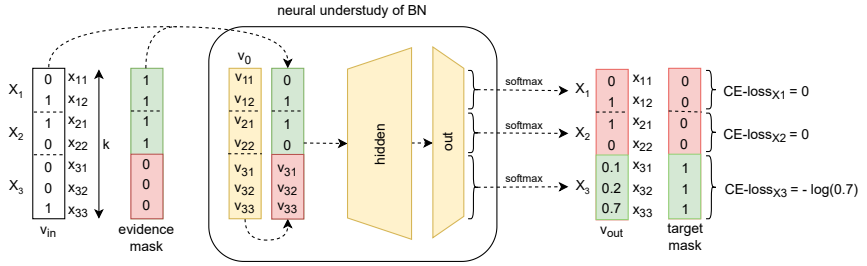


Figure 4.1: Illustration: training with evidence $\mathcal{E} = \{X_1, X_2\}$ and targets $\mathcal{T} = \{X_3\}$, for $P(X_3|X_1 = x_{12}, X_2 = x_{21})$. The mask at the input (output) indicates selection of evidence (target) entries.

We focus on problems involving a set of N discrete variables $\mathcal{V} = \{X_1, X_2, ..., X_N\}$, in which X_i can take n_i possible values in $\mathcal{X}_i = \{x_{i1}, ..., x_{in_i}\}$. Given a set $\mathcal{E} \subset \mathcal{V}$ of m evidence variables ($m < N$), each with an assigned value $\{E_1 = e_1, ..., E_m = e_m\}$ (for convenience written as $\mathcal{E} = e$), our goal is to train a neural network to predict the probability distribution $P(X|\mathcal{E} = e)$ for each of the remaining variables X. The latter are called the target variables, aggregated in the set $\mathcal{T} = \mathcal{V} \setminus \mathcal{E}$.

After training on an observed set of individual samples of the form $\{X_1 = x_1, \ldots, X_N = x_N\}$, our model should be able to deal with any selection of evidence variables. The key idea to achieve that is through dynamic masking (i.e., a different mask for any variable split $\mathcal{E}$ vs. $\mathcal{T}$): all variables are present at the model input and output, but target variables are filtered out of the inputs through an evidence mask, whereas predictions for the evidence variables are filtered from the outputs through a target mask. This is illustrated in Fig. 4.1.

Model architecture: Each input to the model is a vector v_{in} of dimension $k = \sum_{i=1}^{N} n_i$, and is formed by concatenating the n_i-sized one-hot representation of the value held by each of the N variables X_i ($i = 1, \ldots, N$). The masked positions of the target variables in v_{in} are substituted by the corresponding components of a static vector v_0. The input v_{in} is subsequently passed through one or more hidden layers, resulting at the output in a k-dimensional vector of logits. These are then locally normalised through softmax activations, for each variable X_i considering its corresponding n_i entries. The output vector v_{out} is finally constructed by replacing the entries at the positions of evidence variables, by the corresponding one-hot representations from v_{in}. The values corresponding to the target variables can be interpreted as the predicted target distributions given the evidence. This is illustrated in Fig. 4.1. Note that the k-dimensional vector v_0 is obtained by applying the linear output layer with per-variable softmax normalisation on a randomly initialised trainable vector.

Model training: Whereas a BN is naturally able to answer any query once its Conditional Probability Tables (CPTs) are specified, we need to explicitly train our neural network for that ability. To that end, we iterate over the observed samples, randomly dividing the variables into evidence set $\mathcal{E}$ and targets $\mathcal{T}$, and minimise the average cross-entropy loss for the target variables. The training loss can be expressed as $\mathcal{L}^T = \frac{1}{|\mathcal{T}|} \sum_{X_i \in \mathcal{T}} -\log \hat{p}_{ij}$, in which $\hat{p}_{ij}$ denotes the entry in v_{out} corresponding to the actual observed input value x_{ij} of target variable X_i, i.e., the predicted output probability for the target class of X_i.

Even though the model only receives discrete data samples and therefore never observes the class probabilities it is meant to predict, this training strategy steers the predicted probabilities towards the empirical (conditional) class probabilities. It only requires that each sample's frequency of occurrence during training corresponds to its relative frequency in the set of observations. Appendix 4.6.1 proves this for the canonical case of two binary variables, but the presented derivation can be extended to the multi-variable case.

4.3.2 Training with causal structure

We now propose the following two training strategies to augment the neural network with knowledge of the causal structure between the variables. They are based on the assumption that the causal knowledge can be represented as a DAG, reflecting the (conditional) independence relations that exist between the variables.

1. Injecting independence relations through regularisation (REG): In general, we can say a given DAG implies M pairwise conditional independence relations between two variables, conditioned on a set of observed variables, shortly written as $X \perp Y \mid \mathcal{A}$, with $X, Y \in \mathcal{V}$ and the conditioning set $\mathcal{A} \subseteq \mathcal{V} \setminus \{X, Y\}$. The first proposed training strategy is to inject these independence relations by constructing regularisation loss terms $\mathcal{L}^{\text{REG}}_{X \perp Y \mid \mathcal{A}}$ that quantify how strongly the model violates them:

$$\mathcal{L}^{\text{REG}}_{X \perp Y \mid \mathcal{A}} = \frac{1}{n} \sum_{j=1}^{n} \left(\hat{p}(X = x_j \mid \mathcal{A} = a, Y = y) - \hat{p}(X = x_j \mid \mathcal{A} = a, Y = y') \right)^2 \tag{4.1}$$

in which the summation runs over all n possible values of X. The model's predicted probability for $X = x_j$ with as evidence the assignment $\mathcal{A} = a$ and $Y = y$, is denoted as $\hat{p}(X = x_j \mid \mathcal{A} = a, Y = y)$. The loss term expresses that if the independence relation were satisfied by the model, its predicted probability for any value of X, given any evidence $\mathcal{A} = a$, should be independent of the value assumed by Y. The conditioning set $\mathcal{A}$ is randomly instantiated every time the corresponding loss term applies

during training, and the values y and y' of Y are chosen randomly (with $y \neq y'$).

During training, the regular loss $\mathcal{L}^T$ per data sample is augmented with the regularisation loss $\mathcal{L}^{\text{REG}}$ of one sampled independence relation, weighted with a hyperparameter α.

2. Injecting independence relations through evidence corruption (COR): The second training strategy is based on the intuition that for particular observed data samples, the value of some of the evidence variables may no longer matter besides the other observed variables, when accounting for the relevant independence relations. During training, we detect these cases, and randomly corrupt (i.e., re-sample) those values in the evidence presented to the network.

Consider training on a particular observed sample, the variables divided into evidence $\mathcal{E}$ and targets $\mathcal{T}$. We then go through the known independence relations $X \perp Y \mid \mathcal{A}$, to see which ones are relevant to the training instance. This is the case if either of the following two conditions hold: (1) $(\{Y\} \cup \mathcal{A}) = \mathcal{E}$ and $X \in \mathcal{T}$, or (2) $(\{X\} \cup \mathcal{A}) = \mathcal{E}$ and $Y \in \mathcal{T}$. When condition (1) holds, the predicted outcome of target X should not depend on the observed value of Y, as prescribed by the independence relation, since the conditioning set that is needed for this relation to hold is indeed part of the evidence. The observed input value for Y can hence be disregarded, and we randomly assign a new value from its possible classes. Such a corruption of the input will condition the model to ignore Y when predicting X, given that all variables in $\mathcal{A}$ are also provided as evidence. Note that predictions for other variables in $\mathcal{T}$ need to be done based on the original (i.e., non-corrupted) value for Y, as there is no guarantee that the same independence relation holds for those targets as well. We apply a similar reasoning when condition (2) holds, now corrupting the value of X instead of Y. If no relevant independence relation is found for a given selection of evidence and targets, the input sample is not corrupted and passed as-is to the model. In Appendix 4.6.2, we consider the basic two-variable case and show that the corruption strategy leads to the desired predicted probabilities.

4.4 Neural understudy of Bayesian network: proof-of-concept

The experiments presented in this section aim at answering the following research questions:

- **RQ4.1**: How does the basic neural network perform in comparison with a Bayesian network trained on the same set of samples, in terms of accuracy in predicted probabilities and in terms of sample efficiency? (Section 4.4.2)

- **RQ4.2**: What is the effect of injecting causal structure knowledge into our neural network on its performance? (Section 4.4.2)
- **RQ4.3**: How does each model's performance change when faced with a partially incorrect specification of the causal structure? (Section 4.4.3)

To this end, we compare the following models, trained on an artificial dataset of samples generated from a given joint distribution, as specified in the next section:

- **Bayesian network baseline (BN)**: Bayesian network with the correct DAG, its joint probability distribution estimated from the training samples using Maximum Likelihood Estimation with a K2 prior (see Appendix 4.6.3).
- **Neural network baseline (NN)**: Basic neural network approach as presented in Section 4.3.1, implemented in a single-layer feed-forward model (see Appendix 4.6.4 for details).
- **NN with DAG-based regularisation (NN+REG)**: This model extends the basic NN model with independence relation information, through the regularisation technique REG (Section 4.3.2).
- **NN with DAG-based corruption (NN+COR)**: This model, also based on NN, injects independence relations during training with the corrupted inputs strategy introduced in Section 4.3.2.

4.4.1 Evaluation

For the experiments in this section, we make use of the "Asia" network based on [Lauritzen and Spiegelhalter, 1988], as depicted in Fig. 4.2. Despite its simplicity, with only 7 binary variables, it has sufficient connectivity between the variables to make the problem non-trivial. Following the rules of D-separation [Verma and Pearl, 1990], we can in total extract 191 unique independence relations from its DAG, for the training strategies NN+REG and NN+COR.

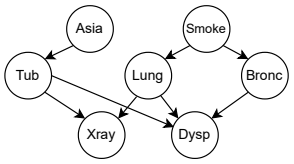


Figure 4.2: The "Asia" BN model, describing a basic lung cancer detection system (for details, see Appendix 4.6.3).

From the ground-truth "Asia" model (detailed in Appendix 4.6.3), we can draw data samples to use as training instances. Through variable elimination [Russell and Norvig, 2009], we can build test queries with the ground-truth conditional probabilities of target variables, for any particular assignment of evidence variables. Just like during training, this means that one query may contain multiple target variables. We can then evaluate models by iterating through the test queries, and calculating the Mean Absolute Error (MAE) between the predicted target distribution and the desired one.

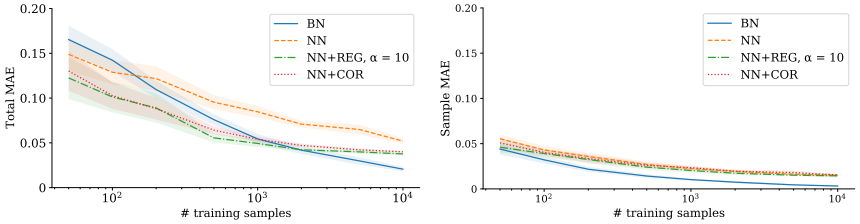


Figure 4.3: Comparison of 4 models in terms of total MAE and sample MAE for different sizes of the training set. The lines reflect the mean across 10 random seeds, used to sample the training set and initialise the neural networks. The shading represents the 95% confidence interval.

For each query, we compute the MAE per target variable, sum all their contributions and divide by the number of targets in the query. We use two different test sets. The first set contains all possible evidence assignments, in total 2,059 queries, and leads to the **total MAE**. The second set contains 1,000 queries, each constructed by drawing a sample from the joint probability distribution, then randomly selecting the set of evidence variables, and obtaining the conditional probabilities for the target variables. It measures the **sample MAE**, assigning more weight to the model's performance for queries with commonly observed evidence values, while the total MAE puts equal focus on common as well as rare assignments of the evidence variables.

4.4.2 Performance of Bayesian network vs. neural understudy

In Fig. 4.3 we compare the sample efficiency of our 4 models in terms of total and sample MAE. For the NN+REG model, we set the regularisation parameter α to 10. The training hyperparameters (hidden layer dimensions, batch size, learning rate, number of training epochs) are kept fixed for all neural models to allow for a direct comparison. Details on the training process and hyperparameters are given in Appendix 4.6.4. The results allow us to answer the first two research questions.

RQ4.1 (Sample efficiency) The proposed training strategy leads to neural understudy models that can approximately infer conditional probabilities. For smaller training sets, the basic NN model shows similar performance as the BN baseline in terms of total MAE. As the training set grows larger, the BN model's knowledge of the DAG structure allows it to significantly overtake the basic NN model in terms of performance. The BN baseline outperforms the neural models across the board when looking at sample MAE, though the gap is not large.

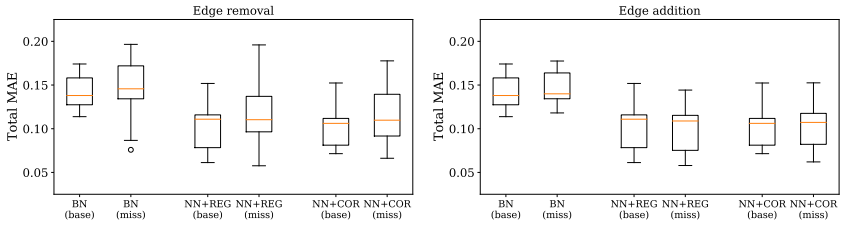


Figure 4.4: Boxplots visualising the total MAE for models receiving the correct DAG ("base") vs. a partially miss-specified DAG ("miss"), i.e., either one edge removed (left) or added (right), for different models (BN; NN+REG; NN+COR) trained on 100 observed samples. All train runs are done for 5 random seeds, and for "miss" runs, 5 random DAG corruptions are done (resulting in 25 runs per model for the "miss" setting).

RQ4.2 (Effect of including DAG info) Including information on the DAG structure in the neural model improves its performance, with a more visible effect on the total MAE. In terms of that metric, injecting the independence relations during training allows the neural model to outperform the BN baseline for smaller sample sets. The NN+REG model performs slightly better than the NN+COR model, albeit not significantly. We hypothesise that this stems from the REG technique systematically iterating over all conditional independence relations, whereas the COR technique only applies a relation when the randomly sampled evidence forms a match. The strength of the neural models appears to lie in their improved performance for rare evidence combinations, which are more heavily disadvantaged in smaller datasets.

4.4.3 Robustness against miss-specification of causal structure

We now explore the impact of injecting partially incorrect information on the causal structure. To this end, we create 10 new DAGs, by randomly removing or adding[2] one edge at a time to the ground-truth DAG from Fig. 4.2. While still being trained on samples from the correct "Asia" network, the BN, NN+REG and NN+COR models now get their conditional independence relations from a partially incorrect DAG. The results displayed in Fig. 4.4 allow answering the third research question.

RQ4.3 (Partially incorrect DAG) While the inclusion of DAG information in the neural understudy brings clear additional benefits in terms of total MAE compared to its most basic form, we must be careful to correctly specify this causal structure. When we assume two variables to be independent when they are not (i.e. by removing an edge in the DAG), both the Bayesian baseline and the neural models become less stable, showing

[2]We were careful not to add edges which might introduce cycles in the network.

higher variation across sample sets. Adding an edge in the DAG seems to do no harm, since this makes for a more conservative estimate of the DAG structure (we assume two variables to have some dependency when they do not).

4.5 Conclusions and future work

We presented ideas for building neural networks that behave like Bayesian networks. In future research we aim to combine the benefits of neural networks (i.e., encoding unstructured data) with some advantages of Bayesian networks, such as their ability to combine uncertainty with causal structure. As a first contribution in that direction, we presented a method to learn a neural network model on observational sample data, and two strategies to encode known causal relationships between the variables, by injecting independence relations.

We tested our method on a single small-scale example, and saw that our proposed training strategy generally works: our neural understudy models (w.r.t. to their Bayesian Network counterpart trained on the same samples) were able to make approximate predictions of conditional probabilities. The inclusion of causal structure resulted in similar performance of the neural understudy compared to the BN baseline, with the neural understudy slightly outperforming the BN under low-sample regimes when testing on all possible queries. We saw the performance of the NN models become less stable as soon as incorrect independence relations were injected, though this behaviour was also observed for the BN.

We see multiple avenues for future work. We will first extend our experiments to larger, more realistic settings, to see whether the conclusions based on our small example still hold. We then aim to investigate how our models can be extended towards continuous variables as well as unstructured data, to be able to answer probabilistic queries concerning a combination of any of these inputs. For example, we envision the use of pre-trained language encoders, although that will require considerable adaptation of the straightforward model with corresponding discrete nodes at input and output.

4.6 Appendix

4.6.1 Sample-based training of neural network

We provide a small-scale proof of how optimising the cross-entropy loss for the observed samples leads to probabilistic outcomes. While the illustrative setting for which we provide the proof only concerns two binary variables, the proof can be extended to a more general setting.

Assume we have two binary variables, and observe ν samples $\{X = x, Y = y\}$ during every training epoch. The distribution of the training data is shown in Table 4.1. Our model simultaneously optimises three partial training losses, each corresponding to a possible evidence mask: $\mathcal{L}_{X\to Y}$ (target Y, evidence X), $\mathcal{L}_{Y\to X}$ (target X, evidence Y), $\mathcal{L}_{XY}$ (target X and Y, no evidence). The random evidence/target mask decides which loss is optimised during which iteration, and the sum of these three losses forms the overall training objective. We will calculate the optimum for each partial training loss and show that this leads to the desired predictions for the targets at hand.

Table 4.1: Absolute frequency table showing the occurrence of $X = x$ and $Y = y$ in the hypothetical training set ($\nu = n_{00} + n_{01} + n_{10} + n_{11}$).

x	y	abs. freq.
0	0	n_{00}
0	1	n_{01}
1	0	n_{10}
1	1	n_{11}

Evidence X, target Y: Optimising the partial loss $\mathcal{L}_{X\to Y}$ should lead us to an estimate for query $P(Y = y|X = x)$. The model returns a prediction $\hat{y}_{x\to y}$ for target $Y = y$ taking evidence $X = x$ as an input. Taking into account the frequencies of each sample as listed in Table 4.1, we obtain the partial loss $\mathcal{L}_{X\to Y}$ as shown in Eq. (4.3). Here, we use the definition of binary cross-entropy loss as listed in Eq. (4.2) and simplify by filling in the possible values (0 or 1) for target y.

$$\mathcal{L}_{\mathcal{CE}} = -ylog(\hat{y}) - (1-y)log(1-\hat{y}) \tag{4.2}$$

$$\begin{aligned}\mathcal{L}_{X\to Y} &= -n_{01}log(\hat{y}_{0\to 1}) - n_{00}log(\hat{y}_{0\to 0}) - n_{11}log(\hat{y}_{1\to 1}) - n_{10}log(\hat{y}_{1\to 0}) \\ &= -n_{01}log(\hat{y}_{0\to 1}) - n_{00}log(1-\hat{y}_{0\to 1}) - n_{11}log(\hat{y}_{1\to 1}) - n_{10}log(1-\hat{y}_{1\to 1})\end{aligned} \tag{4.3}$$

When given $X = 0$ as evidence, we will optimise this loss for $\hat{y}_{0\to 1}$. Equation (4.4) illustrates how calculating the partial derivative and setting it to zero, leads to the optimum for $\hat{y}_{0\to 1}$ (whereby $\hat{y}_{0\to 0}$ = 1 - $\hat{y}_{0\to 1}$). A similar derivation for $X = 1$ leads to the optimum for $\hat{y}_{1\to 1}$ (and $\hat{y}_{1\to 0}$ = 1 - $\hat{y}_{1\to 1}$). In other words, the predicted value for target Y moves towards its relative frequency in the training set, conditioned on the observed evidence values.

$$\frac{\partial \mathcal{L}_{X\to Y}}{\partial \hat{y}_{0\to 1}} = 0 \Rightarrow \hat{y}_{0\to 1} = \frac{n_{01}}{n_{00}+n_{01}}; \quad \frac{\partial \mathcal{L}_{X\to Y}}{\partial \hat{y}_{1\to 1}} = 0 \Rightarrow \hat{y}_{1\to 1} = \frac{n_{11}}{n_{10}+n_{11}} \tag{4.4}$$

Evidence Y, target X: Optimising the partial loss $\mathcal{L}_{Y\to X}$ should lead us to an estimate for query $P(X = x|Y = y)$. The derivation is symmetrical to the previous case, with the roles of X and Y switched.

No evidence, targets X and Y: By optimising $\mathcal{L}_{XY}$, the model jointly optimises its predicted outcome for queries $P(X = x)$ and $P(Y = y)$. A training case only contributes to this partial loss when the evidence mask chooses both variables as targets. Now, we can write the loss as in Eq. (4.5), where $\hat{x}$ is the prediction for $X = 1$ given no evidence, and analogous for $\hat{y}$. We use the same notation for the relative frequencies as before, and again use the definition of binary CE-loss from Eq. (4.2).

$$\begin{aligned}\mathcal{L}_{XY} = &-(n_{10} + n_{11})log(\hat{x}) - (n_{00} + n_{01})log(1 - \hat{x}) \\ &-(n_{01} + n_{11})log(\hat{y}) - (n_{00} + n_{10})log(1 - \hat{y})\end{aligned} \tag{4.5}$$

Again, we can optimise the loss above for $\hat{x}$ and $\hat{y}$. This leads to the optima shown in Eq. (4.6). The optimum for $\hat{x}$ simply corresponds to the relative frequency of seeing $X = 1$ in the training set, which is indeed what we want as a prediction for $P(X = 1)$. The same goes for $\hat{y}$.

$$\begin{aligned}\frac{\partial \mathcal{L}_{XY}}{\partial \hat{x}} = 0 \Rightarrow \hat{x} = \frac{n_{10} + n_{11}}{n_{00} + n_{01} + n_{10} + n_{11}} \\ \frac{\partial \mathcal{L}_{XY}}{\partial \hat{y}} = 0 \Rightarrow \hat{y} = \frac{n_{01} + n_{11}}{n_{00} + n_{01} + n_{10} + n_{11}}\end{aligned} \tag{4.6}$$

We emphasize that the results above only hold if training happens uniformly over the available training instances, so that the frequency of occurrence of a target given a particular evidence actually corresponds with the empirical probability in the training data. We ensure that this is the case by properly shuffling our batches within each epoch.

4.6.2 Injecting independence relations through evidence corruption

We again consider the simple setting of two binary variables X and Y, to show that the NN+COR method works as expected. We now add the knowledge that X and Y are independent. The total loss is made up of $\mathcal{L}_{X\to Y}$, $\mathcal{L}_{Y\to X}$ and $\mathcal{L}_{XY}$, as defined in Section 4.6.1. Since the DAG-based corruption is only executed when the evidence set is not empty, only the partial losses $\mathcal{L}_{X\to Y}$ and $\mathcal{L}_{Y\to X}$ are affected. We zoom in on how to adapt the first one according to this new setting and how this affects the predicted outputs. The derivation for the other partial loss is symmetrical.

Say we receive a sample $\{X = x, Y = y\}$ during training and the mask indicates that X is evidence, while Y is the target. Since $X \perp Y$, the

value of the target y should be independent of the observed evidence value x. Therefore, we corrupt the value of x, setting it to 1 with probability γ and to 0 with probability $1 - \gamma$. We can use the frequencies from Table 4.1 to write out the contribution of each training sample to the partial loss $\mathcal{L}_{X \to Y}$, taking into account that the evidence is corrupted for a fraction of the training samples. This is depicted in Eq. (4.7). The predicted targets $\hat{y}_{0 \to 1}$ and $\hat{y}_{1 \to 1}$ are defined in the same way as described in Section 4.6.1.

$$\begin{aligned}\mathcal{L}_{X \to Y} = &-(1-\gamma)(n_{01}+n_{11})log(\hat{y}_{0\to1}) - (1-\gamma)(n_{00}+n_{10})log(1-\hat{y}_{0\to1}) \\ &-\gamma(n_{11}+n_{01})log(\hat{y}_{1\to1}) - \gamma(n_{10}+n_{00})log(1-\hat{y}_{1\to1})\end{aligned} \tag{4.7}$$

When taking the partial derivative of the loss as shown in Eq. (4.8), we get the optima for $Y = 1$ with either value of X as evidence. Due to the corruptions we implemented in the training process, we now get $\hat{y}_{0 \to 1} = \hat{y}_{1 \to 1}$. The prediction for Y is indeed independent of the value of X (in accordance to $X \perp Y$) and simply equal to the relative frequency of observing $Y = 1$ in the training set.

$$\begin{aligned}\frac{\partial \mathcal{L}_{X \to Y}}{\partial \hat{y}_{0 \to 1}} = 0 \Rightarrow \hat{y}_{0 \to 1} = \frac{n_{01} + n_{11}}{n_{00} + n_{01} + n_{10} + n_{11}} \\ \frac{\partial \mathcal{L}_{X \to Y}}{\partial \hat{y}_{1 \to 1}} = 0 \Rightarrow \hat{y}_{1 \to 1} = \frac{n_{01} + n_{11}}{n_{00} + n_{01} + n_{10} + n_{11}}\end{aligned} \tag{4.8}$$

Note that the parameter γ plays no role in the optimum for $\hat{y}$. It does not matter according to which distribution we corrupt the evidence. In our implementation, we sample uniformly over all possible classes for the variable in question to corrupt its value. We could also opt to pull a random sample from the training set and switch out the value of the evidence variable to its value in this sample.

4.6.3 Bayesian network implementation

We use the *pgmpy* Python library [Ankur Ankan and Abinash Panda, 2015] (version 0.1.17) for sampling, training and inference in our BNs.

To train our BNs from observed samples, we use the Maximum Likelihood Estimator. This estimator studies the co-occurrence of particular values of each variable and its parents in the training set, filling up the CPTs as such. We use a K2 prior as a smoothing strategy, to counteract the extremely skewed probability distributions that might appear in the CPTs when particular combinations of variables are never observed in the training set.

Artificial samples are generated from the ground-truth BN using the method of forward sampling with a particular seed. As a ground-truth

Tub	Asia	
	yes	no
yes	0.05	0.01
no	0.95	0.99

Lung	Smoke	
	yes	no
yes	0.1	0.01
no	0.9	0.99

Bronc	Smoke	
	yes	no
yes	0.6	0.3
no	0.4	0.7

Asia	
yes	0.01
no	0.99

Smoke	
yes	0.5
no	0.5

Xray	Lung = yes		Lung = no	
	Tub = yes	Tub = no	Tub = yes	Tub = no
yes	0.98	0.98	0.98	0.05
no	0.02	0.02	0.02	0.95

Dysp	Bronc = yes				Bronc = no			
	Lung = yes		Lung = no		Lung = yes		Lung = no	
	Tub = yes	Tub = no	Tub = yes	Tub = no	Tub = yes	Tub = no	Tub = yes	Tub = no
yes	0.9	0.9	0.9	0.8	0.7	0.7	0.7	0.1
no	0.1	0.1	0.1	0.2	0.3	0.3	0.3	0.9

Figure 4.5: CPTs defining the ground-truth "Asia" BN, based on [Lauritzen and Spiegelhalter, 1988].

Bayesian model, from which our artificial training and test sets are created, we use the "Asia" model (see Fig. 4.2 for its DAG structure). The CPTs that define its joint distribution with 20 parameters are shown in Fig. 4.5.

We use the technique of variable elimination to perform exact inference on our BNs. There are some cases where this method fails because the query contains some evidence combination it has never seen before. When coming across such a case during test time, we simply throw out the evidence for this particular query and take $P(X)$ as an estimate. We believe this makes for a fairer comparison than simply ignoring those queries, since our NN is in fact able to provide an estimate for all queries, even if it has never seen a particular evidence set before.

4.6.4 Neural network training details

Our neural network architecture and its training procedure are implemented in *Pytorch* [Paszke et al., 2019].

As shown in Fig. 4.1, our neural network is made up of 3 layers. The network receives an input vector of dimension k. In our "Asia" example, k is equal to 14 (all 7 variables have 2 classes). First, the values of the target variables (as selected by the evidence mask) are substituted by their respective value from v_0. The full input is then transformed to dimension h, using an input-to-hidden linear layer with tanh activation. Then, a hidden-to-hidden linear layer of size h, again with tanh activation is applied. Finally, the hidden-to-output layer transforms the activations back to dimension k. This layer applies N softmax functions to transform the activations belonging to each variable separately into normalised probability values, as depicted in Fig. 4.1.

For initialisation of the input vector on the positions of the target variables, we use the vector v_0 of size k, obtained by applying the hidden-to-

output layer to a trainable vector of size h, followed by the per-variable softmax normalisation. When no evidence is present, v_0 serves as the full input vector, and leads to the model predicting the empirical mean probability for all variables.

We chose a hidden size h of 50, since we noticed this allowed enough flexibility while still constraining the parameter space sufficiently to avoid overfitting. With these dimensions, our network has 4014 trainable parameters in total. We use the Adam optimiser [Kingma and Ba, 2015] with a learning rate of 0.001 and otherwise default parameters in Pytorch. The training samples are fed to the neural model in batches of size 16, and the model is trained for 500 epochs. In the NN+REG model, we additionally use a regularisation batch size of 16.

The hyperparameter α controls the trade-off between training and regularisation loss. The MSE naturally has a smaller order of magnitude than the cross-entropy loss, therefore rather large choices for α (10, 100, 1000) work best. We settled for α equal to 10 since this seemed to lead to the best performance of the NN+REG method, though we did not observe a big difference between any choice of α within the range of 1 to 1000.

5

Clinical Reasoning over Tabular Data and Text with Bayesian Networks

This chapter shows how a Bayesian network can automate clinical reasoning in a use case emulating the diagnosis of respiratory diseases in primary care. To this end, we develop a benchmark dataset linking tabular data and text, which we introduce here and expand further in the next chapter. We extend the Bayesian network with neural text representations, allowing it to process both structured tabular evidence and unstructured textual input. We explore both generative and discriminative strategies for incorporating neural text representations into the Bayesian network, integrating their outputs directly into Bayesian inference. Our results highlight the strengths of Bayesian networks for modeling the clinical reasoning process and the added value of including textual evidence alongside standard tabular features.

* * *

Paloma Rabaey, Johannes Deleu, Stefan Heytens, Thomas Demeester

Proceedings of Artificial Intelligence in Medicine (AIME) 2024

Abstract

Bayesian networks are well-suited for clinical reasoning on tabular data, but are less compatible with natural language data, for which neural networks

provide a successful framework. This paper compares and discusses strategies to augment Bayesian networks with neural text representations, both in a generative and discriminative manner. This is illustrated with simulation results for a primary care use case (diagnosis of pneumonia) and discussed in a broader clinical context.

5.1 Introduction

The process of clinical reasoning lies at the heart of many interactions between a clinician and their patient [Yazdani and Hoseini Abardeh, 2019]. Clinical reasoning is the process by which a clinician integrates their own knowledge with patient information (like symptoms, objective medical evidence, background, medical history...), to arrive at a diagnosis and subsequent therapeutic options [Gruppen, 2017]. Cognitive biases and knowledge deficits can cause errors in clinical reasoning, causing the clinician to arrive at an incorrect diagnosis [Norman et al., 2017]. To help clinicians avoid these pitfalls, it can help to (partially) automate the process of clinical reasoning [Chin-Yee and Upshur, 2018, Yanase and Triantaphyllou, 2019]. Bayesian Networks (BNs) are ideally suited for this task, given (i) their ability to model complex problems involving uncertainty, (ii) their ability to combine data and expert knowledge, and (iii) their interpretable graphical structure [Kyrimi et al., 2021g]. However, a key factor limiting the adoption of BNs in clinical practice is their inadequacy to deal with realistic medical data [Kyrimi et al., 2021a], often a mix between structured tabular variables (disease codes, timestamps, demographic features, lab results...) and unstructured text (consultation notes, discharge summaries...) [Ford et al., 2016]. Encoding the information contained in the unstructured text into structured variables not only requires (considerable) effort, but also inevitably results in loss of information.

In this work, we explore how to integrate unstructured text data in BNs, to facilitate joint clinical reasoning over structured tabular data and unstructured text. To this end, we investigate a relevant use case in primary care: diagnosis of pneumonia. We create an artificial yet realistic dataset, allowing us to control several aspects of the data generation process. This allows us to investigate the impact of different modeling approaches to integrate text in the clinical reasoning process, and discuss their advantages and pitfalls. By keeping the use case highly tangible for a clinical audience, we aim to lower the bar toward real-world medical applications of the presented technology.

Our main contribution is the study of different approaches to integrate the neural representation of a textual variable in the BN. In particular, we compare the properties of adding the text with a generative model (in the space of neural text representations, fitted alongside the BN) vs. a discriminative model (a text classifier jointly trained with the BN). We evaluate

the performance of both approaches on the prediction of pneumonia in a toy setting, and compare with baselines which are either missing the text component or the BN structure. Based on the presented results, we discuss (i) the advantages of including unstructured text, (ii) the properties of different approaches to achieve this, and (iii) the overall idea of performing Bayesian inference for automated clinical reasoning involving textual data.

5.2 Related work

Since the topic of this paper touches on multiple different research domains, we position our work in regards to the most relevant domains, without providing an exhaustive overview of all related research.

Clinical reasoning This work follows the interpretation of clinical reasoning as an analytical process, where a clinician weighs up every piece of evidence to reject or confirm a certain diagnostic hypothesis [Strauss et al., 2018, Yazdani et al., 2017]. Starting from a set of differential diagnoses, each with their own prior probability reflecting their prevalence in the population, clinical reasoning comes down to updating the likelihood of each diagnosis with every new piece of evidence that comes in, using Bayes' rule. This results in a posterior likelihood for each diagnosis, which the clinician takes into account for planning further steps.

Bayesian networks BNs form the perfect tool to formalize the process outlined above [Kyrimi et al., 2021g]. Their interpretable graph structure can help keep track of independencies between certain types of evidence and particular diagnoses, and inference in BNs follows Bayes' rule. BNs have been used to model a wide range of medical conditions in research settings [McLachlan et al., 2020a], including respiratory diseases such as pneumonia and Covid-19 [Edye et al., 2021]. However, their deployment for clinical decision making in practice remains limited, partly due to real-world data challenges [Kyrimi et al., 2021a].

Clinical unstructured text The last few decades have seen an abundance of electronic medical records being collected in clinical practice, which form a useful source of data to build Clinical Decision Support Systems (CDSSs). These records are usually made up of structured data (disease codes, dates, treatment codes...), as well as free text [Ford et al., 2016]. Studies have shown that ignoring the information present in free text records can results in data loss and bias in CDSSs [Price et al., 2016]. Nevertheless, a large majority of CDSSs either completely disregards this unstructured text [Peiffer-Smadja et al., 2020a] or applies information extraction techniques to turn the text into tabular format, which then serves as input to the CDSS [Ford et al., 2016]. Turning unstructured text into structured

variables using information extraction methods (see, e.g., [Sterckx et al., 2020]) and then building a predictive model on top of the structured features has been applied to learning clinical BNs in the past [Ye et al., 2014, Rotmensch et al., 2017]. Our work focuses on integrating the full unstructured text, removing the need for this information extraction step. Some CDSSs are built on raw unstructured text, yet they often fail to integrate it with the structured portion of the data [Peiffer-Smadja et al., 2020a, Mujtaba et al., 2019]. Zhang et al. manage to successfully integrate both through a multi-modal recurrent neural network that combines embeddings of clinical text with static and time-varying features in the electronic medical record, outputting a full patient representation that can be used for further downstream prediction [Zhang et al., 2020]. We also represent clinical text through neural representations, though we use a BN in combination with feed-forward neural components to integrate these text embeddings with the static tabular features in the reasoning process.

Neuro-Symbolic AI The integration of reasoning and learning has seen considerable progress in recent years, in the field of Neuro-Symbolic (NeSy) AI [Marra et al., 2024]. One strongly related contribution is the DeepProbLog (DPL) framework [Manhaeve et al., 2018a]. The authors show how a probabilistic logic program can be extended with neural predicates, whereby a neural network converts an unstructured data item (like an image) into class probabilities, that are then treated as regular predicates in the logic program. Importantly, the parameters of the logic program and of the neural networks encoding unstructured data can be jointly trained. The discriminative model for integrating text nodes into BNs in this paper corresponds to the neural predicates approach, since a BN can be seen as a special case of a probabilistic program. In contrast, we also compare this approach with a generative model in text representation space, and provide a targeted discussion from the perspective of clinical reasoning.

5.3 Use case and data description

Our use case focuses on diagnostic clinical reasoning performed by a General Practitioner (GP) [Yazdani et al., 2017]. One non-trivial task in primary care is distinguishing pneumonia from an upper respiratory tract infection (also known as the common cold), where the former is more serious and calls for treatment with antibiotics. When a patient presents with respiratory symptoms, a GP will apply clinical reasoning based on these symptoms and a short clinical examination, ordering the necessary additional testing or starting a treatment only if the probability for pneumonia exceeds a certain threshold.

We create an artificial dataset that allows us to study automation of the clinical reasoning process for the pneumonia use case, in the presence of

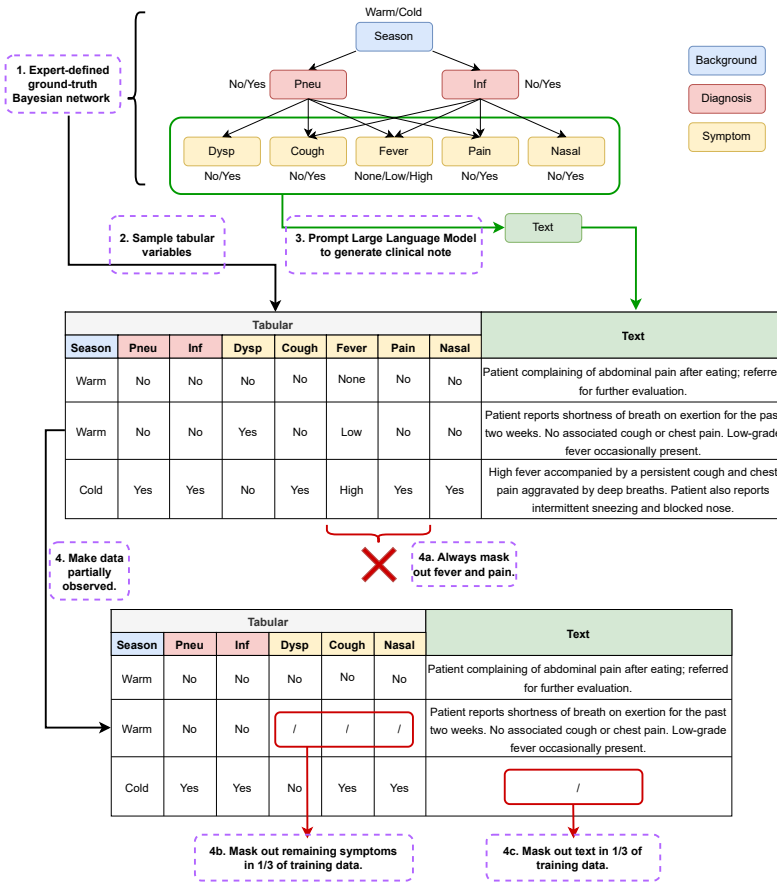


Figure 5.1: Key steps in generating the artificial dataset, where each sample consists of both tabular variables and corresponding clinical text descriptions. With help of an expert, we define a BN simulating the pneumonia use case (**step 1**). We sample the tabular variables (background, diagnoses and symptoms) from the distribution defined by this BN (**step 2**), prompting an LLM (GPT3.5 [Ouyang et al., 2022]) to generate realistic but fictitious consultation notes based on the sampled symptoms (**step 3**). We repeat steps 1 to 3 to generate 4000 training samples and 1000 test samples. Finally, to induce property (ii) of realistic medical data (see Section 5.3), we remove two symptoms, $fever$ and $pain$, from the tabular portion of the data, ensuring they are never encoded and only observed through the text (**step 4a**). From now on, when we talk about symptoms, we take this to mean the symptoms $dysp$, $cough$, $nasal$, unless explicitly stated otherwise. For the training set only, we partially mask out the remaining symptoms (**step 4b**) and the text (**step 4c**) in a complementary subset of the training samples. Each sample now represents a fictional patient encounter, consisting of one background feature ($season$), two diagnoses ($pneu$ and inf), three symptoms ($dysp$, $cough$ and $nasal$, partially unobserved) and a textual description ($text$, partially unobserved). The text contains additional context on the three encoded symptoms, as well as describing two additional unencoded symptoms $fever$ and $pain$.

unstructured text resembling consultation notes taken by the GP during a patient encounter. Figure 5.1 shows the data generation process. Its caption describes the four key steps, and a more detailed explanation is given in Appendices 5.7.1 and 5.7.2. We aim to mirror the following properties of realistic medical data: (i) the data contains structured tabular variables and/or unstructured text, (ii) information in the text is only partially encoded in the structured variables, and (iii) the text contains additional context on the patient's background and symptoms, complementing the information encoded in the tabular variables. The final train and test datasets are available in our Github repository: `https://github.com/prabaey/bn-text`.

5.4 Augmenting BNs with text representations

We propose two model architectures that are able to integrate text in a Bayesian network: `BN-gen-text` (Section 5.4.1) and `BN-discr-text` (Section 5.4.2). Both models incorporate text through a single-vector text embedding, either modeling its distribution directly or learning classifiers with these representations as an input. As shown in Figure 5.2, we compare them with three baseline models. Our first baseline `BN` is a standard Bayesian network without text variables, trained only on the partially observed tabular features. Its extension `BN`$^{++}$ is trained on a version of the training set where the symptoms $fever$ and *pain* are exceptionally *not* masked out, forming an upper bound to the performance of all other models, which never get to directly observe these two symptoms. The last baseline `FF-discr-text` is a discriminative feed-forward neural network which takes both tabular features (one-hot encoded) and text (as a *BioLORD* embedding) as an input, and outputs a prediction for *pneu* or inf. Details on the baseline models `BN`, `BN`$^{++}$ and `FF-discr-text` can be found in Appendix 5.7.3.1 and 5.7.3.2.

All models are trained on the final dataset shown in Figure 5.1 (with only the symptoms *dysp*, *cough* and *nasal*, partially observed) except for `BN`$^{++}$ (where $fever$ and *pain* are added, as described above). During inference, each model computes a posterior distribution for each diagnosis given some set of evidence. For readability, we represent the diagnoses by D_i ($i \in \{0, 1\}$) with D_0 (*pneu*) and D_1 (inf), symptoms as S_0 (*dysp*), S_1 (*cough*) and S_2 (*nasal*), background as B (*season*) and text as T. We discuss how each model is able to calculate the following posterior diagnostic probabilities:

- $\mathcal{P}(D_i \mid B, S_0, S_1, S_2)$: take only background and symptoms as evidence.
- $\mathcal{P}(D_i \mid B, S_0, S_1, S_2, T)$: take background, symptoms and text as evidence.
- $\mathcal{P}(D_i \mid B, T)$: take background and text as evidence.

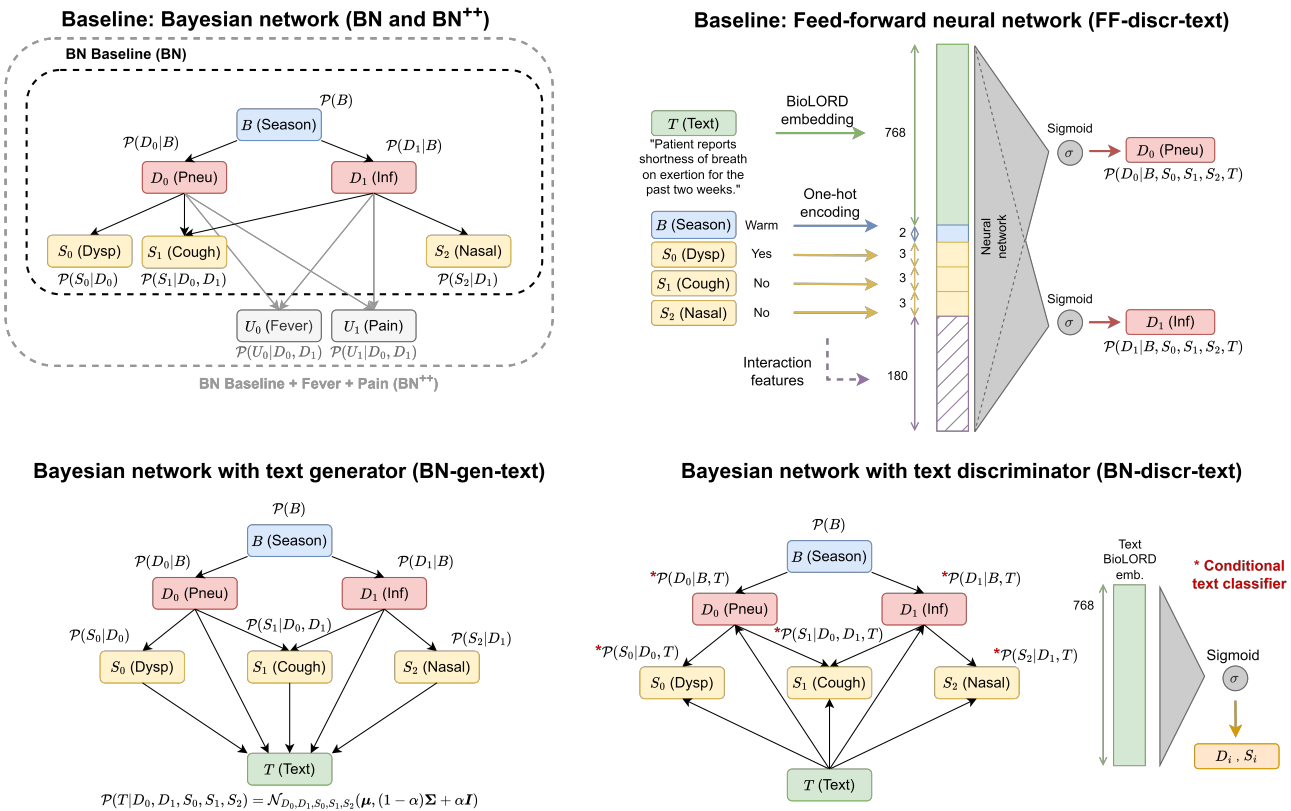


Figure 5.2: Schematic depiction of all models. The top row presents our baselines `BN`, `BN++` and `FF-discr-text`. The bottom row shows `BN-gen-text` and `BN-discr-text`, two variants of a Bayesian network augmented with text representations.

5.4.1 Bayesian network with text generator (BN-gen-text)

Training: In the BN-gen-text model, a text node is added to the BN baseline, conditioned on all diagnoses and symptoms. The conditional distributions for all tabular variables are trained using Maximum Likelihood Estimation, as a standard Bayesian network (see Appendix 5.7.3.1). To obtain a vector representation for the text, we use *BioLORD*, which is a pre-trained language model that produces semantic single-vector representations for clinical sentences and biomedical concepts [Remy et al., 2024]. 32 separate multivariate Gaussians, one for each possible combination of the values for the two diagnoses and three symptoms, are fitted to the text embeddings to obtain the distribution $\mathcal{P}(T \mid D_0, D_1, S_0, S_1, S_2)$. This basic model allows us to get the probability density of unseen text embeddings and even sample new ones, although those cannot be directly decoded into text. To learn each Gaussian, we select all samples in the training set that match a particular condition and fit the mean μ and covariance matrix $\mathbf{\Sigma}$ to the corresponding text embeddings. The estimated covariance matrix $\mathbf{\Sigma}$ is regularized as follows

$$\mathcal{P}(T \mid D_0, D_1, S_0, S_1, S_2) = \mathcal{N}_{D_0,D_1,S_0,S_1,S_2}(\boldsymbol{\mu}, (1-\alpha)\mathbf{\Sigma} + \alpha \boldsymbol{I}) \tag{5.1}$$

where the hyperparameter α allows tuning the contribution of the individual variances of the text representation dimensions.

Inference: After training, we can calculate the posterior for either diagnosis D_i given a set of evidence by applying Bayes' rule and marginalizing over the learned joint distribution. For $\mathcal{P}(D_i \mid B, S_0, S_1, S_2)$ the calculation is the same as in a standard BN, since the conditional text distribution $\mathcal{P}(T \mid D_0, D_1, S_0, S_1, S_2)$ is integrated out. $\mathcal{P}(D_i \mid B, S_0, S_1, S_2, T)$ and $\mathcal{P}(D_i \mid B, T)$ on the other hand do evaluate the conditional text distribution. The detailed equations for all posteriors can be found in Appendix 5.7.3.3.

5.4.2 Bayesian network with text discriminator (BN-discr-text)

Training: In the BN-discr-text model, we augment the BN baseline by conditioning all diagnoses and symptoms on the text embedding. This contrasts with the BN-gen-text approach, where we augment the BN baseline with distributions of text embeddings conditioned on the diagnoses and symptoms. While this arc reversal renders the two BNs semantically non-equivalent, all independence relations between the non-text nodes remain intact. Each of the conditional distributions is modeled as a set of discriminative neural text classifiers, one for each configuration of the tabular parent variables, meaning there are 12 in total. For example, we model $\mathcal{P}(D_0 \mid B = warm, T)$ and $\mathcal{P}(D_0 \mid B = cold, T)$ as two separate feed-forward neural networks that take the *BioLORD* text embedding as an

input, and learn to predict the diagnosis probability for D_0 at the output. All parameters are optimized jointly by maximizing the likelihood $\mathcal{P}(B, D_0, D_1, S_0, S_1, S_2 \mid T)$ (see Appendix 5.7.3.4) based on the training data. By making this likelihood conditional on text, we refrain from having to learn a prior distribution $\mathcal{P}(T)$ of the text embeddings.

Inference: $\mathcal{P}(D_i \mid B, S_0, S_1, S_2, T)$ is again obtained by applying Bayes' rule and marginalizing over the joint distribution (conditional on text). The trained classifiers are used to evaluate the probabilities needed during inference. Strictly speaking, conditioning on the text node means that $\mathcal{P}(D_i \mid B, S_0, S_1, S_2)$ cannot be computed. We circumvent this issue by conditioning on the embedding of the empty text "" in case no text is observed. The classifiers learn to take this into account, since an empty text occurs in 1/3 of the training data. Finally, $\mathcal{P}(D_i \mid B, T)$ is simply the output of one of the two diagnosis classifiers. As before, the detailed equations for all posteriors can be found in Appendix 5.7.3.4.

5.5 Empirical results and analysis

Evaluation: There are various ways to measure the models' ability of estimating diagnostic probabilities, given the observed background, symptoms and/or textual inputs. This section assumes the real-world scenario with a known (binary) diagnosis on the test set, but no knowledge of ground truth conditional probabilities. We therefore rank all patients in the test set according to the estimated probability of the considered diagnosis, and measure the area under the precision-recall curve for that ranking by comparing with the binary ground truth, i.e., we report the average precision. Results are averaged over 5 training runs with different model initializations. The full code is available in our Github repository: `https://github.com/prabaey/bn-text`.

Results: Table 5.1 presents average precision results for the diagnosis of pneumonia (D_0), while Table 5.2 shows these results for the prediction of upper respiratory tract infection (D_1). Ablation results in terms of connectivity in the network are provided and discussed in Appendix 5.7.4.2, and details on training and hyperparameter optimization are given in Appendix 5.7.4.1.

Analysis: Comparing $\mathcal{P}(D_0 \mid B, S_0, S_1, S_2, T)$ and $\mathcal{P}(D_0 \mid B, S_0, S_1, S_2)$ in Table 5.1, we note that both `BN-gen-text` and `BN-discr-text` improve over the baseline `BN`. This improvement is thanks to the incorporation of text, which contains information on the symptoms $fever$ and *pain* that is otherwise never explicitly encoded in the tabular features, yet very useful for diagnosing *pneu*. Indeed, when `BN-discr-text` takes both symptoms and text into account, in $\mathcal{P}(D_0 \mid B, S_0, S_1, S_2, T)$, its average precision comes close to the upper bound set by the baseline `BN`$^{++}$, with direct access to all 5 encoded symptoms. Furthermore, the ablation study in Appendix 5.7.4.2

Table 5.1: Average precision over test set of three posterior probabilities for the diagnosis D_0 (*pneu*), each taking a different set of evidence into account (various combinations of background, symptoms and text). We show mean (± std) over 5 initialization seeds.

Model	**Average precision for *pneu***		
	$\mathcal{P}(D_0 \mid B, S_0, S_1, S_2, T)$	$\mathcal{P}(D_0 \mid B, S_0, S_1, S_2)$	$\mathcal{P}(D_0 \mid B, T)$
`BN`	-	0.0914 (± 0.0000)	-
`BN`$^{++}$	-	**0.8326** (± 0.0000)	-
`FF-discr-text`	0.6574 (± 0.0118)	0.1090 (± 0.0020)	0.6220 (± 0.0121)
`BN-gen-text`	0.5870 (± 0.0000)	0.0892 (± 0.0007)	0.4434 (± 0.0000)
`BN-discr-text`	**0.7538** (± 0.0323)	0.1079 (± 0.0011)	**0.6922** (± 0.0273)

Table 5.2: Average precision over test set of three posterior probabilities for the diagnosis D_1 (inf), each taking a different set of evidence into account (various combinations of background, symptoms and text). We show mean (± std) over 5 initialization seeds.

Model	**Average precision for *inf***		
	$\mathcal{P}(D_1 \mid B, S_0, S_1, S_2, T)$	$\mathcal{P}(D_1 \mid B, S_0, S_1, S_2)$	$\mathcal{P}(D_1 \mid B, T)$
`BN`	-	0.8884 (± 0.0000)	-
`BN`$^{++}$	-	0.9009 (± 0.0000)	-
`FF-discr-text`	0.9042 (± 0.0018)	0.8813 (± 0.0003)	0.8821 (± 0.0014)
`BN-gen-text`	0.7968 (± 0.0007)	0.8889 (± 0.0000)	0.7624 (± 0.0011)
`BN-discr-text`	0.9016 (± 0.0007)	0.8889 (± 0.0000)	0.8738 (± 0.0018)

shows a dramatic performance drop for both `BN-discr-text` and `BN-gen-text` when omitting the direct relation between diagnoses and text in the network, rendering the model unable to incorporate any complementary text-only information (i.e., on $fever$ and *pain*) during inference.

Though the `BN-gen-text` model performs better than the baseline `BN`, we see two reasons for why it is not on par with `BN-discr-text`. First, the distribution $\mathcal{P}(T \mid D_0, D_1, S_0, S_1, S_2)$ is made up of 32 conditional Gaussians, each trained on a different subset of text embeddings that occur with a particular (possibly rare) combination of symptom and diagnosis values. The `BN-discr-text` model has a more modular architecture and does not suffer as much from limited relevant training samples to fit each of its classifiers. Second, a multivariate normal distribution is not the best fit for the text embeddings. This is probably also why we see a bigger performance gap between the prediction that incorporates only text ($\mathcal{P}(D_0 \mid B, T)$) and the one that incorporates both text and symptoms ($\mathcal{P}(D_0 \mid B, S_0, S_1, S_2, T)$) for the `BN-gen-text` model.

Interestingly, the `FF-discr-text` baseline performs worse than `BN-discr-text`. While the two *pneu* classifiers in `BN-discr-text` can focus on modeling the text given one particular value of the background variable, the `FF-discr-text` classifier needs to deal with various configurations of the background and symptoms, missing or present, as well as their interactions

with the text, in a single model. This is why `BN-discr-text` already improves over `FF-discr-text`, even when it only incorporates text during inference. When `BN-discr-text` incorporates symptoms on top of text ($\mathcal{P}(D_0 \mid B, S_0, S_1, S_2, T)$), it improves over `FF-discr-text` with almost 10 percentage points in average precision, proving the merit of learning separate symptom classifiers, as well as diagnosis classifiers, and incorporating all their predictions through a Bayesian inference procedure.

Table 5.2 shows higher average precision measures for the prediction of inf, which is much more common in our dataset than *pneu*. Note that including the text in the prediction does not improve performance. Indeed, the small gap between `BN` and `BN`$^{++}$ shows that knowledge on the symptoms $fever$ and *pain* does not improve the prediction of inf much. Their textual representation is therefore expected to have little impact as well.

5.6 Discussion and conclusion

We conclude with a discussion of how the results from the previous section can be seen in a broader context, by answering three key questions on the integration of text in Bayesian networks for clinical reasoning.

What are different ways to integrate text into a BN, to allow for joint reasoning over unstructured text and structured tabular features?

We compared two architectures belonging to complementary model families: a BN with text generator (`BN-gen-text`) and a BN with text discriminator (`BN-discr-text`). An advantage of the `BN-gen-text` model is that it follows the causal structure of the text generation process, making it more intuitive to understand. However, to fit a generative model for the text embeddings, we need to make assumptions on the distribution which do not hold in practice. Conditional fitting of the Gaussians for every combination of diagnoses and symptoms also leads to a bad fit for rarer combinations. Both of these downsides translate to inferior performance of the generative model on our use case. However, alternative generative architectures are worth exploring in future research. The `BN-discr-text` model can benefit from the flexibility of the neural classifiers without requiring any assumptions on the distribution of the text embeddings. Its modular approach (with separate classifiers for the diagnoses and symptoms) allows for an intuitive integration of the observed symptoms into the Bayesian inference procedure.

What are the advantages of using unstructured text for clinical reasoning, compared to only using structured tabular features?

Reducing clinical text to a set of structured variables can be challenging, and inherently causes loss of information. By retaining the raw text

and training the model to deal with this, the information extraction step is no longer required. This avoids the need to determine up front which variables are (1) relevant for any set of diagnoses, (2) mentioned frequently enough, and (3) can be extracted with sufficient accuracy. We simulated the presence of complementary information in the text with the symptoms $fever$ and $pain$. The models that included the text, were able to leverage information on those symptoms without explicitly including them as variables in the BN, which turned out especially beneficial for the rarer disease (pneumonia). This aligns with the intuition that specific symptoms related to rare diseases may not be encoded due to their infrequent presence, while at the same time being indispensable for accurate diagnosis.

What are the advantages of Bayesian inference for clinical reasoning, compared to approaches that don't contain a BN component?

BNs model each conditional distribution separately. This is not the case for the `FF-discr-text` baseline, which directly outputs a prediction for the diagnosis instead. This modular approach has multiple advantages. First of all, it helps the model deal with missing data during the training process: conditional distributions for variables that are not observed in a particular sample are simply not updated. The `FF-discr-text` baseline deals with unobserved symptoms by incorporating a special category in its one-hot encoding, which is much less natural. Second, and even more important, this modularity improves the interpretability of the prediction, which is essential in medical applications. An end user of the `BN-discr-text` model can inspect the outputs of the symptom classifiers as well as the diagnosis classifier, and see how all these probabilities contribute to the predicted posterior through the Bayesian inference process.

Acknowledgements

Paloma Rabaey's research is funded by the Research Foundation Flanders (FWO-Vlaanderen) with grant number 1170124N. This research also received funding from the Flemish government under the "Onderzoeksprogramma Artificiële Intelligentie (AI) Vlaanderen" programme.

5.7 Appendix

5.7.1 Data generation process

Figure 5.1 in the main text shows the steps we take to generate our dataset.

Step 1. Expert-defined Bayesian network With help of an expert general practitioner, we define a BN that can be used for diagnosis of two diseases: pneumonia ($pneu$) and upper respiratory tract infection (inf).

We model the effect of one background factor, *season* of the year, on both diagnoses. Additionally, five symptoms are added as nodes to the network: *dysp* (dyspnea, also known as shortness of breath), *cough*, $fever$, *pain* (chest pain and/or pain attributed to airways) and *nasal* (nasal congestion and/or sneezing). All variables are binary ($warm/cold$ for *season* and no/yes for all others), except for $fever$, which can take on three levels ($none/low/high$). According to the expert, these five symptoms are the most informative to diagnose the two respiratory diseases in a primary care setting. Figure 5.3 shows the conditional probability tables (CPTs, defining the distribution of any child conditioned on its possible parent values), which were filled in according to the expert's own knowledge and experience. The product of all conditional distributions forms the joint ground truth distribution $\mathcal{P}_{GT}$ modeled by the BN, as shown in Equation 5.2.

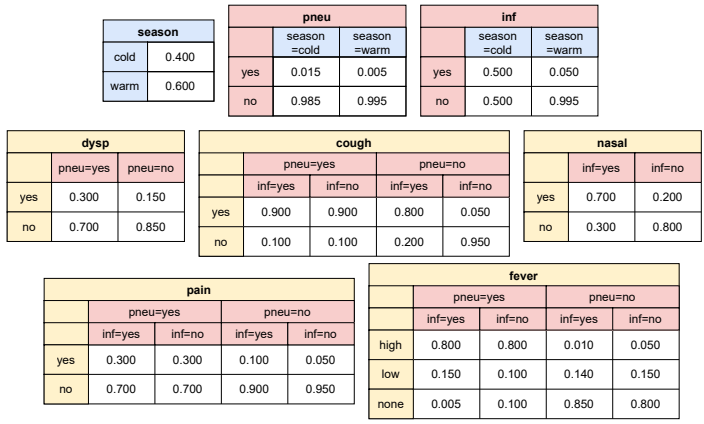
$$\begin{aligned}
&\mathcal{P}_{GT}(season, pneu, inf, dysp, cough, fever, pain, nasal) \\
&= \mathcal{P}_{GT}(season)\mathcal{P}_{GT}(pneu \mid season)\mathcal{P}_{GT}(inf \mid season)\mathcal{P}_{GT}(dysp \mid pneu) \\
&\qquad \mathcal{P}_{GT}(cough \mid pneu, inf)\mathcal{P}_{GT}(fever \mid pneu, inf) \\
&\qquad\qquad \mathcal{P}_{GT}(pain \mid pneu, inf)\mathcal{P}_{GT}(nasal \mid inf) \quad (5.2)
\end{aligned}$$

Step 2. Sample tabular variables We can easily sample from the joint distribution $\mathcal{P}_{GT}$ in a top-down fashion, starting from the root node *season*, generating the 2 diagnoses conditioned on *season*, and finally generating the 5 symptoms conditioned on the diagnoses. This way, we obtain 4000 train samples and 1000 test samples, each with 8 tabular features. We use the *pgmpy* library for implementing our BN and sampling from it [Ankur Ankan and Abinash Panda, 2015].

Step 3. Generate clinical consultation notes We then prompt an LLM (in our case, the GPT-3.5-turbo model from OpenAI [Ouyang et al., 2022]) to generate textual descriptions for each sample, given the presence or absence of the tabular symptoms. We want these textual descriptions to resemble clinical consultation notes made by a general practitioner for each fictitious patient encounter, which means the LLM only gets to observe the symptoms, not the diagnoses. Appendix 5.7.2 outlines the full prompting strategy.

Step 4. Make data partially observed While our dataset now fulfills desired properties (i) and (iii) as outlined in Section 5.3 in the main text, we still need to enforce property (ii), which we split up into 3 parts.

- **Step 4a:** Some symptoms are never encoded in the tabular portion of the data at all. To mimic this, we completely remove features $fever$ and *pain* from the dataset. This way, none of our models (except for baseline model `BN++`, see later) ever observe these variables directly in tabular format, instead only seeing indirect mentions of them in



season	
cold	0.400
warm	0.600

pneu	season =cold	season =warm
yes	0.015	0.005
no	0.985	0.995

inf	season =cold	season =warm
yes	0.500	0.050
no	0.500	0.995

dysp	pneu=yes	pneu=no
yes	0.300	0.150
no	0.700	0.850

cough	pneu=yes		pneu=no	
	inf=yes	inf=no	inf=yes	inf=no
yes	0.900	0.900	0.800	0.050
no	0.100	0.100	0.200	0.950

nasal	inf=yes	inf=no
yes	0.700	0.200
no	0.300	0.800

pain	pneu=yes		pneu=no	
	inf=yes	inf=no	inf=yes	inf=no
yes	0.300	0.300	0.100	0.050
no	0.700	0.700	0.900	0.950

fever	pneu=yes		pneu=no	
	inf=yes	inf=no	inf=yes	inf=no
high	0.800	0.800	0.010	0.050
low	0.150	0.100	0.140	0.150
none	0.005	0.100	0.850	0.800

Figure 5.3: CPTs for all parent-child relations in the ground truth BN, as defined by an expert general practitioner.

the text. Both the symptoms *fever* and *pain* are highly informative for the prediction of pneumonia, and any model that can extract information from the text should reap the benefits.

- **Step 4b:** Other symptoms are only encoded in a subset of the data. We simulate this situation by masking out the remaining symptoms in a subset of the training data. For 1/3 of the training data (1333 samples), we leave out the values for variables *dysp*, *cough* and *nasal*, rendering them unobserved. Note that we either observe all 3 symptoms or none at all, thereby avoiding the need to model missingness and simplifying reality.

- **Step 4c:** Furthermore, real data might not contain textual descriptions for all samples. For this reason, we mask out the textual description for another 1/3 of the training data (1333 samples). This leaves the remaining 1/3 of the training data (1334 samples), with fully observed symptoms and text.

We assume that the background variable is always observed – in a real setting, it could be extracted from the timestamp of the electronic record – and therefore never mask it out. The diagnoses variables are never masked out either, mostly to simplify the setup. We don't mask out anything in the test set, to leave full flexibility during the evaluation process in deciding what to include as input to the predictive models.

5.7.2 Prompting strategies

There are 5 symptoms in our tabular dataset, forming a total of 48 possible combinations. To mimic a realistic setting, we want each sample to have a unique textual description, meaning we need to generate a wide variety

of different texts for each of these combinations of symptoms. We asked an expert general practitioner to provide us with some example patient encounter notes that could be used to prompt the LLM and encourage some variety in its responses. The expert received a symptom configuration and was asked to describe the patient encounter like they normally would in practice. We manually translated these notes from Dutch to English. We ensured that the top 10 most occurring symptom combinations in the training set (for which we will need to generate the highest number of unique textual descriptions) have at least one clinical example note. We had 20 annotated example notes in total. Since some of these notes are based on real encounters the general practitioner remembered from their own clinical practice, we do not make these public.

We now describe our LLM prompting strategy. We structure all prompts according to the OpenAI Chat Completions interface with the GPT-3.5-turbo model, using a temperature of 1 and a frequency penalty of 0.5. The full code to reproduce our prompting strategy is available in our Github repository: `https://github.com/prabaey/bn-text`.

Suppose we need to generate a text description for a symptom combination $\{dysp = d, cough = c, fever = f, pain = p, nasal = n\}$ that needs m unique textual descriptions. First, we check whether this combination is present in the set of examples. If one or more examples are found, we start our prompt with the requested symptom combination, followed by the example responses, see `https://platform.openai.com/playground/p/poSdvoy9dipYIwVPXepRrUyL?model=gpt-3.5-turbo&mode=chat`. If no example is found, we randomly pick two unrelated examples and prompt the language model by listing one after the other, preceded by their corresponding symptom combinations, see `https://platform.openai.com/playground/p/6KOm6pP6DXmMwDxUJWdGGHP1?model=gpt-3.5-turbo&mode=chat`. In both scenarios, after showing the examples, we ask the LLM to generate 5 clinical notes. We repeat the prompt as many times as needed to build up a set of m notes. To further encourage diversity in the responses, we only mention symptoms with positive values in the prompt in 50% of the cases, while in the other 50% we mention all symptoms and their values. We execute the entire pipeline separately for the train and test set. A random sample of the resulting notes were checked for coherence and correctness by the authors, which were deemed sufficient for this proof-of-concept setting.

We use a separate prompting strategy for the combination where all symptoms are absent. This combination occurs most often out of all, though we still only have 4 example notes for it. If we were to exclusively use the prompting strategy from scenario 1, the notes would have little variety. For this reason, we use 5 different strategies that each account for a different number of generated notes. While some strategies encourage the model to mention the absence of respiratory symptoms, others encourage the model to invent a completely unrelated patient encounter. We once again conduct the entire process for the train and test set separately, with the train set

needing 1032 descriptions and the test set needing 388. We describe the 5 strategies we used, together with the proportion of textual descriptions we generated using each strategy and how many samples this comes down to in both the train and test set. These proportions were decided arbitrarily based on how useful we deemed each strategy to be.

1. Provide two in-context examples for symptom combination $\{dysp = no, cough = no, fever = none, pain = no, nasal = no\}$: `https://platform.openai.com/playground/p/Zw9y4EZ8RRfGaZTgBCPu5DPT?model=gpt-3.5-turbo&mode=chat`. (40%, Train: #400, Test: #151)

2. Provide two random out-of-context examples for other symptom combinations: `https://platform.openai.com/playground/p/jYJeFfzXVZ5akndHGkB6cmPk?model=gpt-3.5-turbo&mode=chat`. (5%, Train: #50, #19)

3. Similar to strategy 1, but do not mention the absent symptoms explicitly, thereby encouraging the model to describe cases outside of the respiratory domain: `https://platform.openai.com/playground/p/Ikszv18Zbqr162kF0iSErbCT?model=gpt-3.5-turbo&mode=chat`. We manually go over the generated cases and filter those out where *dysp*, *cough*, *fever*, *pain* and *nasal* are described as present in the patient. (10%, Train: #107, Test: #39)

4. Same as strategy 3, but don't show any examples: `https://platform.openai.com/playground/p/8KaVpr7chxLMHyeKz6Y2mMVE?model=gpt-3.5-turbo&mode=chat`. We manually go over the generated cases and filter those out where *dysp*, *cough*, *fever*, *pain* and *nasal* are described as present in the patient. (10%, Train: #111, Test: #43)

5. Same as stragegy 4, still without showing any examples, but this time telling the model that the patient is not experiencing the symptoms *dysp*, *cough*, *fever*, *pain* and *nasal*: `https://platform.openai.com/playground/p/eXDqpwMkqcvUA8wWiW1HO6et?model=gpt-3.5-turbo&mode=chat`. (35%, Train: #364, Test: #136)

5.7.3 Augmenting BNs with text representations

5.7.3.1 Baseline: Bayesian network (BN and BN^{++})

Training We train a simple Bayesian network (BN) where the Directed Acyclic Graph (DAG), which defines the structure between all the tabular variables, is the same as the one used to generate the data (see Figure 5.1, excluding the unobserved symptoms). This Bayesian network defines the joint distribution as a product of six conditional distributions, one for each

variable, as shown in Equation 5.3. These distributions are learned from the training data using maximum likelihood estimation. This method studies the co-occurrence of particular values of each variable and its parents in the training set, filling up the CPTs as such. We use a K2 prior as a smoothing strategy, to counteract the extremely skewed probability distributions that might be learned when particular combinations of variables are never observed in the training set. We use the *pgmpy* Python library to learn the Bayesian network [Ankur Ankan and Abinash Panda, 2015].

$$\mathcal{P}(B, D_0, D_1, S_0, S_1, S_2) = \mathcal{P}(B)\mathcal{P}(D_0 \mid B)\mathcal{P}(D_1 \mid B)\mathcal{P}(S_0 \mid D_0) \\ \mathcal{P}(S_1 \mid D_0, D_1)\mathcal{P}(S_2 \mid D_1) \quad (5.3)$$

Inference The baseline Bayesian network can only include background and symptoms as evidence (no text). We can calculate the posterior for either diagnosis D_i by applying Bayes' rule and performing marginalization over the variables which are not included in the evidence, as shown in Equation 5.4. We use the Variable Elimination method from *pgmpy* to perform exact inference.

$$\mathcal{P}(D_i \mid B, S_0, S_1, S_2) = \frac{\sum_{D_{1-i}} \mathcal{P}(B, D_0, D_1, S_0, S_1, S_2)}{\sum_{D_0, D_1} \mathcal{P}(B, D_0, D_1, S_0, S_1, S_2)} \quad (5.4)$$

Inclusion of unobserved symptoms We also build a second variant of this baseline (`BN++`) where we additionally include the unobserved symptoms $fever$ and *pain* in the DAG. As opposed to all other models, this model is trained on a version of the training data where these two variables are *not* masked out. This baseline serves as an upper bound to the performance of all other models, which never get to directly observe these two symptoms. Equation 5.5 shows modeled the joint distribution, where U_0 and U_1 represent the unobserved symptoms $fever$ and *pain* respectively. Equation 5.4 can be trivially extended to obtain $\mathcal{P}(D_i \mid B, S_0, S_1, S_2, U_0, U_1)$, where evidence does not only include the partially observed symptoms S_i, but also the unobserved symptoms U_i.

$$\mathcal{P}(B, D_0, D_1, S_0, S_1, S_2, U_0, U_1) = \mathcal{P}(B)\mathcal{P}(D_0 \mid B)\mathcal{P}(D_1 \mid B)\mathcal{P}(S_0 \mid D_0) \\ \mathcal{P}(S_1 \mid D_0, D_1)\mathcal{P}(S_2 \mid D_1)\mathcal{P}(U_0 \mid D_0, D_1)\mathcal{P}(U_1 \mid D_0, D_1) \quad (5.5)$$

5.7.3.2 Baseline: feed-forward neural network (`FF-discr-text`)

Training We train two discriminative feed-forward neural networks (`FF-discr-text`) which receive a vector representation of both the tabular features and the text at the input, and transform it into a one-dimensional representation which is turned into a prediction for $\mathcal{P}(D_i \mid B, S_0, S_1, S_2, T)$

by applying a *sigmoid* non-linearity. We build two completely separate models, one for *pneu* and one for inf, and optimize the neural network weights using maximum-likelihood estimation with a binary cross-entropy loss. As a vector representation for the text, we use *BioLORD* [Remy et al., 2024], which returns a 768-dimensional embedding of the text description. The tabular features are turned into a one-hot encoding, with 11 dimensions in total. Note that each symptom is encoded into a three-dimensional vector, to be able to model the case where the symptom is unobserved, next to its two possible classes (*yes*/*no*). We also experimented with including pairwise, three-way and four-way interactions of background and symptom representations at the input, which adds another 180 dimensions. See Appendix 5.7.4.1 for the final hyperparameter configuration.

Inference The model is trained to maximize the likelihood $\mathcal{P}(D_i \mid B, S_0, S_1, S_2, T)$, so we can directly obtain this probability as an output to the model when we input a test sample. To get a prediction for $\mathcal{P}(D_i \mid B, S_0, S_1, S_2)$, we replace the text at the input by an empty string (simply "") and use its *BioLORD* embedding. Note that the model is equipped to deal with this, since these empty texts occur in 1/3 of the training data as well. Finally, to get a prediction for $\mathcal{P}(D_i \mid B, T)$, we set all symptoms to unobserved and use their corresponding one-hot encoding at the input of the model, instead of the original encoding.

5.7.3.3 Bayesian network with text generator (`BN-gen-text`)

The joint probability distribution modeled by the Bayesian network with text generator is given in Equation 5.6. We parameterize each conditional distribution as a Bernoulli distribution with one trainable parameter per conditional parent configuration, except for the text T, which fits a Gaussian distribution to the text embeddings as explained in Section 5.4.1. We then learn all trainable parameters using maximum likelihood estimation where the likelihood $\mathcal{P}(B, D_0, D_1, S_0, S_1, S_2)$ (Equation 5.6 without factor $\mathcal{P}(T \mid D_0, D_1, S_0, S_1, S_2)$) is maximized based on the training data. This essentially comes down to filling in the CPTs like in a normal Bayesian network.

$$\begin{aligned}\mathcal{P}(B, D_0, D_1, S_0, S_1, S_2, T) = \mathcal{P}(B)\mathcal{P}(D_0 \mid B)\mathcal{P}(D_1 \mid B)\mathcal{P}(S_0 \mid D_0) \\ \mathcal{P}(S_1 \mid D_0, D_1)\mathcal{P}(S_2 \mid D_1)\mathcal{P}(T \mid D_0, D_1, S_0, S_1, S_2) \quad (5.6)\end{aligned}$$

Note that the generative model can easily deal with missing data: if the symptoms are unobserved, only the parameters for $\mathcal{P}(B)$, $\mathcal{P}(D_0 \mid B)$ and $\mathcal{P}(D_1 \mid B)$ are updated. Similarly, samples where the text is missing still contribute to the learned CPTs, while $\mathcal{P}(T \mid D_0, D_1, S_0, S_1, S_2)$ is fitted separately to the observed text embeddings only.

Equations 5.7, 5.8 and 5.9 show how we calculate the posterior likelihood for the diagnoses through Bayesian inference, for different sets of evidence.

Note that we write sums for clarity, but strictly speaking marginalization over T is done by integration over the normally distributed text embedding variable.

$$\begin{aligned}\mathcal{P}(D_i \mid B, S_0, S_1, S_2) &= \frac{\sum_{D_{1-i},T} \mathcal{P}(B, D_0, D_1, S_0, S_1, S_2, T)}{\sum_{D_0,D_1,T} \mathcal{P}(B, D_0, D_1, S_0, S_1, S_2, T)} \\ &= \frac{\begin{array}{r}\sum_{D_{1-i}} \mathcal{P}(B)\mathcal{P}(D_0 \mid B)\mathcal{P}(D_1 \mid B)\mathcal{P}(S_0 \mid D_0)\mathcal{P}(S_1 \mid D_0, D_1) \\ \mathcal{P}(S_2 \mid D_1) \sum_T \mathcal{P}(T \mid D_0, D_1, S_0, S_1, S_2)\end{array}}{\begin{array}{r}\sum_{D_0,D_1} \mathcal{P}(B)\mathcal{P}(D_0 \mid B)\mathcal{P}(D_1 \mid B)\mathcal{P}(S_0 \mid D_0)\mathcal{P}(S_1 \mid D_0, D_1) \\ \mathcal{P}(S_2 \mid D_1) \sum_T \mathcal{P}(T \mid D_0, D_1, S_0, S_1, S_2)\end{array}} \\ &= \frac{\sum_{D_{1-i}} \mathcal{P}(B)\mathcal{P}(D_0 \mid B)\mathcal{P}(D_1 \mid B)\mathcal{P}(S_0 \mid D_0)\mathcal{P}(S_1 \mid D_0, D_1)\mathcal{P}(S_2 \mid D_1)}{\sum_{D_0,D_1} \mathcal{P}(B)\mathcal{P}(D_0 \mid B)\mathcal{P}(D_1 \mid B)\mathcal{P}(S_0 \mid D_0)\mathcal{P}(S_1 \mid D_0, D_1)\mathcal{P}(S_2 \mid D_1)}\end{aligned} \tag{5.7}$$

$$\begin{aligned}\mathcal{P}(D_i \mid B, S_0, S_1, S_2, T) &= \frac{\sum_{D_{1-i}} \mathcal{P}(B, D_0, D_1, S_0, S_1, S_2, T)}{\sum_{D_0,D_1} \mathcal{P}(B, D_0, D_1, S_0, S_1, S_2, T)} \\ &= \frac{\begin{array}{r}\sum_{D_{1-i}} \mathcal{P}(B)\mathcal{P}(D_0 \mid B)\mathcal{P}(D_1 \mid B)\mathcal{P}(S_0 \mid D_0)\mathcal{P}(S_1 \mid D_0, D_1) \\ \mathcal{P}(S_2 \mid D_1)\mathcal{P}(T \mid D_0, D_1, S_0, S_1, S_2)\end{array}}{\begin{array}{r}\sum_{D_0,D_1} \mathcal{P}(B)\mathcal{P}(D_0 \mid B)\mathcal{P}(D_1 \mid B)\mathcal{P}(S_0 \mid D_0)\mathcal{P}(S_1 \mid D_0, D_1) \\ \mathcal{P}(S_2 \mid D_1)\mathcal{P}(T \mid D_0, D_1, S_0, S_1, S_2)\end{array}}\end{aligned} \tag{5.8}$$

$$\begin{aligned}\mathcal{P}(D_i \mid B, T) &= \frac{\sum_{D_{1-i},S_0,S_1,S_2} \mathcal{P}(B, D_0, D_1, S_0, S_1, S_2, T)}{\sum_{D_0,D_1,S_0,S_1,S_2} \mathcal{P}(B, D_0, D_1, S_0, S_1, S_2, T)} \\ &= \frac{\begin{array}{r}\sum_{D_{1-i},S_0,S_1,S_2} \mathcal{P}(B)\mathcal{P}(D_0 \mid B)\mathcal{P}(D_1 \mid B)\mathcal{P}(S_0 \mid D_0)\mathcal{P}(S_1 \mid D_0, D_1) \\ \mathcal{P}(S_2 \mid D_1)\mathcal{P}(T \mid D_0, D_1, S_0, S_1, S_2)\end{array}}{\begin{array}{r}\sum_{D_0,D_1,S_0,S_1,S_2} \mathcal{P}(B)\mathcal{P}(D_0 \mid B)\mathcal{P}(D_1 \mid B)\mathcal{P}(S_0 \mid D_0)\mathcal{P}(S_1 \mid D_0, D_1) \\ \mathcal{P}(S_2 \mid D_1)\mathcal{P}(T \mid D_0, D_1, S_0, S_1, S_2)\end{array}}\end{aligned} \tag{5.9}$$

5.7.3.4 Bayesian network with text discriminator (`BN-discr-text`)

The joint probability distribution modeled by the Bayesian network with text discriminator is given by Equation 5.10. Like before, $\mathcal{P}(B)$ is parameterized as a Bernoulli distribution with one trainable parameter. All other conditional probability distributions are parameterized by one neural text discriminator per conditional parent configuration, resulting in 2 classifiers per conditional distribution, except for $\mathcal{P}(S_1 \mid D_0, D_1, T)$, which has 4 (due to *cough* having both *pneu* and *inf* as a parent). Each text classifier is modeled as a discriminative feed-forward neural network that takes a vector

representation of the text as an input (once again, we use 768-dimensional *BioLORD* embeddings), transforming it into a one-dimensional representation which is then turned into a prediction for $\mathcal{P}(X \mid \mathcal{Y} = y, T)$ by applying a *sigmoid* non-linearity. Here, X represents the child variable (either D_0, D_1, S_0, S_1 or S_2), while $\mathcal{Y}$ represents the parent variable (either B, D_0, D_1 or $\{D_0, D_1\}$) taking on the configuration y. The Bernoulli parameter and neural network weights are trained by jointly maximizing the likelihood in Equation 5.10 over the training set.

$$\begin{aligned}\mathcal{P}(B, D_0, D_1, S_0, S_1, S_2 \mid T) = \mathcal{P}(B)\mathcal{P}(D_0 \mid B, T)\mathcal{P}(D_1 \mid B, T) \\ \mathcal{P}(S_0 \mid D_0, T)\mathcal{P}(S_1 \mid D_0, D_1, T)\mathcal{P}(S_2 \mid D_1, T)\end{aligned} \tag{5.10}$$

Note that the discriminative model can easily deal with missing data: if the symptoms are unobserved, only the parameters for $\mathcal{P}(B)$, $\mathcal{P}(D_0 \mid B, T)$ and $\mathcal{P}(D_1 \mid B, T)$ are updated. When the text is unobserved, we input the *BioLORD* embedding for an empty text into each classifier. For that particular input, the classifiers will simply learn the co-occurrence of child and parent values in the 1/3 of the training data where text is empty, their outputs essentially mimicking the CPTs in a normal Bayesian network (like in the `BN` baseline model).

Equations 5.11 and 5.12 show how we calculate the posterior likelihood for the diagnoses through Bayesian inference, taking the symptoms and text (empty or not) as evidence. $\mathcal{P}(D_i \mid B, T)$ can simply be taken directly as the output of the relevant diagnosis classifier.

$$\begin{aligned}\mathcal{P}(D_i \mid B, S_0, S_1, S_2, T) &= \frac{\sum_{D_{1-i}} \mathcal{P}(B, D_0, D_1, S_0, S_1, S_2 \mid T)}{\sum_{D_0, D_1} \mathcal{P}(B, D_0, D_1, S_0, S_1, S_2 \mid T)} \\ &= \frac{\begin{array}{r}\sum_{D_{1-i}} \mathcal{P}(B)\mathcal{P}(D_0 \mid B, T)\mathcal{P}(D_1 \mid B, T)\mathcal{P}(S_0 \mid D_0, T) \\ \mathcal{P}(S_1 \mid D_0, D_1, T)\mathcal{P}(S_2 \mid D_1, T)\end{array}}{\begin{array}{r}\sum_{D_0, D_1} \mathcal{P}(B)\mathcal{P}(D_0 \mid B, T)\mathcal{P}(D_1 \mid B, T)\mathcal{P}(S_0 \mid D_0, T) \\ \mathcal{P}(S_1 \mid D_0, D_1, T)\mathcal{P}(S_2 \mid D_1, T)\end{array}}\end{aligned} \tag{5.11}$$

$$\mathcal{P}(D_i \mid B, S_0, S_1, S_2) = \mathcal{P}(D_i \mid B, S_0, S_1, S_2, T = \text{“”}) \tag{5.12}$$

5.7.4 Empirical results and analysis

5.7.4.1 Training, hyperparameter tuning and evaluation

All models, except the baselines `BN` and `BN++`, have multiple hyperparameters to tune. We optimized these separately for each model using a train-validation split on the train set (3200/800 sample split out of 4000 samples in total), choosing the hyperparameters that maximize the average precision of $\mathcal{P}(D_i|B, S_0, S_1, S_2, T)$ on the validation set. The full implementation can be

found in our Github repository: `https://github.com/prabaey/bn-text`.

- `FF-discr-text` We optimized the number of epochs, the number of layers (including their width), the batch size, learning rate and weight decay of the Adam optimizer, dropout and whether to include interaction features at the input or not. We optimized these hyperparameters separately for the *pneu* and inf classifier. For the *pneu* classifier, the final configuration we landed on was the following: 200 epochs, 2 layers (dimensions $768 \rightarrow 256 \rightarrow 1$, with a ReLU activation in the middle), batch size 256, learning rate 1e−2, weight decay 1e−3, dropout of 70% in every layer and no interaction features. For the inf classifier, the optimal settings were the same, except that it had 1 layer (dimensions $768 \rightarrow 1$). To make up for the lower complexity of the model (and limited ability to mix features in a single layer), it proved optimal to include the interaction features at the input of this classifier.

- `BN-discr-text` To make for a fair comparison, we used the same layer and dropout configurations that were chosen after tuning the `FF-discr-text` model for the $\mathcal{P}(D_0|B,T)$ classifier (*pneu*) and $\mathcal{P}(D_1|B,T)$ classifier (inf). The symptom classifiers already achieved perfect performance with only 1 layer (dimensions $768 \rightarrow 1$) and without dropout, so we kept these settings. We again used the Adam optimizer to learn the neural weights for all classifiers, with learning rate 1e−2 and weight decay 1e−3. We used a separate learning rate of 0.05 (without weight decay) for learning the $\mathcal{P}(B)$ distribution, which is modeled with a single Bernoulli parameter. Other hyperparameters were also chosen in accordance with the `FF-discr-text` model: 200 epochs and batch size 256.

- `BN-gen-text` For learning the CPT parameters in the Bayesian network, we used an Adam optimizer with a learning rate of 0.05 and no weight decay. We trained for 15 epochs with a batch size of 256. Hyperparameter α, which regularizes the covariance matrix in Equation 5.1, was found to be optimal at 0.85. We use the same α for all 32 Gaussians.

We trained all models with their optimal hyperparameter configurations over the train set of 4000 samples. We repeated this process 5 times, each time with a different initialization seed (except for the `BN` baseline, which is deterministic). For each trained model, we calculated the three posterior diagnosis probabilities for all 1000 samples in the test set. We then obtained the average precision (area under the precision-recall curve) by comparing each prediction to the known label for the diagnosis. We report average precision rather than area under the ROC curve (another metric often used to

assess classification performance), since the former is better suited to evaluate predictive performance in extremely imbalanced datasets [Davis and Goadrich, 2006], which is the case for pneumonia.[1] Furthermore, balancing precision and recall (catching as many cases of pneumonia as possible without including too many false positives) describes the diagnostic task of the GP in the practical use case well.

5.7.4.2 Ablation study

In designing the DAG for models `BN-gen-text` and `BN-discr-text`, we explicitly included an arc between each diagnosis and text. This modeling decision makes sense when one assumes the presence of some unknown and unobserved symptoms in the text. In this section, we investigate how the models would perform if these relations were left out. We first introduce our generative and discriminative ablated models, and then discuss the empirical results.

Ablated BN with Text Generator (`BN-gen-text`$^-$) We remove the arcs $D_0 \rightarrow T$ and $D_1 \rightarrow T$ from the `BN-gen-text` model shown in Figure 5.2, forming the `BN-gen-text`$^-$ model. The text node now has only three parents (symptoms S_0, S_1 and S_2), meaning only 8 conditional Gaussians have to be fitted. Note that this means there are more text embeddings available to fit each Gaussian than there were for the `BN-gen-text` model. The new joint distribution modeled by this Bayesian network is shown in Equation 5.13. Note that it differs from Equation 5.6 only in its definition of the conditional text distribution. We train this model in the same way as before, with the hyperparameters described in Section 5.7.4.1.

$$\begin{aligned}\mathcal{P}(B, D_0, D_1, S_0, S_1, S_2, T) = \mathcal{P}(B)\mathcal{P}(D_0 \mid B)\mathcal{P}(D_1 \mid B)\mathcal{P}(S_0 \mid D_0) \\ \mathcal{P}(S_1 \mid D_0, D_1)\mathcal{P}(S_2 \mid D_1)\mathcal{P}(T \mid S_0, S_1, S_2) \quad (5.13)\end{aligned}$$

Bayesian inference over the ablated DAG partially differs from inference over the original DAG. The calculation of both $\mathcal{P}(D_i \mid B, S_0, S_1, S_2)$ and $\mathcal{P}(D_i \mid B, T)$ incurs only minimal changes: just swap out $\mathcal{P}(T \mid D_0, D_1, S_0, S_1, S_2)$ for $\mathcal{P}(T \mid S_0, S_1, S_2)$ in Equations 5.7 and 5.9. However, the DAG shows that D_i is independent of T when all symptoms are known (no unblocked paths), meaning that $\mathcal{P}(D_i \mid B, S_0, S_1, S_2, T) = \mathcal{P}(D_i \mid B, S_0, S_1, S_2)$.

Ablated BN with Text Discriminator (`BN-discr-text`$^-$) We remove the arcs $T \rightarrow D_0$ and $T \rightarrow D_1$ from the `BN-discr-text` model shown in

[1] We have a positive pneumonia label for only 34 out of 4000 samples in the training set and 14 out of 1000 samples in the test set.

Table 5.3: Average precision over test set for the ablated text models, which do not explicitly include the relation between diagnoses and text.

Model	Average precision for *pneu*		
	$\mathcal{P}(D_0 \mid B, S_0, S_1, S_2, T)$	$\mathcal{P}(D_0 \mid B, S_0, S_1, S_2)$	$\mathcal{P}(D_0 \mid B, T)$
`BN-gen-text`$^-$	0.0892 ($\pm$ 0.0007)	0.0892 ($\pm$ 0.0007)	0.0933 ($\pm$ 0.0009)
`BN-discr-text`$^-$	0.1017 ($\pm$ 0.0008)	0.1041 ($\pm$ 0.0072)	0.0302 ($\pm$ 0.0000)
Model	**Average precision for** *inf*		
	$\mathcal{P}(D_1 \mid B, S_0, S_1, S_2, T)$	$\mathcal{P}(D_1 \mid B, S_0, S_1, S_2)$	$\mathcal{P}(D_1 \mid B, T)$
`BN-gen-text`$^-$	0.8889 ($\pm$ 0.0000)	0.8889 ($\pm$ 0.0000)	0.8914 ($\pm$ 0.0002)
`BN-discr-text`$^-$	0.8065 ($\pm$ 0.0004)	0.8889 ($\pm$ 0.0000)	0.4441 ($\pm$ 0.0000)

Figure 5.2, forming the `BN-discr-text`$^-$ model. This means that there are only 8 classifiers to be learned, as $\mathcal{P}(D_0 \mid B)$ and $\mathcal{P}(D_1 \mid B)$ can now be modeled as simple CPTs, just like $\mathcal{P}(B)$. The joint distribution for the ablated model is shown in Equation 5.14. We train this model with the hyperparameters described in Section 5.7.4.1, using a learning rate of 0.05 to learn the parameters of the CPTs for D_0, D_1 and B.

$$\mathcal{P}(B, D_0, D_1, S_0, S_1, S_2 \mid T) = \mathcal{P}(B)\mathcal{P}(D_0 \mid B)\mathcal{P}(D_1 \mid B)\mathcal{P}(S_0 \mid D_0, T) \\ \mathcal{P}(S_1 \mid D_0, D_1, T)\mathcal{P}(S_2 \mid D_1, T) \quad (5.14)$$

Again, Bayesian inference over the ablated DAG partially differs from inference over the original DAG. $\mathcal{P}(D_i \mid B, S_0, S_1, S_2, T)$ is calculated analogously to Equation 5.11, but with $\mathcal{P}(D_i \mid B)$ instead of $\mathcal{P}(D_i \mid B, T)$. Finally, it is clear from the ablated DAG that the diagnoses are independent of the text if no symptoms are observed (all paths between D_i and T are blocked by unobserved colliders). Therefore $\mathcal{P}(D_i \mid B, T)$ equals $\mathcal{P}(D_i \mid B)$, meaning the `BN-discr-text`$^-$ model cannot extract any information from the text without any observed symptoms.

Analysis Comparing the *pneu* portion of Table 5.3 with Table 5.1, we immediately note that performance drops dramatically in the ablated versions of both the generative and discriminative model. While $\mathcal{P}(D_0 \mid B, S_0, S_1, S_2)$ is still very similar to the `BN` baseline, including text in the prediction $\mathcal{P}(D_0 \mid B, S_0, S_1, S_2, T)$ now does not improve performance. Since we do not model the relation between diagnoses and text, the model can only extract information from the text *through* the three symptoms we explicitly include in the DAG: *dysp*, *cough* and *nasal*. Information regarding other useful symptoms, *pain* and $fever$, cannot be extracted.

While the `BN-gen-text`$^-$ model is able to extract the necessary information on the symptoms S_0, S_1 and S_2 from the text alone ($\mathcal{P}(D_0 \mid B, T) \sim$

$\mathcal{P}(D_0 \mid B, S_0, S_1, S_2)$), `BN-discr-text`$^-$ performs abysmally when only text is included in the evidence. This comes as no surprise when we actually study the DAG: the diagnoses are independent of the text if no symptoms are observed. These independence assumptions do not match the reality we are trying to capture.

Comparing the *inf* portion of Table 5.3 with Table 5.2 shows lower performance of `BN-discr-text`$^-$ compared to `BN-discr-text` when only taking text into account ($\mathcal{P}(D_1 \mid B, T)$). Conversely, `BN-gen-text`$^-$ actually improves over `BN-gen-text` on both $\mathcal{P}(D_1 \mid B, S_0, S_1, S_2, T)$ and $\mathcal{P}(D_1 \mid B, T)$. Since the text node T now only has three parents instead of five, there's more text embeddings available to fit each conditional Gaussian. Combined with the fact that there is no additional information in the text that can help to predict *inf* anyway, modeling the direct relation between diagnosis and text will only result in a less reliable fit of the text distribution by the `BN-gen-text` model.

6

SimSUM – Simulated Benchmark with Structured and Unstructured Medical Records

This chapter introduces SimSUM, a benchmark dataset that links structured tabular data with unstructured text for the domain of respiratory diseases. It is an extension of the benchmark dataset introduced in the previous chapter. Existing multi-modal datasets are too complex for controlled experimentation, motivating the creation of a simpler, self-contained alternative. In SimSUM, both modalities are grounded in the same clinical concepts through a Bayesian network that encodes relevant domain knowledge. The tabular data are generated directly from this Bayesian network, while the textual notes are produced by a prompt-steered large language model, ensuring full transparency of the data-generation process. We evaluate the dataset through expert review of the clinical notes, descriptive analysis of their content, and baseline tabular-only and text-only models for a basic symptom-prediction task. The SimSUM dataset can support research on clinical information extraction (as explored in the next chapter), but also causal effect estimation and synthetic data generation.

* * *

Paloma Rabaey, Stefan Heytens, Thomas Demeester

Journal of Biomedical Semantics

Abstract

Background: Clinical information extraction, which involves structuring clinical concepts from unstructured medical text, remains a challenging problem that could benefit from the inclusion of tabular background information available in electronic health records. Existing open-source datasets lack explicit links between structured features and clinical concepts in the text, motivating the need for a new research dataset.
Methods: We introduce SimSUM, a benchmark dataset of 10,000 simulated patient records that link unstructured clinical notes with structured background variables. Each record simulates a patient encounter in the domain of respiratory diseases and includes tabular data (e.g., symptoms, diagnoses, underlying conditions) generated from a Bayesian network whose structure and parameters are defined by domain experts. A large language model (GPT-4o) is prompted to generate a clinical note describing the encounter, including symptoms and relevant context. These notes are annotated with span-level symptom mentions. We conduct an expert evaluation study to assess note quality and run baseline predictive models on both the tabular and textual data.
Conclusion: The SimSUM dataset is primarily designed to support research on clinical information extraction in the presence of tabular background variables, which can be linked through domain knowledge to concepts of interest to be extracted from the text—namely, symptoms in the case of SimSUM. Secondary uses include research on the automation of clinical reasoning over both tabular data and text, causal effect estimation in the presence of tabular and/or textual confounders, and multi-modal synthetic data generation. SimSUM is not intended for training clinical decision support systems or production-grade models, but rather to facilitate reproducible research in a simplified and controlled setting. The dataset is available at `https://github.com/prabaey/SimSUM`.

6.1 Background

Electronic Health Records (EHRs) are a gold mine of information, containing a mix of structured tabular variables[1] (medication, diagnosis codes, lab results...) and free unstructured text (detailed clinical notes from physicians, nurses...) [Ford et al., 2016]. These EHRs form a valuable basis for training clinical decision support systems, (partially) automating essential processes in the clinical world, such as diagnosis, writing treatment plans, and more [Peiffer-Smadja et al., 2020a, Mujtaba et al., 2019, Rasmy et al., 2021, Li et al., 2020, Xu et al., 2019]. While LLMs can help leverage the potential of the unstructured text portion of the EHR [Zhang et al., 2020,

[1] In the remainder of our work, we use the terms "structured tabular data" and "tabular features" interchangeably.

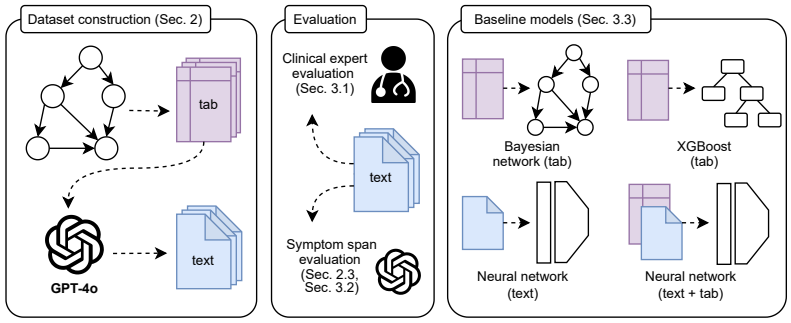


Figure 6.1: General overview of the structure of this paper. We first describe the construction of SimSUM, a simulated dataset combining structured tabular data and unstructured clinical text (Section 6.2). We then evaluate the clinical notes through both a clinical expert review (Section 6.3.1) and automated span-level symptom extraction (Sections 6.2.3 and 6.3.2). Finally, we present four baseline predictive models that take as input the tabular data, the textual data, or both (Section 6.3.3).

Liu et al., 2022b, Huang et al., 2019, Lehman and Johnson, 2023, Singhal et al., 2023, Labrak et al., 2024], these black box systems lack interpretability [Quinn et al., 2022, Zhao et al., 2024, Tian et al., 2024]. In high-risk clinical applications, it can be argued that one should prefer more robust and transparent systems built on simpler, feature-based models, like regression models, decision trees, or Bayesian networks [Rudin, 2019, Sanchez et al., 2022, Lundberg et al., 2020]. However, such models cannot directly deal with unstructured text and require tabular features as an input. For this reason, automated Clinical Information Extraction (CIE) [Ford et al., 2016, Wang et al., 2018] is an essential tool for building large structured datasets that can serve as training data for such systems.

Clinical information extraction remains a challenging task due to the complex nature of clinical notes [Hahn and Oleynik, 2020]. These often leave out important contextual details which an automated system would need in order to correctly extract concepts from the text. State-of-the-art CIE systems are predominantly built on pre-trained language models. Earlier methods relied on models pre-trained for entity and relation extraction in clinical text [Wang et al., 2023a], while more recent approaches draw on LLMs that have absorbed substantial medical knowledge during training [Xu et al., 2024b]. In both paradigms, domain knowledge is acquired implicitly rather than being explicitly encoded. Moreover, current systems typically overlook complementary information available in structured patient records. We propose that CIE could benefit from leveraging two additional sources of information, apart from the unstructured text itself. On the one hand, a range of tabular features are already encoded in the EHR.

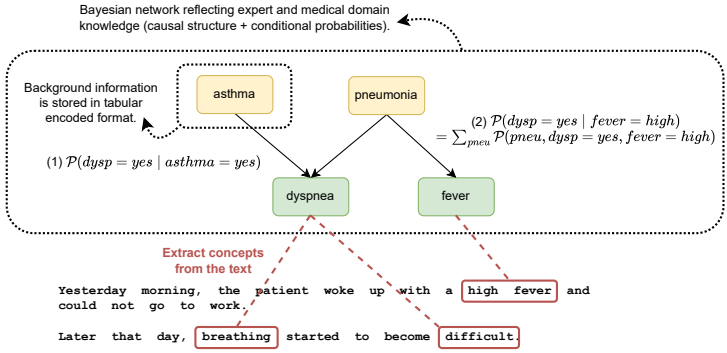


Figure 6.2: We have a clinical description of a patient encounter from which we want to extract some concepts, in this case the symptoms experienced by the patient. Some symptoms might be easy to extract using text-matching, like "high fever". Other symptoms are not mentioned verbatim and are therefore harder to extract, like dyspnea. In this case, additional information on the patient, present in encoded format in the tabular portion of the EHR, together with domain knowledge, may help. We illustrate this idea with two examples.

Example 1: We know that the patient has asthma. Domain knowledge may tell us that the probability of experiencing dyspnea when one has asthma (Equation (1)) is 90%, thereby increasing the prior probability of encountering dyspnea in the text. By integrating this knowledge in the information extraction module, it can more accurately predict the posterior probability of encountering dyspnea in the text.

Example 2: We know that the patient is experiencing high fever. Domain knowledge may tell us that a high fever often co-occurs with dyspnea due to their common cause, which is pneumonia. Even if we do not know that the patient has pneumonia, the probability of dyspnea being mentioned in the text increases as a result of observing high fever. By modeling the joint probability of dyspnea, fever and pneumonia using a Bayesian network, we can get the exact probability of $\mathcal{P}(dyspnea = yes \mid fever = high)$ by summing over the possible presence and absence of pneumonia in a procedure called Bayesian inference (Equation (2), Koller and Friedman [2009]).

These contain information related to a particular patient visit (e.g. partially encoded symptoms or diagnosis codes), as well as information on the medical history of the patient. On the other hand, we can connect this encoded background information with the concepts we are trying to extract from the text, using a Bayesian Network (BN) that represents medical domain knowledge. While ontologies and controlled vocabularies are commonly used to capture standardized concepts and semantic relationships [Sirocchi et al., 2024, Wu et al., 2023], they lack the ability to model causal dependencies or uncertainty. An expert-defined BN complements these resources by modeling how clinical factors influence one another and with what probabilistic strength [Kyrimi et al., 2021d]. A visual example that helps illustrate this idea is shown in Fig. 6.2.

To investigate and implement this idea, we need a **clinical dataset which (i) contains a mix of tabular data and unstructured text, where (ii) the tabular data and the concepts we aim to extract from the text can be linked through domain knowledge.** While open-source datasets like MIMIC-III [Johnson et al., 2016] and MIMIC-IV [Johnson et al., 2023] contain this mix, they are not a perfect fit. First, the area of intensive care in which the data was collected is very extensive, making it hard to isolate a specific use-case for which the domain knowledge could be listed. Second, the portion of the dataset which is encoded into tabular features is often driven by billing needs [Johnson et al., 2023], rather than completeness or accuracy, and does not contain any encoded symptoms, which are concepts that could be interesting to extract from the text for application in clinical decision support systems. Third, the link between the tabular features and the concepts mentioned in the text might be inconsistent due to system design or human errors [Kwon et al., 2024, Lin et al., 2023]. Finally, the EHRs in MIMIC are time series, and while temporal information is a standard aspect of EHRs, it adds significant modeling complexity, making simpler datasets preferable for prototyping novel methods. Other existing datasets linking unstructured clinical text to structured features include BioDEX [D'Oosterlinck et al., 2023], a large set of papers describing adverse drug events, as well as TCGA-Reports [Kefeli and Tatonetti, 2024], a set of cancer pathology reports, both accompanied by tabular patient descriptors and extracted biological features. However, in both cases, it is not trivial to devise a BN representing the relevant expert knowledge, partially because that knowledge is not fully understood yet.

In this work, we simulate a sufficiently realistic dataset that addresses some of these shortcomings, enabling research on incorporating domain knowledge for improved CIE in the presence of tabular variables. Our dataset, called **SimSUM (Simulated Structured and Unstructured Medical records)** is a self-contained set of artificial EHRs in a primary care setting, fulfilling the following requirements:

- Each record in SimSUM is an extract representing a single patient encounter, eliminating the time aspect for simplicity.
- Each record is made up of structured tabular data and unstructured text.
- By design, clinical concepts expressed in the text and encoded in the tabular portion of SimSUM are connected through a Bayesian network representing domain knowledge. In this case, the domain is respiratory diseases, with their associated symptoms and underlying conditions.
- The unstructured text in SimSUM contains additional context on some of the encoded tabular variables. For example, symptoms are stored as structured tabular data, while the text elaborates on the nature of these symptoms, including their severity, onset, and other details.

SimSUM is constructed in the domain of respiratory diseases, simulating patient visits to a primary care doctor. We mimic the scenario where the doctor notes down the patient's symptoms in a clinical note, along with some additional context, and stores this in the EHR together with the encoded diagnosis, as well as the encoded symptoms. Additionally, the EHR stores tabular background information on the underlying health conditions of the patient.

It is important to emphasize that **the SimSUM dataset is not intended for training clinical decision support systems or deploying predictive models in real-world healthcare**. Instead, it provides a controlled research environment for developing and prototyping methods. Although designed to be domain-relevant, SimSUM is entirely synthetic: tabular data is generated via an expert-defined BN, and clinical notes are produced by a Large Language Model (LLM). It simulates a specific use-case in a limited domain – respiratory diseases in primary care – without modeling any temporal dynamics. SimSUM therefore does not reflect the full variability or complexity of real EHRs, and should not be used to evaluate clinical model performance or train production models. Its value lies in enabling research on integrating background information into CIE, within a simplified setting with known and controllable ground-truth structure.

The remainder of this work is structured as shown in Figure 6.1. In short, Section 6.2 describes how we generated SimSUM by simulating 10,000 patient encounters by sampling from an expert-defined BN and prompting an LLM. Section 6.3 analyses the utility of our dataset and discusses its intended use. Finally, Section 6.4 states our main conclusions and highlights SimSUM's limitations.

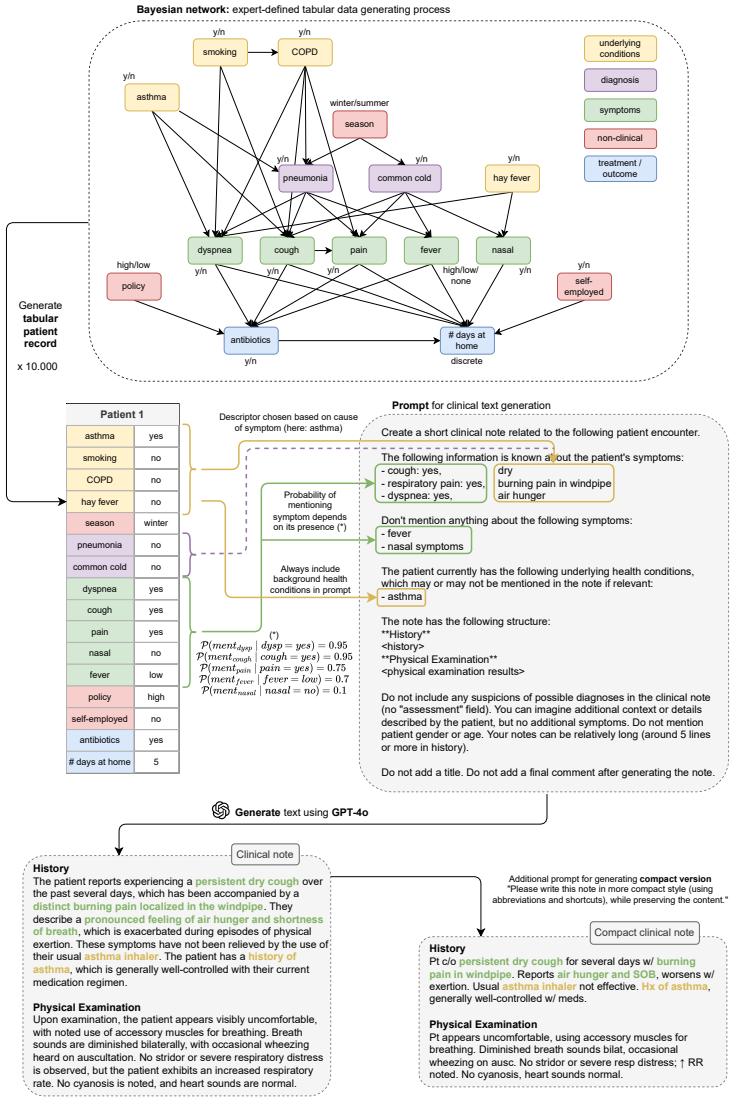


Figure 6.3: Overview of the full data generating process for the SimSUM dataset. First, the tabular portion of the artificial patient record is sampled from a BN, where both the structure and the conditional probability distributions were defined by an expert. Afterwards, we construct a prompt describing the symptoms experienced by the patient, as well as their underlying health conditions (but no diagnoses). We ask the LLM to generate a clinical note describing this patient encounter. Finally, we ask to generate a more challenging compact version of the note, mimicking the complexity of real clinical notes by prompting the use of abbreviations and shortcuts. We generate 10,000 of these artificial patient records in total.

6.2 Construction and content

Our general methodology for generating the artificial patient records is shown in Fig. 6.3. We now zoom in on the two major parts of this data generating process. First, Section 6.2.1 describes how we generated the structured tabular variables through an expert-defined BN. Then, Section 6.2.2 dives into how the clinical notes were generated by the LLM. Finally, Section 6.2.3 describes how we enhanced our dataset by automatically annotating each note with all spans mentioning each symptom.

6.2.1 Modeling structured tabular variables with a Bayesian network

For this part of the process, we enlisted an expert with over 30 years of experience in primary care. The expert has a background in developing and applying BNs in clinical settings and received support from the authors for any technical questions that arose.

Causal structure We asked the expert to define a Directed Acyclic Graph (DAG) which (partially) models the domain of respiratory diseases in primary care, shown in Fig. 6.3. In this DAG, a directed arrow between two variables models a causal relation between them. Central to the model are the diagnoses of *pneumonia* and *common cold*, which may give rise to five symptoms (*dyspnea*, *cough*, *pain*, *fever* and *nasal*). The expert also modeled some relevant underlying conditions which may render a patient more predisposed to certain diagnoses or symptoms: *asthma*, *smoking*, *COPD* and *hay fever*. Based on the symptoms experienced by a patient, a primary care doctor decides whether to prescribe *antibiotics* or not. The presence and severity of the symptoms, as well as the prescription of antibiotics as a treatment, influence the outcome, which is the total number of days that the patient eventually stays home as a result of illness (*days at home*). Finally, there are some non-clinical variables which exert an external influence on the diagnoses, treatment and outcome (*season*, *policy* and *self-employed*). Table 6.1 summarizes all variables and their meaning, as well as their possible values. While this model is a simplification of a real-world clinical process, we believe it is sufficiently realistic to fulfill our purpose of generating a simulated dataset.

Probability distribution To define a data generating mechanism from which we can sample simulated patients, we turn the DAG from Fig. 6.3 into a BN by defining a joint probability distribution. This distribution, shown in Equation 6.1, factorizes into 16 conditional distributions, one for each variable.

Table 6.1: Description of all 16 tabular variables in our dataset, including their type (underlying condition, diagnosis, symptom, non-clinical, treatment or outcome), and their possible values.

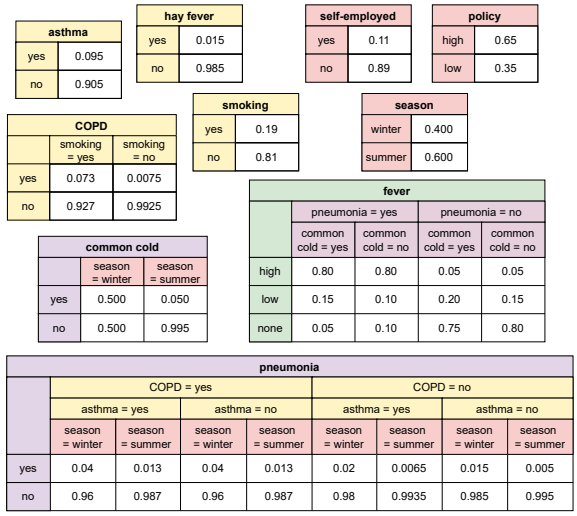
Name	Type	Description	Values
Asthma	underlying condition	Chronic lung disease in which the airways narrow and swell.	yes/no
Smoking	underlying condition	Whether the patient is a regular smoker of tobacco.	yes/no
COPD	underlying condition	Chronic Obstructive Pulmonary Disease, where airflow from the lungs is obstructed.	yes/no
Hay fever	underlying condition	Allergic rhinitis, irritation of the nose caused by an allergen (e.g. pollen).	yes/no
Season	non-clinical	Season of the year.	winter/summer
Pneumonia	diagnosis	Infection that inflames the air sacs in one or both lungs.	yes/no
Common cold	diagnosis	Upper respiratory tract infection, irritation and swelling of the upper airways.	yes/no
Dyspnea	symptom	Shortness of breath, the feeling of not getting enough air.	yes/no
Cough	symptom	Any type of cough.	yes/no
Pain	symptom	Pain related to the airways or chest area.	yes/no
Fever	symptom	Elevation of body temperature.	high/low/none
Nasal	symptom	Nasal symptoms, such as runny nose or sneezing.	yes/no
Policy	non-clinical	Whether the clinician has higher or lower prior inclination to prescribe antibiotics. Can be influenced by many factors, such as local policy in their general practice, their own caution towards antibiotics or level of experience.	high/low
Self-employed	non-clinical	Whether the patient is self-employed, rendering them less inclined to stay home from work for longer periods.	yes/no
Antibiotics	treatment	Whether any type of antibiotics are prescribed to the patient.	yes/no
Days at home	outcome	How many days the patient ends up staying home as a result of their symptoms and treatment.	discrete (0 – ...)

$$
\begin{aligned}
&\mathcal{P}_{joint}(\textit{asthma, smoking, ..., antibiotics, days at home}) = \\
&\mathcal{P}(\textit{asthma})\mathcal{P}(\textit{smoking})\mathcal{P}(\textit{COPD} \mid \textit{smoking})\mathcal{P}(\textit{hay fever})\mathcal{P}(\textit{season})\mathcal{P}(\textit{common cold} \mid \textit{season}) \\
&\mathcal{P}(\textit{pneumonia} \mid \textit{asthma, COPD, season})\mathcal{P}(\textit{dyspnea} \mid \textit{asthma, smoking, COPD, pneumonia, hay fever}) \\
&\mathcal{P}(\textit{cough} \mid \textit{asthma, smoking, COPD, pneumonia, common cold})\mathcal{P}(\textit{nasal} \mid \textit{common cold, hay fever}) \\
&\mathcal{P}(\textit{pain} \mid \textit{cough, pneumonia, COPD, common cold})\mathcal{P}(\textit{fever} \mid \textit{pneumonia, common cold}) \\
&\mathcal{P}(\textit{policy})\mathcal{P}(\textit{self-employed})\mathcal{P}(\textit{antibiotics} \mid \textit{policy, dyspnea, cough, pain, fever}) \\
&\mathcal{P}(\textit{days at home} \mid \textit{antibiotics, dyspnea, cough, pain, fever, nasal, self-employed})
\end{aligned}
\tag{6.1}
$$

We use four different approaches to parameterize these conditional distributions. For this process, we again rely on our expert primary care practitioner. Given that SimSUM is intended as a simplified, controlled benchmark rather than a reflection of real EHR variability, we find input from a single domain expert sufficient to guide the design of its probability distribution.

1. Conditional probability table When the variable is discrete and has a limited number of parents, we ask the expert to define a Conditional Probability Table (CPT). We provide these tables for the variables *asthma*, *smoking*, *hay fever*, *COPD*, *season*, *pneumonia*, *common cold*, *fever*, *policy* and *self-employed* in Figure 6.4. The probabilities in the tables were filled in based on the expert's own experience, as well as demographics in their local general practice and Belgium. While we do not expect these probabilities to generalize to the global patient population as a whole, an expert-informed distribution which contains realistic elements suffices for our use-case.

2. Noisy-OR distribution For categorical variables with many parents, it becomes infeasible to manually fill in the CPT in a clinically meaningful



asthma	
yes	0.095
no	0.905

hay fever	
yes	0.015
no	0.985

self-employed	
yes	0.11
no	0.89

policy	
high	0.65
low	0.35

smoking	
yes	0.19
no	0.81

season	
winter	0.400
summer	0.600

COPD	smoking = yes	smoking = no
yes	0.073	0.0075
no	0.927	0.9925

common cold	season = winter	season = summer
yes	0.500	0.050
no	0.500	0.995

fever	pneumonia = yes		pneumonia = no	
	common cold = yes	common cold = no	common cold = yes	common cold = no
high	0.80	0.80	0.05	0.05
low	0.15	0.10	0.20	0.15
none	0.05	0.10	0.75	0.80

pneumonia	COPD = yes				COPD = no			
	asthma = yes		asthma = no		asthma = yes		asthma = no	
	season = winter	season = summer	season = winter	season = summer	season = winter	season = summer	season = winter	season = summer
yes	0.04	0.013	0.04	0.013	0.02	0.0065	0.015	0.005
no	0.96	0.987	0.96	0.987	0.98	0.9935	0.985	0.995

Figure 6.4: CPTs for the variables *asthma*, *smoking*, *hay fever*, *COPD*, *season*, *pneumonia*, *common cold*, *fever*, *policy* and *self-employed*.

way, because of the large number of possible combinations of parent values. This is the case for the symptoms *dyspnea*, *cough*, *pain* and *nasal* in our BN. To circumvent this problem, we define a Noisy-OR distribution, which assumes an independent causal mechanism behind the activation of a symptom through any of its parents [Koller and Friedman, 2009]. This is a reasonable assumption to make in the case of symptoms with multiple possible causes (parents in the BN): a symptom arises in a patient if any of its possible causes succeeds in activating the symptom through its own independent mechanism [Oniśko et al., 2001, Koller and Friedman, 2009]. As shown in Equation 6.2, the parameterization of the Noisy-OR distribution rests on choosing each parameter p_i, which is the probability that a possible cause X_i activates symptom Y. As a special case, p_0, also known as the leak probability, is the probability that symptom Y is activated as the result of another unmodeled cause (outside of all X_i's). Note that x_i is 1 when the cause X_i is present in the patient ("yes" in our BN), and 0 if not.

$$\begin{aligned}
&\mathcal{P}(Y = 1 \mid X_1 = x_1, \ldots, X_k = x_k) \\
&= 1 - \mathcal{P}(Y = 0 \mid X_1 = x_1, \ldots, X_k = X_k) \\
&= 1 - (1 - p_0)(1 - p_1)^{x_1} \ldots (1 - p_k)^{x_k} \\
&= \text{Noisy-OR}(p_0, p_1, \ldots, p_k; x_1, \ldots, x_k), \text{ with } x_1, \ldots, x_k \in \{0, 1\}
\end{aligned} \tag{6.2}$$

Equations 6.3 through 6.6 define such a Noisy-OR distribution for the symptoms *dyspnea*, *cough*, *pain* and *nasal*. Note that the symptom *fever* is fully

defined through a CPT, since the expert was able to provide intuition on all possible combinations of its two parent values, eliminating the need for a Noisy-OR distribution.

$$\mathcal{P}(\textit{dysp} \mid \textit{asthma, smoking, COPD, hayf, pneu}) = \text{Noisy-OR}(p_0 = 0.05, p_{asthma} = 0.9, p_{smoking} = 0.3, p_{COPD} = 0.9, p_{hayf} = 0.2, p_{pneu} = 0.3) \tag{6.3}$$

$$\mathcal{P}(\textit{cough} \mid \textit{asthma, smoking, COPD, pneu, cold}) = \text{Noisy-OR}(p_0 = 0.07, p_{asthma} = 0.3, p_{smoking} = 0.6, p_{COPD} = 0.4, p_{pneu} = 0.85, p_{cold} = 0.7) \tag{6.4}$$

$$\mathcal{P}(\textit{pain} \mid \textit{COPD}, \textit{cough}, \textit{pneu}, \textit{cold}) = \text{Noisy-OR}(p_0 = 0.05, p_{COPD} = 0.15, p_{cough} = 0.2, p_{pneu} = 0.3, p_{cold} = 0.1) \tag{6.5}$$

$$\mathcal{P}(\textit{nasal} \mid \textit{hayf, cold}) = \text{Noisy-OR}(p_0 = 0.1, p_{hayf} = 0.85, p_{cold} = 0.7) \tag{6.6}$$

3. Logistic regression We model the prescription of *antibiotics* by mimicking the way the clinician's suspicion of pneumonia (which needs to be treated with antibiotics) rises when a higher number of symptoms is present in the patient, with some symptoms weighing more than others. Once their level of suspicion reaches a certain threshold, they decide to prescribe treatment. As shown in Equation 6.7, this process can be approximated in a simplified way through a logistic regression model, taking as an input the symptoms *dyspnea*, *cough*, *pain* and *fever*, as well as the variable *policy*. Here, x_{po} (*policy*) can take on the values 1 (high) or 0 (low), x_d (*dyspnea*), x_c (*cough*) and x_{pa} (*pain*) can take on the value 1 (yes) or 0 (no), and x_f (*fever*) can be 2 (high), 1 (low) or 0 (none).

$$\begin{aligned}&\mathcal{P}(\textit{antibio} = \textit{yes} \mid \textit{policy} = x_{po}, \textit{dysp} = x_d, \textit{cough} = x_c, \textit{pain} = x_{pa}, \textit{fever} = x_f) \\ &= \text{Sigmoid}(-3 + 1 \times x_{po} + 0.8 \times x_d + 0.665 \times x_c + 0.665 \times x_{pa} + 0.9 \times (x_f == 1) \\ &+ 2.25 \times (x_f == 2)), \text{ with } x_{po}, x_d, x_c, x_{pa} \in \{0, 1\}, \text{ and } x_f \in \{0, 1, 2\}\end{aligned} \tag{6.7}$$

We decided on the weights in this model with help of the expert. First, we set the bias term and the coefficient for *policy* based on the following constraint: if there's no symptoms at all, and *policy* is low, then the probability of prescribing antibiotics (due to some other unmodeled cause) should be around 5% . If *policy* is high, it should be around 10%. All other coefficients were then chosen by the expert based on the relative importance

of the symptoms when deciding to prescribe antibiotics. As a final sanity-check, we asked the expert to label a set of test cases with whether or not they would prescribe antibiotics, and compared these with the probability predicted by the model. Table 6.6 in Appendix 6.5.1 shows these results. We see that the predictions made by the model mostly correspond well with the desired input-output behavior, indicating that the proposed coefficients make sense.

4. Poisson regression In a similar fashion, we model the number of *days at home* as a result of the patient's complaints with a Poisson regression model. Assuming a non-linear effect of prescribing antibiotics, we define two separate models: one where antibiotics were not prescribed (Equation 6.8), and one where they were (Equation 6.9). Both models take as an input the symptoms *dysp*, *cough*, *pain*, *nasal* and *fever*, as well as the variable *self-employed*, and predict a mean number of days λ, which parameterizes the Poisson distribution.

$$\begin{aligned}
&\mathcal{P}(\mathit{days\ at\ home} \mid \mathit{dysp} = x_d, \mathit{cough} = x_c, \mathit{pain} = x_{pa}, \mathit{nasal} = x_n, \mathit{fever} = x_f, \\
&\mathit{self\text{-}empl} = x_{se}, \mathit{antibio} = \mathit{no}) = \mathit{Poisson}(\lambda_0) \\
&\lambda_0 = \mathit{exp}(0.010 + 0.64 \times x_d + 0.35 \times x_c + 0.47 \times x_{pa} + 0.011 \times x_n \\
&+ 0.81 \times (x_f == 1) + 1.23 \times (x_f == 2) - 0.5 \times x_{se}) \\
&\text{with } x_d, x_c, x_{pa}, x_n, x_{se} \in \{0, 1\}, \text{ and } x_f \in \{0, 1, 2\}
\end{aligned} \tag{6.8}$$

$$\begin{aligned}
&\mathcal{P}(\mathit{days\ at\ home} \mid \mathit{dysp} = x_d, \mathit{cough} = x_c, \mathit{pain} = x_{pa}, \mathit{nasal} = x_n, \mathit{fever} = x_f, \\
&\mathit{selfempl} = x_{se}, \mathit{antibio} = \mathit{yes}) = \mathit{Poisson}(\lambda_1) \\
&\lambda_1 = \mathit{exp}(0.16 + 0.51 \times x_d + 0.42 \times x_c + 0.26 \times x_{pa} + 0.0051 \times x_n \\
&+ 0.24 \times (x_f == 1) + 0.57 \times (x_f == 2) - 0.5 \times x_{se}) \\
&\text{with } x_d, x_c, x_{pa}, x_n, x_{se} \in \{0, 1\}, \text{ and } x_f \in \{0, 1, 2\}
\end{aligned} \tag{6.9}$$

The coefficients for each model were tuned using gradient descent based on the train cases shown in Table 6.7 in Appendix 6.5.1. The expert was asked to (loosely) label these cases with how long they suspected the patient to stay home as a result of these symptoms. The coefficient for the variable *self-employed* was tuned manually, based on the assumption that being self-employed would shave some days off the predicted number, regardless of the particular symptoms experienced by the patient. As a sanity check, we compared the mean number of days predicted by the model (parameter λ in the Poisson model) with the number of days estimated by the expert for a small set of test cases which were not seen during training. The results

are shown in Table 6.8 in Appendix 6.5.1.

Sampling Once the joint probability distribution from Equation 6.1 is fully specified, we can use this BN to randomly sample the tabular portion of a patient record top-down. Hereby, we start from the root variables without parents at the top and continue further down. Each value is sampled conditionally on the variable's parents' values, using the conditional distributions we have defined. We repeat this process 10,000 times, leaving us with 10,000 artificial patient records consisting of 16 tabular features.

6.2.2 Generating unstructured text with a large language model

Once the tabular patient record has been generated, we prompt an LLM (in this case GPT-4o) to write a clinical note based on this fictional patient encounter and its associated tabular variables. Only the background variables and symptoms may be directly mentioned in the prompt, while the diagnoses *pneumonia* and *common cold* would not yet be known to a clinician who is taking notes during a consultation and are therefore left out. Realistically, they can still influence the content of the note through the descriptions of the symptoms that are included in the prompt, as will be explained later. The treatment and outcome are left out of the prompt as well, just like the non-clinical variables, as all of these are typically either unknown or irrelevant at the time of writing the note.

The prompt is generally structured like the example shown in Fig. 6.3, and is made up of the following parts. Further details and additional example prompts can be found in Appendix 6.5.2.1.

- First, we list the **symptoms** which are experienced by the patient. We do not exhaustively list the full set of symptoms, but rather sample the probability that a symptom is mentioned according to a distribution defined by the expert. This renders the notes more realistic, since real notes do not exhaustively mention the presence or absence of all possible symptoms, instead following the narrative of the patient and the subsequent probing of the clinician.
- While we encourage the LLM to invent additional context around the patient's symptoms, we want this context to at least partially relate to the underlying cause of the symptom. For this reason, we include a **descriptor** next to each symptom, which is randomly sampled from a set of expert-defined phrases that describe the symptom in the case where it results from a particular cause. This way, the diagnoses can influence the content of the note, even when they are not mentioned explicitly in the prompt. The full list of descriptors, as well as further information on how we sample them, is included in Table 6.9 in the Appendix.

- The next part of the prompt lists the **underlying health conditions**, which are assumed to be known up-front, and could therefore influence the content of the note.
- At the end, the prompt lists some **additional instructions**. We tell the LLM that the note must be structured with a "History" portion and a "Physical examination" portion. We also ask it not to mention any suspicions of possible diagnoses, nor patient gender or age. Further motivation for these requests can be found in Appendix 6.5.2.1.

Furthermore, we introduce two additional prompts to extend our dataset and make the notes more realistic.

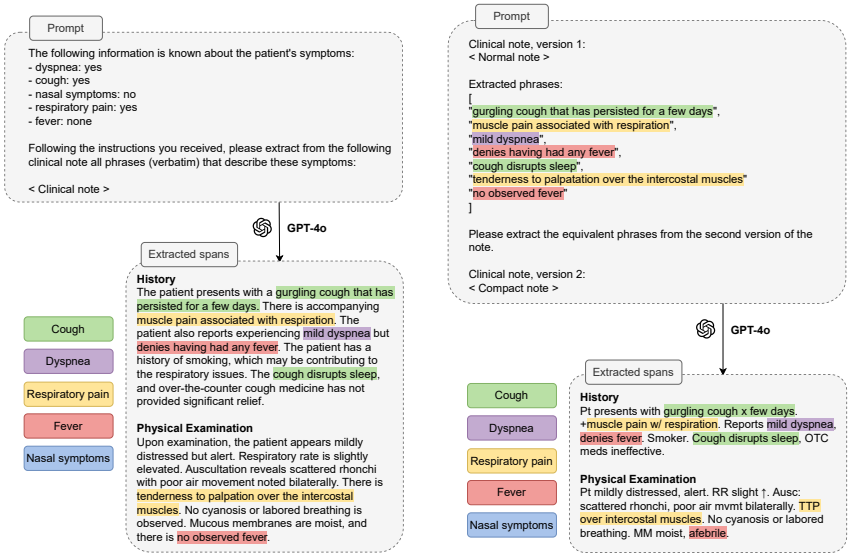
- Real clinical notes can be challenging, often containing abbreviations, shortcuts and denser sentence structure. To make the notes more challenging, we ask the LLM to create a **compact version** of each note through an additional prompt, as can be seen in the bottom part of Fig. 6.3.
- We devise an alternative prompting strategy which is used to generate clinical notes in the special case where **no respiratory symptoms** are present in the patient record. This occurs for around one third of our dataset. In these cases, we encourage the LLM to imagine an alternative reason for the patient visit, unrelated to the respiratory domain. Examples and further details relating to this alternative strategy are included in Appendix 6.5.2.3.

According to the prompts outlined above, we ask the LLM to generate both a clinical note and its compact version for each of the 10,000 patients in the dataset. After generating each note, we use the LLM to perform an additional **consistency check** to ensure the quality of the generated notes. More specifically, we prompt the LLM once again, showing it the same information on the patient's symptoms and underlying health conditions as in the original prompt. We then ask the LLM to decide whether the generated note is consistent with this information or not. This way, the LLM identified 52 out of 10,000 notes as inconsistent with the prompt, which were regenerated and manually checked.

6.2.3 Automated extraction of symptom spans

To further enhance the dataset, we annotated the notes with all spans that mention a symptom. We automatically extracted these phrases from the note using additional LLM calls.

Normal notes Fig. 6.5a shows the prompt and the extracted phrases for one of the normal notes. We first tell the LLM what symptoms the patient does and does not have, and then ask it to extract phrases that describe these symptoms. We use the system message to send extensive



(a) LLM-based annotation of a normal clinical note with symptom spans. The model is prompted to return a JSON object of the form `{"symptom": symptom, "text": extracted phrase}`.

(b) LLM-based annotation of a compact clinical note using spans previously extracted from the corresponding normal note. For clarity, we highlight the extracted spans in the note rather than displaying the raw JSON.

Figure 6.5: Prompting strategy for extracting symptom spans from (a) normal and (b) compact clinical notes using an LLM. Full task instructions are in Appendix 6.5.3

task instructions to the LLM, which we have included in Appendix 6.5.3. We further process the extracted phrases by matching them to the note, disregarding capital letters and punctuation. If no exact match is found, we use another LLM call to correct the mistake, asking it to map the extracted phrase to an exact phrase in the text.

Compact notes For the compact notes, we proceed differently, as shown in Fig. 6.5b. We first tell the LLM which phrases were extracted from the normal note, and ask it to match these with the compact version of the note. We found that this worked better than asking it to extract phrases from the compact note from scratch. As with the normal notes, we post-process the extracted phrases by matching them to the text and asking the LLM to correct itself when no exact match is found.

Using the strategy outlined above, we extract all text spans mentioning each of the 5 symptoms from all 10,000 notes in our dataset. In Section 6.3.2, we use these fine-grained extracted spans to analyze the content of

the notes and quantify the presence of each symptom in the textual portion of SimSUM.

6.3 Utility and discussion

We now present a series of experiments to illustrate the utility and potential of our dataset. First, Section 6.3.1 reports the results of an expert evaluation study assessing the quality of the artificial clinical notes generated by the LLM. Next, Section 6.3.2 uses the extracted symptom spans to further analyze the content of the notes. Following this, Section 6.3.3 evaluates the performance of simple symptom prediction models applied to both the tabular and textual portions of the dataset. Finally, Section 6.3.4 explores the broader research applications of the dataset, including but not limited to CIE.

6.3.1 Expert evaluation of generated notes

We asked five general practitioners to evaluate the quality of a random sample of 30 generated notes. The primary aim of our evaluation study was to assess whether the generated notes adhered to the instructions provided in the prompt. As a secondary aim, we sought to gain an indication of how realistic the content of the notes appeared. We invited general practitioners with an affiliation to the Department of Primary Care at our institution to take part in the evaluation study. The participants were recruited via email, and included based on their willingness to participate when provided information about the task's duration. The evaluators got to see the two versions of the generated note (normal and compact), as well as the prompt that was used to generate them.

6.3.1.1 Evaluation criteria

The evaluators were asked to rate four criteria for each clinical note:

1. **Consistency**: The description of the patient's symptoms and underlying health conditions must correspond with the instructions provided in the prompt.
2. **Realism**: The context and details invented by the LLM and added to the "history" portion of the note should be realistic given the symptoms experienced by the patient, as detailed in the prompt. Furthermore, the elements that are checked in the "physical examination" must be realistic in light of the patient information described in the "history". Both aspects are evaluated separately, focusing on content (what information is included in the note), rather than format (how the information is written down).

Table 6.2: Results of the expert evaluation study. We report average scores (ranging from 1-5) over 30 notes, together with their standard deviation over the five evaluators.

	Normal				**Compact**	
	Consistency	Realism (hist)	Realism (phys)	Clinical accuracy	Content	Readability
mean	4.69	4.53	4.15	4.92	4.88	4.02
std	0.12	0.21	0.30	0.07	0.10	0.31

3. **Clinical accuracy**: The findings described in the "physical examination" must be clinically accurate, both in a standalone fashion and in relation to the patient's symptoms described in the "history".
4. **Quality of compact version**: The content of the compact version of the note must correspond well with the original note, and remain readable despite the use of abbreviations and shortcuts. We evaluate both aspects separately.

To measure consistency, the evaluators were asked to assign a penalty for every element in the note that does not correspond with the requested information in the prompt, and for every symptom requested in the prompt that was missing from the generated note. These penalties were then turned into ratings from 1 (>3 penalties) to 5 (0 penalties). All other aspects were directly rated on a scale of 1 (very bad) to 5 (perfect). To ensure the clarity and intuitiveness of the evaluation criteria, we conducted a pilot run of the evaluation process and obtained feedback from a general practitioner who did not participate in the final study. For more details on the meaning of each rating in each criterion, as well as inter-annotator agreement scores, we refer to Appendix 6.5.4.

6.3.1.2 Evaluation results

Table 6.2 shows the results of the evaluation study. For each rated criterion, we calculate the average score over all 30 notes, and report the mean and standard deviation over the five evaluators. Our artificial notes were rated as highly consistent with the prompt, and therefore with the information present in the tabular portion of the dataset. As shown in Appendix 6.5.4.2, a large majority of the inconsistencies arise due to a violation of the additional instructions (usually by inventing additional symptoms), while the key information included in the prompt was still conveyed correctly in the note. High consistency between the tabular variables and the concepts mentioned in the text makes our dataset a reliable resource for information extraction tasks.

The evaluators also deem the notes sufficiently realistic, though the realism of the history section is rated higher on average than the realism of the physical examination. Out of those notes that scored worse, many included a clinical test that seemed unnecessary to the evaluators, while a few forgot a test that was deemed important. At the same time, the very high score for clinical accuracy is an important indication that the notes do not contain falsehoods. Multiple evaluators mentioned that while the content of the notes seemed realistic, the format did not, as their own notes would be more complex as opposed to the artificial notes, which use clean language and full sentences. This underscores the fact that the dataset should not be used to train any systems which will later be deployed on real notes, and should instead fulfill the role of a research benchmark only.

The compact versions of the notes score very well on content, mostly conveying the same information as the original. They score a little lower in terms of readability, which evaluators often attributed to the extensive use of abbreviations.

6.3.2 Analysis of generated notes with symptom spans

The symptom spans extracted in Section 6.2.3 allow us to quantify the presence of each of the 5 symptoms in the notes. We first evaluate the accuracy of the extracted spans with a small manual evaluation, after which we use the spans to analyze the content of the notes.

Manual evaluation To assess the quality of the automated span extraction process, we conducted an evaluation study over 100 randomly selected notes. We manually annotated each of these notes with phrases that were missed and phrases that were extracted incorrectly. From this, we calculated precision (how many extracted phrases are correct, over the total number of extracted phrases per note) and recall (how many extracted phrases are correct, over the total number of phrases that should have been extracted per note). Averaging these metrics over this set of 100 notes yields a precision of 94.06% and recall of 98.78% for the normal notes, and a precision of 93.55% and recall of 94.50% for the compact notes.

Descriptive statistics The extracted spans allow us to calculate statistics on the presence of each of the 5 symptoms in the notes, which are reported in Table 6.3. We highlight some of the most interesting findings. Table 6.3(a) shows that on average, less spans were extracted for the compact notes than for the normal notes. The explanation is probably two-fold: while some descriptions could be missing from the compact notes entirely, their challenging nature also means more mistakes will be made in the automated extraction, leading to some spans being missed (as reflected by

Table 6.3: Descriptive statistics on the symptoms mentioned in the notes, calculated via the extracted symptom spans.

(a) Statistics on the average number of spans per note, distribution of spans over symptoms, average length of spans, and location of spans ("history" vs. "physical examination" portion of the note). For each metric, we report normal / compact.

	Dyspnea	**Cough**	**Pain**	**Nasal**	**Fever**	**All**
Avg. # spans / note	0.96 / 0.90	1.03 / 0.96	0.52 / 0.49	0.85 / 0.81	0.79 / 0.74	4.16 / 3.90
Total % of spans	23.2% / 23.1%	24.9% / 24.6%	12.6% / 12.5%	20.4% / 20.7%	19.0% / 19.0%	100% / 100%
Avg. # words / span	8.34 / 4.86	8.83 / 5.22	9.34 / 5.96	8.92 / 5.98	7.04 / 4.07	8.46 / 5.17
Avg. # chars / span	57.21 / 30.45	56.70 / 32.14	63.54 / 36.57	63.05 / 41.74	44.82 / 23.95	56.71 / 32.73
% of spans in *hist*	79.3% / 78.8%	92.5% / 91.4%	81.6% / 80.8%	63.0% / 62.1%	65.9% / 65.2%	76.8% / 76.1%
% of spans in *phys*	20.7% / 21.2%	8.5% / 8.6%	18.4% / 19.2%	37.0% / 37.9%	34.1% / 34.8%	23.2% / 23.9%

(b) We show the percentage of notes where a symptom is mentioned (meaning there is at least one span mentioning the symptom in the note), and the average number of spans per note in which a symptom is mentioned. These are reported conditional on the value of the symptom (whether it is present in the patient or not).

		Dyspnea		**Cough**		**Pain**		**Nasal**		**Fever**		
		yes	no	yes	no	yes	no	yes	no	high	low	none
% of notes where sympt. mentioned ($\geq$ 1 span / note)	Normal	99.1%	66.5%	97.0%	64.5%	81.7%	39.9%	95.7%	26.6%	95.9%	71.6%	44.4%
	Compact	98.1%	62.5%	95.4%	59.4%	79.4 %	37.1%	95.4%	25.1%	95.7%	71.4%	42.4%
Avg. # spans where sympt. mentioned / note	Normal	2.03	0.70	1.78	0.65	1.28	0.40	2.47	0.30	2.46	1.55	0.52
	Compact	1.91	0.65	1.66	0.60	1.21	0.37	2.38	0.28	2.33	1.48	0.48

the lower average recall in our manual evaluation). The symptom *pain* is mentioned much less than the other 4 symptoms, even when it is present in the patient. Unsurprisingly, the average span length is much shorter in the compact notes, both in terms of words and characters. According to the extracted spans, most of the symptoms are described in the "history" portion of the clinical note, with proportions differing considerably between symptoms. The percentages of symptoms being mentioned at least once in the note (see Table 6.3(b)) correspond quite well with the probabilities used to build the prompt, as defined in Section 6.2.2. The average number of times a symptom is mentioned per note is much lower when the symptom is not present in the patient. In total, 16.3% of all normal notes do not mention any of the 5 symptoms at all (no spans extracted), while this percentage is 16.4% for the compact notes.

6.3.3 Symptom predictor baselines

In order to set a baseline for future information extraction tasks, such as the one envisioned in Fig. 6.2, we run various prediction models on both the tabular and textual parts of the dataset. These models are trained to predict each of the five symptoms: *dyspnea*, *cough*, *pain*, *fever* and *nasal*. Symptom prediction is a useful task to show the utility of our dataset, since information about the symptoms can be inferred from both the tabular portion of the dataset (through the other variables), as well as the textual portion of our dataset (the clinical notes which partially describe the symptoms).

Section 6.3.3.1 describes the task setup and gives an overview of the various types of baselines we run on our dataset. Section 6.3.3.2 provides a detailed overview of each baseline model, followed by Section 6.3.3.3, which analyzes their performance on the task of automated symptom extraction using the SimSUM dataset.

6.3.3.1 Task overview

Tabular-only baselines Two of our baselines only get to see the tabular portion of the dataset at the input: Bayesian network (**BN-tab**) and XGBoost (**XGBoost-tab**). We include XGBoost because it has consistently demonstrated state-of-the-art performance on a wide range of tabular prediction tasks, often serving as a strong baseline in machine learning benchmarks [Shwartz-Ziv and Armon, 2022]. We use both models to predict each symptom in three settings, differing from one another in the set of tabular features that are taken as an input, which we call the evidence:

- $\mathcal{P}(sympt \mid all)$: Predict the symptom given all other tabular features as evidence. This set includes the background, diagnoses, non-clinical, treatment and outcome variables, as well as the other symptoms.

- $\mathcal{P}(sympt \mid no\text{-}sympt)$: Predict the symptom given all other tabular features as evidence, except for the other symptoms. This mimics the setting where we have tabular features available in the patient record, but have not extracted any symptoms from the text yet. This set includes the background, diagnoses, non-clinical, treatment and outcome variables.
- $\mathcal{P}(sympt \mid realistic)$: Predict the symptom given a more realistic set of tabular features as evidence. We do not expect *policy*, *self-employed* and *days at home* to be recorded in any kind of realistic patient record, and therefore leave them out of this evidence set. As in the *no-sympt* setting, we do not include the symptoms either. In other words, this set includes the background, diagnoses, season and treatment variables.

Text baselines Apart from the tabular-only baselines, we also train baselines that get to see the text. Our **neural-text** classifier takes only the text as an input (in the form of a pretrained clinical sentence embedding) and outputs the probability that a symptom is mentioned in the text. We extend this text-only baseline by concatenating a numerical representation of the tabular features to the text embedding at the input, forming the **neural-text-tab** baseline. Again, we do this for each of the three evidence settings outlined above. Note that this is the only model that combines both the background knowledge available in the tabular features with the unstructured text, and it does so in a naive way. Future work will focus on improving the performance of this model by exploiting the relations between the tabular concepts, as envisioned in Fig. 6.2.

6.3.3.2 Models

BN-tab We provide the causal structure in Fig. 6.3 to the BN, and learn all parameters in the CPTs, Noisy-OR distributions, logistic regression model and Poisson regression model from the training data. In each case, we use maximum likelihood estimation to estimate the parameters. Where we do not directly learn a CPT (for the variables *dyspnea*, *cough*, *pain*, *nasal*, *antibiotics* and *days at home*), we evaluate the learned distribution for each combination of child and parent values to obtain a CPT. For more details, we refer to Appendix 6.5.5.1. We then use variable elimination over the full joint distribution to evaluate the capability of the learned BN to predict each of the symptoms, taking different sets of variables as evidence according to the three settings described earlier (*all*, *no-sympt* and *realistic*).

XGBoost-tab We train an XGBoost classifier for each symptom in combination with each of the three evidence settings, meaning each classifier sees a different set of tabular features at the input. We optimize the hy-

perparameters separately for each combination (15 in total) using 5-fold cross-validation. For more details, we refer to Appendix 6.5.5.2.

Neural-text We train a neural classifier that takes only the text as an input and is trained to predict the probability a symptom is mentioned. We train separate classifiers for each symptom. We first split the text into sentences, and transform these into an embedding using the pretrained clinical representation model BioLORD-2023 [Remy et al., 2024]. We explore four settings for turning these sentence embeddings into a single note embedding:

- *hist*: We average all sentence embeddings for the sentences in the "history" portion of the note.
- *phys*: We average all sentence embeddings for the sentences in the "physical examination" portion of the note.
- *mean*: To get a single representation for the full note, we take the average of the *hist* and *phys* embeddings.
- *concat*: Idem as previous, but now the embeddings for the two portions are concatenated.

Additionally, we create symptom-specific *span* embeddings for each note as well. To this end, we take the set of spans mentioning the symptom in each note (as extracted automatically in Section 6.2.3), transform each of these spans into their BioLORD embedding, and take the average to get one span embedding per symptom per note. No matter the type, each note embedding is fed into a multi-layer perceptron with one hidden layer, followed by a Sigmoid activation for the symptoms *dyspnea*, *cough*, *pain* and *nasal*, and a Softmax activation with 3 outputs heads for the symptom $fever$. We trained each model (i.e. each combination of symptom and embedding type) using the binary or multiclass cross-entropy objective over the symptom labels. For more details, we refer to Appendix 6.5.5.3.

Neural-text-tab We extend the **neural-text** baseline by concatenating the *mean* text embeddings with the tabular variables at the input of each neural classifier. All categorical tabular variables were first transformed to a one-hot encoding, while the variable *days at home* was preprocessed using standard scaling. We used the same architecture as the **neural-text** baseline (only changing the dimension of the input layer), and again trained separate classifiers for each symptom combined with each evidence setting (*all*, *no-sympt* and *realistic*). For more details, we refer to Appendix 6.5.5.4.

Training setup We use a random 8,000/2,000 split to obtain a train and test subset of SimSUM. We use cross-validation on the train set to tune any hyperparameters associated with each model, and report the average F1-score over the test set after training with 5 different seeds. Seeding randomizes the initialization of the model weights, as well as batching. For the binary symptoms, we use a 0.5 decision threshold. For fever, which has three possible categories, the class with the highest predicted probability is

Table 6.4: F1-score obtained over the test set for each of our baseline models. For fever, we report the macro F1 score over all three categories. The results for the text classifiers trained over the normal vs. the compact version of the notes are grouped together for readability. We report results for the *mean* embedding type, while **neural-text** baseline results for the other embedding types can be found in Table 6.5. We underline the best result obtained by the tabular-only models, while the best overall result per symptom and note version is in shown in **bold**.

	dyspnea	**cough**	**pain**	**nasal**	**fever**
BN-tab					
- all	0.7370	0.7816	0.2386	0.7146	0.4864
- no-sympt	0.7153	0.7776	0.1312	0.7146	0.4384
- realistic	0.6698	0.7763	0.0280	0.7146	0.3594
XGBoost-tab					
- all	0.6639	0.7848	0.4070	0.7130	0.4111
- no-sympt	0.6612	0.7779	0.3638	0.7146	0.4015
- realistic	0.6626	0.7798	0.3698	0.7146	0.3951
neural-text					
- normal	**0.9617**	**0.9603**	0.8143	**0.9628**	**0.9096**
neural-text-tab					
- normal + all	0.9544	0.9555	0.8191	0.9613	0.9091
- normal + no-sympt	0.9594	0.9577	0.8155	0.9611	0.8996
- normal + realistic	0.9569	0.9596	**0.8266**	0.9591	0.9029
neural-text					
- compact	0.9444	0.9397	**0.7940**	0.9622	0.9010
neural-text-tab					
- compact + all	**0.9532**	0.9395	0.7873	0.9589	0.8962
- compact + no-sympt	0.9429	0.9456	0.7856	**0.9632**	0.8973
- compact + realistic	0.9379	**0.9480**	0.7922	0.9630	**0.9025**

chosen.

6.3.3.3 Symptom predictor results

Table 6.4 compares the results obtained for all baselines, using the *mean* embedding type for the text-based classifiers. The tabular-only baselines (**BN-tab** and **XGBoost-tab**) perform consistently worse than the baselines that include text (**neural-text** and **neural-text-tab**). The evidence setting where *all* other features are included as evidence usually performs better than the other settings.

For the text baselines, we note that there is little room for improvement in the dyspnea, cough and nasal classifiers. On the other hand, the symptoms pain and fever are harder to predict. We can assume that this is largely because these symptoms are mentioned less often in the notes, even

Table 6.5: Results for the **neural-text** baseline using different embedding types at the input. We report F1-score over the test set for both the normal and compact versions of the notes. The best results per symptoms and per note version is shown in **bold**, the runner-up is underlined. Fig. 6.11 in the Appendix compares these results visually.

Normal	**dyspnea**	**cough**	**pain**	**nasal**	**fever**
Sentence embedding					
- hist	0.9520	0.9652	0.8089	0.9504	0.9040
- phys	0.8969	0.8226	0.6956	0.9565	0.8523
- mean	0.9617	0.9603	0.8143	0.9628	0.9096
- concat	0.9504	0.9572	0.8268	0.9589	0.9044
Span embedding	**0.9730**	**0.9785**	**0.8731**	**0.9712**	**0.9226**
Compact	**dyspnea**	**cough**	**pain**	**nasal**	**fever**
Sentence embedding					
- hist	0.9307	0.9523	0.7734	0.9515	0.9001
- phys	0.8824	0.7802	0.6514	0.9531	0.8315
- mean	0.9444	0.9397	0.7940	0.9622	0.9010
- concat	0.9522	0.9473	0.7934	0.9411	0.9007
Span embedding	**0.9685**	**0.9719**	**0.8613**	**0.9705**	**0.9207**

if they are present in the patient, as supported by Table 6.3. We also see a consistent gap in performance between the normal and compact notes, which can be attributed to the higher complexity of the latter. While the **neural-text-tab** baseline does not manage to outperform the **neural-text** baseline for the normal notes, we do see marginal improvements by including the tabular features in the case of the compact notes.

Table 6.5 further breaks down the results for the **neural-text** classifier over the different embedding types. We clearly see that the *span* embeddings perform best, since this method preselects phrases that are important for the prediction of a certain symptom. While the same phrases are still implicitly embedded in the other types, the important information gets diluted as sentence embeddings are averaged, and might even be lost entirely. However, in a realistic setting, one would rarely have access to these symptom-specific *span* embeddings, as this would require either a manual annotation effort, or a (small) local LLM to facilitate automated annotation in a way that preserves patient privacy. We do not expect such a local LLM to achieve near the same span detection performance as GPT-4o, which we were only able to use in this case due to the artificial nature of our notes. The performance achieved with these *span* embeddings should therefore be seen as a best-case reference within our benchmark, rather than an achievable goal in a practical setting.

Comparing the four different sentence embedding types, the performance difference between the *mean* and *concat* settings is usually small, with *mean*

slightly outperforming *concat* overall. While the score for *hist* comes close to those for *mean* and *concat*, the latter two usually still outperform the former for the normal notes, showing that there is some complementary information in the "history" and "physical examination" portions. At the same time, we note that using only the *hist* embedding clearly outperforms the *phys*-only setting. This makes sense, as the "history" section of the note outlines the symptoms experienced by the patient more clearly.

6.3.4 Intended uses

The primary purpose of the SimSUM dataset is to enable research on CIE in multimodal contexts, where effective integration of structured and unstructured data is required. A key objective for future work is to realize the approach illustrated in Fig. 6.2, in which domain knowledge links structured features to textual concepts, enabling more accurate information extraction. As demonstrated in the previous section, even a simple method—concatenating tabular features with text embeddings at the input of a neural classifier—already improves symptom extraction performance, particularly on the more challenging, compact notes. This suggests that background information can help recover symptoms that are mentioned less explicitly in the text, underscoring the utility of our dataset for developing and evaluating such approaches.

Apart from this, we also foresee multiple secondary uses of the dataset. First, the dataset could facilitate research on the automation of clinical reasoning over tabular data and text, following our previous work in Rabaey et al. [2024c]. Second, it could be used to benchmark causal effect estimation methods in the presence of textual confounders, as proposed by Arno et al. [2024] and Ma et al. [2025]. This is possible thanks to the purposeful inclusion of both a treatment and outcome variable in our dataset. Third, there has been increasing interest in clinical synthetic data [Hernandez et al., 2022], where a set of patient characteristics is turned into a synthetic version that is meant to protect the privacy of individuals in the original dataset. Our dataset could serve as a benchmark for comparing synthetic data generation methods that jointly generate tabular variables and text [Lee, 2018, Guan et al., 2019, Wang and Sun, 2022, Ibrahim et al., 2025]. In short, any area of research focusing on the intersection of tabular data and text in healthcare can potentially benefit from our proposed benchmark.

Conversely, there are important limitations to the intended use of SimSUM. As it is fully synthetic—constructed from expert-defined tabular data and language model–generated notes—it does not capture the full complexity, variability, or noise characteristic of real electronic health records. For this reason, we strongly advise against using SimSUM to evaluate clinical model performance or to train models intended for production. Instead, its value lies in supporting research scenarios, such as those outlined above, which benefit from the availability of a known and controllable ground-truth

structure.

6.4 Conclusions

We introduced SimSUM, a simulated dataset of structured and unstructured medical records designed to support research on clinical information extraction using background tabular data. By explicitly linking structured features and textual concepts through a Bayesian network, SimSUM provides a controlled setting to explore how domain knowledge can enhance information extraction from clinical notes. Indeed, our baseline results suggest that integrating tabular background features with text improves symptom extraction, particularly for harder-to-predict cases like pain and fever. Future research can explore more advanced hybrid models that leverage domain knowledge to link structured and unstructured data. Additionally, the annotated symptom spans, though imperfect, offer a foundation for developing lightweight span detection models and explainable extraction methods.

Limitations Despite aiming for realism, SimSUM intentionally simplifies many aspects of real-world clinical data to support its role as a controlled research benchmark. The dataset is synthetic by design, with known ground-truth relationships, and is not intended to generalize to real clinical notes or settings. Expert reviewers confirmed that while the content of the notes is generally plausible, the writing style does not reflect how clinicians typically document patient encounters. Additionally, SimSUM represents static patient snapshots rather than evolving time series, which limits its applicability in domains like intensive care where temporal dynamics are critical.

Acknowledgements Paloma Rabaey's research is funded by the Research Foundation Flanders (FWO Vlaanderen) with grant number 1170124N. This research also received funding from the Flemish government under the "Onderzoeksprogramma Artificiële Intelligentie (AI) Vlaanderen" programme. The authors would like to thank Géraldine Deberdt, Thibault Detremerie, An De Sutter, Veerle Piessens and Florian Stul for participating in the expert evaluation of the artificial clinical notes. The authors would also like to thank Henri Arno for his early contributions to the dataset design.

6.5 Appendix

6.5.1 Bayesian network

Table 6.6 shows the comparison of the probability predicted by the *antibiotics* model (Equation 6.7 in Section 6.2.1) with test cases labeled by the expert. Table 6.7 shows the expert-labeled train cases that were used to fit

Table 6.6: Test cases labeled by the expert on whether to prescribe antibiotics or not (all assume *policy* = *low*). "label" indicates the expert's decision, while "pred" indicates the probability predicted by the antibiotics model, according to Equation 6.7.

Symptoms				**Antibiotics**	
dysp	cough	pain	fever	label	pred.
no	yes	no	high	no	0.48
no	yes	yes	high	yes	0.64
yes	yes	no	high	yes	0.67
yes	yes	yes	high	yes	0.80
yes	no	no	high	yes	0.51
no	no	yes	high	yes	0.48
no	no	no	high	no	0.32
no	no	no	low	no	0.11
no	yes	no	low	no	0.19
yes	yes	no	low	no	0.35
no	yes	yes	low	no	0.31
yes	yes	yes	low	yes	0.51
yes	yes	yes	none	yes	0.30
yes	no	yes	none	no	0.18
yes	yes	no	none	no	0.18
yes	no	no	none	no	0.10
no	yes	no	none	no	0.09
no	no	yes	none	no	0.09

the coefficients in the # *days* model (Equation 6.8 and 6.9 in Section 6.2.1). Table 6.8 shows the comparison of the probability predicted by the # *days* model with test cases labeled by the expert.

6.5.2 LLM prompt

6.5.2.1 Prompt details

We prompt an LLM to generate a clinical note based on the tabular variables associated with each fictional patient record. The scenario we simulate artificially is as follows. The patient goes to the primary care physician, telling them their symptoms and possible underlying conditions, along with additional context on the severity of these symptoms, when they started, and other details. The physician takes descriptive notes during this consultation, writing down the (recent) history prescribed by the patient. Then, based on the patient's described complaints, they conduct a physical examination, writing down all findings. Both parts together then form the textual description of the patient encounter.

Table 6.7: Train cases labeled by the expert on how many days they expect a patient to stay home (all assuming *self-employed* = *no*), with and without prescribing antibiotics. "label" indicates the expert's estimation, while "pred" indicates the mean number of days λ predicted by the *days at home* models (Equation 6.8 and 6.9).

Symptoms					**Days at home**			
					antibio = no		**antibio = yes**	
dysp	cough	pain	nasal	fever	label	pred.	label	pred.
no	no	no	no	none	1.5	1	1	1
no	yes	no	no	high	4	5	4	3
no	yes	no	no	low	2	3	2	2
no	yes	yes	no	high	9	8	4	4
yes	yes	no	no	high	10	9	5	5
yes	yes	yes	no	high	14	15	7	7
no	yes	yes	no	low	5	5	3	3
yes	yes	no	no	low	6	6	4	4
yes	yes	yes	no	low	10	10	5	5
yes	yes	yes	no	none	4	4	4	4
no	yes	yes	no	none	2	2	2	2
yes	yes	no	no	none	3	3	3	3
yes	no	yes	no	none	3	3	3	3
no	no	no	yes	none	2	1	2	1
no	yes	no	yes	high	4	5	4	3
no	yes	no	yes	low	2	3	2	2
no	yes	yes	yes	high	9	8	4	4
yes	yes	no	yes	high	10	9	5	5
yes	yes	yes	yes	high	14	15	7	7
no	yes	yes	yes	low	5	5	3	3
yes	yes	no	yes	low	6	6	4	4
yes	yes	yes	yes	low	10	10	5	5
yes	yes	yes	yes	none	4	4	4	4
no	yes	yes	yes	none	2	2	2	2
yes	yes	no	yes	none	3	3	3	3
yes	no	yes	yes	none	3	3	3	3

Table 6.8: Test cases (not part of training set) labeled by the expert on how many days they expect a patient to stay home (all assuming *self-employed* = *no*), with and without prescribing antibiotics. "label" indicates the expert's estimation, while "pred" indicates the mean number of days λ predicted by the *days at home* models (Equation 6.8 and 6.9).

Symptoms					**Days at home**			
					antibio = no		**antibio = yes**	
dysp	cough	pain	nasal	fever	label	pred.	label	pred.
yes	no	no	no	high	6	7	4	4
no	no	yes	no	high	6	6	3	3
yes	no	yes	no	high	12	11	5	5
yes	no	no	no	low	4	4	3	3
no	no	yes	no	low	4	4	3	2
yes	no	yes	no	low	6	7	5	3

We now describe the different parts of the prompt.

Presence of symptoms The first block of information in the prompt concerns the symptoms experienced by the patient. We do not list the full set of symptoms exhaustively, since there is a possibility that a patient does not mention a symptom to the clinician, or that the clinician does not find it noteworthy enough to write down. We therefore ask our expert to list the probability of each symptom being mentioned in a clinical note, both when the symptom is positive and when it is negative. While we do not expect these arbitrary probabilities to generalize to all settings, it helps to bring some variety and realism in the notes we generate. The probabilities are as follows:

- $\mathcal{P}(ment_{dysp} = yes \mid dysp = yes) = 0.95$, $\mathcal{P}(ment_{dysp} = yes \mid dysp = no) = 0.75$
- $\mathcal{P}(ment_{cough} = yes \mid cough = yes) = 0.95$, $\mathcal{P}(ment_{cough} = yes \mid cough = no) = 0.9$
- $\mathcal{P}(ment_{pain} = yes \mid pain = yes) = 0.75$, $\mathcal{P}(ment_{pain} = yes \mid pain = no) = 0.3$
- $\mathcal{P}(ment_{fever} = yes \mid fever = high) = 0.95$, $\mathcal{P}(ment_{fever} = yes \mid fever = low) = 0.7$, $\mathcal{P}(ment_{fever} = yes \mid fever = none) = 0.4$
- $\mathcal{P}(ment_{nasal} = yes \mid nasal = yes) = 0.95$, $\mathcal{P}(ment_{nasal} = yes \mid nasal = no) = 0.1$

For each symptom, we sample whether it is to be mentioned in the prompt, conditional on its value, according to the probabilities stated above. As can be seen in Fig. 6.3, we explicitly tell the model what symptoms to mention and which to steer clear from. We randomly permute the ordering of the symptoms in each prompt.

Table 6.9: Descriptors used in the prompt to describe each symptom when it is present in the patient. Depending on the cause(s) of the symptom (as present in the tabular patient record), we randomly sample from a different set of descriptors.

Symptom	Cause	Descriptors
dyspnea	asthma	attack-related, at night, in episodes, wheezing, difficulty breathing in, feeling of suffocation, nighttime stuffiness, provoked by exercise, light, severe, not able to breathe properly, air hunger
	smoking	during exercise, worse in morning, mild
	COPD	chronic, worse during flare-up, worse when lying down, difficulty sleeping, air hunger
	hay fever	light, mild, stuffy feeling, all closed up
	pneumonia	light, mild, severe, no clear cause
cough	asthma	attack-related, dry
	smoking	productive, mostly in morning, during exercise, gurgling
	COPD	phlegm, sputum, gurgling, worse when lying down
	pneumonia	for over 7 days, light, mild, severe, non-productive at first and later purulent
	common cold	prickly, irritating, dry, phlegm, sputum, light, mild, severe, constant, day and night
pain	asthma	tension behind sternum
	COPD	light, mild
	cough	muscle pain, burning pain in trachea, burning pain in windpipe, scraping pain in trachea, scraping pain in windpipe
	pneumonia	light, mild, severe, localized on right side, localized on left side, associated with breathing
	common cold	burning pain in trachea, burning pain in windpipe, scraping pain in trachea, scraping pain in windpipe, light, mild

Symptom descriptors To make the note realistic, the LLM must invent some context regarding the patient's symptoms when writing the history portion of the note. We want this context to indirectly relate to the cause of these symptoms, as they would in a real patient encounter. For example, a cough induced by asthma would likely be momentarily and attack-related, while a cough resulting from pneumonia might be more persistent over the longer term. We therefore ask the expert to write down a list of adjectives or phrases describing each symptom, conditioned on the cause of the symptom. These descriptors can be found in Table 6.9. The list of possible causes for a symptom is simply the list of parents in the BN.

For each symptom which is present in the patient and selected to be mentioned in the note, we check the tabular patient record for the possible causes. For example, for the symptom cough, the possible causes are asthma, smoking, COPD, pneumonia and common cold (its parents in the BN). In the example in Fig. 6.3, asthma is the only cause present in the patient. We therefore randomly sample a descriptor from the list of descriptors for cough in the presence of asthma, in this case the adjective "dry". This adjective is added in the prompt. If multiple causes are present, we find the strongest cause, and sample from that list. The strongest cause is pneumonia, followed by common cold, followed by all other causes. If neither pneumonia nor common cold is part of the multiple causes, we simply make a bag of all descriptors associated to the causes which are present in the patient, and sample from that bag. In the event that no causes are present, yet a symptom is still observed (which is possible due to the leak probabilities in the Noisy-OR distributions), we do not add a descriptor. Note that while the diagnoses pneumonia and common cold should not be mentioned explicitly, they indirectly and subtly influence the content of the note through the descriptors, adding another realistic dimension to the content of the note.

Underlying health conditions While the diagnoses should not be mentioned directly in the note, it is realistic to assume that the note would mention underlying health conditions the patient may have. Since these health conditions are assumed to be known up-front, as they are part of the history of the patient, they may contribute a lot to the interpretation of the symptoms by both the patient themselves and the clinician writing down the note. We therefore add them to the prompt as well, as can be seen in Fig. 6.3. We do not force the LLM to explicitly mention these in the note, since it seems feasible that a clinician would not mention them every time. Should there be more than one underlying condition, we mention them all, randomly permuting the order. If there are no underlying health conditions, we simply remove this part of the prompt.

Additional instructions We tell the LLM that the note must be structured with a "History" portion and a "Physical examination" portion.

While the "History" portion describes the patient's self-reported symptoms and underlying health conditions, which are in large part dictated by the prompt, the "Physical examination" portion leaves the LLM with more freedom to imagine additional clinical examinations which were performed on the patient. As such, the "Physical examination" portion has a lot of potential for adding complexity, clinical terminology and realism to the note.

We also add some additional instructions to the prompt, asking it not to mention any suspicions of possible diagnoses. We further tell the LLM it can imagine context or details, but no additional symptoms. We noticed that if we left this part out, the LLM would sometimes mention the symptoms we specifically asked to leave out. We ask not to mention patient gender or age, because our initial experiments without this instruction revealed that the LLM often used the same age and gender, which could confound or bias the notes. Finally, we add that the notes may be long (around 5 lines or more), to avoid the LLM being too succinct.

Model details and parameters As an LLM, we opted for OpenAI's GPT-4o model, using the version released in May 2024 [OpenAI, 2024]. We set the temperature to 1.2 to encourage some more variation in the notes. Before providing the case-specific prompt, we set the following system message: "You are a general practitioner, and need to summarize the patient encounter in a clinical note. Your notes are detailed and extensive." We set the *max_tokens* parameter to 1000. All other parameters were set to their default values. Generating all 10,000 notes and their compact version cost around $150.

6.5.2.2 Additional example prompts

Fig. 6.6 and Fig. 6.7 show two additional example prompts.

6.5.2.3 Prompting strategy for special cases

Around one third of all patients in our dataset (3629 out of 10,000 patients sampled from the BN) do not experience any respiratory symptoms at all (all symptoms are "no"). If we used the same prompt as before, this would result in an unrealistic clinical note, since the note would simply list all symptoms the patient does not have, without giving an actual reason for the patient's visit. Furthermore, there would be little variation in these notes. An example of what would happen should we use the original prompt is shown in Fig. 6.8.

In these cases, it makes more sense to assume that the patient visits the doctor because of a complaint unrelated to the respiratory domain, such as back pain, stomach issues, a skin rash, etc. To generate these special cases, we use a special prompt, telling the LLM the patient does not experience any of the 5 respiratory symptoms. When the patient has at least one un-

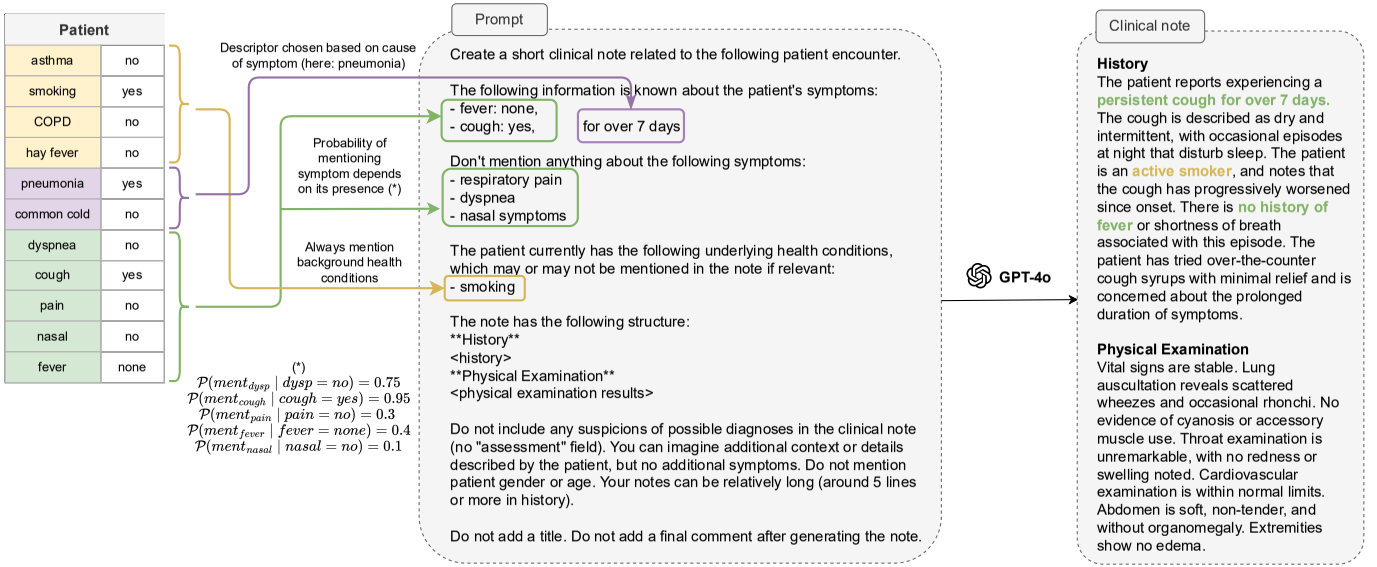


Figure 6.6: Additional example prompt with generated note.

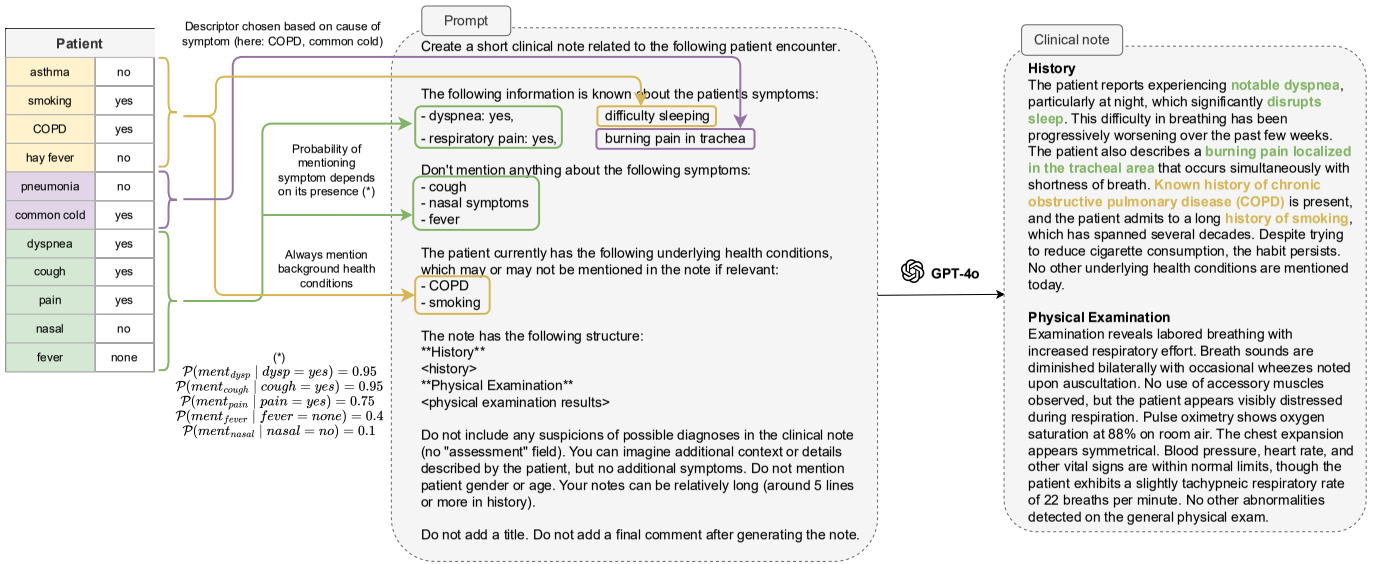


Figure 6.7: Additional example prompt with generated note. Since there are multiple causes for the symptom dyspnea (smoking, COPD) which are present in the patient, the descriptor "difficulty sleeping" was chosen randomly out of the bag of descriptors for smoking and COPD from Table 6.9. For respiratory pain, the descriptor "burning pain in the trachea" was chosen randomly out of the descriptors for common cold. While cough and COPD are also possible causes for respiratory pain, and are present in the patient, common cold overrules the two according to our strategy outlined in Section 6.2.2.

derlying health condition (which is the case in 239 out of 3629 special cases), we add this to the prompt in the same way as before, like the example in Fig. 6.9. If not (i.e., for the remaining 3390 out of 3629 special cases), we tell the LLM not to mention any of those health conditions either, as shown in Fig. 6.10. The latter prompt asks for three clinical notes at once, encouraging the LLM to be more creative and not repeat the same scenario every time, as well as being a little more cost-effective. This is possible because of the prompt being non-specific to any of these 3390 patients. We then randomly distribute all generated notes to each tabular patient record within this subset.

6.5.3 Automated span extraction

To extract spans mentioning each symptom from the normal clinical notes, we send the following instructions to the LLM in the system message:

```
I will show you a clinical note containing information on a patient's symptoms. For each
      symptom, I will tell you whether the patient suffers from this symptom or not.

Your task is to extract phrases from the note that mention these symptoms.
The annotation must have the following JSON structure:
[
   {
      "symptom": one of the symptoms ("dyspnea", "cough", "respiratory pain",
      "fever" or "nasal symptoms")
      "text": phrase in the text that mentions the symptom and whether it is
      present or absent
   }
   {
      "symptom": ...
      "text":...
   }
   ...
]

Keep the following instructions in mind:
- The same symptom may be mentioned multiple times. Include all phrases in which a
      symptom is mentioned. Consider both the "history" portion of the note, and the "
      physical examination" portion of the note.
- Also annotate a symptom if the note mentions that the patient does not suffer from it.
- The phrases do not need to be full sentences, but need to be verbatim as they appear
      in the note. You are not allowed to alter any words. If you leave out words, use
      ...
- Order does not matter.
- You will reply only with the JSON itself, and you will not wrap in JSON markers.
- You can only extract phrases from the "clinical note", not from any of the other text
      in the prompt.
- Not all symptoms are necessarily mentioned in the note. Do not include a symptom in
      the JSON if you cannot find any implicit or explicit mention of it in the clinical
       note.
```

We then ask the LLM to extract the spans for each note, using the prompt shown in Fig. 6.5a.

For the compact notes, we tell the LLM to match the spans extracted from the normal note to the text in the compact note. We send the following instructions to the LLM in the system message:

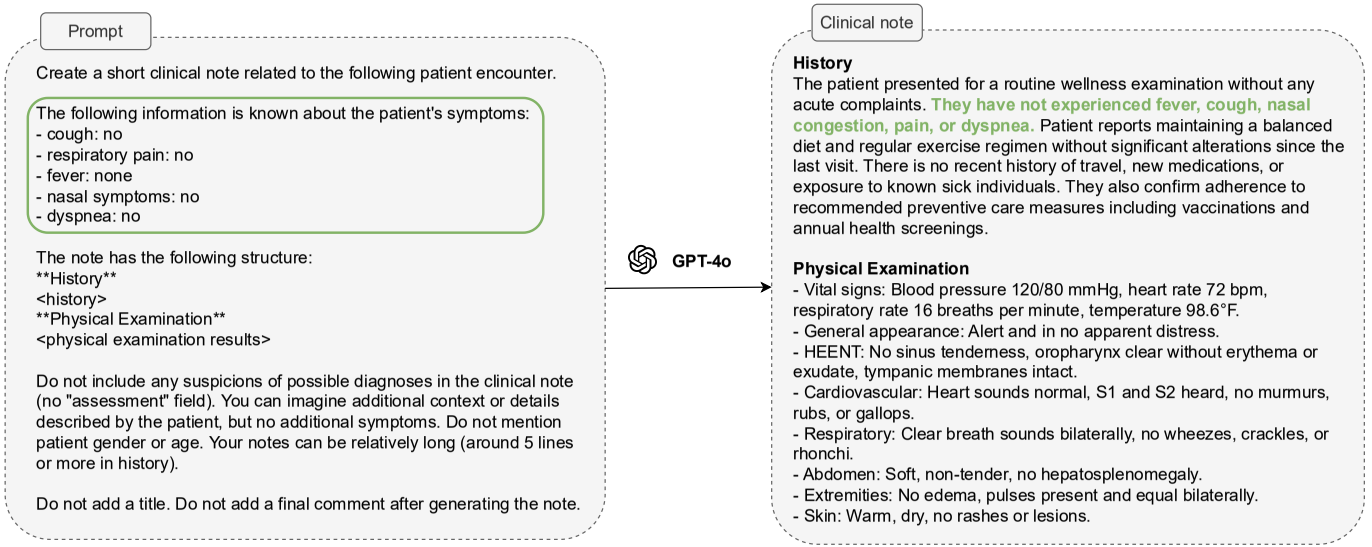


Figure 6.8: Example of what would happen if we simply extended the general prompt to the 3629 cases where the patient does not experience any respiratory symptoms. There would be little variation in these notes, as these patients seem to visit the doctor's office for no reason.

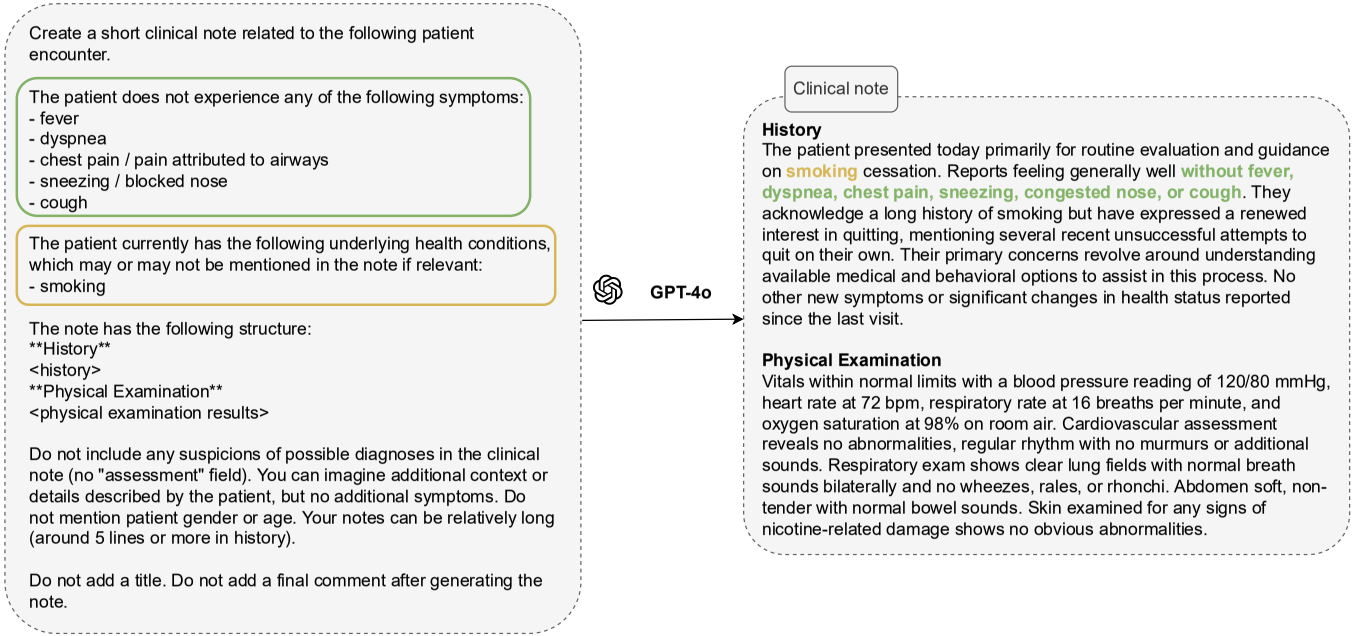


Figure 6.9: Example prompt and generated note for case where patient does not experience any respiratory symptoms, but does have an underlying respiratory health condition (here: smoking).

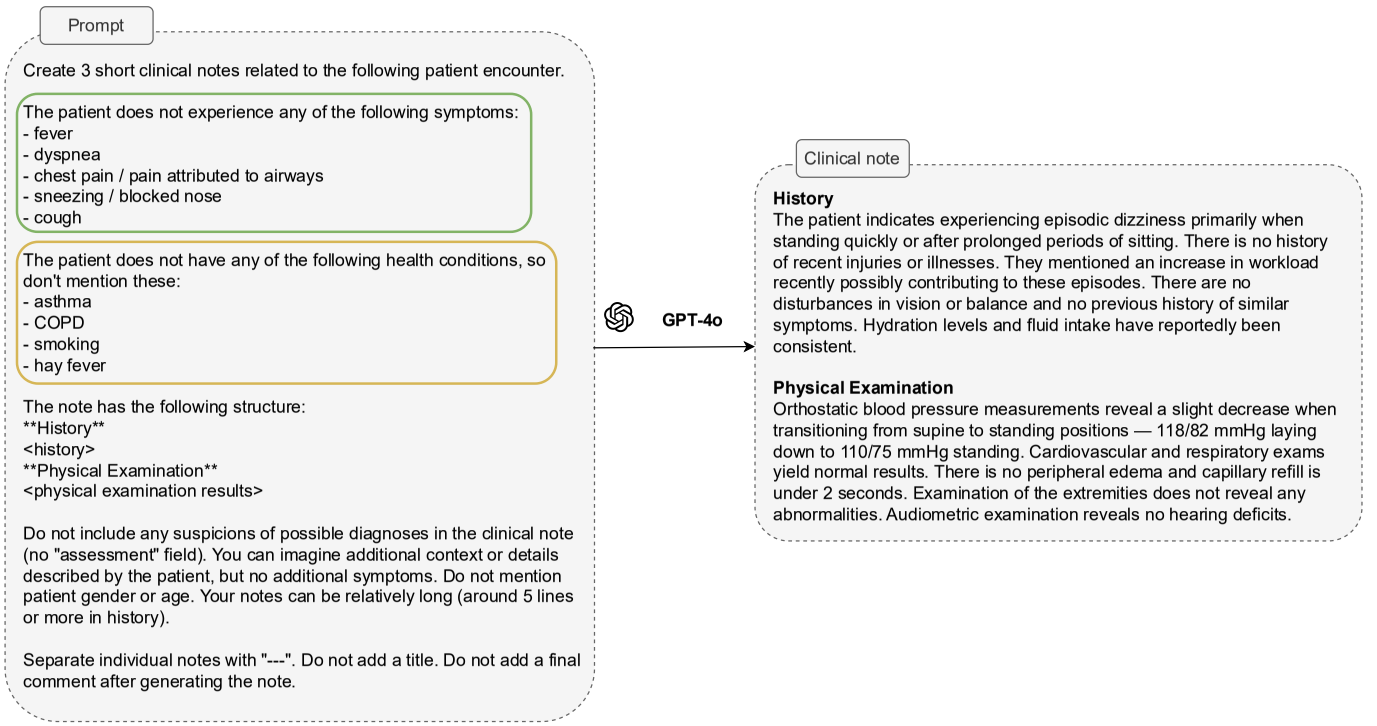


Figure 6.10: Prompt used for cases where patient does not experience any respiratory symptoms or have any underlying respiratory health condition. We show one of the three generated notes.

```
I will show you two versions of a clinical note. The first version describes a patient's
      visit to the doctor's office. The second one describes the same visit, but in a
     more compact style (using abbreviations and shortcuts), while preserving the
     overall message.

I will show you a set of phrases which were extracted from the first version of the note
     . Your task is to map these to phrases in the second version of the note.

You must reply with the following JSON format.
{
   phrase in version 1 : corresponding phrase extracted in version 2
}

Keep the following instructions in mind:
- Please extract phrases verbatim.
- Please use the empty string if you cannot find a phrase with the same meaning.
- The phrases you extract must have the same meaning, you cannot simply copy phrases
     that are in the same spot in the text.
- You will reply only with the JSON itself, and you will not wrap in JSON markers.
```

We then ask the LLM to match the phrases for each pair of notes, using the prompt shown in Fig. 6.5b. After the first response, we ask it to double-check if the extracted phrase has the same meaning. We included this step because it would otherwise select phrases that were in the same location in the text as the original phrase, even if they did not mean the same. We used the following prompt for this:

```
Please check if each extracted phrase has the same meaning as the original phrase. If
     not, substitute it by the empty string (''). The rest of the JSON must remain
     unchanged.
```

We post-process all phrases extracted from both the normal and compact notes by matching them to the text. If a phrase does not match the text exactly, we ask the LLM to correct itself. For this, we use the following prompt:

```
I will show you a clinical note, together with one or more phrases that were extracted
     from it. However, some mistakes were made in extracting these phrases. You must
     correct them.

---

The following is a clinical note:
< note >

The following phrases were extracted from this note. However, they do not exactly match
     the text:
- < phrase 1 >
- < phrase 2 >
- ...

Please correct the phrases so they map exactly to a phrase in the text.
You must reply with the following JSON format:
{
   original phrase: corrected phrase
}

You will reply only with the JSON itself, and you will not wrap in JSON markers.
```

For all these prompts, we use OpenAI's GPT-4o model. We set the temperature to 0.2 to encourage correctness over creativity. The full span

annotation process cost around $80. For a full walkthrough of the span extraction process, we refer to the tutorial notebook on our Github repository: https://github.com/prabaey/SimSUM/blob/main/src/span_annotations.ipynb.

6.5.4 Expert evaluation

We picked a random subset of 30 generated notes and show them to 5 general practitioners. All evaluators got to see the same 30 notes, together with the prompts that were used to generate them, in a random order that differed between evaluators. All evaluators received detailed instructions on what was expected of them in the form of a PDF[2], which was orally explained by the authors. The authors then met with each evaluator separately to complete three example notes (different from the 30 notes that were to be evaluated). Afterwards, the evaluators were asked to rate the notes in their own time, without the authors' involvement. Since it is infeasible to evaluate the whole dataset of 10,000 notes, we opted for a small subset of 30 notes, each going through a relatively extensive evaluation process that considered various measures of quality (evaluators took around $5 - 10$ minutes to evaluate each note). We decided to show all 5 evaluators the same set of 30 notes, to get a broader range of expert opinions in the evaluation of each note. This also allowed for the calculation of inter-annotator agreement.

We now provide further details on each dimension along which we evaluated the notes and the specific meaning assigned to each rating. The aspects of consistency, realism and clinical accuracy are only evaluated based on the normal note, while the quality of the compact note is evaluated using the last two dimensions (content and readability).

6.5.4.1 Evaluation dimensions

Consistency We subdivided the prompt into four different sections, and asked the evaluators to assign penalties for each section. A penalty was assigned if the requested information in that section was incorrectly mentioned in the note (e.g. a particular symptom was said to be present in the patient, when the prompt particularly requested the symptom to be absent), or if the requested information was absent from the note (e.g. a symptom descriptor is not mentioned in the text). The four parts of the prompt were as follows: (i) the symptoms to mention (can be present or absent), (ii) the symptoms not to mention, (iii) the underlying health conditions, and (iv) the additional instructions.

As explained in Section 6.2.2, around one third of the notes were generated using a second type of prompt, where the LLM is told that the patient

[2] https://github.com/prabaey/SimSUM/blob/main/eval/Instructions_clinical_evaluation.pdf

does not suffer from any respiratory symptoms or underlying health conditions. Following the prompt in Fig. 6.10, there are three parts of the prompt which can be violated (leading to penalties): (i) the respiratory symptoms, which the patient does not have, (ii) the underlying conditions, which the patient does not have, and (iii) the additional instructions. In our random set of 30 notes, 20 notes belonged to the first type, and 10 to the second type.

Once penalties were assigned, we summed them into a total number of penalties, and converted these into scores from 1 to 5. Notes with no penalties get a perfect score of 5, one penalty corresponds to 4, two penalties to 3, three penalties to 2 and more than three penalties to 1.

Realism The LLM is allowed to invent context and details in light of the information it receives in the prompt, but this must be realistic and relevant to the symptoms experienced by the patient. While some clinical facts might not seem technically incorrect, one might not expect to see them in the note, or it might be unlikely that they would be written down by a real physician. For example, if a patient has a runny nose and no other complaints, most clinicians would not check for abnormalities in lung capacity. Another example is asking whether the patient has recently traveled to an exotic destination because they have a cough.

We ask to score realism of the "history" section using the ratings below. The evaluators are specifically instructed to take into account the information mentioned in the prompt.

5 – All pieces of additional context and details (i.e. outside of the symptoms and background provided in the prompt) are realistic and seem like they belong in the note.

4 – There are one or two pieces of additional context or details that I would not have mentioned as a physician, or that do not seem relevant (even though they do seem like they belong).

3 – There are one or two pieces of additional context or details that do not seem like they belong in the note, or do not seem relevant, given the symptoms and background provided in the prompt.

2 – There are multiple pieces of additional context or details that do not seem like they belong in the note, or do not seem relevant, given the symptoms and background provided in the prompt.

1 – (Almost) all of the additional context is nonsensical given the symptoms and background provided in the prompt.

We ask to score realism of the "physical examination" section using the ratings below. The evaluators are specifically instructed to take into account the information mentioned in the "history" section of the note.

5 – All elements in the physical examination are things I would check, given the history and symptoms of the patient, and no important elements are missing.

4 – There are one or two elements in the physical examination that I probably would not have checked, given the history and symptoms of the patient, but I could see it happen. Some minor elements might be missing, but nothing major.
3 – There are one or two elements in the physical examination that I would not have checked, or some important elements are missing, given the history and symptoms of the patient.
2 – There are multiple elements in the physical examination that make no sense given the history and symptoms of the patient, or many important elements are missing.
1 – The physical examination portion of the note seems totally unrealistic.

Clinical accuracy While the previous section talks about evaluating the realism of the presence of all examinations described in the "physical examination" section, here we talk about evaluating the clinical accuracy of these findings. Clinical inaccuracies may depend on the context, like physical findings which are not congruent with the history and symptoms of the patient. For example, if the "history" portion mentions that the patient has no fever, then this should not be contradicted in the "physical examination" portion with a temperature of 39°C. Clinical inaccuracies may also stand alone. For example, a blood pressure reading of 20/10 mm Hg is impossible to encounter in any patient.

We ask the evaluators to score clinical accuracy of the findings that are mentioned in the "physical examination" portion of the note, using the ratings below.

5 – There are no mistakes, all reported clinical information is plausible in light of the patient's symptoms and history.
4 – There are one or two minor mistakes, or some details seem less plausible in light of the patient's symptoms and history, while the overall picture painted by the note is still correct.
3 – There are more than two minor mistakes, or multiple details which seem implausible in light of the patient's symptoms and history, but no major inaccuracies.
2 – There is a major mistake (on top of possibly some minor ones), or many details seem implausible given the patient's symptoms and history.
1 – There are multiple major mistakes and many details seem totally implausible given the patient's symptoms and history.

Quality of compact version While all the previous evaluations concerned the original note, here we evaluate the quality of the compact version of the note.

The **content** of the compact version should convey the same information as the original text, albeit in a shorter format. This is evaluated jointly for "history" and "physical examination" using the scoring system below.

5 – The compact version conveys the exact same information as the original text.
4 – The compact version conveys all key points of the original text, leaving out some details here and there.
3 – The compact version conveys some of the key points of the original text, but misses some as well.
2 – The compact version conveys some of the same information as the original text, but misses many key points.
1 – The compact version does not convey the same information as the original text, leaving out almost all key points.

While we purposefully want these notes to be harder to read and understand for both humans and machines, mimicking the complexity of some real clinical notes, the use of abbreviations should not be excessive. We evaluate **readability** jointly for "history" and "physical examination", using the scoring system below.

5 – The compact version seems understandable without seeing the original.
4 – The compact version seems mostly understandable without seeing the original, though there are some abbreviations that I would not immediately understand.
3 – Some parts of the compact version seem understandable without seeing the original, but other parts are not. There are some abbreviations that seem far-fetched or are used incorrectly (i.e. these are known to refer to other clinical terms than the way they are used in the text).
2 – Many parts of the compact version would not be understandable without seeing the original. Many abbreviations seem far-fetched or are used incorrectly (i.e. these are known to refer to other clinical terms than the way they are used in the text).
1 – The compact version is impossible to understand without seeing the original.

6.5.4.2 Results

Extended results for consistency We list the total number of penalties assigned by all evaluators to each of the prompt sections. For the 20 notes belonging to the first prompt type, 41 penalties were assigned in total for section (iv) of the prompt, which are the additional instructions. Sections (i), (ii), and (iii), which describe the symptoms and underlying health conditions of the patients, received no penalties at all. For the 10 notes belonging to the second prompt type, 1 penalty was assigned for part (i), 2 penalties for part (ii) and another 2 penalties for part (iii).

Inter-Rater Reliability Krippendorff's alpha is a statistical measure of inter-rater reliability that quantifies the extent to which observed agreement among raters exceeds what would be expected by chance. Usually, an alpha of 0.667 is cited as the desired level in order to draw tentative

conclusions [Krippendorff, 2018]. We calculate Krippendorff's alpha with ordinal distance measure between the 5 evaluators in our study, for each of the measured criteria:

- Consistency: 0.44
- Realism (history): 0.25
- Realism (physical examination): 0.32
- Clinical accuracy: 0.21
- Content of compact version: -0.02
- Readability of compact version: 0.36

With an alpha of 1 indicating perfect agreement, and 0 indicating agreement at the level of chance, we can say that there is some agreement between the 5 evaluators. While we do not reach the desired level of 0.667, in our case we must consider that due to the high scores assigned by all evaluators (mostly 4 and 5), a deviation from 5 can more easily be attributed to random chance rather than to a shared opinion of the evaluators. The lower standard deviations in Table 6.2 show that the evaluators consistently assign high scores to all aspects of the evaluation, even if they do not agree on which particular notes deviate from these higher scores.

6.5.5 Symptom predictor baselines

6.5.5.1 BN-tab

We learn a BN over the training data, providing the structure over all variables as in Fig. 6.3. For the variables *asthma*, *smoking*, *hay fever*, *COPD*, *season*, *pneumonia*, *common cold*, *fever* and *self-employed*, we learn the CPTs from the training data using maximum likelihood estimation (which comes down to counting co-occurrences of child and parent values for each entry in the CPT). We use the `pgmpy` library [Ankur Ankan and Abinash Panda, 2015] with a K2 prior as a smoothing strategy to initialize empty CPTs.

As support for learning Noisy-OR distributions is not provided in `pgmpy`, we learn these parameters with a custom training loop. We formulate the likelihood as in Equation 6.2 in Section 6.2.1, and learn the parameters p_i in Equations 6.3 through 6.6 for the variables *dyspnea*, *cough*, *pain* and *nasal* through maximum likelihood estimation by iterating over the train set for 10 epochs, using an Adam optimizer with a batch size of 50, a learning rate of 0.01 and random initialization of each parameter. To integrate the learned Noisy-OR distributions in the BN, we turn them into fully specified CPTs. To obtain these, we simply evaluate Equation 6.2 in Section 6.2.1 for all possible combinations of child and parent values. While this results in large and inefficient CPTs, the automated inference engine built into the `pgmpy` library does not support Noisy-OR distributions directly. Note that both versions of the conditional distribution are equivalent, so we do not incur a loss in precision.

Similarly, the coefficients in the logistic regression model for *antibiotics* and the Poisson regression model for *days at home* are learned using maximum likelihood estimation over the training set of 8,000 examples. The likelihood is expressed as in Equation 6.7, 6.8 and 6.9 in Section 6.2.1 respectively, with learnable parameters in place of each coefficient. We iterate over the train set for 15 epochs, again using an Adam optimizer with a batch size of 50, a learning rate of 0.01 and random initialization of each parameter. Finally, we turn the logistic regression and Poisson regression models into CPTs by evaluating Equation 6.7, 6.8 and 6.9 for all combinations of parent and child values. For the variable *days at home*, we needed to turn each discrete number of days into a category, because `pgmpy` only provides automated inference for BNs consisting of exclusively categorical variables. This results in a large CPT containing one row per possible number of days, which range from 0 to 15 in our training dataset, and one column for each combination of the 7 parent variables. To allow for a possible larger maximum number of days in the test set, we create a category $\geq$ *15 days*, which is defined as one minus the summed probability of all other days.

Once we have learned all parameters in the joint distribution, we can evaluate the BN's ability to predict each of the symptoms. For each evidence setting (as defined in the main text), we apply variable elimination with each of the symptoms as a target variable. Looking at the causal structure in Fig. 6.3, we note that the model never has to marginalize over the many rows in the learned *days at home* CPT, since it is never a target variable. This makes automated inference feasible in our case.

6.5.5.2 XGBoost-tab

We use the `xgboost` library in combination with `sklearn`. We train separate classifiers per symptom, one for each setting, which means we train 15 classifiers total. We tune the hyperparameters separately for each classifier, using 5-fold cross validation with F1 as a scoring metric (macro-F1 for *fever*).

The classifiers for the symptoms *dysp*, *cough*, *pain* and *nasal* use a binary logistic objective and logloss as an evaluation metric within the XGBoost training procedure, while the classifiers for the symptom *fever* use the multi-softmax objective with multiclass logloss as an evaluation metric. The *scale_pos_weight* parameter is set to the ratio of negative over positive samples for the binary classifiers. For the *fever* classifier, we address class imbalance by setting *class_weight = balanced*, which ensures that samples from less frequent classes (in our case low and high fever) receive higher weight in the loss calculation. We use grid search to find the best hyperparameter configuration, where the following sets of options are explored:

- *n_estimators*: $\{50, 100, 200\}$
- *max_depth*: $\{2, 3, 4, 5\}$
- *learning_rate*: $\{0.01, 0.1, 0.2\}$

- *subsample*: $\{0.8, 1\}$
- *colsample_ bytree*: $\{0.8, 1\}$
- *gamma*: $\{0, 0.1, 0.3\}$
- *min_ child_ weight*: $\{1, 5, 10\}$

6.5.5.3 Neural-text

The pretrained BioLORD encoder [Remy et al., 2024] was obtained through the `huggingface` library. The encoder outputs 768-dimensional sentence embeddings. Since the full text did not fit into the context window, we embedded each sentence separately, and then combined them using our strategies outlined in the main text. To split the text into sentences, we used the `nltk` package. The settings *hist*, *phys*, *mean* and *span* all result in a text embedding of 768 dimensions, while the setting *concat* results in a text embedding of 2*768 dimensions.

These embeddings are fed into a linear layer with 256 neurons, followed by a ReLU activation. The hidden state is then transformed into a single output neuron, followed by a Sigmoid activation. For the classifiers that predict *fever*, three output neurons followed by a Softmax activation are used instead, one for each class. While the embeddings remain fixed, we learn the parameters in the hidden and output layers using cross-entropy as a loss function over the training set. We train a separate classifier for each symptom, setting and difficulty of the text (normal vs. compact). For the binary symptoms, we train for 15 epochs using the Adam optimizer with a batch size of 100, a learning rate of 0.001 and *weight_ decay* set to $1e-5$. The classifier for $fever$ tended to collapse more easily, which is why we train it for 30 epochs with a lower learning rate of $5e-4$ instead. These hyperparameters were obtained using a mix of manual tuning and grid search with 5-fold cross validation over the training set.

Results for the different embedding types are reported numerically in Table 6.5 in Section 6.3.3, and shown visually in Fig. 6.11.

6.5.5.4 Neural-text-tab

For each evidence setting, we select the relevant set of tabular features and transform them into a numerical representation. We use a one-hot encoding for the categorical (binary or multiclass) features, and normalize the *days at home* feature using the StandardScaler from `sklearn`. This tabular feature representation is then concatenated with the text representation we obtained in the previous baseline. Both are fed into the same architecture described in Section 6.5.5.3, adapting the dimension of the input layer accordingly. For example, for the *dyspnea* classifier in the evidence setting *all*, the input dimension becomes $768+17$. We use the same hyperparameters as in the **neural-text** baseline to ensure a fair comparison. All other training details remain the same.

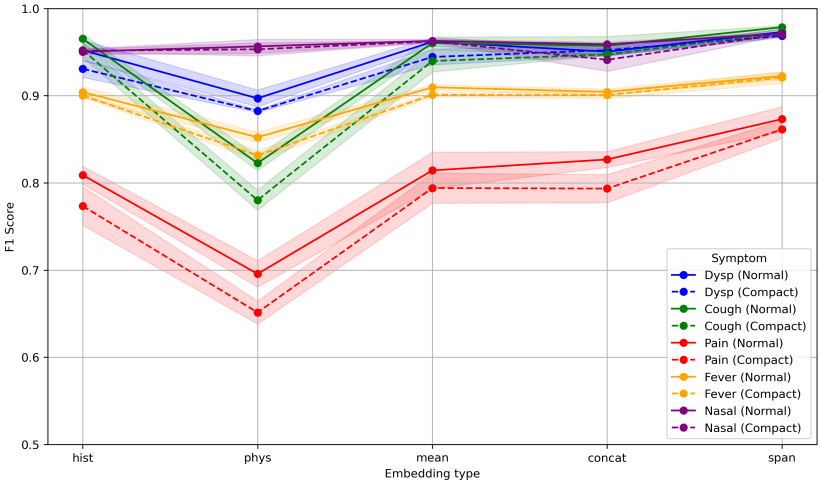


Figure 6.11: Results for the **neural-text** baseline for the five different embedding types. We show F1-scores over the test set for both the normal and compact versions of the notes, averaging over 5 seeds. Figure best viewed in color. For numerical results, see Table 6.5 in Section 6.3.3.

7

Patient-level Information Extraction by Consistent Integration of Textual and Tabular Evidence with Bayesian Networks

This chapter demonstrates how to perform patient-level information extraction by integrating predictions from both a Bayesian network and a neural text classifier on the SimSUM dataset. While conventional clinical information extraction methods focus solely on unstructured text, our approach additionally leverages the structured tabular features already present in a patient's electronic health record. Tabular data are modeled with a Bayesian network, and symptom information is extracted from text using a neural classifier built on pre-trained embeddings. To combine these predictions in an interpretable and probabilistic manner, we introduce the use of virtual evidence with a consistency node within the Bayesian network. This framework allows the Bayesian network to modulate the neural classifier's outputs, handle missing textual information, and resolve contradictions between textual and tabular evidence.

* * *

Paloma Rabaey[*], **Adrick Tench**[*], **Stefan Heytens, Thomas Demeester**

[*]Equal contribution.

Submitted to the Journal of Applied Intelligence

Abstract

Electronic Health Records (EHRs) form an invaluable resource for training clinical decision support systems. To leverage the potential of such systems in high-risk applications, we need large, structured tabular datasets on which we can build transparent feature-based models. While part of the EHR already contains structured information (e.g. diagnosis codes, medications, and lab results), much of the information is contained within unstructured text (e.g. discharge summaries and nursing notes). In this work, we propose a method for multi-modal patient-level information extraction that leverages both the tabular features available in the patient's EHR (using an expert-informed Bayesian network) as well as clinical notes describing the patient's symptoms (using neural text classifiers). We propose the use of *virtual evidence* augmented with a *consistency node* to provide an interpretable, probabilistic fusion of the models' predictions. The consistency node improves the calibration of the final predictions compared to virtual evidence alone, allowing the Bayesian network to better adjust the neural classifier's output to handle missing information and resolve contradictions between the tabular and text data. We show the potential of our method on the SimSUM dataset, a simulated benchmark linking tabular EHRs with clinical notes through expert knowledge.

7.1 Introduction

In healthcare, storing patient information in a structured format is essential for ensuring continuity of care. This information, such as diagnosis codes, medication, and lab results, is usually stored in EHRs. Often, these also contain a large quantity of unstructured free text, such as discharge summaries, nursing notes, procedural descriptions, and more [Ford et al., 2016]. All this information forms an invaluable resource for training clinical decision support systems, which have the potential to partially automate care processes such as diagnosis and treatment planning [Peiffer-Smadja et al., 2020a, Rasmy et al., 2021, Li et al., 2020]. While the recent rise in LLMs has shown great potential for processing clinical notes [Lehman and Johnson, 2023, Singhal et al., 2023, Labrak et al., 2024], these black-box systems lack interpretability [Quinn et al., 2022, Zhao et al., 2024, Tian et al., 2024]. In high-risk clinical applications, one should prefer robust and transparent systems built on simpler, feature-based models, like regression models, decision trees or Bayesian networks [Rudin, 2019, Sanchez et al., 2022, Lundberg et al., 2020]. However, these models require large structured tabular datasets as a training resource to ensure their generalization

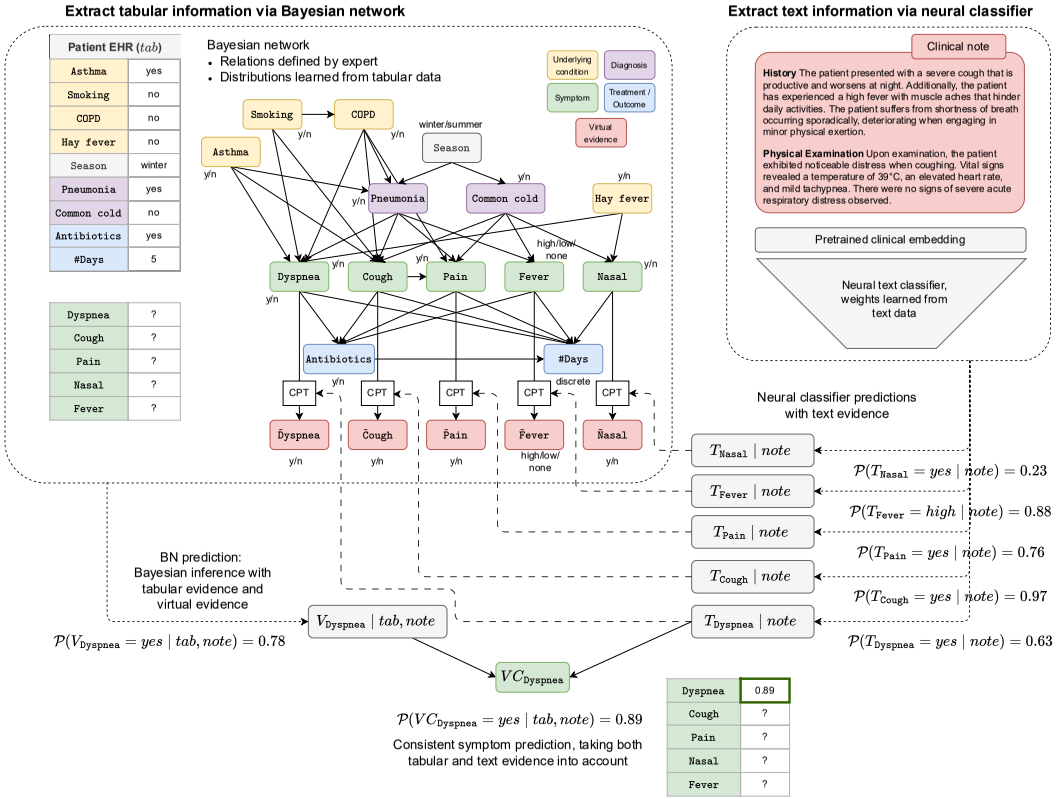


Figure 7.1: Overview of our patient-level information extraction method which integrates both tabular and text evidence. As an example, we show how to extract the probability that a patient suffers from `Dyspnea`, given tabular evidence that is already encoded in the EHR, a clinical note describing the patient's symptoms, and an expert-defined **BN** structure. On the right, the **neural classifiers** infer probabilities that the **text** mentions each symptom, with a 63% confidence for `Dyspnea` (in this case, "dyspnea" is not mentioned verbatim). The classifiers' probabilities for each symptom are provided as **virtual evidence** to the **BN**, via the red "virtual" nodes in the network. Given all **tabular** and **virtual evidence**, the **BN** infers that the patient has a 78% chance of `Dyspnea` – since the patient has both `Pneumonia` and `Asthma`, this probability is high. The consistency node $VC_{\texttt{Dyspnea}}$ combines these probabilities, arriving at an 89% chance that this patient has `Dyspnea`. Part of this figure is adapted from Rabaey et al. [2024a].

to broader patient populations. As such, developing methods to extend the tabular portion of the EHR with as much information as possible could benefit a wide range of downstream applications.

In this work, we propose a method for patient-level information extraction that leverages both the structured tabular features available in a patient's EHR and the unstructured clinical notes documenting physician encounters. We model the tabular portion of the EHR using a Bayesian Network (BN) with an expert-defined structure and learned probabilities, enabling interpretable integration of background information. This network is then connected to the predictions of a text classifier that interprets the clinical notes. By linking these two modalities, the BN can adjust the text classifier's predictions when they conflict with evidence from the tabular data, and infer missing information when the text is incomplete, all in a manner that remains transparent to the end user.

As a proof-of-concept for our method, we focus on a specific use-case built on the SimSUM dataset [Rabaey et al., 2024a]. SimSUM is a simulated benchmark of 10,000 artificial patient records, linking tabular variables (like symptoms, diagnoses and underlying conditions) to associated clinical notes describing the patient encounter in the domain of respiratory diseases. To our knowledge, SimSUM is the only available clinical dataset that provides both tabular and textual data connected through a shared and fully known data-generating process. This inherent alignment between modalities, together with the known BN structure underlying the data, enables fundamental research on integrating tabular and text-based models.

Figure 7.1 shows the overview of our setup and proposed method. On one side, we have a tabular EHR containing encoded background, diagnoses, and treatment. In addition, we have a clinical text describing the symptoms experienced by this patient. Since such symptom information can be valuable for downstream applications (e.g., automated diagnostic systems), our goal is to incorporate it into the EHR in a structured format. In a standard information extraction pipeline, a neural text classifier – however advanced – is typically used to predict whether a symptom is mentioned in the text. Our approach additionally exploits the tabular information already present in the EHR by connecting it to the target concepts for extraction (i.e., symptoms) through a BN. The relations in the BN (i.e. which underlying conditions may give rise to which symptoms) are provided by an expert, while the exact probabilities are learned from the tabular portion of the EHR.

Our main contribution lies in **enabling patient-level information extraction**, by **combining the predictions made by a BN and a text classifier in a probabilistic manner**, augmenting the established approach of virtual evidence with an additional *consistency node*. This combination enables our method to correct for inconsistencies and missing information in the text by leveraging tabular data and background knowledge through the BN. The consistency node improves the calibration of the final

predictions compared to using virtual evidence alone, especially in abnormal cases where information is missing from the text, while retaining high predictive power in common cases. Moreover, by obtaining a probabilistic rather than deterministic encoding of the symptoms within the tabular record, our approach yields extracted information that is more robust for downstream use. Consequently, users of the final tabular dataset can either apply a threshold to obtain hard labels for these concepts or directly utilize the probabilistic outputs, depending on their specific application and system requirements.

The remainder of our work is structured as follows. In Section 7.2 we discuss related work, including an in-depth comparison of our work with previous approaches for integrating BNs with neural text classifiers. Then, Section 7.3 explains the building blocks of our multimodal approach and how these are connected together using the consistency node. In Section 7.4, we report the performance of our method on the SimSUM dataset for various sample sizes, and take a closer look at how our multimodal approach improves over uni-modal baselines. Finally, Sections 7.5 and 7.6 summarize the potential of our method, while also addressing its limitations. The code for this project can be found at `https://github.com/AdrickTench/patient-level-IE`.

7.2 Related work

In Section 7.2.1, we first introduce related work on representation learning for EHRs, in particular focusing on multimodal approaches that integrate tabular data and text, as well as those that incorporate background knowledge. Section 7.2.2 then zooms in on two closely related methods that integrate text in BNs, contrasting them with our approach. Finally, Section 7.2.3 explains the concept of virtual evidence.

7.2.1 Multimodal EHR representation learning

Modeling tabular data and text Many studies focus on combining two modalities commonly found in EHRs – structured tabular data (e.g. disease codes, patient demographics, treatment codes, lab results) and unstructured text (e.g. discharge summaries or nursing notes) – for representation learning. Most methods use state-of-the-art encoders to learn a representation for each modality, combining both through simple concatenation [Zhang et al., 2020], ensemble methods [Xu et al., 2019] or an attention mechanism [Liu et al., 2022a, Nguyen et al., 2024, Wang et al., 2023b]. While our work also focuses on combining these two modalities, our goal is not to learn a black-box patient-level representation for downstream prediction tasks. Instead, we aim to enrich the structured portion of the EHR with information extracted from the text, thereby facilitating its use in interpretable

downstream prediction models such as regression models, decision trees, or BNs [Rudin, 2019]. Furthermore, by providing a probabilistic view on the extracted structured variables, our approach enables intermediate manual inspection of the dataset before it is used in downstream models, which is not possible with black-box patient representations.

Integrating background knowledge Particularly interesting to our work are representation learning methods that include some form of background knowledge to improve the learned multimodal representations. Nguyen et al. [2024] enhance their multimodal encoding for the structured and text portions of the EHR with a clinical reasoning embedding, which is obtained by retrieving relevant clinical documents and asking an LLM to reason over them. Xu et al. [2024c] use a similar information retrieval module to enhance disease code embeddings.

Instead of incorporating relevant information from clinical corpora, we focus on incorporating graph-based background knowledge. In this line of work, both Choi et al. [2018] and Park et al. [2022] model the structured portion of the EHR with a graph attention network, where each admission is represented as a hierarchical graph containing procedure, diagnoses and prescription nodes. As an extension, Choi et al. [2020] learn to automatically construct the causal structure of a patient admission (in particular, modeling which diagnoses lead to the prescription of which treatment), by training a graph convolutional transformer jointly with supervised medical prediction tasks. This approach is conceptually similar to the BN component of our method. However, in our case, we (i) learn a BN rather than a graph convolutional transformer, and (ii) rely on domain experts to specify the medical relationships symbolically, learning only the underlying probability distributions from the data rather than the relations among variables.

Other works incorporate graph-based background knowledge into EHR representation learning by leveraging medical ontologies such as ICD-10, SNOMED-CT, or UMLS, which encode hierarchical clinical concepts and relationships among medical entities [Choi et al., 2017]. Most approaches employ graph neural networks to model the relationships between medical codes, integrating these representations into the structured EHR embedding at various stages of the representation learning pipeline [Liu et al., 2022a, Ma et al., 2018, Ye et al., 2021]. In contrast, we do not rely on knowledge graphs to model relationships between medical variables. Instead, we represent background knowledge through a BN, which not only captures clinical dependencies among variables (as knowledge graphs do) but also enables probabilistic reasoning and prediction.

7.2.2 Integrating Bayesian networks and text

Bayesian networks A BN has the ability to model complex problems involving uncertainty, while combining data with expert knowledge in an

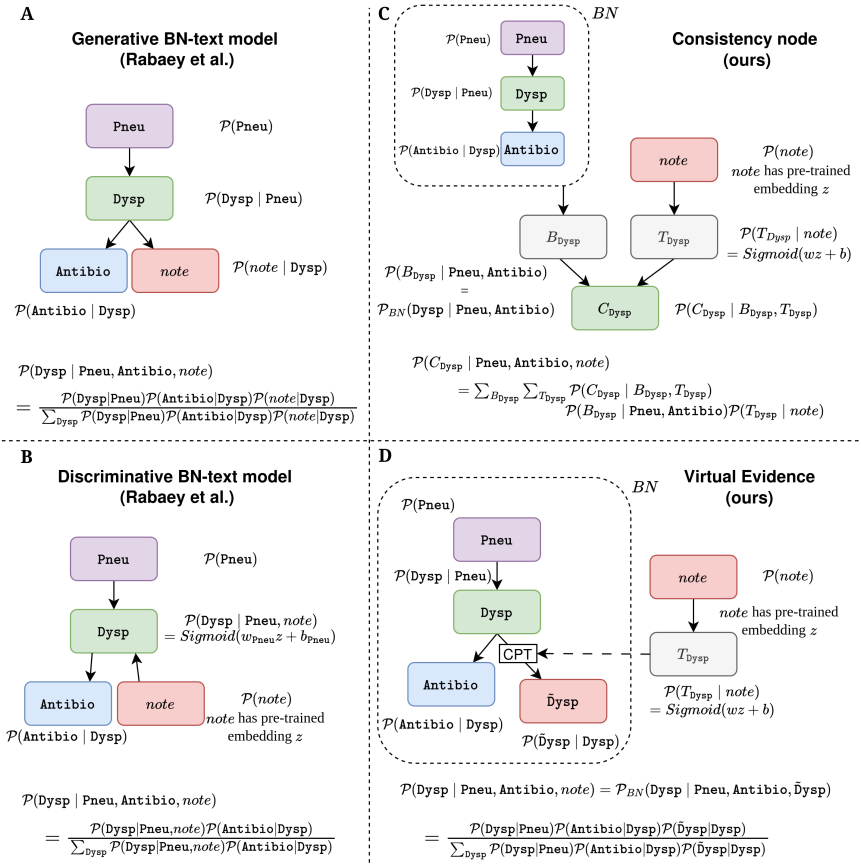


Figure 7.2: Comparison of our work with Rabaey et al. [2024b]. In this running example, we aim to predict the probability $\mathcal{P}(\texttt{Dysp} \mid \texttt{Pneu}, note)$ that a patient suffers from the symptom dyspnea (`Dysp`), given both tabular information (whether the patient has pneumonia, or `Pneu`, and was prescribed antibiotics, or `Antibio`) and a clinical note (*note*). In the generative and discriminative BN-text models proposed by Rabaey et al. [2024b] (**A** and **B**), a *note* node is directly integrated in the BN, allowing one to perform Bayesian inference with mixed textual and tabular evidence. In our method, we split off the BN – where Bayesian inference is performed only over tabular evidence – and the neural text classifier, integrating their predictions through the consistency node $C_{\texttt{Dysp}}$ (**C**) and virtual evidence (**D**). Our method improves on the poor performance of **A** and poor scalability, interpretability, and causal structure of **B**.

interpretable graphical structure [Kyrimi et al., 2021f]. It is made up of two parts: a Directed Acyclic Graph (DAG) modeling the relations between the

variables, and a joint probability distribution factorizing into a set of conditional distributions, one for each variable. BNs have proven useful to model a wide range of medical conditions in clinical research settings [McLachlan et al., 2020a, Arora et al., 2019a], including respiratory diseases such as pneumonia and Covid-19 [Edye et al., 2021]. While one could automatically learn the structure of the BN from data [Choi et al., 2020, Mani and Cooper, 2000], a particular asset of the BN is the possibility to integrate expert knowledge in the prediction process. In our case, we indeed assume that the relations between the clinical variables have been provided by an expert, while we learn the conditional probabilities from the data. Despite their potential, the clinical adoption of BNs remains limited, largely because they struggle to handle realistic medical data, where unstructured text is abundant [Kyrimi et al., 2021b]. Consequently, there is significant potential in developing methods to effectively integrate textual information into BNs. Despite this, there has been limited prior work on integrating text data into BNs. We now zoom in on two highly relevant contributions.

DeepProbLog [Manhaeve et al., 2018b] One contribution comes from the field of Neuro-Symbolic AI, in the form of the DeepProbLog framework [Manhaeve et al., 2018b]. Here, a probabilistic logic program is extended with neural predicates, whereby a neural network is used to learn a representation of an unstructured concept (in our case, clinical text), which is further treated as a regular predicate in the program. Since BNs are a symbolic method, they can be naturally formulated as probabilistic logic programs. However, one limitation of DeepProbLog is that neural predicates can only serve as root nodes within the BN, from which probabilities propagate downward to fully symbolic, categorical variables. In other words, these neural predicates cannot have any symbolic parent variables. In our case, the tabular nodes we aim to predict – namely, the symptoms – occupy arbitrary positions within the BN and typically have multiple parent variables.

BN-text model [Rabaey et al., 2024b] To tackle this limitation, Rabaey et al. [2024b] propose two approaches – a *generative* and a *discriminative* BN-text model – which integrate text directly into a clinical BN, allowing the text to be part of the evidence in the Bayesian inference procedure. Figure 7.2 compares both approaches to our method, making use of a simple example. In this small BN, the disease pneumonia (`Pneu`) gives rise to the symptom dyspnea (`Dysp`), with a clinician deciding whether or not to prescribe antibiotics (`Antibio`) based on the presence of this symptom in the patient. A clinical *note* may describe information about the symptom dyspnea. The first approach proposed by Rabaey et al. [2024b], called the generative BN-text model (Figure 7.2A), directly includes a text node in the BN and models its conditional distribution $\mathcal{P}(note \mid \texttt{Dysp})$ by learning the

parameters of a multivariate Gaussian over pre-trained text embeddings. This parametric assumption proved too stringent to work in practice, leading to inferior performance of this method. Furthermore, the need to learn a separate Gaussian for each possible configuration of parent variables does not scale well to larger networks. For example, in the SimSUM dataset, the text node would be influenced by four underlying conditions, two diagnoses, and five symptoms, resulting in eleven parent nodes, with over 3000 ($2^{10} \times 3$) unique configurations.

Beyond this, Rabaey et al. [2024b] propose a second approach, called the discriminative BN-text model (Figure 7.2B). This method deviates from the causal structure of the problem – the symptom dyspnea influencing the content of the text – and instead uses a neural classifier to model $\mathcal{P}(\texttt{Dysp} \mid \texttt{Pneu}, \mathit{note})$. In other words, it extends the DeepProbLog framework [Manhaeve et al., 2018b] by learning multiple neural predicates, conditional on the possible values of the parent variable `Pneu`. However, this approach comes with two disadvantages. First, the effect of the diagnosis `Pneu` is implicitly encoded in the text classifiers, rather than explicitly injected through the conditional distributions in the BN. Second, learning a text classifier for every combination of parent values per symptom does not scale well when symptoms have many parents, as is the case in realistically sized datasets like SimSUM. Here, up to 90 text classifiers would need to be trained in total (8 for pneumonia, 2 for common cold, 32 for dyspnea, 32 for cough, 8 for pain, 4 for fever, and 4 for nasal).

As illustrated in Figure 7.2, our solution addresses these limitations by assuming a less stringent connection between the BN modeling the structured tabular variables on the one hand, and the neural text classifier modeling the clinical notes on the other. With *virtual evidence* (Figure 7.2D, Sections 7.2.3 and 7.3.4), a child node is introduced to the BN with a Conditional Probability Table (CPT) determined by the neural classifier. With the *consistency node* (Figure 7.2C, Section 7.3.5), the individual predictions of the BN and neural classifier are integrated by a separate module trained to assign appropriate weights to their contributions. Our final method incorporates both *virtual evidence* and a *consistency node* to combine the predictions of the BN with the neural classifiers (see Figure 7.1 for an overview of the combined model).

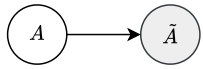
7.2.3 Virtual evidence

Virtual evidence is an established method to incorporate the predictions of neural networks with a BN [Feng and Williams, 1998, Morgan and Bourlard, 1995], and can be understood as evidence with uncertainty [Mrad et al., 2012]. Pearl [1988a] first introduced virtual evidence as a convenient way to incorporate uncertain evidence into a BN. Pearl's method treats virtual evidence as likelihood information. As opposed to the standard hard presence or absence of a piece of evidence in a regular BN, a piece of virtual evidence

is represented by a real number in $[0, 1]$ that indicates the confidence in observing a particular value for a variable V.[1]

To incorporate virtual evidence into a BN, a child node is added to the variable for which virtual evidence is provided. The child node itself is "observed" – i.e., provided as standard, hard evidence – while its CPT is determined by the uncertain evidence. For example, suppose a binary variable A is observed with a probability of 0.8. To include this uncertain observation as virtual evidence, a child node $\tilde{A}$ is added to the BN with a CPT encoding $P(\tilde{A}|A)$:[2]

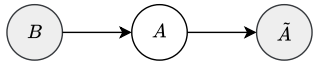
	A	$\neg A$
$\tilde{A}$	0.8	0.2



This method of incorporating uncertain evidence retains the prior probability distribution of the BN. It is possible to provide virtual evidence for multiple variables, or even multiple pieces of virtual evidence for the same variable [Pan et al., 2006]. Furthermore, it is still sensible to query the BN for the variable(s) for which virtual evidence is provided.

To make this concrete, consider a slight extension of our example above. Now A has a single parent B, with the following CPT:[3]

	B	$\neg B$
A	0.3	0.7



We can then use standard variable elimination [Koller and Friedman, 2009] to calculate $P(A|B, \tilde{A}) \approx 0.6316$ (Equation 7.1), where both B and virtual evidence $\tilde{A}$ are provided as evidence to the BN. Note that the final probability is lower than the uncertain evidence of 0.8 provided through $\tilde{A}$, because it factors in the prior probability $P(A|B)$.

$$P(A \mid B, \tilde{A}) = \frac{P(\tilde{A} \mid A)\, P(A \mid B)}{\sum_{a'} P(\tilde{A} \mid a')\, P(a' \mid B)} = \frac{0.8 \cdot 0.3}{(0.8 \cdot 0.3) + (0.2 \cdot 0.7)} \approx 0.6316 \tag{7.1}$$

As shown in Figure 7.1, our method uses virtual evidence but extends it with a *consistency node* (Section 7.3.5), further improving the calibration of the full model.

[1] Note that virtual evidence is sometimes referred to as soft evidence. We avoid using the term soft evidence here, as it also refers to a different type of uncertain evidence [Mrad et al., 2012].

[2] Here we use the convention of shading in the observed nodes in the BN.

[3] See footnote 2.

7.3 Methods

Following the introduction of the dataset and our experimental setup in Section 7.3.1, we present the fundamental components of our model: a Bayesian network (Section 7.3.2) and a neural network (Section 7.3.3). The core contribution of this work is the investigation of two strategies for integrating these models' predictions: virtual evidence (Section 7.3.4) and a consistency node (Section 7.3.5).

7.3.1 Preliminaries

Dataset The SimSUM dataset links artificial tabular patient records with artificial clinical notes describing a patient's visit to the general practitioner's office. By design, the clinical concepts expressed in the text (the symptoms experienced by the patient) and the tabular background information are connected through a BN representing domain knowledge. As shown in Figure 7.1, the BN relates two respiratory diseases (`Pneumonia` and `Common cold`) with their associated symptoms (`Dyspnea`, `Cough`, `Pain`, `Nasal` symptoms and `Fever`), as well as including underlying respiratory conditions (`Asthma`, `Smoking`, `COPD` and `Hay fever`), some background (`Season` of the year), and a treatment and outcome variable (whether `Antibiotics` were prescribed and the `#Days` the patient ended up staying at home as a result of their symptoms).[4] All variables are binary, except for `Fever`, which has three levels (none, low and high) and `#Days`, which is discrete and bounded $(0, \ldots, 15)$.

The SimSUM notes are generated in such a way that they describe the five symptoms and occasionally mention underlying respiratory conditions, but they do not explicitly mention the diagnoses. In this way, both modalities (tabular features and clinical notes) contain complementary information. Every patient has one unique note, with two versions: "normal" and "compact". We will report results on the normal notes.

Setup As shown in Equation 7.2, we have a set $\mathcal{X}$ of n patient records where each patient $X^{(i)}$ is described by their tabular record $tab^{(i)}$, their clinical note $note^{(i)}$ and their set of symptoms $s^{(i)}$.

[4] In the original SimSUM dataset, two other variables are included (`policy` and `self-employed`). We leave these out of our setup as they are non-clinical variables that would not be encoded in the patient record in a realistic setting.

$$
\begin{aligned}
\mathcal{X} =& \{X^{(0)}, ..., X^{(n-1)} \mid X^{(i)} = \{tab^{(i)}, note^{(i)}, s^{(i)}\}\} \\
tab^{(i)} =& \{\texttt{Asthma} = t_0^{(i)}, \texttt{Smoking} = t_1^{(i)}, \texttt{COPD} = t_2^{(i)}, \texttt{Hay fever} = t_3^{(i)}, \texttt{Season} = t_4^{(i)}, \\
& \texttt{Pneumonia} = t_5^{(i)}, \texttt{Common cold} = t_6^{(i)}, \texttt{Antibiotics} = t_7^{(i)}, \texttt{\#Days} = t_8^{(i)}\} \\
s^{(i)} =& \{\texttt{Dyspnea} = s_0^{(i)}, \texttt{Cough} = s_1^{(i)}, \texttt{Pain} = s_2^{(i)}, \texttt{Nasal} = s_3^{(i)}, \texttt{Fever} = s_4^{(i)}\} \\
=& \{s_j = s_j^{(i)} \mid j = 0, \ldots, 4\}
\end{aligned} \tag{7.2}
$$

Here, $s_j^{(i)} \in \mathcal{V}_{s_j}$, where $\mathcal{V}_{s_j} = \{yes, no\}$ for $j = 0 \ldots 3$ (symptoms `Dyspnea`, `Cough`, `Pain` and `Nasal`), and $\mathcal{V}_{s_j} = \{high, low, none\}$ for $j = 4$ (symptom `Fever`).

While the tabular values $t_0^{(i)}, \ldots, t_8^{(i)}$ and the text $note^{(i)}$ are available in the patient record at all times, the symptom values $s_0^{(i)}, \ldots, s_4^{(i)}$ are only available during training. Out of the full set of patient records $\mathcal{X}$, we select a training set $\mathcal{X}_{train}$, and a test set $\mathcal{X}_{test}$. At test time, we aim to predict the probability for each of the symptoms given the tabular data and text note, i.e., the distributions $\mathcal{P}(s_j \mid tab^{(i)}, note^{(i)})$ for $X^{(i)} \in \mathcal{X}_{test}$ and $j = 0 \ldots 4$.

Note that the distinction between the symptoms $s^{(i)}$ and tabular data $tab^{(i)}$ serves merely to denote $s^{(i)}$ as the targets for information extraction. While targeting $s^{(i)}$ is a natural choice given the data-generating process behind the SimSUM dataset, our method does not depend on any specific choice of target variables for extraction.

7.3.2 Bayesian network

We model the tabular portion of the data with a BN, where the relations between the variables (the DAG, as shown in Figure 7.1) are provided upfront by an expert. This DAG naturally prescribes how the joint probability distribution factorizes into conditional probability distributions (CPD), one for each variable conditional on its parents. As in Rabaey et al. [2024a], each CPD is learned independently by fitting it to the tabular portion of the training data $\mathcal{X}_{train}$. We refer to Appendix 7.6.1.1 for more details on this training procedure.

When the full joint distribution defined by the BN has been learned, we can perform Bayesian inference to obtain $\mathcal{P}(s_j \mid tab^{(i)})$ for each patient $i \in \mathcal{X}_{test}$ and each symptom j, by filling in the tabular evidence available for this patient and performing variable elimination [Koller and Friedman, 2009]. For example, for the patient in Figure 7.1, we would calculate $\mathcal{P}(\texttt{Dyspnea} = \text{yes} \mid \texttt{Asthma} = \text{yes}, \ldots, \texttt{\#Days} = 5)$ by summing out the other symptoms `Cough`, `Pain`, `Nasal`, and `Fever` (which are unavailable at test time) from the joint distribution and normalizing. From now on, we will denote the BN's prediction for the symptom s_j as $\mathcal{P}(B_{s_j} \mid tab^{(i)})$.

7.3.3 Neural network

To model the text portion of the data, we use a lightweight neural text classifier. For simplicity, we use the same encoder and architecture as SimSUM ([Rabaey et al., 2024a]), described below.[5] We train separate classifiers for each symptom. At the input, the clinical note is first split into sentences. Each sentence is transformed into an embedding using the pretrained clinical representation model BioLORD-2023 [Remy et al., 2024], after which all sentence embeddings are averaged to obtain a single representation for the full note. This note embedding is then fed into a multi-layer perceptron with one hidden layer, of which the weights are trained using the cross-entropy objective over the symptom labels in $\mathcal{X}_{train}$. We finally obtain class probabilities by applying a Sigmoid activation (or Softmax for `Fever`, which has three classes) to the output layer. At test-time, each trained classifier can provide predictions $\mathcal{P}(s_j \mid note^{(i)})$ for symptom j, for each patient $i \in \mathcal{X}_{test}$. From now on, we will denote the text classifier's prediction for the symptom s_j as $\mathcal{P}(T_{s_j} \mid note^{(i)})$.

7.3.4 Virtual evidence

To combine the probabilities of the BN and neural classifiers, we can provide the latter's predictions to the BN as virtual evidence. That is, for a patient $i \in \mathcal{X}$, the predictions of the neural classifiers for each symptom $\mathcal{P}(T_{s_j} \mid note^{(i)})$ are provided to the BN as virtual evidence, as outlined in Section 7.2.3. We refer to the virtual evidence for a symptom `s` with $\tilde{\texttt{s}}$. For example, $\tilde{\texttt{Dyspnea}}$ refers to the virtual evidence for the presence of `Dyspnea`. The combined tabular and text prediction for each symptom can be noted as follows, where both the tabular evidence $tab^{(i)}$ and the virtual evidence for all symptoms are provided:

$$\mathcal{P}(s_j \mid tab^{(i)}, \tilde{\texttt{Dyspnea}} = 1, \tilde{\texttt{Cough}} = 1,\ \tilde{\texttt{Pain}} = 1, \tilde{\texttt{Nasal}} = 1, \tilde{\texttt{Fever}} = 1)$$

This probability can be obtained through variable elimination, again by summing out the other (non-virtual) symptom nodes, apart from target symptom s_j. From now on, we will denote the prediction of the BN with virtual evidence for the symptom s_j as $\mathcal{P}(V_{s_j} \mid tab^{(i)}, note^{(i)})$.

7.3.5 Consistency node

7.3.5.1 Formulation

After training both the BN and the neural text classifier over $\mathcal{X}_{train}$, we can obtain $\mathcal{P}(B_{s_j} \mid tab^{(i)})$ (prediction based on tabular evidence) and $\mathcal{P}(T_{s_j} \mid note^{(i)})$ (prediction based on text evidence), for any patient $i \in \mathcal{X}_{test}$ and any symptom j. From now on, we will leave out the patient index i to

[5] Note that we only intend to provide an example of a lightweight text classifier for comparison and do not require the best possible model.

avoid cluttering the notation. We now combine both probabilities through a consistency node C_{s_j}, as illustrated in the overview in Figure 7.1. The nodes B_{s_j}, T_{s_j} and C_{s_j} form a probabilistic graphical model, with joint distribution as shown in Equation 7.3.

$$\mathcal{P}(C_{s_j}, B_{s_j}, T_{s_j} \mid tab, note) = \mathcal{P}(B_{s_j} \mid tab)\, \mathcal{P}(T_{s_j} \mid note)\, \mathcal{P}(C_{s_j} \mid B_{s_j}, T_{s_j}) \quad (7.3)$$

To obtain $\mathcal{P}(C_{s_j} \mid tab, note)$, we simply need to marginalize out the BN and text classifier predictions from the joint distribution, as in Equation 7.4. We do this by summing over all possible values of the symptom s_j:

$$\mathcal{P}(C_{s_j} \mid tab, note) = \sum_{b'} \sum_{t'} \left[\mathcal{P}(B_{s_j} = b' \mid tab)\mathcal{P}(T_{s_j} = t' \mid note)\mathcal{P}(C_{s_j} \mid b', t')\right], \quad \text{with } b', t' \in V_{s_j} \quad (7.4)$$

The conditional distribution $\mathcal{P}(C_{s_j} \mid B_{s_j}, T_{s_j})$ can be computed over the training set $\mathcal{X}_{train}$.[6] We first use the BN and neural classifier to obtain the probabilities $\mathcal{P}(B_{s_j} = b' \mid tab^{(k)})$ and $\mathcal{P}(T_{s_j} = t' \mid note^{(k)})$ for each patient $k \in \mathcal{X}_{train}$, each symptom j and each label $b', t' \in V_{s_j}$. As shown in Equations 7.5 and 7.6, we then calculate the agreement of the predicted probabilities with respect to the ground truth label c' as observed in $\mathcal{X}_{train}$:

$$\mathcal{P}(C_{s_j} = c' \mid b', t') = \frac{W\{c', b', t'\}}{\sum_{c''} W\{c'', b', t'\}}, \text{ with } c', c'' \in V_{s_j} \quad (7.5)$$

where the weights are obtained as:

$$W\{c', b', t'\} = \sum_{k \in \mathcal{X}_{train}} \mathcal{P}(B_{s_j} = b' \mid tab^{(k)}) \cdot \mathcal{P}(T_{s_j} = t' \mid note^{(k)}) \cdot \mathbb{1}\left[s_j^{(k)} = c'\right] \quad (7.6)$$

Intuitively, this process calculates the agreement between the BN and the text classifier on the training set $\mathcal{X}_{train}$. When they disagree (meaning $b' \neq t'$), one of the two models will be right more often. For example, if the text classifier agrees with the ground truth label in the training set more often, this means $\mathcal{P}(c' = t' \mid b', t') > \mathcal{P}(c' = b' \mid b', t')$. The terms containing the factor $\mathcal{P}(T_{s_j} = t' \mid note)$ will then receive a higher weight in Equation 7.4, pushing the prediction towards the label t'.

7.3.5.2 Numerical example

To make this more concrete, we provide a simple numerical example. We provide ground truth labels and probabilities from a text classifier and BN for four fictional patients in Table 7.1.

[6] To prevent overfitting on the training set, which would taint the conditional distribution, we train the neural networks using 5-fold cross-validation as described in Appendix 7.6.1.2.

Table 7.1: Initial symptom probabilities predicted for four samples by the text classifier ($\mathcal{P}(T \mid note)$) and the BN ($\mathcal{P}(BN \mid tab)$). GT label indicates the ground truth label for the symptom in the training set.

	GT label	$\mathcal{P}(T \mid note)$	$\mathcal{P}(BN \mid tab)$
1	`yes`	0.1	0.7
2	`no`	0.1	0.2
3	`no`	0.1	0.6
4	`yes`	0.9	0.6

Table 7.2: Consistency node probabilities $\mathcal{P}(C \mid T, BN)$ obtained from the predictions in Table 7.1.

T	BN	$\mathcal{P}(C = \texttt{no} \mid T, BN)$	$\mathcal{P}(C = \texttt{yes} \mid T, BN)$
`no`	`no`	0.78	0.22
`no`	`yes`	0.51	0.49
`yes`	`no`	0.24	0.76
`yes`	`yes`	0.12	0.88

Table 7.2 shows the agreement between the text classifier and BN across these 4 patients. This agreement is calculated through Equation 7.5. For example, the first row of Table 7.2 is calculated using the probabilities provided by the text classifier and BN that the symptom is `no` in both cases of the ground truth label. Concretely, the probabilities from the cases where the ground truth is `no` are $(1-0.1) \times (1-0.2) + (1-0.1) \times (1-0.6)$ (corresponding to patient 2 and 3, yielding a weight of $W\{S = \texttt{no}, BN = \texttt{no}, T = \texttt{no}\} = 1.08$). The probabilities where the ground truth is `yes` are $(1-0.1) \times (1-0.7) + (1-0.9) \times (1-0.6)$ (corresponding to patient 1 and 4, yielding a weight of $W\{S = \texttt{yes}, BN = \texttt{no}, T = \texttt{no}\} = 0.31$). These weights are then normalized by dividing by the sum for the row ($W\{S = \texttt{no}, BN = \texttt{no}, T = \texttt{no}\} + W\{S = \texttt{yes}, BN = \texttt{no}, T = \texttt{no}\} = 1.39$), yielding the probabilities in the first row of Table 7.2.

Given the consistency node probabilities in Table 7.2, the final probability for the presence of a symptom, $\mathcal{P}(C = \texttt{yes} \mid tab, note)$, can be calculated using the probabilities in the `yes` column, following Equation 7.4. For example, given probabilities of 0.1 from the text classifier and 0.8 from the BN, the final probability is calculated as follows:

$$\begin{aligned}\mathcal{P}(C = \texttt{yes} \mid tab, note) = &((1-0.1) \times (1-0.8) \times 0.22) + ((1-0.1) \times 0.8 \times 0.49) \\ &+ (0.1 \times (1-0.8) \times 0.76) + (0.1 \times 0.8 \times 0.88) \approx 0.48\end{aligned}$$

7.3.5.3 Incorporating virtual evidence

From Section 7.3.4 we can obtain $\mathcal{P}(V_{s_j} \mid tab, note)$, denoting the prediction of the BN with virtual evidence for the symptom s_j. In the formulation above, $\mathcal{P}(V_{s_j} \mid tab, note)$ can be used in place of $\mathcal{P}(B_{s_j} \mid tab)$, in particular in Equation 7.3. This leads to our final model shown in Figure 7.1, where

we calculate the consistency between the virtual evidence-enhanced BN and the text classifier prediction, also called **V-C-BN-text** in the next section.

7.4 Empirical Results

We split the SimSUM dataset into a training set $\mathcal{X}_{train}$ of 8000 samples and a test set $\mathcal{X}_{test}$ of 2000 samples. To investigate the performance of our method in various training regimes, we subsample different training sets $\mathcal{X}^n_{train}$ from $\mathcal{X}_{train}$, logarithmically sized between $n = 100$ and 8000, i.e. $n \in \{100, 187, 350, 654, 1223, 2287, 4278, 8000\}$, using 20 different seeds per n. We also use these seeds for initialization of the model parameters and weights. For more details on hyperparameter tuning, we refer to Appendix 7.6.1. The BN and neural text classifier are trained on the tabular and text portion of $\mathcal{X}^n_{train}$, respectively, across these 20 seeds. We compare the following model architectures:

- **BN-only**: The BN predicts $\mathcal{P}(B_{s_j} \mid tab^{(i)})$ for every patient i, taking tabular evidence $tab^{(i)}$ as input (Section 7.3.2).
- **text-only**: The neural text classifier predicts $\mathcal{P}(T_{s_j} \mid note^{(i)})$ for every patient i, taking text evidence $note^{(i)}$ as input (Section 7.3.3).
- **V-BN-text**: The **BN-only** and **text-only** predictions are combined by using the text-only predictions as virtual evidence, obtaining the combined prediction $\mathcal{P}(V_{s_j} \mid tab^{(i)}, note^{(i)})$ (Section 7.3.4).
- **C-BN-text**: We learn the distribution of the consistency node on $\mathcal{X}^n_{train}$ for the **BN-only** and **text-only** predictions, obtaining a final consistent prediction that symptom s_j is present in patient i: $\mathcal{P}(C_{s_j} \mid tab^{(i)}, note^{(i)})$ (Section 7.3.5).
- **V-C-BN-text**: This approach uses both virtual evidence and the consistency node, corresponding to our final model in Figure 7.1. We learn the distribution of the consistency node on $\mathcal{X}^n_{train}$ for the **V-BN-text** and **text-only** predictions, obtaining the combined prediction $\mathcal{P}(VC_{s_j} \mid tab^{(i)}, note^{(i)})$ (Section 7.3.4 and 7.3.5). Note the contrast with **C-BN-text**, which uses $\mathcal{P}(B_{s_j} \mid tab^{(i)})$ rather than $\mathcal{P}(V_{s_j} \mid tab^{(i)}, note^{(i)})$.

The models **C-BN-text** and **V-BN-text**, can be viewed as ablations of the final model **V-C-BN-text**, using only one of the consistency node or virtual evidence to combine the neural classifiers with the BN.

In addition to these ablations, we consider the following additional baseline:

- **Concat-tab-text**: This early-fusion baseline concatenates the tabular features and the text embedding at the input of an MLP with the same architecture as the **text-only** classifier. We use the same implementation as described in Chapter 6, where binary variables are transformed into a one-hot encoding, and the variable `#Days` is prepro-

cessed using standard scaling. In contrast to the other fusion models we explore, this black-box model treats the tabular and text features as a single entity, and does not involve a BN. For more details on the implementation of this baseline, we refer to Appendix 7.6.1.3.

To evaluate these models, we calculate the **average precision** and **Brier score** over $\mathcal{X}_{test}$, across 20 seeds. We report average precision (equivalent to area under the precision-recall curve), rather than some threshold-based metrics like F1 score or accuracy, because our goal is to provide a probabilistic estimate of the symptom s_j being present in the patient, rather than a hard decision. We choose average precision over area under the ROC curve since the former is better equipped to deal with imbalanced datasets [Davis and Goadrich, 2006], which is indeed the case for the symptom labels in SimSUM. Since our classifiers for `Fever` have three classes, we report the macro average precision in that case.

We also report the Brier score [Rufibach, 2010] (equivalent to the MSE between the predicted probabilities and ground truth labels for binary symptoms) to reflect the accuracy of our models' predicted probabilities. It provides a combined measure of calibration and confidence. Like average precision, this metric quantifies the correctness of the predicted probabilities without requiring a hard threshold be set for classification. Note that while our Brier scores for the binary symptoms lie in the $[0, 1]$ range, we did not scale the score for `Fever` (ternary), so it remains in the $[0, 2]$ range.

Section 7.4.1 presents our main results, where we compare our multimodal consistency method with the uni-modal and multimodal baselines. Then, Section 7.4.2 unravels where the improvements lie by examining the models' performance on distinct subsets of the test set. Following this, we investigate how our consistency method handles a shift in text data distribution during inference in Section 7.4.3. Our key takeaways are summarized in Section 7.4.4.

7.4.1 Overall model comparison

As shown in Table 7.3, the naive **text-only** method already performs quite well for the symptoms `Dysp`, `Cough`, and `Nasal`. Therefore, we focus our analysis on the more difficult symptoms `Pain` and `Fever` that show more room for improvement. Tables 7.4 and 7.5, respectively, report the average precision and Brier scores for these symptoms. Detailed results for the other symptoms are available in Appendix 7.6.3.1.

Examining Tables 7.4 and 7.5, we note that the **BN-only** model performs sub-par to the models involving text, which is unsurprising, as this model has no access to the clinical notes containing ample detail on the presence of the symptoms. More importantly, the results show that the **V-C-BN-text** model almost always outperforms the **text-only** and **Concat-tab-text** models. In terms of Brier scores, the **V-C-BN-text** model always outperforms its ablated versions **V-BN-text** and **C-BN-text**. While the

Table 7.3: Average precision (↑↑) for the predictions of the **text-only** model over $\mathcal{X}_{test}$, averaged over 20 seeds for various training sizes n. The symptoms `Pain` and `Fever` show the most room for improvement.

	Training size n							
	100	**187**	**350**	**654**	**1223**	**2287**	**4278**	**8000**
`dysp`	92.46 ± 1.88	94.82 ± 1.14	95.78 ± 0.49	96.7 ± 0.3	97.31 ± 0.28	97.98 ± 0.16	98.31 ± 0.13	98.78 ± 0.12
`cough`	90.2 ± 3.11	94.52 ± 0.65	95.71 ± 0.48	96.89 ± 0.38	97.8 ± 0.23	98.26 ± 0.14	98.63 ± 0.11	98.9 ± 0.08
`pain`	61.81 ± 9.03	72.52 ± 4.21	76.48 ± 2.61	80.4 ± 0.95	82.5 ± 0.85	83.77 ± 0.88	84.71 ± 0.89	86.08 ± 0.24
`nasal`	95.12 ± 0.59	95.84 ± 0.56	96.57 ± 0.25	97.06 ± 0.17	97.43 ± 0.18	97.79 ± 0.11	98.0 ± 0.1	98.16 ± 0.03
`fever`	69.05 ± 4.81	75.01 ± 4.12	79.73 ± 2.34	86.46 ± 0.98	89.86 ± 0.65	91.64 ± 0.51	93.15 ± 0.25	93.93 ± 0.21

Table 7.4: Average precision (↑↑) for the predictions of our models over $\mathcal{X}_{test}$, averaged over 20 seeds for various training sizes n. The best model per training size and per symptom is highlighted in **bold**. The best baseline model for each class is underlined. Cases where a model outperforms the best baseline model significantly are indicated by * ($p < 0.05$ in a one-sided Wilcoxon signed-rank test over 20 seeds). **Change vs. baseline** compares the difference between the strongest baseline and non-baseline models.

		Training size n							
		100	**187**	**350**	**654**	**1223**	**2287**	**4278**	**8000**
pain	**BN-only**	0.3197	0.3277	0.3409	0.3464	0.35	0.3509	0.3517	0.3515
	text-only	**0.6181**	0.7252	0.7648	0.804	0.825	0.8377	0.8471	0.8608
	Concat-tab-text	0.5091	0.6537	0.7422	0.7858	0.8183	0.8412	0.8559	0.868
	C-BN-text	0.6135	***0.7271**	***0.7723**	*0.8124	*0.8323	0.8435	0.8521	0.8653
	V-BN-text	0.5667	0.7027	0.7532	***0.8194**	***0.8463**	***0.8598**	***0.8699**	***0.8826**
	V-C-BN-text	0.6022	0.7244	0.7673	*0.8146	*0.8375	*0.8511	*0.8606	*0.8738
	change vs. baseline	-0.46%	+0.19%	+0.76%	+1.54%	+2.13%	+1.86%	+1.4%	+1.46%
fever	**BN-only**	0.4792	0.4983	0.5167	0.5309	0.5398	0.5437	0.5465	0.5474
	text-only	**0.6905**	0.7501	0.7973	0.8646	0.8986	0.9164	0.9315	0.9393
	Concat-tab-text	0.6605	0.7495	0.7939	0.8526	0.8951	0.922	0.9381	0.9501
	C-BN-text	0.66	0.7472	0.7999	*0.8705	*0.9017	0.9199	0.9345	0.9434
	V-BN-text	0.6521	0.7545	*0.8091	***0.8829**	***0.9141**	***0.931**	***0.9475**	***0.9562**
	V-C-BN-text	0.6644	***0.7595**	***0.8107**	*0.8802	*0.9101	*0.9267	*0.9421	*0.9515
	change vs. baseline	-2.61%	+0.94%	+1.35%	+1.82%	+1.54%	+0.9%	+0.94%	+0.61%

Table 7.5: Brier scores (↓↓) for the predictions of our models over $\mathcal{X}_{test}$, averaged over 20 seeds for various training sizes n. The best model per training size and per symptom is highlighted in **bold**. The best baseline model for each class is underlined. Cases where a model outperforms the best baseline model significantly are indicated by * ($p < 0.05$ in a one-sided Wilcoxon signed-rank test over 20 seeds). **Change vs. baseline** compares the difference between the strongest baseline and non-baseline models.

		Training size n							
		100	**187**	**350**	**654**	**1223**	**2287**	**4278**	**8000**
pain	**BN-only**	0.1138	0.1119	0.1091	0.1078	0.1074	0.1073	0.1071	0.1072
	text-only	**0.0854**	0.0711	**0.0628**	0.0553	0.0491	0.0451	0.0426	0.0383
	Concat-tab-text	0.0942	0.0789	0.0666	0.0623	0.0524	0.0478	0.0436	0.0393
	C-BN-text	0.0915	0.0734	0.0702	0.0604	0.0535	0.0501	0.047	0.0423
	V-BN-text	0.0963	0.0792	0.0717	0.0577	0.0511	0.046	0.0427	0.0385
	V-C-BN-text	0.0866	***0.0704**	0.0647	**0.0539**	**0.0483**	**0.0444**	**0.0414**	**0.0377**
	change vs. baseline	+0.13%	-0.07%	+0.19%	-0.14%	-0.08%	-0.06%	-0.13%	-0.06%
fever	**BN-only**	0.3253	0.3153	0.3083	0.3049	0.3029	0.3016	0.301	0.3008
	text-only	0.256	0.2257	0.201	0.1744	0.1448	0.1256	0.1126	0.0984
	Concat-tab-text	0.2632	0.2208	0.1978	0.1738	0.1449	0.1243	0.1081	0.0962
	C-BN-text	0.2683	0.2253	0.2089	0.1709	0.1441	0.1261	0.1144	0.1038
	V-BN-text	0.2868	0.2327	0.2098	*0.1613	*0.1353	0.1242	0.1067	0.0986
	V-C-BN-text	**0.2535**	***0.2115**	***0.1911**	***0.1511**	***0.1269**	***0.1146**	***0.1014**	**0.0941**
	change vs. baseline	-0.25%	-0.93%	-0.67%	-2.27%	-1.79%	-0.96%	-0.68%	-0.22%

improvements in average precision and Brier score over the baselines are sometimes marginal, they are mostly significant across 20 seeds according to a paired statistical test. As we will show in Section 7.4.2, including knowledge from the tabular portion of the data with help of the BN, allows the **V-C-BN-text** model to flag mistakes made by the **text-only** classifier and thereby reliably improve upon its predictions, without impacting its performance on more straightforward cases.

While one might expect lower training sizes to have the most potential for improvement, as the text classifier does not have many examples to learn from in that case, this is not reflected in the results. This is because the BN's performance suffers in these low regimes as well, rendering its predictions less reliable and negatively affecting the performance of the combined models. However, note that we do see the greatest improvements on "middling" training sizes (654, 1223, and 2287 samples), where the BN begins to fit the data quite well but the neural classifiers still struggle somewhat. Furthermore, in Appendix 7.6.3.2, we show that a combined model which has access to the full ground truth BN (including the probabilities, rather than learning them from data), indeed shows larger improvements for smaller training sizes.

Zooming in on the **Concat-tab-text** baseline, we note that it suffers from overfitting for smaller train sizes, as a result of its naive concatenation of all features. We find that the combined models using the BN are more robust across train sizes and symptoms. These combined models are also much more interpretable, thanks to their modularity, their reliance on the expert-informed BN for the inclusion of the tabular features, and their interpretable late-fusion of the tabular and text predictions using the consistency node and virtual evidence.

7.4.2 Analysis of test subsets

To get a better idea of how our combined model **V-C-BN-text** manages to improve over the **text-only** baseline, we break the test set up into four distinct subsets based on whether the symptom is present in the patient (treating low + high as present for `Fever`) and whether the symptom is mentioned in the text: {*(present, mentioned)*; *(present, not mentioned)*; *(not present, mentioned)*; *(not present, not mentioned)*}.[7]

As each subset now only contains either negative or positive examples, it is no longer possible to report the average precision. Therefore, we look only at the Brier score. Tables 7.6 and 7.7 provides the results for these subsets for `pain` and `fever`. Results for the other symptoms across all subsets and training sizes are available in Appendix 7.6.3.3.

[7] This is possible in SimSUM thanks to the addition of a label that indicates if a given symptom is mentioned in the text of a clinical note; Appendix 7.6.2 outlines the process for obtaining this label.

Table 7.6: Brier scores (↓↓) for our models on the present vs. mentioned subsets across various training sizes (Part 1 of 2). The best model per training size and per symptom is highlighted in **bold**. Cases where a model outperforms **text-only** significantly are indicated by * ($p < 0.05$ in a one-sided Wilcoxon signed-rank test over 20 seeds).

			Training size n							
			100	**187**	**350**	**654**	**1223**	**2287**	**4278**	**8000**
present, mentioned	pain	**text-only**	**0.3809**	**0.2508**	**0.1941**	**0.1393**	**0.1252**	**0.103**	**0.0909**	**0.0706**
		C-BN-text	0.4679	0.3097	0.3086	0.2337	0.1912	0.1696	0.1489	0.1162
		V-BN-text	0.5569	0.401	0.3573	0.2387	0.1921	0.15	0.1247	0.0931
		V-C-BN-text	0.4157	0.28	0.2625	0.1915	0.1608	0.1354	0.1147	0.0885
	fever	**text-only**	**0.7589**	**0.5334**	**0.4449**	**0.2503**	**0.1718**	**0.155**	**0.0998**	**0.0722**
		C-BN-text	0.8994	0.6497	0.5861	0.3683	0.2582	0.2037	0.1579	0.1223
		V-BN-text	1.0834	0.7634	0.6551	0.3741	0.2486	0.2152	0.1294	0.0905
		V-C-BN-text	0.8151	0.5772	0.4998	0.2932	0.1965	0.1643	0.1076	0.0774
present, not mentioned	pain	**text-only**	0.7786	0.8574	0.8387	0.8591	0.8822	0.8734	0.8711	0.8674
		C-BN-text	*0.7251	***0.8035**	***0.7825**	***0.8102**	***0.8236**	***0.8236**	***0.8206**	***0.8178**
		V-BN-text	0.8285	0.8962	0.8896	0.9142	0.9304	0.922	0.9188	0.9092
		V-C-BN-text	***0.724**	*0.8182	*0.8038	0.8442	*0.863	0.8695	0.8682	0.8679
	fever	**text-only**	1.6366	1.5965	1.5683	1.4084	1.39	1.3764	1.306	1.3218
		C-BN-text	***1.4653**	***1.4837**	***1.4503**	**1.3786**	***1.3556**	***1.3157**	**1.2763**	***1.277**
		V-BN-text	1.7423	1.7089	1.701	1.5679	1.5602	1.5568	1.5109	1.5409
		V-C-BN-text	*1.511	*1.5258	*1.4959	1.4464	1.4325	1.3999	1.3855	1.4088

Table 7.7: Brier scores (⇓⇓) for our models on the present vs. mentioned subsets across various training sizes (Part 2 of 2). The best model per training size and per symptom is highlighted in **bold**. Cases where a model outperforms **text-only** significantly are indicated by * ($p < 0.05$ in a one-sided Wilcoxon signed-rank test over 20 seeds).

			Training size n							
			100	**187**	**350**	**654**	**1223**	**2287**	**4278**	**8000**
not present, mentioned	`pain`	**text-only**	0.0293	0.032	0.0305	0.0275	0.0169	0.0127	0.0103	0.0066
		C-BN-text	*0.0246	*0.0253	*0.0208	*0.0184	*0.0134	*0.0107	0.0089	0.007
		V-BN-text	***0.0138**	***0.0144**	***0.0112**	***0.009**	***0.0055**	***0.0043**	***0.0035**	***0.0025**
		V-C-BN-text	*0.0264	*0.0258	*0.0203	*0.0145	*0.009	*0.0063	*0.0051	*0.0036
	`fever`	**text-only**	0.0929	0.1232	0.1025	0.1117	0.0691	0.0368	0.0318	0.019
		C-BN-text	*0.0728	*0.0821	*0.0697	*0.0678	*0.0467	*0.0307	*0.0244	0.0183
		V-BN-text	***0.0278**	***0.0361**	***0.0263**	***0.0329**	***0.0195**	***0.01**	***0.0086**	***0.0057**
		V-C-BN-text	*0.0724	*0.078	*0.0623	*0.0507	*0.0301	*0.0193	*0.0142	*0.0107
not present, not mentioned	`pain`	**text-only**	0.021	0.0164	0.0147	0.0131	0.0097	0.0098	0.0094	0.0081
		C-BN-text	0.0196	*0.015	*0.013	0.0106	0.0091	0.0089	0.0086	0.0082
		V-BN-text	***0.011**	***0.0083**	***0.0059**	***0.0047**	***0.0037**	***0.0041**	***0.0039**	***0.0039**
		V-C-BN-text	0.0205	*0.0147	*0.0117	*0.0081	*0.0063	*0.0058	*0.0053	*0.0049
	`fever`	**text-only**	0.0289	0.0293	0.0265	0.0527	0.05	0.0388	0.0427	0.029
		C-BN-text	0.0367	0.0271	0.0278	*0.0356	*0.0359	*0.0328	*0.0331	0.0276
		V-BN-text	***0.0096**	***0.0088**	***0.0072**	***0.0163**	***0.017**	***0.0122**	***0.0121**	***0.0076**
		V-C-BN-text	0.0311	*0.0223	*0.0213	*0.0249	*0.0246	*0.0211	*0.0186	*0.0138

Table 7.8: Brier scores (⇓⇓) for BN-only vs. text-only on the *present, not mentioned* subset across various training sizes. Note that BN-only performs much better than text-only for all symptoms save fever.

		Training size n							
		100	**187**	**350**	**654**	**1223**	**2287**	**4278**	**8000**
`dysp`	**BN-only**	0.2892	0.2814	0.2608	0.2632	0.2582	0.2555	0.2579	0.2612
	text-only	0.8879	0.8893	0.8958	0.9192	0.8972	0.8517	0.8501	0.847
`cough`	**BN-only**	0.2019	0.1886	0.177	0.1793	0.1791	0.179	0.1803	0.1761
	text-only	0.5749	0.642	0.6555	0.644	0.66	0.6452	0.6715	0.6648
`pain`	**BN-only**	0.5822	0.5994	0.5925	0.5954	0.5975	0.5995	0.5951	0.5954
	text-only	0.7786	0.8574	0.8387	0.8591	0.8822	0.8734	0.8711	0.8674
`nasal`	**BN-only**	0.2982	0.2931	0.2868	0.2868	0.283	0.2813	0.2751	0.2827
	text-only	0.8839	0.9129	0.9198	0.9324	0.9298	0.899	0.9087	0.8948
`fever`	**BN-only**	1.1247	1.0954	1.098	1.0774	1.0818	1.0847	1.0908	1.0905
	text-only	1.6366	1.5965	1.5683	1.4084	1.39	1.3764	1.306	1.3218

As seen in Tables 7.6 and 7.7, **V-C-BN-text** reliably improves over **text-only** on the *not present, mentioned* and *not present, not mentioned* subsets, and displays varied performance on the *present, not mentioned* subset. Performance is slightly worse on the *present, mentioned* subset, but as seen in the Tables 7.4 and 7.5, **V-C-BN-text** still improves overall. (Note that the **text-only** model should be expected to make a good prediction for this subset, as the clinical note contains the required information.) These trends hold for the other symptoms and training sizes as well. Together, these results indicate that the BN provides **V-C-BN-text** with information about the prior distribution, which is biased towards the symptom(s) not being present.

The *present, not mentioned* subset – where the patient experiences a symptom which is not mentioned in the text – is of particular interest, as the BN can provide valuable complementary information in this case. Table 7.8 directly compares the performance of the BN and neural classifiers on this subset. As expected, the BN performs much better on this subset than the **text-only** classifiers. Intuitively, this occurs because the neural classifiers learn that a symptom is almost never present without being mentioned in the text, while the BN looks only at the other tabular data to form its prediction. The superior performance of the BN in this subset in turn should allow the combined **V-C-BN-text** model to improve more over the **text-only** baseline, as incorporating the BN's prediction helps correct for the missing information in the text. In Appendix 7.6.3.4, we break down a concrete example of **V-C-BN-text** providing a higher probability than **text-only** that `pain` is present when the symptom is not mentioned.

However, Table 7.6 shows that while **V-C-BN-text** performs much better than the **text-only** classifiers on the *present, not mentioned* subset for small training sizes (improving by a much larger margin than in other subsets), at larger training sizes this improvement breaks down for `pain` and

`fever`. Table 7.8 partly explains the particularly noticeable degradation in performance for `fever`: the BN is much worse at predicting the occurrence of `fever` in this subset than the other symptoms, leaving less room for it to improve the combined **V-C-BN-text** model. The general degradation is explained by the fact that, as the training size increases, the **text-only** classifiers become more confident in their predictions, making their contributions as virtual evidence weigh more heavily. This is detrimental in the *present, not mentioned* subset where those **text-only** predictions are typically wrong.

As shown in Table 7.6, the consistency node **C-BN-text** significantly improves over the **text-only** classifiers for the *present, not mentioned* subset for all training sizes, while the virtual evidence-only model **V-BN-text** often performs worse than the **text-only** classifiers. **V-C-BN-text**, using both virtual evidence and the consistency node, reliably improves over **V-BN-text**, indicating the consistency node's ability to help offset the fatal weakness of virtual evidence: confident (but wrong) virtual evidence can overwhelm the BN, leading to a more confident (and more wrong) final prediction. A more comprehensive analysis can be found in Appendix 7.6.3.3.

7.4.3 Handling shifts in text data distribution

The performance gaps between the combined models and the **text-only** classifier in Table 7.4 and Table 7.5 are often small, which can be attributed to the limited information gap between the tabular data and what is written in the text notes in the SimSUM dataset. Furthermore, our analysis of performance on the *present, not mentioned* subset shows the potential of our method to correct for faulty predictions of the **text-only** classifiers when information is missing from the text.

To investigate how our method performs when more information is missing from the text, leaving more room for improvement over the **text-only** model, we create a new version of the test dataset, which we call $\mathcal{X}^*_{test}$, containing manipulated notes. In this test set, the tabular variables remain the same, but we randomly mask out sentences from the notes describing a symptom. For this, we use the annotated symptom spans that were released together with the SimSUM dataset [Rabaey et al., 2024a]. For every note in the test set, we go over each phrase that describes any of the symptoms, and drop the sentence containing that phrase with a 50% probability. For example, the full sentence "Patient presents with low-grade fever and significant nasal symptoms" might be dropped to remove the mention of "low-grade fever" or of "significant nasal symptoms". Note that this technique increases the gap between the information contained in the tabular portion of the data (patient background variables causing certain symptoms), and the information contained in the text portion of the data (descriptions of these symptoms in the clinical notes).

We then used the original **BN-only** and **text-only** classifiers (which

Table 7.9: Average precision (↑↑) for the predictions of our models over the test set $\mathcal{X}^*_{test}$ containing **manipulated text notes**. The best model per training size and per symptom is highlighted in **bold**. Cases where a model outperforms **text-only** significantly are indicated by * ($p < 0.05$ in a one-sided Wilcoxon signed-rank test over 20 seeds).

		Training size n							
		100	**187**	**350**	**654**	**1223**	**2287**	**4278**	**8000**
pain	**text-only**	0.5357	0.6195	0.6575	0.6946	0.722	0.7349	0.7422	0.7516
	C-BN-text	***0.5467**	***0.6262**	***0.669**	*0.7031	*0.7285	*0.7404	*0.7493	*0.7595
	V-BN-text	0.5176	0.6106	0.6567	***0.714**	***0.7473**	***0.764**	***0.775**	***0.7868**
	V-C-BN-text	0.538	0.6228	*0.6632	*0.7065	*0.737	*0.7528	*0.7613	*0.7723
	change vs. baseline	+1.1%	+0.67%	+1.15%	+1.94%	+2.53%	+2.91%	+3.27%	+3.52%
fever	**text-only**	0.6023	0.6524	0.6875	0.7361	0.7714	0.7909	0.8072	0.8161
	C-BN-text	0.5975	*0.6562	*0.6943	*0.7424	*0.7761	*0.7981	*0.8132	*0.8243
	V-BN-text	**0.6047**	***0.6782**	***0.7202**	***0.7746**	***0.8086**	***0.8269**	***0.8439**	***0.8543**
	V-C-BN-text	0.6034	*0.6671	*0.7044	*0.7572	*0.7915	*0.8104	*0.829	*0.839
	change vs. baseline	+0.23%	+2.58%	+3.27%	+3.86%	+3.72%	+3.6%	+3.67%	+3.83%

Table 7.10: Brier scores (⇓⇓) for the predictions of our models over the test set $\mathcal{X}^*_{test}$ containing **manipulated text notes**. The best model per training size and per symptom is highlighted in **bold**. Cases where a model outperforms **text-only** significantly are indicated by ***** ($p < 0.05$ in a one-sided Wilcoxon signed-rank test over 20 seeds).

		Training size n							
		100	**187**	**350**	**654**	**1223**	**2287**	**4278**	**8000**
`pain`	**text-only**	0.0955	0.0879	0.0793	0.0753	0.0698	0.0665	0.0657	0.0621
	C-BN-text	0.0982	0.0867	0.082	0.0754	0.0703	0.0679	0.0662	0.0628
	V-BN-text	0.1043	0.0933	0.0869	0.0773	0.0733	0.0694	0.067	0.0639
	V-C-BN-text	**0.0949**	***0.0853**	**0.0789**	***0.072**	**0.0683**	**0.0661**	**0.0641**	**0.0618**
	change vs. baseline	-0.05%	-0.26%	-0.04%	-0.33%	-0.15%	-0.04%	-0.16%	-0.03%
`fever`	**text-only**	0.3071	0.2887	0.2695	0.2613	0.2388	0.2231	0.2134	0.2041
	C-BN-text	0.3029	*0.2743	*0.2612	*0.2401	*0.2222	*0.2102	*0.2012	*0.196
	V-BN-text	0.3233	0.2878	0.2736	*0.2437	*0.2251	0.2171	0.2051	0.2036
	V-C-BN-text	***0.294**	***0.2671**	***0.2515**	***0.2306**	***0.2131**	***0.2034**	***0.1949**	***0.1929**
	change vs. baseline	-1.31%	-2.16%	-1.8%	-3.07%	-2.57%	-1.97%	-1.85%	-1.12%

were trained using the original train set $\mathcal{X}_{train}$ with non-manipulated notes) to evaluate performance on this new test set $\mathcal{X}^*_{test}$. We also use the original consistency node, whose weights were set based on the original train set $\mathcal{X}_{train}$. Tables 7.9 and 7.10 show the average precision and Brier scores, respectively, of these models on $\mathcal{X}^*_{test}$, the set of out-of-distribution samples, for the symptoms `pain` and `fever`. These results demonstrate larger improvements than seen before in Tables 7.4 and 7.5, due to the added room for improvement in the **text-only** classifiers as a result of the text data shift in $\mathcal{X}^*_{test}$. This pattern also holds for the other symptoms. The full results can be found in Appendix 7.6.3.5. The **V-C-BN-text** model significantly improves over the **text-only** baseline for almost all training sizes and all symptoms. This shows that by including the BN and its background knowledge, the **V-C-BN-text** model can fill in gaps of missing information in the text, allowing it to effectively handle shifts in text data distribution (compared to the training phase) during inference.

7.4.4 Key takeaways

Our results show that the methods combining the BN with neural classifiers outperform the **text-only** and **concat-tab-text** baselines, with the **V-C-BN-text** model performing the best overall. From our analysis of present vs. mentioned subsets, we found that **V-C-BN-text** reliably improves over the **text-only** classifiers in cases where the symptom is not present, indicating that incorporating the predictions of the BN better accounts for the true prior (which is biased towards a symptom not being present).

Furthermore, we found that for symptoms and training sizes where the BN performs well relative to the **text-only** classifiers, the largest improvements of the combined **V-C-BN-text** model come from cases where the symptom is present but not mentioned in the text. This hints at the ability of the BN to fill in the gaps where information is missing from the text. This ability is further demonstrated in a data shift experiment, where information is missing from the text at a higher rate at inference time than during training. In this setting, the combined models outperform the **text-only** classifiers by a larger margin, indicating that the BN is able to help correct for the missing information.

Finally, the Brier scores of **V-C-BN-text** relative to **V-BN-text** indicate the consistency node's ability to improve calibration over virtual evidence alone, while retaining strong overall accuracy. The difference in Brier scores are especially notable when the symptom is missing from the text (see *present, not mentioned* in Table 7.6), where virtual evidence is particularly susceptible to confident, incorrect predictions from the neural classifiers.

7.5 Conclusion

In this work, we introduced the concept of patient-level information extraction that leverages both the structured tabular features in a patient's EHR and the unstructured clinical notes describing the patient's symptoms. By augmenting *virtual evidence* with a *consistency node*, we achieved interpretable integration of a Bayesian network with neural text classifiers, enabling coherent and probabilistic fusion of tabular and textual information. Our method resulted in better-calibrated final predictions for the target variables that take into account the true prior encoded in the Bayesian network. At the same time, it highlighted the potential of the Bayesian network to correct for abnormal cases of missing textual data. Our method proved most effective for middling training sizes where the BN approaches its optimal performance prior to the neural classifiers, making it particularly appealing for use-cases where training data is limited.

While the current work focused on the specific use-case of predicting patient-level symptoms from tabular data and text, we foresee a broader use of our method in the future. First, any node in the Bayesian network may be the target of information extraction, as long as the text contains some information about this feature. Second, and more broadly, either of the two modalities (tabular and text) may be swapped out for any other. Virtual evidence and the consistency node are flexible, only expecting probabilities at the input, without assuming any particular underlying method with which these probabilities are obtained. For example, another application of our combined consistency method could be the automated extraction of information from X-Ray images, where predictions of an image classifier that detects radiology features can be easily combined with a Bayesian network that includes tabular background information on the patient (such as age, previous diagnoses, etc.). Furthermore, while virtual evidence requires a Bayesian network, there is no requirement that a neural network be used to model the other modality. The consistency node is completely agnostic to the choice of models and can even be used without a Bayesian network.[8] Finally, with minor adjustments to the consistency node, it becomes possible to include more than two modalities, while virtual evidence is inherently able to handle evidence from multiple modalities for the same node of a Bayesian network. In the previous example, radiology reports might be included as a third modality along with X-ray images and tabular background information.

Ultimately, the consistency node is a promising approach that provides an interpretable fusion of arbitrary models and modalities, while its compatibility with virtual evidence makes it particularly suitable for integration with Bayesian networks.

[8]However, we do find that having a Bayesian network as the tabular data model would be an asset in many medical use-cases, thanks to the implicit inclusion of expert knowledge.

7.6 Limitations

Our method has several limitations. First, we interpret the probabilities at the output of the neural classifier as if they are a reflection of its confidence on the presence of the symptom in the text. However, this is not necessarily the case, as neural classifiers are known to have issues with calibration [Guo et al., 2017]. Still, this is offset by the consistency node, which improves calibration compared to using virtual evidence alone, as evidenced by the elevated Brier scores.

Second, we make strong assumptions on the types of conditional distributions that are learned in the BN. In our case, these assumptions match up perfectly with the true data generating process of the data as described in Rabaey et al. [2024a]. However, in a realistic setting, one would not have access to the true type of probability distribution for each variable in the network, and would instead need to consult an expert.

Third, it can be very challenging to come up with a DAG structure that accurately captures reality. To mitigate this, one could work with a panel of experts who iteratively improve the DAG. Furthermore, future work can focus on using (partial) structure learning algorithms [Scanagatta et al., 2019] to learn the DAG from the data, filling in the gaps where experts are unsure, while still asking experts to validate the final DAG. Note that while the inclusion of the BN in our method might limit its generalization to broader contexts, we explicitly choose to trade in this flexibility for interpretability and expert input.

Finally, and related to the previous point, we only validated our method on a single simulated use-case. While this shows the merit of our method as a proof-of-concept, future work should focus on putting the theory into practice and applying our method to a more realistic and challenging dataset. To this end, the MIMIC-III [Johnson et al., 2016] and MIMIC-IV [Johnson et al., 2023] datasets come to mind.

Acknowledgements

Paloma Rabaey's research is funded by the Research Foundation Flanders (FWO Vlaanderen) with grant number 1170124N. This research has also received funding from the Flemish government under the "Onderzoeksprogramma Artificiele Intelligentie (AI) Vlaanderen" programme. The authors thank Robin Manhaeve and Jaron Maene for their valuable insights during the conception of the consistency node method and their helpful feedback on early versions of the manuscript.

Appendix

7.6.1 Training details

7.6.1.1 Bayesian network

To model the tabular portion of the data, we use a BN. A BN is defined by a DAG, which models the relations between the variables. This DAG also prescribes how the joint distribution factorizes into conditional probability distributions, one for each variable conditional on its parents. In SimSUM, both the DAG and the probability distributions are defined by an expert, together forming a BN from which the tabular data is sampled. In a realistic setting, we cannot assume that we know the full data generating process. Instead, we assume that we can consult an expert to tell us how the variables are related, giving rise to the DAG in Figure 7.1, but that we need to learn the exact conditional probability distributions from data.

Formally, we model the tabular portion of the data by learning the probability distribution in Equation 7.7, where we abbreviated some of the variable names for ease of presentation.

$$
\begin{aligned}
&\mathcal{P}_{tab}(\texttt{Asthma, Smoking, COPD, Hay, Season, Pneu, Cold}\\
&\texttt{Dysp, Cough, Pain, Nasal, Fever, Antibio, \#Days})\\
&= \mathcal{P}(\texttt{Asthma})\mathcal{P}(\texttt{Smoking})\mathcal{P}(\texttt{COPD} \mid \texttt{Smoking})\mathcal{P}(\texttt{Season})\\
&\mathcal{P}(\texttt{Hay})\mathcal{P}(\texttt{Cold} \mid \texttt{Season})\mathcal{P}(\texttt{Pneu} \mid \texttt{Asthma, COPD, Season})\\
&\mathcal{P}(\texttt{Dysp} \mid \texttt{Asthma, Smoking, COPD, Pneu, Hay})\\
&\mathcal{P}(\texttt{Cough} \mid \texttt{Asthma, Smoking, COPD, Pneu, Cold})\\
&\mathcal{P}(\texttt{Pain} \mid \texttt{Cough, Pneu, COPD, Cold})\mathcal{P}(\texttt{Fever} \mid \texttt{Pneu, Cold})\\
&\mathcal{P}(\texttt{Nasal} \mid \texttt{Cold, Hay})\mathcal{P}(\texttt{Antibio} \mid \texttt{Dysp, Cough, Pain, Fever})\\
&\mathcal{P}(\texttt{\#Days} \mid \texttt{Antibio, Dysp, Cough, Pain, Fever, Nasal})
\end{aligned}
\tag{7.7}
$$

In SimSUM, the conditional probability distributions are parameterized in various ways: (i) CPTs for the variables `Asthma`, `Smoking`, `COPD`, `Hay fever`, `Season`, `Pneumonia`, `Common cold` and `Fever`, (ii) Noisy-OR distributions for the symptoms `Dyspnea`, `Cough`, `Pain`, and `Nasal`, (iii) a logistic regression model and two Poisson regression models for the variables `Antibiotics` and `#Days`, respectively.

To obtain the full probability distribution $\mathcal{P}_{tab}$(`Asthma, Smoking, ..., Antibiotics,#Days`), we can learn the parameters for each of these conditional distributions independently, by training them on the tabular portion of the data $\mathcal{X}_{train}$. To this end, we follow the approach outlined in SimSUM [Rabaey et al., 2024a]. In short, all parameters are learned through Maximum Likelihood Estimation, where the exact likelihood that is optimized depends on the particular parametrization approach. We manually tune the hyperparameters (learning rate and number of epochs) for each training set

size, increasing the number of epochs and learning rate for smaller training sets. We keep the batch size fixed at 50.

When all parameters have been learned, we turn these distributions into CPTs by evaluating them for each combination of child and parent values, as described in SimSUM [Rabaey et al., 2024a]. This allows us to do exact inference over the tabular evidence in the learned BN through variable elimination [Koller and Friedman, 2009].

7.6.1.2 Neural text classifier

We follow the approach of Rabaey et al. [2024a] for training the neural text classifier. There is one hidden layer of size 256, followed by a ReLU activation. To deal with the varying training set sizes, we tune the optimal number of epochs with early stopping in a 5-fold cross-validation loop for each training set, with patience of 10 and tolerance of 10^{-3} on the cross-entropy loss over the validation set (with the maximum set to 200 epochs). We then take the median of the number of epochs at which early stopping was applied for each of these cross-validation splits, and retrain with the full training set afterwards for that number of epochs. The other hyperparameters are fixed as follows: a batch size of 50, a learning rate of 0.0005, weight decay (L2 regularization) of 10^{-5} and no dropout.

7.6.1.3 Multimodal baseline: Concat-tab-text

We use the implementation of the neural-text-tab baseline as described in Rabaey et al. [2024a]. According to this implementation, the tabular features are transformed into a vector representation. We use a one-hot encoding for the categorical (binary) features, and normalize the `#Days` feature using a StandardScaler. This tabular feature representation (vector of shape 9) is concatenated with the text embedding representation (vector of shape 768) and then fed into the exact same architecture as was used for the text-only baseline (only its input layer is updated from size 768 to size 777). Accordingly, we use the exact same cross-validation strategy and hyperparameters to train the model as described in Section 7.6.1.2.

7.6.2 Mentions label construction

The SimSUM dataset does not include labels indicating if a symptom is mentioned in the clinical note. We created these labels from the information available in SimSUM, augmented by manual annotation. The labels we used are made available on our Github repository, along with the rest of the code, at `https://github.com/AdrickTench/patient-level-IE`.

SimSUM includes two key pieces of information that we used to create the mentions labels: (1) a label indicating if the LLM that generated the clinical note was instructed to mention (or not mention) the symptom, and

(2) pre-identified span annotations indicating portions of the clinical note that mention the symptom. In cases where (1) and (2) agree, we automatically generated the corresponding label. Disagreements between (1) and (2) were resolved through manual annotation by one of the authors.

Such disagreements can be attributed to either a failure of the LLM to follow the instructions provided in its prompt, or a failure of the span annotation process. Because of the possibility that both (1) and (2) failed on the same symptom and note but were automatically accepted as the true label, and the possibility of human error in the manual annotation process, we cannot guarantee that our mentions labels are completely accurate. However, we expect that inaccuracies are exceptional and the labels are correct in the overwhelming majority of cases.

7.6.3 Extended results

7.6.3.1 Results for all symptoms

We report the average precision (Table 7.11) and Brier scores (Table 7.12) of our models for all symptoms, as calculated on $\mathcal{X}_{test}$ over 20 seeds for various training sizes n. We also report the overall mean of each metric over all five symptoms. We find that **V-C-BN-text** performs the best overall, providing the best Brier scores and a statistically significant improvement in average precision. While the virtual-evidence-only model **V-BN-text** sometimes provides slightly better improvements to average precision, it does not yield statistically significant improvements to the Brier score as reliably as **V-C-BN-text**.

7.6.3.2 Ground truth SimSUM Bayesian network

The tabular portion of the SimSUM dataset was generated using a fully expert-defined BN, where not only the relations between the variables were defined by an expert, but also the conditional probabilities. In a real setting, it would be unrealistic to assume that we have access to this true generating process, which is why we learn the distributions from the data in our work, which gives us the **BN-only** model. However, having access to the true ground truth distributions allows us to swap the **BN-only** model for the **GT-only** model (GT indicating ground truth), and use its predictions to build the fusion models **C-GT-text**, **V-GT-text**, and **V-C-GT-text**. We do not change anything about the **text-only** model. Table 7.13 and Table 7.14 show the results. We note that these fusion models with access to the ground truth BN (**GT-only**) often outperform the models using the learned BN (**BN-only**), especially for smaller training sizes, as the BN better matches the true data generating process.

7.6.3.3 Present vs. mentioned subsets

We report the Brier scores of the **text-only** baseline and our BN-text models on the subsets {*present, mentioned*; *present, not mentioned*; *not present, mentioned*; *not present, not mentioned*} in Table 7.15, Table 7.16, Table 7.17, and Table 7.18 respectively.

As mentioned in Section 7.4.2, **V-C-BN-text** improves significantly over the **text-only** classifiers in the *present, not mentioned* subset for small training sizes, but this breaks down at larger training sizes for `pain`, `nasal`, and `fever`. We attribute this to two factors: (1) the BN does not increase in accuracy as much as the **text-only** classifiers with more training data, leading the consistency node to favor the predictions of the neural classifiers, and (2) the **text-only** classifiers become more confident at higher training sizes, leading their contributions as virtual evidence to be weighed more heavily.

Evidence of (1) can be seen in Table 7.11, where the performance of the BN does not improve as much as the **text-only** classifiers at higher training sizes. Note this table also helps indicate why `dysp` and `cough` continue to display larger improvements on the *present, not mentioned* than the other symptoms: the BN is better at predicting those symptoms relative to the neural classifiers at higher training sizes.

As evidence of (2), we define a standard confidence measure using normalized entropy [Wu et al., 2021b]: using standard Shannon entropy to determine the uncertainty of our model's predictions:

$$H(\mathbf{p}) = -\sum_{i=1}^{K} p_i \log p_i \tag{7.8}$$

we define confidence as 1 minus the normalized entropy:

$$1 - \frac{H(\mathbf{p})}{\log K} \tag{7.9}$$

We report the confidence of the **text-only** classifiers (substituting $\mathcal{P}(T_{s_j} \mid note)$ for p in Equation 7.9) in Table 7.19, which shows that the neural classifiers become more confident at larger training sizes. As noted in Section 7.4.2, confident virtual evidence can lead to a more confident final prediction, producing a more confidently wrong prediction in the case of faulty virtual evidence. As shown in Table 7.16, the consistency node can help correct for this weakness of virtual evidence at higher training sizes for the more difficult symptoms `Pain` and `Fever`.

7.6.3.4 Illustrative example

As explained in Section 7.4.2, many of the cases contributing to the superior performance of the **V-C-BN-text** model over the **text-only** model are

those where the symptom is present in the patient, but not present in the text. We zoom in on one of these cases to illustrate this point.

Patient 8809 in SimSUM has a common cold, and as a result of this, they suffer from a cough, nasal symptoms, a low fever, and pain. Furthermore, the tabular data record reveals that the patient visited the doctor's office during winter time, and stayed home for 9 days as a result of these symptoms. From the tabular evidence (`Common cold` = yes, `Season` = winter, `#Days` = 9, all other tabular evidence = no), the BN predicts an 82.9% chance for pain. The clinical note is as follows:

```
**History**
The patient presented with a constant cough persisting for the past week. They confirmed
      experiencing mild fever fluctuations, primarily in the evenings, describing the
     fever as low-grade. In addition, the patient reported nasal congestion and mild
     rhinorrhea, which started around the same time as the cough. The patient denied
     any episodes of dyspnea. They have been taking over-the-counter decongestants with
      minimal relief.

**Physical Examination**
Vital signs: Temperature: 37.6C, Blood Pressure: 120/80 mmHg, Pulse: 78 bpm, Respiratory
      Rate: 16 breaths per minute, and SpO2: 98% on room air. The patient appeared
     alert and in no acute distress. Upon auscultation, lungs were clear bilaterally
     with normal breath sounds and no wheezing or crackles. Nasal mucosa appeared
     slightly inflamed and there were no signs of significant throat erythema or
     exudates. Palpation of the cervical lymph nodes was unremarkable. Cardiac
     examination revealed regular rate and rhythm with no murmurs, rubs, or gallops.
     Abdomen was soft and non-tender with no apparent organomegaly.
```

This note does not mention pain anywhere. By training on similar examples, the neural text classifier has learned that there is a very low chance of pain in this patient (since most notes in the training set that are labeled with pain, do indeed mention pain somewhere in the note). As a result, the text classifier predicts only a 4.9% chance for pain.

By combining the BN's prediction with the neural text classifier's prediction using the **V-C-BN-text** model, we arrive at a probability of 40.1% for pain, which is higher than before and closer to the true label.[9]

7.6.3.5 Handling shifts in text data distribution

As explained in Section 7.4.3, we used our original **BN-only** and **text-only** classifiers, as well as the original consistency nodes (which were trained using the original train set $\mathcal{X}_{train}$ with non-manipulated notes) to evaluate performance on the new test set $\mathcal{X}^*_{test}$ with manipulated notes. Tables 7.20 and 7.21 show the results for all symptoms. The **V-C-BN-text** model significantly improves over the **text-only** baseline on both metrics for almost all training sizes and symptoms. These manipulated notes are released on our Github repository, along with the rest of the code, at `https://github.com/AdrickTench/patient-level-IE`.

[9]These results are attained for seed 2014 on training size 187.

Table 7.11: Average precision (↑↑) for the predictions of our models over $\mathcal{X}_{test}$, averaged over 20 seeds for various training sizes n. The best model per training size and per symptom is highlighted in **bold**. The best baseline model for each class is underlined. Cases where a model outperforms the best baseline model significantly are indicated by * ($p < 0.05$ in a one-sided Wilcoxon signed-rank test over 20 seeds).

		Training size n							
		100	**187**	**350**	**654**	**1223**	**2287**	**4278**	**8000**
dysp	**BN-only**	0.7625	0.7644	0.7794	0.7937	0.7972	0.7981	0.7981	0.7989
	text-only	0.9246	0.9482	0.9578	0.967	0.9731	0.9798	0.9831	0.9878
	Concat-tab-text	0.9127	0.9398	0.9533	0.9657	0.9737	0.9801	0.9841	0.987
	C-BN-text	*0.9258	0.948	0.957	0.9669	0.9736	0.9785	0.9817	0.9875
	V-BN-text	0.9186	0.9392	0.9472	0.9665	0.9741	0.9806	0.9841	*0.9882
	V-C-BN-text	***0.928**	***0.9519**	***0.96**	***0.9701**	***0.9759**	***0.9819**	***0.9853**	***0.9892**
	change vs. baseline	+0.35%	+0.36%	+0.22%	+0.31%	+0.22%	+0.18%	+0.12%	+0.13%
cough	**BN-only**	0.7568	0.7718	0.7844	0.7898	0.7929	0.7946	0.7947	0.7942
	text-only	0.902	0.9452	0.9571	0.9689	0.978	0.9826	0.9863	0.989
	Concat-tab-text	0.8866	0.9393	0.9582	0.9705	0.9794	0.9851	0.9886	0.9908
	C-BN-text	*0.9086	*0.9483	*0.9607	0.9701	0.9772	0.9813	0.9858	0.988
	V-BN-text	0.8988	0.9431	0.9578	*0.9727	***0.9825**	***0.9864**	***0.9896**	***0.9918**
	V-C-BN-text	***0.916**	***0.9549**	***0.9647**	***0.9743**	*0.9821	0.9857	0.989	*0.9914
	change vs. baseline	+1.4%	+0.97%	+0.65%	+0.38%	+0.3%	+0.13%	+0.1%	+0.1%
pain	**BN-only**	0.3197	0.3277	0.3409	0.3464	0.35	0.3509	0.3517	0.3515
	text-only	**0.6181**	0.7252	0.7648	0.804	0.825	0.8377	0.8471	0.8608
	Concat-tab-text	0.5091	0.6537	0.7422	0.7858	0.8183	0.8412	0.8559	0.868
	C-BN-text	0.6135	***0.7271**	***0.7723**	*0.8124	*0.8323	0.8435	0.8521	0.8653
	V-BN-text	0.5667	0.7027	0.7532	***0.8194**	***0.8463**	***0.8598**	***0.8699**	***0.8826**
	V-C-BN-text	0.6022	0.7244	0.7673	*0.8146	*0.8375	*0.8511	*0.8606	*0.8738
	change vs. baseline	-0.46%	+0.19%	+0.76%	+1.54%	+2.13%	+1.86%	+1.4%	+1.46%
nasal	**BN-only**	0.6337	0.6357	0.6341	0.6303	0.634	0.6324	0.6328	0.634
	text-only	0.9512	0.9584	0.9657	0.9706	0.9743	0.9779	0.98	0.9816
	Concat-tab-text	0.9444	0.9579	0.9666	0.973	0.9777	0.9818	0.9847	0.9869
	C-BN-text	***0.9567**	*0.9624	0.9686	***0.9769**	*0.9799	0.9817	0.9841	0.9847
	V-BN-text	0.9508	*0.9609	0.9664	*0.976	***0.9819**	***0.9861**	***0.9879**	***0.9885**
	V-C-BN-text	*0.9547	***0.9628**	***0.9699**	*0.9757	*0.9798	*0.9836	*0.9858	0.9869
	change vs. baseline	+0.54%	+0.45%	+0.33%	+0.39%	+0.43%	+0.43%	+0.31%	+0.16%
fever	**BN-only**	0.4792	0.4983	0.5167	0.5309	0.5398	0.5437	0.5465	0.5474
	text-only	**0.6905**	0.7501	0.7973	0.8646	0.8986	0.9164	0.9315	0.9393
	Concat-tab-text	0.6605	0.7495	0.7939	0.8526	0.8951	0.922	0.9381	0.9501
	C-BN-text	0.66	0.7472	0.7999	*0.8705	*0.9017	0.9199	0.9345	0.9434
	V-BN-text	0.6521	0.7545	*0.8091	***0.8829**	***0.9141**	***0.931**	***0.9475**	***0.9562**
	V-C-BN-text	0.6644	***0.7595**	***0.8107**	*0.8802	*0.9101	*0.9267	*0.9421	*0.9515
	change vs. baseline	-2.61%	+0.94%	+1.35%	+1.82%	+1.54%	+0.9%	+0.94%	+0.61%
mean	**BN-only**	0.5904	0.5996	0.6111	0.6182	0.6228	0.6239	0.6247	0.6252
	text-only	**0.8173**	0.8654	0.8885	0.915	0.9298	0.9389	0.9456	0.9517
	Concat-tab-text	0.7827	0.848	0.8828	0.9095	0.9289	0.942	0.9503	0.9566
	C-BN-text	0.8129	*0.8666	*0.8917	*0.9194	*0.933	0.941	0.9477	0.9538
	V-BN-text	0.7974	0.8601	0.8867	***0.9235**	***0.9398**	***0.9488**	***0.9558**	***0.9615**
	V-C-BN-text	0.8131	***0.8707**	***0.8945**	*0.923	*0.9371	*0.9458	*0.9525	*0.9586
	change vs. baseline	-0.42%	+0.53%	+0.6%	+0.85%	+1.0%	+0.68%	+0.55%	+0.49%

Table 7.12: Brier scores (⇓⇓) for the predictions of our models over $\mathcal{X}_{test}$, averaged over 20 seeds for various training sizes n. The best model per training size and per symptom is highlighted in **bold**. The best baseline model for each class is underlined. Cases where a model outperforms the best baseline model significantly are indicated by * ($p < 0.05$ in a one-sided Wilcoxon signed-rank test over 20 seeds).

		Training size n							
		100	**187**	**350**	**654**	**1223**	**2287**	**4278**	**8000**
dysp	**BN-only**	0.0861	0.083	0.0797	0.0777	0.077	0.0766	0.0765	0.0763
	text-only	0.0485	0.0365	0.0314	0.0277	0.023	0.019	0.0172	0.0128
	Concat-tab-text	0.0508	0.0396	0.0339	0.0288	0.0251	0.0196	0.0172	0.0145
	C-BN-text	0.0517	0.0367	0.0324	0.0275	0.0231	0.0192	0.0175	0.0132
	V-BN-text	0.0498	0.0375	0.0333	0.0289	0.0249	0.02	0.0181	0.0145
	V-C-BN-text	***0.0468**	***0.0345**	***0.0301**	***0.0255**	***0.0218**	***0.0177**	***0.0156**	**0.0123**
	change vs. baseline	-0.16%	-0.21%	-0.12%	-0.22%	-0.12%	-0.13%	-0.15%	-0.04%
cough	**BN-only**	0.1267	0.1209	0.1176	0.1169	0.1158	0.1155	0.1155	0.1154
	text-only	0.0882	0.0659	0.0576	0.0494	0.0401	0.0337	0.027	0.0247
	Concat-tab-text	0.0907	0.0659	0.0553	0.0462	0.0398	0.0327	0.0264	0.0237
	C-BN-text	*0.0857	*0.0642	0.0572	0.0487	0.0389	0.0328	0.0268	0.0244
	V-BN-text	*0.0831	*0.0601	*0.0529	*0.044	*0.0349	*0.0296	*0.0242	0.0229
	V-C-BN-text	***0.0779**	***0.0575**	***0.0503**	***0.0417**	***0.0327**	***0.0274**	***0.0228**	***0.0203**
	change vs. baseline	-1.04%	-0.84%	-0.5%	-0.45%	-0.71%	-0.54%	-0.36%	-0.34%
pain	**BN-only**	0.1138	0.1119	0.1091	0.1078	0.1074	0.1073	0.1071	0.1072
	text-only	**0.0854**	0.0711	**0.0628**	0.0553	0.0491	0.0451	0.0426	0.0383
	Concat-tab-text	0.0942	0.0789	0.0666	0.0623	0.0524	0.0478	0.0436	0.0393
	C-BN-text	0.0915	0.0734	0.0702	0.0604	0.0535	0.0501	0.047	0.0423
	V-BN-text	0.0963	0.0792	0.0717	0.0577	0.0511	0.046	0.0427	0.0385
	V-C-BN-text	0.0866	***0.0704**	0.0647	**0.0539**	**0.0483**	**0.0444**	**0.0414**	**0.0377**
	change vs. baseline	+0.13%	-0.07%	+0.19%	-0.14%	-0.08%	-0.06%	-0.13%	-0.06%
nasal	**BN-only**	0.1216	0.1197	0.1185	0.1183	0.1175	0.1177	0.1174	0.1173
	text-only	0.0424	0.0373	0.0315	0.0274	0.0238	0.0221	0.019	0.0181
	Concat-tab-text	0.0513	0.0417	0.035	0.0309	0.0262	0.0224	0.02	0.0177
	C-BN-text	0.0441	*0.037	*0.0311	*0.0267	*0.0235	*0.0218	0.0189	0.018
	V-BN-text	0.045	*0.0344	*0.0289	0.0262	*0.0224	*0.0201	*0.018	*0.0165
	V-C-BN-text	***0.0395**	***0.0336**	***0.0279**	***0.0244**	***0.0214**	***0.019**	***0.0173**	***0.0164**
	change vs. baseline	-0.3%	-0.37%	-0.36%	-0.3%	-0.24%	-0.31%	-0.17%	-0.13%
fever	**BN-only**	0.3253	0.3153	0.3083	0.3049	0.3029	0.3016	0.301	0.3008
	text-only	0.256	0.2257	0.201	0.1744	0.1448	0.1256	0.1126	0.0984
	Concat-tab-text	0.2632	0.2208	0.1978	0.1738	0.1449	0.1243	0.1081	0.0962
	C-BN-text	0.2683	0.2253	0.2089	0.1709	0.1441	0.1261	0.1144	0.1038
	V-BN-text	0.2868	0.2327	0.2098	*0.1613	*0.1353	0.1242	0.1067	0.0986
	V-C-BN-text	**0.2535**	***0.2115**	***0.1911**	***0.1511**	***0.1269**	***0.1146**	***0.1014**	**0.0941**
	change vs. baseline	-0.25%	-0.93%	-0.67%	-2.27%	-1.79%	-0.96%	-0.68%	-0.22%
mean	**BN-only**	0.1547	0.1502	0.1466	0.1451	0.1441	0.1437	0.1435	0.1434
	text-only	0.1041	0.0873	0.0768	0.0668	0.0562	0.0491	0.0437	0.0384
	Concat-tab-text	0.11	0.0894	0.0777	0.0684	0.0577	0.0494	0.0431	0.0383
	C-BN-text	0.1082	0.0873	0.08	0.0668	0.0566	0.05	0.0449	0.0404
	V-BN-text	0.1122	0.0888	0.0793	*0.0636	*0.0537	0.048	*0.0419	0.0382
	V-C-BN-text	***0.1009**	***0.0815**	***0.0728**	***0.0593**	***0.0502**	***0.0446**	***0.0397**	***0.0362**
	change vs. baseline	-0.32%	-0.58%	-0.4%	-0.75%	-0.6%	-0.45%	-0.34%	-0.21%

Table 7.13: Average precision (↑↑) for the predictions of our models using the ground truth Bayesian network (**GT-only**) over $\mathcal{X}_{test}$, averaged over 20 seeds for various training sizes n. The best model per training size and per symptom is highlighted in **bold**. The best baseline model for each class is underlined. Cases where a model outperforms the best baseline model significantly are indicated by ***** ($p < 0.05$ in a one-sided Wilcoxon signed-rank test over 20 seeds).

		Training size n							
		100	**187**	**350**	**654**	**1223**	**2287**	**4278**	**8000**
dysp	**GT-only**	0.7996	0.7996	0.7996	0.7996	0.7996	0.7996	0.7996	0.7996
	text-only	0.9246	0.9482	0.9578	0.967	0.9731	0.9798	0.9831	0.9878
	Concat-tab-text	0.9127	0.9398	0.9533	0.9657	0.9737	0.9801	0.9841	0.987
	C-GT-text	*0.9292	*0.9499	*0.9584	0.9676	0.974	0.9786	0.982	0.9875
	V-GT-text	0.9283	*0.9514	0.9572	0.9677	0.9742	0.9805	0.984	*0.9881
	V-C-GT-text	***0.9313**	***0.9533**	***0.9606**	***0.97**	***0.9757**	***0.9817**	***0.9851**	***0.989**
	change vs. baseline	+0.67%	+0.5%	+0.29%	+0.3%	+0.2%	+0.16%	+0.1%	+0.12%
cough	**GT-only**	0.797	0.797	0.797	0.797	0.797	0.797	0.797	0.797
	text-only	0.902	0.9452	0.9571	0.9689	0.978	0.9826	0.9863	0.989
	Concat-tab-text	0.8866	0.9393	0.9582	0.9705	0.9794	0.9851	0.9886	0.9908
	C-GT-text	*0.9175	*0.9501	*0.9612	0.9704	0.9776	0.9816	0.986	0.9882
	V-GT-text	***0.9279**	***0.9575**	*0.9658	***0.9754**	***0.9831**	***0.9868**	***0.9898**	***0.9919**
	V-C-GT-text	*0.9269	*0.9571	***0.9663**	*0.9752	*0.9823	*0.9859	0.9891	*0.9914
	change vs. baseline	+2.59%	+1.24%	+0.81%	+0.49%	+0.37%	+0.17%	+0.12%	+0.11%
pain	**GT-only**	0.3603	0.3603	0.3603	0.3603	0.3603	0.3603	0.3603	0.3603
	text-only	0.6181	0.7252	0.7648	0.804	0.825	0.8377	0.8471	0.8608
	Concat-tab-text	0.5091	0.6537	0.7422	0.7858	0.8183	0.8412	0.8559	0.868
	C-GT-text	*0.6259	***0.7328**	***0.7739**	*0.8121	*0.8322	0.8431	0.852	0.8651
	V-GT-text	0.6119	*0.7308	0.7628	***0.8215**	***0.8473**	***0.8601**	***0.8699**	***0.8825**
	V-C-GT-text	***0.6296**	*0.7326	*0.7708	*0.8149	*0.8375	*0.8508	*0.8602	*0.8734
	change vs. baseline	+1.15%	+0.75%	+0.92%	+1.76%	+2.23%	+1.89%	+1.4%	+1.45%
nasal	**GT-only**	0.6477	0.6477	0.6477	0.6477	0.6477	0.6477	0.6477	0.6477
	text-only	0.9512	0.9584	0.9657	0.9706	0.9743	0.9779	0.98	0.9816
	Concat-tab-text	0.9444	0.9579	0.9666	0.973	0.9777	0.9818	0.9847	0.9869
	C-GT-text	***0.9559**	*0.9625	0.9689	*0.9771	*0.9799	0.9817	0.9842	0.9847
	V-GT-text	*0.9538	***0.9644**	***0.9715**	***0.9785**	***0.983**	***0.9861**	***0.9879**	***0.9885**
	V-C-GT-text	*0.9548	*0.9632	*0.9703	*0.9758	*0.9799	*0.9836	*0.9857	0.987
	change vs. baseline	+0.47%	+0.6%	+0.49%	+0.55%	+0.53%	+0.43%	+0.31%	+0.16%
fever	**GT-only**	0.5465	0.5465	0.5465	0.5465	0.5465	0.5465	0.5465	0.5465
	text-only	0.6905	0.7501	0.7973	0.8646	0.8986	0.9164	0.9315	0.9393
	Concat-tab-text	0.6605	0.7495	0.7939	0.8526	0.8951	0.922	0.9381	0.9501
	C-GT-text	0.681	0.7536	*0.803	*0.8711	*0.9018	0.9199	0.9344	0.9434
	V-GT-text	***0.7149**	***0.7869**	***0.8224**	***0.8861**	***0.9154**	***0.9313**	***0.9476**	***0.9563**
	V-C-GT-text	0.6936	*0.7756	*0.8179	*0.8814	*0.9103	*0.9266	*0.942	*0.9515
	change vs. baseline	+2.44%	+3.68%	+2.52%	+2.15%	+1.68%	+0.93%	+0.95%	+0.62%
mean	**GT-only**	0.6302	0.6302	0.6302	0.6302	0.6302	0.6302	0.6302	0.6302
	text-only	0.8173	0.8654	0.8885	0.915	0.9298	0.9389	0.9456	0.9517
	Concat-tab-text	0.7827	0.848	0.8828	0.9095	0.9289	0.942	0.9503	0.9566
	C-GT-text	*0.8219	*0.8698	*0.8931	*0.9197	*0.9331	0.941	0.9477	0.9538
	V-GT-text	***0.8273**	***0.8782**	*0.8959	***0.9258**	***0.9406**	***0.949**	***0.9558**	***0.9615**
	V-C-GT-text	*0.8272	*0.8764	***0.8972**	*0.9234	*0.9372	*0.9457	*0.9524	*0.9585
	change vs. baseline	+1.01%	+1.28%	+0.87%	+1.08%	+1.08%	+0.69%	+0.55%	+0.49%

Table 7.14: Brier scores (⇓) for the predictions of our models using the ground truth Bayesian network (**GT-only**) over $\mathcal{X}_{test}$, averaged over 20 seeds for various training sizes n. The best model per training size and per symptom is highlighted in **bold**. The best baseline model for each class is underlined. Cases where a model outperforms the best baseline model significantly are indicated by ***** ($p < 0.05$ in a one-sided Wilcoxon signed-rank test over 20 seeds).

		Training size n							
		100	**187**	**350**	**654**	**1223**	**2287**	**4278**	**8000**
dysp	**GT-only**	0.0761	0.0761	0.0761	0.0761	0.0761	0.0761	0.0761	0.0761
	text-only	0.0485	0.0365	0.0314	0.0277	0.023	0.019	0.0172	0.0128
	Concat-tab-text	0.0508	0.0396	0.0339	0.0288	0.0251	0.0196	0.0172	0.0145
	C-GT-text	0.0508	0.0365	0.0322	0.0274	0.023	0.0192	0.0175	0.0132
	V-GT-text	0.0468	*0.0352	0.0324	0.0287	0.0248	0.02	0.0183	0.0148
	V-C-GT-text	***0.0458**	***0.0337**	***0.0298**	***0.0254**	***0.0218**	***0.0178**	***0.0157**	**0.0124**
	change vs. baseline	-0.27%	-0.29%	-0.16%	-0.23%	-0.12%	-0.13%	-0.15%	-0.03%
cough	**GT-only**	0.1146	0.1146	0.1146	0.1146	0.1146	0.1146	0.1146	0.1146
	text-only	0.0882	0.0659	0.0576	0.0494	0.0401	0.0337	0.027	0.0247
	Concat-tab-text	0.0907	0.0659	0.0553	0.0462	0.0398	0.0327	0.0264	0.0237
	C-GT-text	*0.085	*0.064	0.0571	0.0486	0.0388	0.0328	0.0268	0.0244
	V-GT-text	***0.0727**	***0.0555**	*0.0496	*0.042	*0.0341	*0.0292	*0.0239	0.0224
	V-C-GT-text	*0.0742	*0.0558	***0.049**	***0.0408**	***0.0324**	***0.0272**	***0.0228**	***0.0202**
	change vs. baseline	-1.55%	-1.04%	-0.62%	-0.54%	-0.74%	-0.55%	-0.37%	-0.35%
pain	**GT-only**	0.1066	0.1066	0.1066	0.1066	0.1066	0.1066	0.1066	0.1066
	text-only	**0.0854**	0.0711	**0.0628**	0.0553	0.0491	0.0451	0.0426	0.0383
	Concat-tab-text	0.0942	0.0789	0.0666	0.0623	0.0524	0.0478	0.0436	0.0393
	C-GT-text	0.0919	0.0736	0.0702	0.0604	0.0536	0.0501	0.047	0.0423
	V-GT-text	0.0952	0.0767	0.0705	0.0573	0.0508	0.0457	0.0427	0.0384
	V-C-GT-text	0.0866	***0.0699**	0.0644	**0.0538**	**0.0482**	**0.0444**	**0.0414**	***0.0377**
	change vs. baseline	+0.13%	-0.12%	+0.16%	-0.14%	-0.09%	-0.07%	-0.12%	-0.07%
nasal	**GT-only**	0.1167	0.1167	0.1167	0.1167	0.1167	0.1167	0.1167	0.1167
	text-only	0.0424	0.0373	0.0315	0.0274	0.0238	0.0221	0.019	0.0181
	Concat-tab-text	0.0513	0.0417	0.035	0.0309	0.0262	0.0224	0.02	0.0177
	C-GT-text	0.0441	*0.037	*0.0312	*0.0268	*0.0235	*0.0218	0.0189	0.018
	V-GT-text	0.0417	***0.0331**	*0.0281	*0.0255	*0.0222	*0.0201	*0.0179	*0.0164
	V-C-GT-text	***0.039**	*0.0335	***0.0278**	***0.0245**	***0.0214**	***0.019**	***0.0173**	***0.0164**
	change vs. baseline	-0.35%	-0.42%	-0.37%	-0.29%	-0.25%	-0.31%	-0.17%	-0.14%
fever	**GT-only**	0.3006	0.3006	0.3006	0.3006	0.3006	0.3006	0.3006	0.3006
	text-only	0.256	0.2257	0.201	0.1744	0.1448	0.1256	0.1126	0.0984
	Concat-tab-text	0.2632	0.2208	0.1978	0.1738	0.1449	0.1243	0.1081	0.0962
	C-GT-text	0.268	0.2253	0.2088	0.1709	0.1441	0.1261	0.1144	0.1038
	V-GT-text	0.2766	0.2241	0.2054	*0.1589	*0.1338	0.124	0.1065	0.0985
	V-C-GT-text	***0.2503**	***0.2092**	***0.1897**	***0.1505**	***0.1266**	***0.1148**	***0.1015**	**0.0942**
	change vs. baseline	-0.58%	-1.16%	-0.81%	-2.34%	-1.81%	-0.95%	-0.67%	-0.21%
mean	**GT-only**	0.1429	0.1429	0.1429	0.1429	0.1429	0.1429	0.1429	0.1429
	text-only	0.1041	0.0873	0.0768	0.0668	0.0562	0.0491	0.0437	0.0384
	Concat-tab-text	0.11	0.0894	0.0777	0.0684	0.0577	0.0494	0.0431	0.0383
	C-GT-text	0.108	0.0873	0.0799	0.0668	0.0566	0.05	0.0449	0.0404
	V-GT-text	0.1066	*0.0849	0.0772	*0.0625	*0.0531	*0.0478	*0.0419	0.0381
	V-C-GT-text	***0.0992**	***0.0804**	***0.0721**	***0.059**	***0.0501**	***0.0446**	***0.0397**	***0.0362**
	change vs. baseline	-0.49%	-0.69%	-0.47%	-0.78%	-0.61%	-0.45%	-0.33%	-0.21%

Table 7.15: Brier scores (⇓) for our models on the *present, mentioned* subset across various training sizes. The best model per training size and per symptom is highlighted in **bold**. Cases where a model outperforms **text-only** significantly are indicated by ***** ($p < 0.05$ in a one-sided Wilcoxon signed-rank test over 20 seeds).

		Training size n							
		100	**187**	**350**	**654**	**1223**	**2287**	**4278**	**8000**
dysp	**text-only**	0.1174	**0.08**	**0.0719**	0.0648	**0.0562**	**0.0369**	**0.0355**	**0.0283**
	C-BN-text	0.1411	0.0912	0.083	0.0695	0.0592	0.0431	0.0419	0.0309
	V-BN-text	0.1477	0.1031	0.0931	0.0831	0.0719	0.0487	0.0465	0.0364
	V-C-BN-text	**0.115**	0.0807	0.0742	**0.064**	0.0567	0.0433	0.04	0.0308
	change vs. baseline	-0.24%	+0.07%	+0.23%	-0.08%	+0.05%	+0.62%	+0.45%	+0.25%
cough	**text-only**	0.1153	0.0659	0.0592	0.0402	0.0283	0.0182	0.017	0.0098
	C-BN-text	0.1259	0.0733	0.0634	0.0473	0.0328	0.0217	0.0183	0.0135
	V-BN-text	0.1228	0.0668	0.0563	***0.0365**	***0.0243**	***0.0145**	***0.0139**	**0.0088**
	V-C-BN-text	***0.1031**	**0.0627**	***0.0521**	0.0377	0.025	0.0161	0.0147	0.0098
	change vs. baseline	-1.22%	-0.32%	-0.71%	-0.38%	-0.4%	-0.37%	-0.31%	-0.1%
pain	**text-only**	**0.3809**	**0.2508**	**0.1941**	**0.1393**	**0.1252**	**0.103**	**0.0909**	**0.0706**
	C-BN-text	0.4679	0.3097	0.3086	0.2337	0.1912	0.1696	0.1489	0.1162
	V-BN-text	0.5569	0.401	0.3573	0.2387	0.1921	0.15	0.1247	0.0931
	V-C-BN-text	0.4157	0.28	0.2625	0.1915	0.1608	0.1354	0.1147	0.0885
	change vs. baseline	+3.48%	+2.91%	+6.84%	+5.22%	+3.56%	+3.24%	+2.38%	+1.79%
nasal	**text-only**	**0.0521**	**0.0349**	**0.0205**	0.019	**0.0124**	**0.0065**	**0.0057**	0.004
	C-BN-text	0.0681	0.041	0.0281	0.0222	0.0163	0.0112	0.0086	0.007
	V-BN-text	0.0924	0.0501	0.0328	0.0287	0.0178	0.009	0.0067	**0.0038**
	V-C-BN-text	0.054	0.037	0.0234	**0.0189**	0.014	0.0083	0.0067	0.0046
	change vs. baseline	+0.19%	+0.21%	+0.29%	-0.01%	+0.16%	+0.18%	+0.09%	-0.03%
fever	**text-only**	**0.7589**	**0.5334**	**0.4449**	**0.2503**	**0.1718**	**0.155**	**0.0998**	**0.0722**
	C-BN-text	0.8994	0.6497	0.5861	0.3683	0.2582	0.2037	0.1579	0.1223
	V-BN-text	1.0834	0.7634	0.6551	0.3741	0.2486	0.2152	0.1294	0.0905
	V-C-BN-text	0.8151	0.5772	0.4998	0.2932	0.1965	0.1643	0.1076	0.0774
	change vs. baseline	+5.62%	+4.38%	+5.49%	+4.29%	+2.47%	+0.93%	+0.78%	+0.52%
mean	**text-only**	**0.2849**	**0.193**	**0.1581**	**0.1027**	**0.0788**	**0.0639**	**0.0498**	**0.037**
	C-BN-text	0.3405	0.233	0.2139	0.1482	0.1115	0.0898	0.0751	0.058
	V-BN-text	0.4007	0.2769	0.2389	0.1522	0.1109	0.0875	0.0642	0.0465
	V-C-BN-text	0.3006	0.2075	0.1824	0.1211	0.0906	0.0735	0.0567	0.0422
	change vs. baseline	+1.57%	+1.45%	+2.43%	+1.83%	+1.18%	+0.96%	+0.69%	+0.52%

Table 7.16: Brier scores (⇓⇓) for our models on the *present, not mentioned* subset across various training sizes. The best model per training size and per symptom is highlighted in **bold**. Cases where a model outperforms **text-only** significantly are indicated by ***** ($p < 0.05$ in a one-sided Wilcoxon signed-rank test over 20 seeds).

		Training size n							
		100	**187**	**350**	**654**	**1223**	**2287**	**4278**	**8000**
dysp	**text-only**	0.8879	0.8893	0.8958	0.9192	0.8972	0.8517	0.8501	0.847
	C-BN-text	*0.7689	*0.7967	*0.7684	*0.804	*0.7996	*0.7884	*0.7733	*0.7704
	V-BN-text	***0.7117**	***0.709**	***0.7026**	***0.7335**	***0.7047**	***0.6328**	***0.6443**	***0.6146**
	V-C-BN-text	*0.7241	*0.7441	*0.7389	*0.7826	*0.7559	*0.7284	*0.7318	*0.7038
	change vs. baseline	-17.62%	-18.03%	-19.31%	-18.57%	-19.25%	-21.89%	-20.58%	-23.24%
cough	**text-only**	0.5749	0.642	0.6555	0.644	0.66	0.6452	0.6715	0.6648
	C-BN-text	***0.4496**	*0.5267	***0.5064**	***0.535**	*0.56	*0.561	*0.5787	*0.5909
	V-BN-text	*0.4654	***0.5249**	*0.5272	*0.5376	***0.5587**	***0.544**	***0.5533**	***0.5292**
	V-C-BN-text	*0.4714	*0.5556	*0.5518	*0.5751	*0.5946	*0.5921	*0.6008	*0.6111
	change vs. baseline	-12.53%	-11.71%	-14.91%	-10.89%	-10.13%	-10.12%	-11.81%	-13.56%
pain	**text-only**	0.7786	0.8574	0.8387	0.8591	0.8822	0.8734	0.8711	0.8674
	C-BN-text	*0.7251	***0.8035**	***0.7825**	***0.8102**	***0.8236**	***0.8236**	***0.8206**	***0.8178**
	V-BN-text	0.8285	0.8962	0.8896	0.9142	0.9304	0.922	0.9188	0.9092
	V-C-BN-text	***0.724**	*0.8182	*0.8038	0.8442	*0.863	0.8695	0.8682	0.8679
	change vs. baseline	-5.46%	-5.39%	-5.62%	-4.89%	-5.86%	-4.98%	-5.05%	-4.96%
nasal	**text-only**	0.8839	0.9129	0.9198	0.9324	0.9298	0.899	0.9087	0.8948
	C-BN-text	***0.7955**	***0.8495**	***0.8482**	***0.8553**	***0.8609**	*0.8463	*0.8503	*0.8434
	V-BN-text	*0.8381	*0.8655	*0.8625	*0.8741	*0.8785	***0.8297**	***0.8262**	***0.809**
	V-C-BN-text	*0.8217	*0.8654	*0.8654	*0.8763	*0.8915	*0.8813	*0.8843	*0.885
	change vs. baseline	-8.84%	-6.34%	-7.15%	-7.71%	-6.89%	-6.93%	-8.25%	-8.57%
fever	**text-only**	1.6366	1.5965	1.5683	1.4084	1.39	1.3764	1.306	1.3218
	C-BN-text	***1.4653**	***1.4837**	***1.4503**	**1.3786**	***1.3556**	***1.3157**	**1.2763**	***1.277**
	V-BN-text	1.7423	1.7089	1.701	1.5679	1.5602	1.5568	1.5109	1.5409
	V-C-BN-text	*1.511	*1.5258	*1.4959	1.4464	1.4325	1.3999	1.3855	1.4088
	change vs. baseline	-17.14%	-11.28%	-11.8%	-2.98%	-3.44%	-6.07%	-2.96%	-4.48%
mean	**text-only**	0.9524	0.9796	0.9756	0.9526	0.9518	0.9291	0.9215	0.9191
	C-BN-text	***0.8409**	***0.892**	***0.8712**	***0.8766**	***0.8799**	***0.867**	***0.8598**	***0.8599**
	V-BN-text	*0.9172	*0.9409	*0.9366	*0.9255	*0.9265	*0.8971	*0.8907	*0.8806
	V-C-BN-text	*0.8504	*0.9018	*0.8912	*0.9049	*0.9075	*0.8942	*0.8941	*0.8953
	change vs. baseline	-11.15%	-8.76%	-10.44%	-7.6%	-7.19%	-6.21%	-6.16%	-5.92%

Table 7.17: Brier scores (⇓⇓) for our models on the *not present, mentioned* subset across various training sizes. The best model per training size and per symptom is highlighted in **bold**. Cases where a model outperforms **text-only** significantly are indicated by ***** ($p < 0.05$ in a one-sided Wilcoxon signed-rank test over 20 seeds).

		Training size n							
		100	**187**	**350**	**654**	**1223**	**2287**	**4278**	**8000**
dysp	**text-only**	0.0358	0.028	0.0227	0.0188	0.0137	0.0135	0.0105	0.0053
	C-BN-text	*0.0339	*0.0253	*0.0215	0.0176	0.0134	*0.0124	0.0096	0.0057
	V-BN-text	***0.0302**	***0.0243**	***0.0211**	*0.0167	0.0137	0.0134	0.0105	0.0076
	V-C-BN-text	0.0352	*0.0258	*0.0213	**0.0166**	**0.0128**	***0.0105**	***0.0076**	**0.005**
	change vs. baseline	-0.56%	-0.37%	-0.16%	-0.22%	-0.09%	-0.3%	-0.28%	-0.03%
cough	**text-only**	0.0929	0.0792	0.0656	0.0612	0.0473	0.0393	0.0239	0.0222
	C-BN-text	*0.0819	*0.0725	0.0648	0.0571	0.044	0.0374	0.0253	0.0217
	V-BN-text	***0.0771**	*0.0666	*0.0588	*0.0538	*0.0413	*0.0362	0.0241	0.0242
	V-C-BN-text	*0.0803	***0.064**	***0.056**	***0.0469**	***0.0348**	***0.0287**	***0.0192**	***0.0158**
	change vs. baseline	-1.58%	-1.52%	-0.96%	-1.43%	-1.25%	-1.05%	-0.47%	-0.64%
pain	**text-only**	0.0293	0.032	0.0305	0.0275	0.0169	0.0127	0.0103	0.0066
	C-BN-text	*0.0246	*0.0253	*0.0208	*0.0184	*0.0134	*0.0107	0.0089	0.007
	V-BN-text	***0.0138**	***0.0144**	***0.0112**	***0.009**	***0.0055**	***0.0043**	***0.0035**	***0.0025**
	V-C-BN-text	*0.0264	*0.0258	*0.0203	*0.0145	*0.009	*0.0063	*0.0051	*0.0036
	change vs. baseline	-1.55%	-1.76%	-1.94%	-1.85%	-1.13%	-0.84%	-0.67%	-0.41%
nasal	**text-only**	0.0584	0.0587	0.0514	0.0371	0.0299	0.0247	0.0155	0.0141
	C-BN-text	*0.0517	*0.0528	*0.0445	*0.0336	*0.0269	*0.0215	*0.0144	*0.0129
	V-BN-text	***0.0317**	***0.0323**	***0.0281**	***0.0222**	***0.0186**	*0.0172	*0.0123	*0.011
	V-C-BN-text	*0.0465	*0.043	*0.0354	*0.0268	*0.0197	***0.0144**	***0.0099**	***0.009**
	change vs. baseline	-2.66%	-2.63%	-2.33%	-1.49%	-1.13%	-1.03%	-0.56%	-0.52%
fever	**text-only**	0.0929	0.1232	0.1025	0.1117	0.0691	0.0368	0.0318	0.019
	C-BN-text	*0.0728	*0.0821	*0.0697	*0.0678	*0.0467	*0.0307	*0.0244	0.0183
	V-BN-text	***0.0278**	***0.0361**	***0.0263**	***0.0329**	***0.0195**	***0.01**	***0.0086**	***0.0057**
	V-C-BN-text	*0.0724	*0.078	*0.0623	*0.0507	*0.0301	*0.0193	*0.0142	*0.0107
	change vs. baseline	-6.51%	-8.71%	-7.62%	-7.88%	-4.96%	-2.68%	-2.32%	-1.34%
mean	**text-only**	0.0619	0.0642	0.0545	0.0513	0.0354	0.0254	0.0184	0.0135
	C-BN-text	*0.053	*0.0516	*0.0443	*0.0389	*0.0289	*0.0225	*0.0165	0.0131
	V-BN-text	***0.0361**	***0.0347**	***0.0291**	***0.0269**	***0.0197**	*0.0162	*0.0118	*0.0102
	V-C-BN-text	*0.0522	*0.0473	*0.0391	*0.0311	*0.0213	***0.0159**	***0.0112**	***0.0088**
	change vs. baseline	-2.57%	-2.95%	-2.55%	-2.43%	-1.57%	-0.95%	-0.72%	-0.47%

Table 7.18: Brier scores (⇓⇓) for our models on the *not present, not mentioned* subset across various training sizes. The best model per training size and per symptom is highlighted in **bold**. Cases where a model outperforms **text-only** significantly are indicated by ***** ($p < 0.05$ in a one-sided Wilcoxon signed-rank test over 20 seeds).

		Training size n							
		100	**187**	**350**	**654**	**1223**	**2287**	**4278**	**8000**
dysp	**text-only**	0.0107	0.0091	0.0061	0.0049	0.0042	0.0044	0.0047	0.0039
	C-BN-text	0.0111	*0.0082	0.0061	0.0048	0.0042	*0.004	0.0042	0.0039
	V-BN-text	***0.0074**	***0.0057**	***0.0039**	***0.003**	***0.0027**	*0.0028	*0.0031	*0.0034
	V-C-BN-text	0.0101	*0.0073	*0.0049	*0.0038	*0.0033	***0.0027**	***0.0031**	***0.0032**
	change vs. baseline	-0.33%	-0.34%	-0.22%	-0.19%	-0.14%	-0.17%	-0.16%	-0.07%
cough	**text-only**	**0.0158**	**0.0085**	**0.0058**	**0.0054**	0.0057	0.0074	0.0061	0.0093
	C-BN-text	0.018	0.0106	0.0093	0.007	0.0065	0.0076	0.0068	0.0087
	V-BN-text	0.0186	0.0121	0.0098	0.0075	0.0062	0.007	*0.0055	*0.0082
	V-C-BN-text	0.0173	0.0097	0.0078	0.0057	**0.005**	***0.0053**	***0.0045**	***0.0058**
	change vs. baseline	+0.15%	+0.12%	+0.2%	+0.03%	-0.07%	-0.21%	-0.16%	-0.35%
pain	**text-only**	0.021	0.0164	0.0147	0.0131	0.0097	0.0098	0.0094	0.0081
	C-BN-text	0.0196	*0.015	*0.013	0.0106	0.0091	0.0089	0.0086	0.0082
	V-BN-text	***0.011**	***0.0083**	***0.0059**	***0.0047**	***0.0037**	***0.0041**	***0.0039**	***0.0039**
	V-C-BN-text	0.0205	*0.0147	*0.0117	*0.0081	*0.0063	*0.0058	*0.0053	*0.0049
	change vs. baseline	-1.01%	-0.81%	-0.88%	-0.84%	-0.6%	-0.57%	-0.55%	-0.42%
nasal	**text-only**	0.0112	0.0086	0.0066	0.0044	0.0033	0.0053	0.0028	0.0027
	C-BN-text	0.0118	0.009	0.0068	0.0049	0.0038	0.0051	0.0033	0.0031
	V-BN-text	***0.0083**	***0.0068**	0.006	0.0044	0.0034	0.0049	0.0038	0.0032
	V-C-BN-text	0.0106	*0.0074	**0.0056**	**0.004**	**0.0026**	***0.0028**	***0.0018**	***0.0014**
	change vs. baseline	-0.29%	-0.18%	-0.1%	-0.04%	-0.07%	-0.25%	-0.1%	-0.13%
fever	**text-only**	0.0289	0.0293	0.0265	0.0527	0.05	0.0388	0.0427	0.029
	C-BN-text	0.0367	0.0271	0.0278	*0.0356	*0.0359	*0.0328	*0.0331	0.0276
	V-BN-text	***0.0096**	***0.0088**	***0.0072**	***0.0163**	***0.017**	***0.0122**	***0.0121**	***0.0076**
	V-C-BN-text	0.0311	*0.0223	*0.0213	*0.0249	*0.0246	*0.0211	*0.0186	*0.0138
	change vs. baseline	-1.92%	-2.05%	-1.93%	-3.64%	-3.3%	-2.66%	-3.06%	-2.15%
mean	**text-only**	0.0175	0.0144	0.0119	0.0161	0.0146	0.0131	0.0131	0.0106
	C-BN-text	0.0194	0.014	0.0126	*0.0126	*0.0119	*0.0117	0.0112	0.0103
	V-BN-text	***0.011**	***0.0084**	***0.0065**	***0.0072**	***0.0066**	***0.0062**	***0.0057**	***0.0053**
	V-C-BN-text	0.0179	*0.0123	*0.0103	*0.0093	*0.0083	*0.0075	*0.0067	*0.0058
	change vs. baseline	-0.65%	-0.6%	-0.54%	-0.89%	-0.8%	-0.7%	-0.75%	-0.53%

Table 7.19: The confidence of the **text-only** model, as measured by 1 - the normalized Shannon entropy (7.9) on $\mathcal{X}_{test}$ over 20 seeds.

		Training size n							
		100	**187**	**350**	**654**	**1223**	**2287**	**4278**	**8000**
dysp	**text-only**	0.7987	0.8466	0.8499	0.8794	0.8998	0.9025	0.9022	0.9237
cough	**text-only**	0.6497	0.7263	0.7333	0.7833	0.8341	0.8637	0.8811	0.8914
pain	**text-only**	0.6589	0.7571	0.7176	0.7494	0.7833	0.7898	0.8096	0.8279
nasal	**text-only**	0.8164	0.872	0.876	0.8963	0.903	0.9058	0.9189	0.9243
fever	**text-only**	0.6781	0.7484	0.7475	0.7855	0.8196	0.8534	0.8585	0.868

Table 7.20: Average precision (↑↑) for the predictions of our models over the test set $\mathcal{X}^*_{test}$ containing **manipulated text notes**. The best model per training size and per symptom is highlighted in **bold**. Cases where a model outperforms **text-only** significantly are indicated by ***** ($p < 0.05$ in a one-sided Wilcoxon signed-rank test over 20 seeds).

		Training size n							
		100	**187**	**350**	**654**	**1223**	**2287**	**4278**	**8000**
dysp	**text-only**	0.8762	0.9068	0.9206	0.9342	0.943	0.9519	0.9552	0.961
	C-BN-text	*0.8891	*0.9122	*0.9245	*0.9366	*0.9455	0.9515	0.9551	0.9616
	V-BN-text	*0.8905	*0.9114	0.9204	*0.94	***0.948**	*0.9553	*0.9587	*0.9643
	V-C-BN-text	***0.8915**	***0.9172**	***0.9278**	***0.9404**	*0.9478	***0.9555**	***0.9592**	***0.9645**
	change vs. baseline	+1.53%	+1.04%	+0.72%	+0.62%	+0.5%	+0.36%	+0.41%	+0.35%
cough	**text-only**	0.825	0.8651	0.881	0.8992	0.9144	0.9251	0.9328	0.9345
	C-BN-text	*0.8515	*0.8847	*0.902	*0.9146	*0.9274	*0.9353	*0.9428	*0.9431
	V-BN-text	*0.8574	***0.8942**	***0.9106**	***0.9254**	***0.938**	***0.9451**	***0.9519**	***0.955**
	V-C-BN-text	***0.8586**	*0.8928	*0.9071	*0.9193	*0.9313	*0.9394	*0.9471	*0.95
	change vs. baseline	+3.35%	+2.92%	+2.97%	+2.63%	+2.36%	+2.0%	+1.91%	+2.05%
pain	**text-only**	0.5357	0.6195	0.6575	0.6946	0.722	0.7349	0.7422	0.7516
	C-BN-text	***0.5467**	***0.6262**	***0.669**	*0.7031	*0.7285	*0.7404	*0.7493	*0.7595
	V-BN-text	0.5176	0.6106	0.6567	***0.714**	***0.7473**	***0.764**	***0.775**	***0.7868**
	V-C-BN-text	0.538	0.6228	*0.6632	*0.7065	*0.737	*0.7528	*0.7613	*0.7723
	change vs. baseline	+1.1%	+0.67%	+1.15%	+1.94%	+2.53%	+2.91%	+3.27%	+3.52%
nasal	**text-only**	0.8788	0.8869	0.9013	0.9072	0.9117	0.9097	0.9101	0.9034
	C-BN-text	*0.9032	*0.9073	*0.9205	*0.927	*0.9308	*0.9263	*0.9288	*0.9238
	V-BN-text	***0.9057**	***0.9116**	***0.9229**	***0.9307**	***0.9369**	***0.9371**	***0.9388**	***0.9337**
	V-C-BN-text	*0.8986	*0.9052	*0.919	*0.9238	*0.9288	*0.928	*0.9296	*0.9231
	change vs. baseline	+2.69%	+2.47%	+2.16%	+2.35%	+2.52%	+2.74%	+2.87%	+3.03%
fever	**text-only**	0.6023	0.6524	0.6875	0.7361	0.7714	0.7909	0.8072	0.8161
	C-BN-text	0.5975	*0.6562	*0.6943	*0.7424	*0.7761	*0.7981	*0.8132	*0.8243
	V-BN-text	**0.6047**	***0.6782**	***0.7202**	***0.7746**	***0.8086**	***0.8269**	***0.8439**	***0.8543**
	V-C-BN-text	0.6034	*0.6671	*0.7044	*0.7572	*0.7915	*0.8104	*0.829	*0.839
	change vs. baseline	+0.23%	+2.58%	+3.27%	+3.86%	+3.72%	+3.6%	+3.67%	+3.83%
mean	**text-only**	0.7436	0.7861	0.8096	0.8342	0.8525	0.8625	0.8695	0.8733
	C-BN-text	*0.7576	*0.7973	*0.822	*0.8447	*0.8617	*0.8703	*0.8778	*0.8825
	V-BN-text	*0.7552	***0.8012**	***0.8262**	***0.857**	***0.8758**	***0.8857**	***0.8937**	***0.8988**
	V-C-BN-text	***0.758**	*0.801	*0.8243	*0.8495	*0.8673	*0.8772	*0.8853	*0.8898
	change vs. baseline	+1.44%	+1.51%	+1.66%	+2.27%	+2.32%	+2.32%	+2.42%	+2.55%

Table 7.21: Brier scores (⇊) for the predictions of our models over the test set $\mathcal{X}^*_{test}$ containing **manipulated text notes**. The best model per training size and per symptom is highlighted in **bold**. Cases where a model outperforms **text-only** significantly are indicated by * ($p < 0.05$ in a one-sided Wilcoxon signed-rank test over 20 seeds).

		Training size n							
		100	**187**	**350**	**654**	**1223**	**2287**	**4278**	**8000**
dysp	**text-only**	0.062	0.051	0.0452	0.0419	0.0377	0.0337	0.032	0.028
	C-BN-text	0.063	*0.0497	*0.0443	*0.0401	*0.0365	*0.0326	*0.031	*0.0275
	V-BN-text	0.0606	*0.0492	0.0441	0.0406	0.0373	0.0336	0.0326	0.0305
	V-C-BN-text	***0.058**	***0.0469**	***0.0419**	***0.038**	***0.0349**	***0.0312**	***0.0296**	***0.0274**
	change vs. baseline	-0.4%	-0.41%	-0.34%	-0.39%	-0.28%	-0.25%	-0.24%	-0.05%
cough	**text-only**	0.137	0.1212	0.1141	0.105	0.0973	0.0889	0.087	0.0847
	C-BN-text	*0.12	*0.1084	*0.0992	*0.094	*0.0872	*0.081	*0.0786	*0.0778
	V-BN-text	*0.1213	***0.1039**	***0.0958**	***0.0886**	***0.0818**	***0.0754**	***0.0721**	***0.0694**
	V-C-BN-text	***0.1173**	*0.105	*0.0965	*0.091	*0.0838	*0.0783	*0.0756	*0.0743
	change vs. baseline	-1.97%	-1.73%	-1.82%	-1.64%	-1.55%	-1.35%	-1.49%	-1.53%
pain	**text-only**	0.0955	0.0879	0.0793	0.0753	0.0698	0.0665	0.0657	0.0621
	C-BN-text	0.0982	0.0867	0.082	0.0754	0.0703	0.0679	0.0662	0.0628
	V-BN-text	0.1043	0.0933	0.0869	0.0773	0.0733	0.0694	0.067	0.0639
	V-C-BN-text	**0.0949**	***0.0853**	**0.0789**	***0.072**	**0.0683**	**0.0661**	**0.0641**	**0.0618**
	change vs. baseline	-0.05%	-0.26%	-0.04%	-0.33%	-0.15%	-0.04%	-0.16%	-0.03%
nasal	**text-only**	0.0905	0.0847	0.0767	0.0751	0.0708	0.0682	0.0677	0.0688
	C-BN-text	*0.0869	*0.0817	*0.0737	*0.0713	***0.0678**	***0.0657**	***0.065**	***0.0661**
	V-BN-text	0.0915	0.0832	0.0761	0.0758	0.0712	0.0679	0.0673	*0.0668
	V-C-BN-text	***0.0834**	***0.0798**	***0.0727**	***0.071**	*0.069	0.068	0.0683	0.0694
	change vs. baseline	-0.71%	-0.48%	-0.4%	-0.42%	-0.3%	-0.24%	-0.27%	-0.27%
fever	**text-only**	0.3071	0.2887	0.2695	0.2613	0.2388	0.2231	0.2134	0.2041
	C-BN-text	0.3029	*0.2743	*0.2612	*0.2401	*0.2222	*0.2102	*0.2012	*0.196
	V-BN-text	0.3233	0.2878	0.2736	*0.2437	*0.2251	0.2171	0.2051	0.2036
	V-C-BN-text	***0.294**	***0.2671**	***0.2515**	***0.2306**	***0.2131**	***0.2034**	***0.1949**	***0.1929**
	change vs. baseline	-1.31%	-2.16%	-1.8%	-3.07%	-2.57%	-1.97%	-1.85%	-1.12%
mean	**text-only**	0.1384	0.1267	0.117	0.1117	0.1028	0.0961	0.0932	0.0895
	C-BN-text	*0.1342	*0.1202	*0.1121	*0.1042	*0.0968	*0.0915	*0.0884	*0.0861
	V-BN-text	0.1402	*0.1235	*0.1153	*0.1052	*0.0977	*0.0927	*0.0888	*0.0868
	V-C-BN-text	***0.1295**	***0.1168**	***0.1083**	***0.1005**	***0.0938**	***0.0894**	***0.0865**	***0.0852**
	change vs. baseline	-0.89%	-0.99%	-0.87%	-1.12%	-0.9%	-0.67%	-0.67%	-0.44%

8

Modeling Clinical Uncertainty in Radiology Reports: from Explicit Uncertainty Markers to Implicit Reasoning Pathways

This chapter addresses the challenge of modeling both implicit and explicit uncertainty in clinical text. Instead of requiring experts to specify a complete Bayesian network, we relax the requirements and ask them to define diagnostic pathways: deterministic, top-down relations from differential diagnoses to significant findings. We illustrate how these pathways allow us to handle implicit uncertainty in clinical reports, filling in gaps when clinicians omit parts of their reasoning. To capture explicit uncertainty, we analyze the language used in clinical reports to quantify the confidence expressed for each finding. By combining these approaches, we extract structured, uncertainty-aware information from unstructured text. We illustrate our methods on radiology reports, where both implicit and explicit uncertainty are common and critical for understanding the reporting clinician's reasoning.

* * *

Paloma Rabaey*, Jong Hak Moon*, Jung-Oh Lee, Min Gwan Kim, Hangyul Yoon, Thomas Demeester, Edward Choi

*Equal contribution.

Submitted to Language Resources and Evaluation Conference (LREC) 2026

Abstract

Radiology reports are invaluable for clinical decision-making and hold great potential for automated analysis when structured into machine-readable formats. These reports often contain uncertainty, which we categorize into two distinct types: (i) Explicit uncertainty reflects doubt about the presence or absence of findings, conveyed through *hedging* phrases. These vary in meaning depending on the context, making rule-based systems insufficient to quantify the level of uncertainty for specific findings; (ii) Implicit uncertainty arises when radiologists omit parts of their reasoning, recording only key findings or diagnoses. Here, it is often unclear whether omitted findings are truly absent or simply unmentioned for brevity. We address these challenges with a two-part framework. We quantify *explicit uncertainty* by creating an expert-validated, LLM-based *reference ranking* of common hedging phrases, and mapping each finding to a probability value based on this reference. In addition, we model *implicit uncertainty* through an expansion framework that systematically adds characteristic sub-findings derived from expert-defined diagnostic pathways for 14 common diagnoses. Using these methods, we release LUNGUAGE^{++}, an expanded, uncertainty-aware version of the LUNGUAGE benchmark of fine-grained structured radiology reports. This enriched resource enables uncertainty-aware image classification, faithful diagnostic reasoning, and new investigations into the clinical impact of diagnostic uncertainty.

8.1 Motivation and related work

Radiology reports play a central role in clinical decision-making, serving as the primary medium through which radiologists communicate their interpretations and diagnostic impressions to referring physicians. These reports influence downstream diagnostic reasoning, and often determine treatment trajectories. As AI models have been increasingly used for both automated radiology report interpretation and generation, structuring frameworks have been developed to convert free-text reports into machine-readable formats that can be used for training and evaluation [Moon et al., 2025, Wu et al., 2021a, Jain et al., 2021b, Khanna et al., 2023]. A key challenge that needs to be addressed is that **radiology reports inherently contain uncertainty**. This uncertainty arises in two distinct forms: explicit uncertainty, expressed directly through the language radiologists use to qualify their findings or diagnoses [Bruno et al., 2017, Hobby et al., 2000], and implicit uncertainty, which emerges from the selective and often incomplete nature of what is recorded in the report [Turner et al., 2021]. Figure 8.1 shows how these two forms of uncertainty appear in Chest X-Ray (CXR) reports, which are especially prone to contain uncertainty [Callen et al., 2020, Irvin et al., 2019].

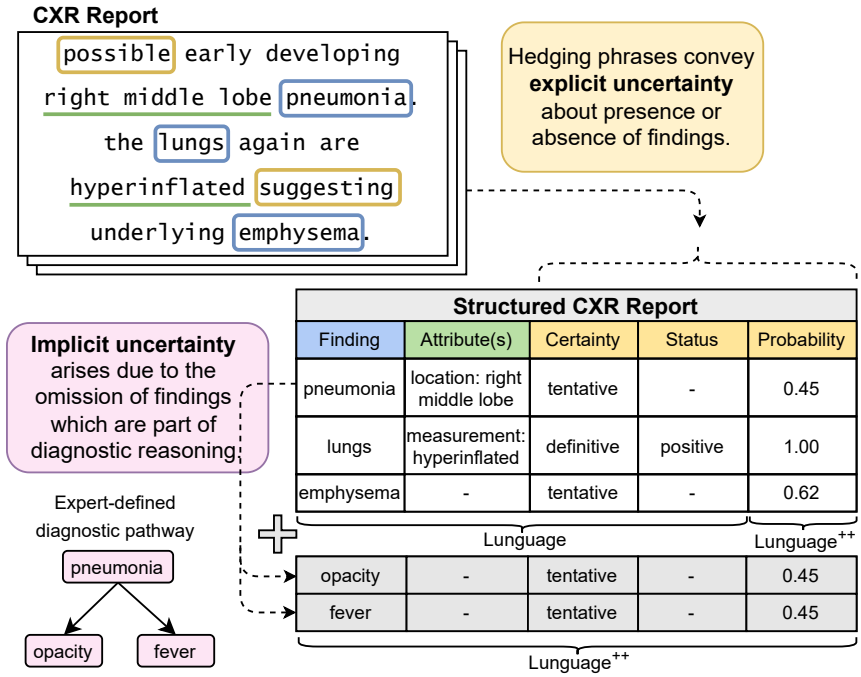


Figure 8.1: **Two types of uncertainty in radiology reports** that we address during structuring, expanding the LUNGUAGE dataset of structured CXR reports to form LUNGUAGE^{++}. Explicit uncertainty is conveyed by hedging phrases that indicate tentative findings, whose (un)certainty we quantify with probabilities. Implicit uncertainty stems from findings that are not explicitly mentioned; we mitigate this by applying expert-defined diagnostic pathways to expand stated diagnoses with their characteristic sub-findings.

Explicit uncertainty arises when radiologists convey doubt **about the presence or absence of a finding or diagnosis**, typically through *hedging* phrases such as "probably", "possible", "suggesting", or "may represent", among many others. Prior work examining these expressions has highlighted their prevalence, revealing the mention of uncertain diagnoses in a high proportion of CXR reports [Irvin et al., 2019, Moon et al., 2025]. The use of hedging in radiology reports is deliberate and meaningful: it enables radiologists to communicate diagnostic uncertainty that is crucial for appropriate clinical interpretation and decision-making [Reiner, 2018, Bruno et al., 2017]. Automated systems should similarly be designed to recognize and account for this uncertainty, just as human readers of radiology reports are trained to do.

Previous work has extracted uncertainty using rule-based systems [Irvin

et al., 2019, Johnson et al., 2019a, Zhang et al., 2023] in ternary classification settings (positive / uncertain / negative). These approaches are limited in scope, using predefined vocabularies of hedging phrases that trigger an uncertain label when detected in the report. In this work, we go beyond discrete labeling by **quantifying uncertainty as a continuous probability between 0 and 1**, taking into account the specific hedging phrases and their sentence-level context (see Figure 8.1). Earlier attempts to assign probabilistic meanings to hedging expressions have typically relied on human judgments, asking experts to rate phrases or position them along a scale of certainty [Hobby et al., 2000, Shinagare et al., 2019]. However, these approaches proved unreliable, as the interpretation of hedging varies widely across radiologists. To address this, we introduce an automated approach that estimates the probability of a finding by leveraging LLMs to perform pairwise comparisons of uncertainty expressions, constructing a relative ranking that is then mapped to a continuous probability.

Beyond overt language, reports also embody a subtler form of **implicit uncertainty** that has received little systematic attention. Radiologists frequently **omit portions of their diagnostic reasoning, documenting only key findings** or final impressions to maintain conciseness and ensure that the main message is easily understood by the reader [Lee et al., 2013]. For example, a report may state "congestive heart failure" without mentioning sub-findings that are commonly associated with the condition, such as "consolidation" or "cardiomegaly". Consequently, it is often unclear whether unmentioned findings are truly absent or simply unrecorded, even though knowing this distinction is crucial, since misinterpreting unrecorded evidence as true absence can systematically bias the data and distort the inferred diagnostic reasoning [Gärtner et al., 2020, Lee et al., 2013, Gunderman, 2009, Berlin, 2000].

In other words, implicit uncertainty does not stem from linguistic ambiguity, but from the selective and incomplete nature of clinical reporting [Turner et al., 2021] – what evidence is stated, abstracted, or left unstated within the report. Disentangling these possibilities requires contextual understanding of diagnostic logic and domain expertise, making implicit uncertainty difficult to model and largely unaddressed in prior research. Understanding and modeling these uncertainties is not only critical for structuring radiology reports but also for developing AI systems that faithfully capture the reasoning process of radiologists. By explicitly representing both what is uncertain and what is implied, such structured resources enable more reliable training and evaluation of medical AI models in uncertainty-aware report generation and interpretation.

To address this gap, we aim to fill in the missing findings that are not explicitly mentioned but are implied by the stated diagnoses when structuring radiology reports, as is shown in Figure 8.1. We **construct expert-defined diagnostic pathways for 14 common CXR conditions**, capturing characteristic sub-findings typically observed with high likelihood

(>80%). These pathways are then used to enrich the original reports by deterministically adding sub-findings that support explicitly mentioned diagnoses, with diagnostic certainty (derived from our explicit uncertainty extraction pipeline) propagated to each added finding. This approach results in expanded structured reports that more accurately reflect the radiologist's underlying reasoning.

In summary, our contributions are as follows:

- **Quantifying explicit uncertainty**: We introduce a comprehensive framework to estimate the probability of findings in radiology reports, taking into account hedging phrases and their sentence-level context. As part of this framework, we publish an expert-validated reference ranking of common hedging phrases.
- **Addressing implicit uncertainty**: We present the first framework to model implicit uncertainty in radiology reports by releasing diagnostic pathways for 14 common CXR diagnoses and integrating them into a rule-based framework. This framework reconstructs omitted diagnostic evidence by inferring the sub-findings that support each diagnosis.
- **Releasing LUNGUAGE^{++}**, which extends the LUNGUAGE dataset of structured radiology reports [Moon et al., 2025] by incorporating our techniques for capturing explicit and implicit uncertainty.

Code can be found at our Github repository[1], while LUNGUAGE^{++} and related resources will be made available via Physionet.

8.2 Lunguage dataset

We demonstrate our methods on the LUNGUAGE dataset [Moon et al., 2025], a benchmark dataset containing 1,473 annotated CXR reports from the MIMIC-CXR dataset [Johnson et al., 2019b]. Each report has been structured into fine-grained *(finding, relation, attribute)* triplets, where findings represent core clinical concepts (e.g. "opacity", "pneumonia"), and relations specify their contextual links (e.g., "location", "severity") with corresponding attributes (e.g., "right lower lobe", "moderate"). We use the LUNGUAGE dataset because of its extensive granularity in the included findings, relations and attribute types. Furthermore, each annotated finding includes a binary label (*tentative* and *definitive*) quantifying the confidence expressed by the radiologist – hedging phrases (*e.g.,* "suggests", "cannot exclude") lead to a *tentative* label, while findings that lack such phrases are labeled *definitive*.

In this work, we represent each report $\mathcal{R}$ as a structured set of $n_{\mathcal{R}}$ findings extracted from $m_{\mathcal{R}}$ sentences:

$$\mathcal{R} = (\{(f_i, s_i, c_i, a_i)\}_{i=1}^{n_{\mathcal{R}}}, \{t_j\}_{j=1}^{m_{\mathcal{R}}}), \tag{8.1}$$

[1] github.com/prabaey/lunguage_uncertainty

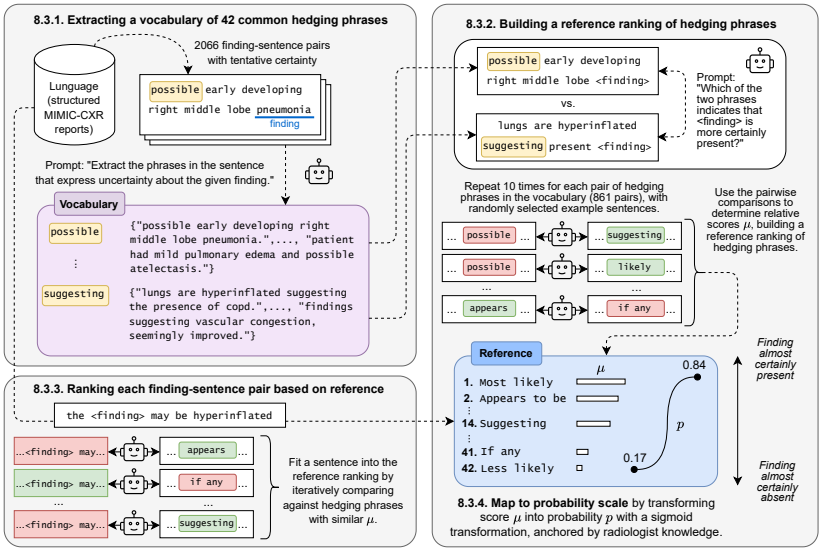


Figure 8.2: **Strategy for assigning probabilities to finding-sentence pairs with tentative certainty in the LUNGUAGE dataset**: We first build a vocabulary of common hedging phrases and the sentences in which these are used (Section 8.3.1). Next, we leverage LLMs to construct a reference ranking of these phrases, by performing pairwise comparisons of examples sentences (Section 8.3.2). Each finding-sentence pair is then compared against this reference (Section 8.3.3) and is finally mapped to a probability (Section 8.3.4). This approach ensures that the probability assigned to each finding reflects not only the hedging phrase itself but also the broader context in which it appears.

Here, t_j denotes the textual form of each **sentence**, f_i indicates the **finding**, s_i indicates the **status** (*positive* if the finding is present and *negative* if it is absent), c_i indicates the **certainty** (*definitive* or *tentative*), and a_i represents the **attributes** (e.g., location, morphology) associated with each finding. In LUNGUAGE, there are 14,049 such structured findings. In our study, we exclude sentences from the *history* section of the reports, as this section primarily contains information about patient symptoms or prior clinical records, whereas our analysis focuses on the *findings* and *impression* sections that convey diagnostic observations and reasoning.

8.3 Explicit uncertainty

We assume we have a report $\mathcal{R}$, where the sentence t_j expresses some uncertainty about finding f_i, in other words $c_i = tentative$. Across all reports in LUNGUAGE, this leaves us with a subset of 2,066 tentative finding-sentence

pairs. Our goal is to quantify this uncertainty by assigning a probability p_i between 0 (finding certainly absent) and 1 (finding certainly present). In the remainder of this section, we ignore the explicit status s_i, as the target probability p_i inherently captures the presence or absence of the finding. Furthermore, we disregard all findings where $c_i = definitive$, as there is no uncertainty in this case. From this point onward, a *sentence* refers to the complete text t_j, which contains a target *finding* f_i (e.g. "pneumonia") and one or more corresponding *(hedging) phrase(s)* (e.g., "possible"), expressing the degree of uncertainty associated with f_i.

Previous attempts to map hedging phrases to probabilities relied on expert ratings of individual phrases, an approach shown to be unreliable due to inconsistency across experts [Hobby et al., 2000, Shinagare et al., 2019]. Furthermore, the context of the sentence beyond individual phrases should be taken into account: "probably pneumonia" conveys a different certainty than "probably pneumonia given patient history", even though both use the phrase "probably". To address these limitations, we adopt a different strategy, which relies on LLMs to perform in-context pairwise comparisons between sentences conveying uncertainty. An overview of our framework is shown in Figure 8.2. We now describe each step of the process in detail. Additional details can be found in Appendix 8.7.1.

8.3.1 Extracting a vocabulary of common hedging phrases

For each finding-sentence pair, we automatically extract the hedging phrases that convey uncertainty about the finding f_i. We do this by prompting Gemini [Comanici et al., 2025]. The main part of the prompt is shown in Listing 8.1, while the full prompt contains a system message, ten in-context examples and additional instructions (see Appendix 8.7.1.1). The prompt specifies the finding f_i to avoid extraction of hedging phrases from the sentence which have nothing to do with that particular finding. We include in our vocabulary all hedging phrases that were extracted ten times or more, resulting in a vocabulary of 42 hedging phrases. Each phrase is associated with a list of finding-sentence pairs where it was extracted. The five most common extracted hedging phrases include *or* (373 times), *likely* (239 times), *may* (215 times), *suggesting* (74 times), and *cannot be excluded* (71 times); the full vocabulary is found in Appendix 8.7.1.1.

Listing 8.1: Prompt for hedging phrase extraction

```
Your task is to identify and extract only the words or phrases in the sentence that
     express uncertainty specifically about the given finding.
```

8.3.2 Building a reference ranking of hedging phrases

We construct a reference ranking of the 42 common hedging phrases, where the top rank corresponds to a probability near 1 and the bottom rank near 0.

TrueSkill We draw inspiration from ranking systems in competitive gaming, where player skill is inferred from the outcomes of matches. Specifically, we employ the TrueSkill algorithm [Herbrich et al., 2006], a Bayesian rating system that updates each item's mean skill level μ and confidence σ based on pairwise comparisons with other items. This way, TrueSkill efficiently infers a global ranking that reflects the relative skill level μ of each item. In our case, the items are hedging phrases.

LLM as a judge To obtain a reliable and well-calibrated ranking, a large number of pairwise comparisons between phrases is required. We therefore leverage LLMs to perform these comparisons automatically, employing three general-purpose LLMs (Gemini [Comanici et al., 2025], GPT-4o [Hurst et al., 2024], and Claude [Anthropic, 2025]) and one domain-specific medical LLM (MedGemma [Sellergren et al., 2025]).[2] Since the preferred phrase can vary depending on the sentence context, we conduct ten comparisons for each pair of the 42 hedging phrases, where in each comparison, we randomly sample a sentence in which the phrase occurs. We ask each LLM to identify which phrase conveys greater certainty about the presence of a finding, ensuring that such phrases will eventually receive a higher μ. The main portion of the prompt is shown in Listing 8.2; the complete prompt additionally includes a system instruction, detailed task description, and three in-context examples (Appendix 8.7.1.2). To ensure a more neutral comparison, the referenced finding f_i within each sentence is masked as *<finding>*.

Listing 8.2: Prompt for hedging phrase comparison

```
You will be given two sentences from radiology reports. Each sentence contains a
     placeholder <finding>, which represents a medical observation. Your task is to
     identify which sentence expresses a higher degree of certainty that the finding is
      present.
```

This procedure yields 4×8610 pairwise comparisons, each treated as an independent comparison by the TrueSkill algorithm. We execute TrueSkill 10 times with different random seeds that affect the order of matches, and then average the resulting μ values across runs. The final averaged scores produce a stable reference ranking of hedging phrases, which is shown in Figure 8.3. Appendix 8.7.1.2 assesses the robustness of this reference ranking and explores inter-LLM agreement across the set of comparisons.

[2]To deal with the sensitive nature of the reports in Lunguage we (i) ran a HIPAA-compliant GPT-4o model provided by Azure, (ii) revoked data retention rights for Gemini and Claude, and (iii) ran MedGemma locally.

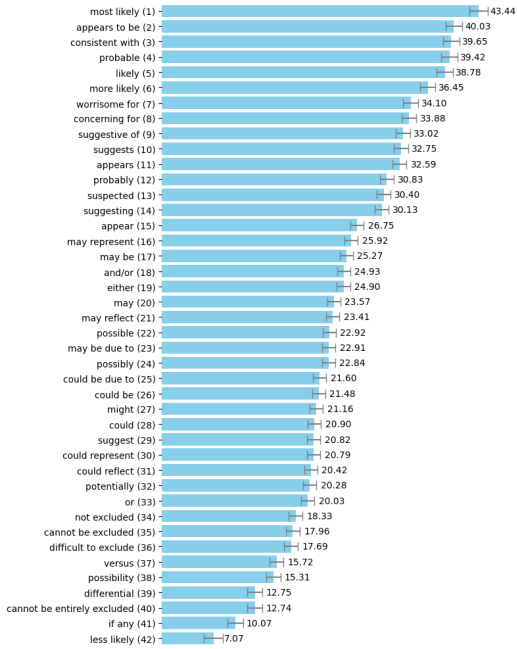


Figure 8.3: **Reference ranking of the 42 common hedging phrases in our vocabulary.** The mean skill level μ for each phrase is shown on the right, with the confidence σ represented by the error bars. Phrases at the top of the ranking correspond to a high likelihood that the finding is present, while phrases at the bottom correspond to a high likelihood that the finding is absent.

Table 8.1: **Expert agreement with the reference ranking and individual LLMs.** We define agreement as the proportion of the 50 phrase pairs where expert and model judgments are concordant. GP = General Practitioner.

Expert	Ref.	Gemini	GPT-4o	Claude	MedGem.
Radiologist	0.80	0.80	0.82	0.72	0.84
Radiologist	0.80	0.76	0.82	0.76	0.80
Internist	0.86	0.86	0.88	0.82	0.90
Oncologist	0.72	0.68	0.74	0.76	0.72
GP	0.66	0.70	0.68	0.66	0.62
GP	0.82	0.82	0.84	0.78	0.82
Average	0.78	0.77	0.80	0.75	0.77

Expert evaluation To validate the final reference ranking, we conducted an expert evaluation study. We recruited two expert *writers* (radiologists) and four expert *readers* (internist, oncologist, and two general practitioners). Each participant was presented with 50 pairs of hedging phrases, with five example sentences per phrase, all randomly sampled from our vocabulary. Participants were asked to select the phrase that conveyed a higher degree of certainty that the finding was present. All participants evaluated the same set of 50 phrase pairs. See Appendix 8.7.1.2 for additional details.

Table 8.1 presents the agreement between each expert and the reference ranking. Agreement is defined as the proportion of phrase pairs (out of 50) for which the expert's relative ordering of the two phrases matches that of the reference. In addition, we assess the agreement between each expert and each LLM. Since each LLM evaluated every phrase pair ten times using different example sentences, we first derive a consensus decision for each pair through majority voting, resolving ties at random.

We see that the experts agree well with both the reference ranking and the individual LLMs. Furthermore, the inter-expert Fleiss' κ is 0.72, indicating substantial agreement between experts. Among individual LLMs, GPT-4o shows the highest agreement with the experts, while Claude shows the lowest. Despite Claude's lower agreement on this limited subset of 50 pairs, we retain all LLMs in the reference ranking, to leverage the diversity of judgments across models, thereby improving the robustness of the TrueSkill-based ranking. Moreover, the expert evaluation covers only a small fraction of the full dataset and uses a different evaluation protocol than the sentence-level LLM comparisons, so occasional disagreements by a single LLM are not sufficient reason for exclusion.

8.3.3 Ranking each finding-sentence pair based on the reference ranking

With the reference ranking established, we fit each of the 2,066 tentative finding-sentence pairs from the LUNGUAGE dataset into it. Note that we cannot simply use the hedging phrases extracted from the sentence to assign a rank directly, because (i) some phrases are too rare to appear in the reference vocabulary, and (ii) the context in which a phrase occurs can significantly alter its implied certainty. Here, we once again draw inspiration from competitive gaming: when a new player enters a game, TrueSkill quickly estimates their rank by identifying existing players with similar skill levels μ. This is done by selecting opponents with the highest *draw probability*, iteratively playing those games, and updating the skill level of the new player based on the outcome of each game [Herbrich et al., 2006].

To fit a finding–sentence pair into the reference ranking, we initialize the target sentence's parameter μ to the default value of 25 and compute its draw probability against all opponent phrases in the reference ranking. The phrase with the highest draw probability is selected, and a corresponding sentence is randomly sampled from the vocabulary. Using the prompt from Listing 8.2, the target and opponent sentence are compared. During the first K iterations, all four LLMs perform the comparison; thereafter, one LLM is selected at random to reduce cost. Based on the outcome, the TrueSkill algorithm updates μ for the target sentence. Draw probabilities are then recomputed, and the next opponent is selected accordingly, with each opponent limited to N comparisons. The procedure terminates once the target sentence's rank remains stable for ten consecutive steps, or after 100 iterations, whichever occurs first. Through hyperparameter tuning, we select $K = 10$ and $N = 5$. Applying our algorithm to the 2,066 tentative finding-sentence pairs in LUNGUAGE incurred a total cost of \$92.16, averaging \$0.045/pair. The full algorithm, including experiments to validate our opponent selection strategy using *draw probability*, can be found in Appendix 8.7.1.3.

8.3.4 Map to probability scale

In the final step, we map the TrueSkill score μ, which determines the position of each finding-sentence pair in the ranking, to a probability $p \in [0, 1]$. This transformation is achieved using the sigmoid function $p = 1/(1 + e^{-\alpha(\mu-\mu_0)})$, where $\alpha \geq 0$ controls the steepness of the curve and μ_0 is the inflection point corresponding to a probability of 0.5.

We determine α and μ_0 using two anchor points: the desired probability p_{bottom} for the phrase *less likely* (lowest-ranked, $\mu = 7.07$), and p_{top} for the phrase *most likely* (highest-ranked, $\mu = 43.44$). To obtain these anchors, two radiologists independently reviewed ten example sentences per phrase and assigned a probability between 0 ("certainly absent") and 1 ("certainly

present"); see Appendix 8.7.1.4 for full instructions. Averaged across radiologists and examples, the resulting values were $p_{bottom} = 0.170$ (95% CI: $[0.156, 0.185]$) and $p_{top} = 0.839$ (95% CI: $[0.818, 0.860]$). These anchors yield $\alpha = 0.089$ and $\mu_0 = 24.89$.

After applying this mapping, the average probability for the 2,066 tentative finding-sentence pairs in LUNGUAGE is 0.459, with a standard deviation of 0.185, a maximum of 0.892, and a minimum of 0.102. Table 8.2 summarizes statistics for common CXR findings in the LUNGUAGE dataset. For each finding, we report proportions of positive and negative cases with definitive certainty, and the mean, standard deviation, minimum, and maximum probability for cases with tentative certainty.

8.4 Implicit uncertainty

Radiology reports often encode diagnostic reasoning implicitly [Gunderman, 2009, Lee et al., 2013, Berlin, 2000, Gärtner et al., 2020], documenting high-level diagnoses while omitting intermediate supporting findings. Although such abstraction enhances efficiency, it introduces ambiguity arising from unmentioned findings. To resolve this, we propose a **Pathway Expansion Framework** that reconstructs omitted diagnostic evidence by systematically expanding structured findings along predefined **diagnostic pathways**. This framework yields LUNGUAGE^{++}, an expanded version of LUNGUAGE, integrating explicit and implicit reasoning into a unified structured representation (Figure 8.4).

8.4.1 Diagnostic Pathway Construction

We formalize radiologists' implicit diagnostic reasoning into explicit, machine-friendly formats that describe how each diagnosis decomposes into its characteristic radiographic findings. To preserve interpretability and clinical fidelity, we established three principles through expert consensus. (i) **Exclusive mutual independence** ensures that each pathway differs from others by at least one defining piece of evidence. (ii) **Specificity** ensures that observations directly contributing to diagnostic differentiation are included, emphasizing findings that are concrete and fine-grained. (iii) **High-certainty features** retain only clinically consistent and reliable findings to minimize ambiguity and preserve clarity. Pathways were then constructed through a two-stage expert-in-the-loop process grounded in these principles. A radiologist and an AI researcher first defined subfindings for 14 common CXR diagnoses, referencing established radiological interpretation principles [Goodman, 2014, Webb and Higgins, 2011] to identify diagnostic criteria, imaging patterns, and hierarchical relations among findings. These preliminary pathways were subsequently reviewed and refined through consensus by another radiologist and an oncologist.

Table 8.2: **Certainty statistics for common CXR findings in the LUNGUAGE dataset.** For each finding, we collect the instances in LUNGUAGE where f_i is mapped to the finding through its various lexical and contextual variants, following the approach described in Section 8.4.2 (Finding Deduplication).

	Certainty level		**Definitive status**		**Tentative probability**			
Finding	**Definitive**	**Tentative**	**Positive**	**Negative**	**Avg.**	**Std.**	**Min.**	**Max.**
Pleural effusion	1165 (86.2%)	186 (13.8%)	497 (42.7%)	668 (57.3%)	0.465	0.201	0.102	0.862
Pneumothorax	897 (98.8%)	11 (1.2%)	43 (4.8%)	854 (95.2%)	0.480	0.151	0.269	0.709
Atelectasis	419 (62.2%)	255 (37.8%)	408 (97.4%)	11 (2.6%)	0.536	0.156	0.145	0.876
Pulmonary edema	516 (83.5%)	102 (16.5%)	264 (51.3%)	252 (48.7%)	0.479	0.186	0.102	0.841
Consolidation	406 (84.1%)	77 (15.9%)	90 (22.2%)	316 (77.8%)	0.447	0.182	0.113	0.808
Pneumonia	225 (51.3%)	214 (48.7%)	32 (14.2%)	193 (85.8%)	0.423	0.221	0.102	0.872
Cardiomegaly	313 (97.8%)	7 (2.2%)	308 (98.4%)	5 (1.6%)	0.518	0.141	0.365	0.706
Congestive heart failure	49 (89.1%)	6 (10.9%)	23 (46.9%)	26 (53.1%)	0.591	0.248	0.102	0.765
Emphysema	40 (80.0%)	10 (20.0%)	40 (100.0%)	0 (0.0%)	0.518	0.134	0.314	0.660
COPD	18 (60.0%)	12 (40.0%)	18 (100.0%)	0 (0.0%)	0.537	0.050	0.453	0.594
Fracture	12 (85.7%)	2 (14.3%)	12 (100.0%)	0 (0.0%)	0.219	0.165	0.102	0.336
Lung cancer	6 (46.2%)	7 (53.8%)	6 (100.0%)	0 (0.0%)	0.531	0.171	0.249	0.662
Tuberculosis	3 (60.0%)	2 (40.0%)	2 (66.7%)	1 (33.3%)	0.413	0.051	0.377	0.449
Bronchitis	1 (50.0%)	1 (50%)	1 (100.0%)	0 (0.0%)	0.475	0.000	0.475	0.475

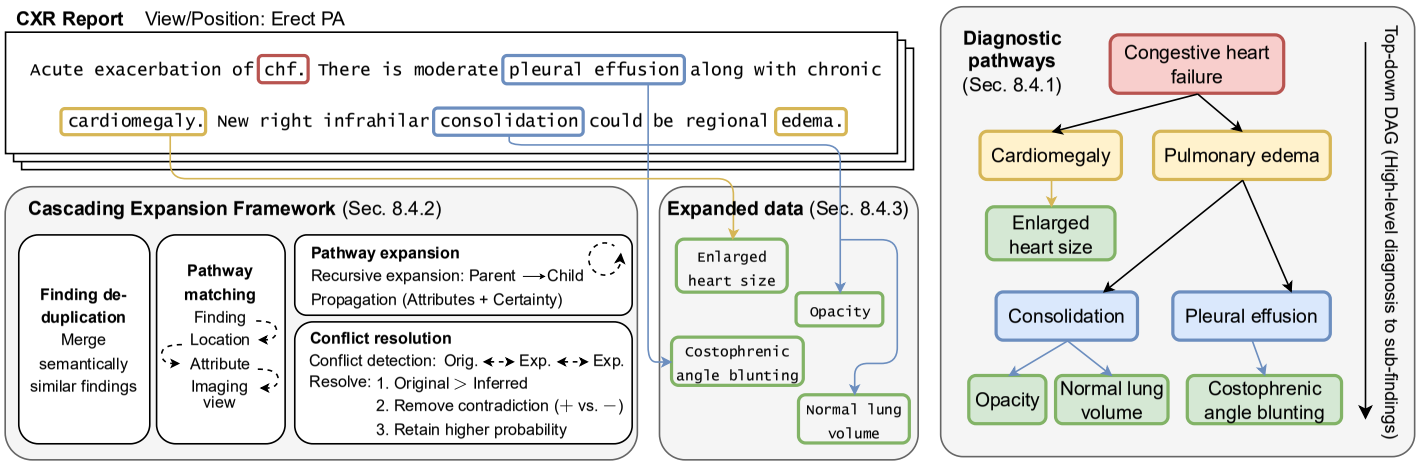


Figure 8.4: **Overview of the Pathway Expansion Framework.** The framework expands structured findings from LUNGUAGE along diagnostic pathways (Section 8.4.1) to reconstruct omitted diagnostic evidence. It comprises four stages—finding deduplication, pathway matching, pathway expansion, and conflict resolution (Section 8.4.2)—that jointly ensure semantic coherence and clinical validity. The resulting representation connects high-level diagnoses with their underlying evidence, forming LUNGUAGE^{++}, which is further analyzed in Section 8.4.3.

Pathways are represented as Directed Acyclic Graphs (DAGs) [Pearl, 2009], capturing how higher-level diagnoses require lower-level manifestations within hierarchical reasoning. An example of a DAG corresponding to (part of) the diagnostic pathways for *congestive heart failure* (*CHF*), *pulmonary edema*, *pleural effusion*, and *cardiomegaly* is shown in Figure 8.4. Each node corresponds to an observable finding, and directed edges (u, v) represent diagnostic relations $p_{path}(v|u)$ established through radiologist consensus, indicating that the sub-finding v is typically observed when its parent u is present, with likelihood greater than 0.8. This top-down structure models the cascading sequence of radiographic evidence that underlies diagnostic inference. Distinct pathways are defined for each diagnosis and further refined by imaging view and patient position, since these factors influence the findings one expects to observe. Table 8.3 reports statistical summaries of the diagnostic pathways, while the full set of pathways is provided in Appendix 8.7.2.1 and on our Github repository.

8.4.2 Pathway Expansion Framework

Building upon the diagnostic pathways, we design a *Pathway Expansion Framework* that implements the top-down reasoning process of radiological interpretation to fill in missing sub-findings. Given structured findings from LUNGUAGE [Moon et al., 2025], the framework reconstructs omitted intermediate findings by expanding each diagnosis along its predefined diagnostic pathway, following the recursive procedure detailed in Algorithm 1. It consists of four stages, each of which ensure that reconstructed structures remain clinically faithful and logically consistent.

1. Finding Deduplication To ensure that subsequent reasoning operates on a consistent and non-redundant set of findings, overlapping or synonymous mentions referring to the same diagnostic concept at the same location (e.g., "opacity in the basal segment of the lung" and "lung base opacity") are identified and merged within each report. Pairwise cosine similarity is computed between clinical embeddings (BioLORD [Remy et al., 2024]) of all report findings, where each finding is linearized into a phrase combining its *"entity, location, attributes"* following the LUNGUAGESCORE [Moon et al., 2025] formulation. Pairs exceeding a similarity threshold of 0.9 are merged into a single finding, while manually defined *blacklist pairs* (e.g., left vs. right) are excluded to prevent erroneous merges of semantically distinct findings. This precision-oriented threshold and manual blacklist correction minimize erroneous merges while ensuring that each finding is represented only once before expansion. The detailed blacklist pairs are provided in Appendix 8.7.2.2.

2. Pathway Matching After deduplication, the framework aligns each finding in LUNGUAGE with its most appropriate diagnostic pathway (Sec-

Algorithm 1 Recursive Finding Expansion

Input: Parent node f_i in pathway $\mathcal{P}$ with s_i, c_i, p_i, a_i
Output: Set $\mathcal{E}$ of all sub-findings expanded to leaf nodes

function EXPAND($f_i, s_i, c_i, p_i, a_i, \mathcal{P}$)
  $\mathcal{E} \leftarrow \emptyset$
  if $s_i \neq$ *positive* **then**
    return $\emptyset$ *// Expand only positive findings*
  end if
  for f_j where $(f_i, f_j) \in \mathcal{P}$ **do** *// For each child f_j in $\mathcal{P}$*
    $(s_j, c_j, p_j) \leftarrow (s_i, c_i, p_i)$ *// Inherit f_i properties to f_j*
    if pathway $(f_i \rightarrow f_j)$ defines refinements **then**
      $a_j \leftarrow a_i \cup$ attributes specified for f_j
    else
      $a_j \leftarrow a_i$ *// Inherit as-is*
    end if
    // Add expanded child f_j to $\mathcal{E}$
    $\mathcal{E} \leftarrow \mathcal{E} \cup \{(f_j, s_j, c_j, p_j, a_j)\}$
    if HasChildren($f_j, \mathcal{P}$) **then** *// recurse for f_j*
      $\mathcal{E} \leftarrow \mathcal{E} \cup$ EXPAND($f_j, s_j, c_j, p_j, a_j, \mathcal{P}$)
    end if
  end for
  return $\mathcal{E}$
end function

tion 8.4.1) through a sequential process based on its *finding*, *location*, *attributes*, and *imaging view*. All terms are first normalized using the LUNGUAGE vocabulary to ensure consistent terminology. The process begins with *finding-level matching*, which identifies the core diagnostic concept but may yield ambiguous interpretations. For instance, "fracture" can denote either a medical device fracture (e.g., pacemaker lead) or a thoracic skeletal fracture (e.g., rib, spine). Such ambiguity is resolved through *location matching*, which determines whether the finding pertains to an anatomical structure within the thorax or not, thereby selecting the appropriate pathway (e.g., device vs. skeletal fracture). Subsequently, *attribute matching* refines the alignment by specifying diagnostic stages such as "acute", "chronic", or "healed". Finally, *imaging view* and *patient orientation* (e.g., "PA, erect") define the view-specific pathway variant, ensuring that inferred findings remain anatomically observable within the imaging context. For the detailed process, see Appendix 8.7.2.2.

3. Pathway Expansion Building on the matched diagnostic pathways, the framework recursively expands structured findings in LUNGUAGE by inferring omitted sub-findings through parent–child relations defined in each pathway (Algorithm 1). The expansion begins with a parent finding f_i, which corresponds to one of the 14 diagnoses in the diagnostic pathways

$\mathcal{P}$. Each finding is represented by its diagnostic status s_i (*positive* or *negative*), certainty c_i (*definite* or *tentative*), probability p_i (quantified in Section 8.3), and attributes a_i (e.g., location, morphology). During expansion, every child node in $\mathcal{P}$ inherits all diagnostic properties from its parent—s_i, c_i, p_i, and a_i—ensuring that each descendant finding reflects its parent's context. For example, if f_i is labeled as *positive, tentative* with $p_i = 0.6$, all inferred sub-findings inherit the same label and probability. However, findings with $s_i = \textit{negative}$ are not expanded (line 3 in Algorithm 1), since the pathway $\mathcal{P}$ is not reversible; for instance, "no pneumonia" does not imply "no opacity." This inheritance (lines 7–11) reflects the propagation of diagnostic properties and the pathway-specific modification of attributes. Comprehensive expansion is achieved through recursive chaining (line 16): if a child finding f_j corresponds to one of the 14 diagnoses in $\mathcal{P}$, it becomes a new parent node and triggers further expansion until reaching leaf nodes. For example, when "pulmonary edema" acts as a parent node, it expands to "consolidation," which in turn recursively expands to its subfinding "opacity." This recursive expansion ensures that all findings along the diagnostic pathway maintain consistent uncertainty estimates and coherent hierarchical relations.

4. Conflict Resolution Although pathway relations capture generally valid diagnostic dependencies (Section 8.4.1), clinical exceptions and inconsistent uncertain expression can lead to conflicting findings during expansion. These conflicts typically arise when multiple pathways generate overlapping findings, leading to discrepancies in diagnostic status (positive vs. negative) or certainty (definitive vs. tentative). Conflicts were classified as: (i) *Original vs. Expansion* — discrepancies between the original report and expansion (e.g., a resolved "pulmonary edema" reappearing from the CHF pathway); and (ii) *Expansion vs. Expansion* — contradictions among expansion (e.g., "volume loss" from atelectasis vs. "normal volume" from consolidation). To ensure clinical plausibility, all potential conflicts were reviewed with radiologist input and resolved through a rule-based consistency protocol. Resolution followed a sequential protocol: (1) original findings take precedence to preserve factual accuracy; (2) contradictory entries (e.g., positive vs. negative) are removed; and (3) among uncertain cases, the instance with higher probability p_i is retained. This process ensures logical coherence and clinical consistency across all expanded findings. The detailed procedure is presented in Appendix 8.7.2.2.

8.4.3 Analysis of LUNGUAGE^{++}

We analyze the diagnostic pathways constructed in Section 8.4.1 and the expanded dataset generated by the framework in Section 8.4.2. This analysis quantifies the extent to which the framework reconstructs implicit diagnostic

reasoning, propagates uncertainty, and maintains report-level coherence. Table 8.3 reports statistics for the full set of diagnostic pathways.

Diagnostic Pathway Diversity Pathway diversity arises from variations in *imaging view*, *finding*, *location*, and *attributes*, which inherently capture differences in disease stage (e.g., acute vs. chronic). Across **14 diagnostic categories** (e.g., pneumonia), we identified **98 disease variants** (e.g., hospital-acquired pneumonia) organized into **33 unique diagnostic pathways** (e.g., pneumonia vs. lobar pneumonia), as shown in Table 8.3. Each pathway is characterized by its **depth**, representing the maximum hierarchical expansion depth, and its **width**, denoting the number of leaf-level findings at the terminal layer. On average, pathways exhibit a depth of 1.4 and a width of 3.2, indicating that most reasoning chains consist of one to three inferential layers. Among them, *pleural effusion* shows the greatest diversity (**10 pathways**) due to view-dependent fluid distribution, whereas *congestive heart failure* exhibits the deepest hierarchy (depth 3), cascading through cardiomegaly, pulmonary edema, and pleural effusion (Figure 8.4). In contrast, simpler findings (e.g., *cardiomegaly*) follow single-layer mappings (depth 1, width 1), directly linking diagnosis to observation.

Expansion Coverage As shown in Table 8.3, applying the Pathway Expansion Framework to 14,049 structured findings in LUNGUAGE identified **2,639 findings (18.7%)** that aligned with predefined diagnostic pathways, from which **4,761 additional sub-findings (+33.9%)** were inferred. This shows that roughly one in five findings contained implicit diagnostic structure recoverable through pathways. Expansion was dominated by interdependent categories such as *atelectasis* (+**1,215**), *pleural effusion* (+**915**), and *consolidation* (+**801**), which together accounted for more than half of all inferred findings. These conditions frequently co-occur or appear in hierarchical cascades naturally producing richer expansions, e.g. effusion with edema or atelectasis.

Conflict Resolution To assess the stability of the expanded structures, we further examined conflicts that arose during expansion. Across the entire LUNGUAGE dataset, such conflicts were rare, appearing in approximately **3.2%** of cases overall: **0.9%** arose between original and expanded findings (64.8% status, 35.2% certainty), and **2.3%** between expansions (22.8% status, 77.2% certainty). After conflict resolution, all remaining inconsistencies were resolved, confirming that the expanded dataset maintains both logical consistency and clinical validity. Detailed analyses are presented in Appendix 8.7.2.2.

Table 8.3: **Statistics of Diagnostic Pathways and LUNGUAGE^{++}.** The left panel summarizes the hierarchical structure of 14 expert-defined diagnostic pathways (98 variants, 33 unique DAGs). The right panel quantifies their application within the 14,049 structured findings in LUNGUAGE. A total of **2,636 findings (18.7%)** were eligible for expansion (i.e., directly aligned with a pathway node) serving as anchors for hierarchical reasoning. From these anchors, the framework inferred **4,761 additional sub-findings (+33.9%)**, representing previously implicit but clinically consistent observations reconstructed from the pathway hierarchy. †*Percentages are relative to the 14,049 original findings in* LUNGUAGE. *"+" denotes additional findings inferred during expansion.*

	Diagnostic Pathway Structure				**Expansion within LUNGUAGE†**	
Diagnosis	**Variants**	**Pathways**	**Depth**	**Width**	**Expandable Finding**	**Inferred Sub-findings**
Pleural effusion	2	10	1	3	683 (4.9%)	+915 (6.6%)
Atelectasis	7	1	1	2	663 (4.7%)	+1,215 (8.6%)
Pulmonary edema	12	2	2	4	366 (2.6%)	+622 (4.4%)
Consolidation	4	1	1	2	167 (1.2%)	+801 (5.7%)
Pneumonia	22	3	2	3	249 (1.8%)	+311 (2.2%)
Cardiomegaly	1	1	1	1	315 (2.2%)	+340 (2.4%)
Congestive heart failure	5	1	3	6	29 (0.2%)	+79 (0.6%)
Pneumothorax	5	2	1	4	54 (0.4%)	+160 (1.1%)
Emphysema	6	2	1	3	50 (0.4%)	+98 (0.7%)
COPD	3	1	2	5	30 (0.2%)	+88 (0.6%)
Fracture	15	4	1	5	14 (0.1%)	+25 (0.2%)
Lung cancer	7	1	1	1	13 (0.1%)	+12 (0.1%)
Tuberculosis	5	2	1	3	4 (0.0%)	+2 (0.0%)
Bronchitis	4	2	1	3	2 (0.0%)	+2 (0.0%)
Total / Avg.	**98**	**33**	**1.4**	**3.2**	**2,639 (18.7%)**	**+4,761 (33.9%)**

8.5 Conclusion

In this work, we present the first systematic approach to addressing both explicit and implicit uncertainty in radiology reports. We rigorously quantify the degree of **explicit uncertainty** of findings in CXR reports, through an LLM-based automated framework. While demonstrated on the LUNGUAGE dataset, this framework can be applied to any CXR report corpus, enabling the enrichment of widely used benchmarks such as CheXpert [Irvin et al., 2019] and MIMIC-CXR [Johnson et al., 2019a] (of which LUNGUAGE^{++} is only a subset) with continuous uncertainty measures.

In parallel, we expose and address **implicit uncertainty** in radiology reports arising from omitted elements of diagnostic reasoning. We introduce a rule-based expansion framework based on expert-defined diagnostic pathways for 14 common CXR diagnoses, which add characteristic sub-findings that may have been left unstated in the original reports. The diagnostic pathways can easily be reused for other tasks, while the expansion framework is applicable to any dataset that has been structured using the LUNGUAGE framework proposed by Moon et al. [2025].

Together with our reusable frameworks, we release LUNGUAGE^{++}, a benchmark dataset of structured radiology reports that includes continuous probabilities for all extracted findings and pathway-based expansions that expose previously omitted findings. This enriched resource supports a range of future research directions, including training uncertainty-aware CXR image classifiers, guiding vision-language models toward uncertainty-aware report interpretation and generation, and studying how diagnostic uncertainty influences downstream clinical outcomes.

8.6 Limitations

For the **explicit uncertainty framework**, a key limitation lies in our reliance on LLM-based pairwise comparisons to construct the reference ranking of hedging phrases. While this approach offers the advantage of scalability (enabling 8,610 comparisons across four LLMs, far beyond what would be feasible with human raters) it also introduces dependency on model behavior. Moreover, our mapping from skill level μ to probability p is based on assessments from only two radiologists; incorporating a larger expert pool would better capture the variability in how uncertainty is interpreted, as noted in prior studies [Hobby et al., 2000, Shinagare et al., 2019]. Finally, the current framework incurs nontrivial costs when assigning probabilities to new sentences, due to repeated LLM comparisons with the reference ranking. Future work could mitigate this by training a lightweight, locally hosted model, fine-tuned on our high-quality LLM comparison data.

For the **implicit uncertainty framework**, the limitations include both structural and clinical aspects. First, the diagnostic pathways are con-

structed as a top-down DAG that maps high-level diagnoses to lower-level sub-findings. While this design captures hierarchical reasoning, it remains unidirectional and cannot represent bottom-up or cyclic dependencies that naturally arise in clinical reasoning – such as when multiple findings interact or reinforce one another to revise a diagnosis. Future work could extend this structure into a bidirectional or dynamically learnable graph representation (e.g., a Bayesian network) that allows reasoning to flow in both directions, thereby capturing the iterative nature of diagnostic interpretation.

Second, when expanding findings along diagnostic pathways, we currently assign the same probability to all child nodes. For instance, if *congestive heart failure (CHF)* has a probability of 0.8, its inferred sub-findings—*cardiomegaly* and *pulmonary edema*—are each assigned the same value ($p = 0.8$), even though in practice, one may be more certain (e.g., $p = 0.9$) while the other less so (e.g., $p = 0.6$). This simplification overlooks interdependence and uncertainty calibration among findings, which future work could address by modeling probabilistic propagation that accounts for relative diagnostic confidence.

Lastly, despite being defined through high-probability relations ($p(v|u) > 0.8$), the pathways inherently encompass clinical exceptions. Even strongly associated findings may not hold under atypical imaging conditions or in the presence of comorbidities, occasionally causing conflicts between pathway-inferred and explicitly reported findings. These exceptions reflect the inherent variability of radiological practice rather than modeling errors, but they underscore the need for incorporating image-level verification and cross-modal reasoning to refine the framework. Future iterations could leverage visual grounding to reconcile such exceptions, ensuring that inferred reasoning aligns more closely with true clinical evidence.

Ethics Statement

This research made use of and expanded the LUNGUAGE dataset [Moon et al., 2025]. This dataset is derived from MIMIC-CXR [Johnson et al., 2019b], a public dataset of chest radiographs and free-text radiology reports. As this dataset contains sensitive patient information, we will follow Physionet's guidelines by publishing LUNGUAGE^{++} under the same agreement as the source data. Furthermore, to avoid passing sensitive patient data to public LLM APIs, we followed Physionet's recommendations by (i) running a HIPAA-compliant GPT-4o model provided by Azure, (ii) revoking data retention rights for Gemini and Claude, and (iii) running MedGemma locally.

Acknowledgments

The authors would like to thank Géraldine Deberdt, Stefan Heytens and Johan Decruyenaere for participating in the expert evaluation study. Additionally, we are grateful to Stefan Heytens and Johan Decruyenaere for their input on the conceptual design of the expert evaluation study and sharing their view on uncertainty in medical reporting.

8.7 Appendix

8.7.1 Explicit uncertainty

8.7.1.1 Vocabulary of hedging phrases

The full prompt that was used to extract hedging phrases for each finding-sentence pair in Lunguage, is shown in Listing 8.3. We use this to prompt the Gemini model *gemini-2.5-flash* with the maximum number of output tokens set to 1100, where 1000 tokens were assigned as a thinking budget. *Temperature* and *top_p* parameters were set to their default values of 1. The total price to extract all hedging phrases was around $1.

Listing 8.3: Full prompt for hedging phrase extraction

```
SYSTEM: You are a radiologist who is given pairs of entities and sentences. Each entity
      appears in the corresponding sentence. Your task is to identify and extract only
      the words or phrases in the sentence that express uncertainty specifically about
      the given entity.

TASK: You are given pairs of entities and sentences. Each entity appears in the
      corresponding sentence. Your task is to identify and extract only the words or
      phrases in the sentence that express uncertainty specifically about the given
      entity.

Important notes:
- The sentence may mention multiple entities, but you should extract uncertainty clues
      only for the specified entity. Please refer to the examples below that show how
      you should handle such cases.
- Look for words or phrases that suggest uncertainty, speculation, approximation,
      possibility, or lack of definitiveness (e.g., "might", "possibly", "suggests", "
      appears to", "in some cases").
- Return a list of such uncertainty clues found in the sentence and relevant to the
      query entity.

Return your output as a list: ["<word or phrase 1>", "<word or phrase 2>", ...]
If there are no uncertainty clues related to the given entity, return an empty list.

Below are 10 examples, after which you must complete the task for an unseen query.

INPUT:
{
  "entity": "pulmonary edema",
  "sentence": "overall, however, there is a more focal airspace opacity in the left mid
       and lower lung, which may reflect asymmetric pulmonary edema or an infectious
       process, less likely atelectasis."
}
OUTPUT:
```

```
["may", "or"]

INPUT:
{
  "entity": "atelectasis",
  "sentence": "overall, however, there is a more focal airspace opacity in the left mid
       and lower lung, which may reflect asymmetric pulmonary edema or an infectious
       process, less likely atelectasis."
}
OUTPUT:
["less likely"]

... <redacted for brevity>

INPUT:
{
  "entity": "pleural effusion",
  "sentence": "a left pleural effusion and atelectasis obscure the left cardiac and
       hemidiaphragmatic contours more than the prior day."
}
OUTPUT:
[]

Examine the entity and sentence pair below. If the sentence also talks about other
     entities, first identify the part of the sentence that is talking about the query
     entity, and then extract phrases expressing uncertainty specifically related to
     that entity.
Return your output as a list: ["<word or phrase 1>", "<word or phrase 2>", ...]. If
     there are no uncertainty clues related to the given entity, return an empty list.

INPUT:
{
  "entity": {entity}
  "sentence": {sentence}
}
OUTPUT:
```

We manually reviewed and corrected all cases where more than one phrase was extracted (568 finding-sentence pairs), since these were prone to mistakes. Furthermore, we noticed that the LLM occasionally extracted phrases which did not relate to the uncertainty of the finding, but rather to other relational attributes which were already extracted in LUNGUAGE: *onset*, *measurement* and *severity*. Phrases that matched these attributes for the same finding-sentence pair were filtered out. The phrase "difficult to assess" was extracted in 11 cases, but was removed since it relates more to visual limitations rather than uncertainty.

Figure 8.5 shows the number of times each of the 42 most common hedging phrases (i.e., occurring more than 10 times) were extracted from LUNGUAGE.

8.7.1.2 Reference ranking of hedging phrases

TrueSkill details The TrueSkill ranking system maintains a Bayesian belief in every item's skill by estimating two parameters: the average skill (μ) and the degree of confidence in the skill (σ), together characterizing a Gaussian distribution $\mathcal{N}(\mu, \sigma)$. Initially, we set $\mu = 25$ and $\sigma = \frac{25}{3}^2$, which is the default. TrueSkill additionally defines a parameter β^2, which controls

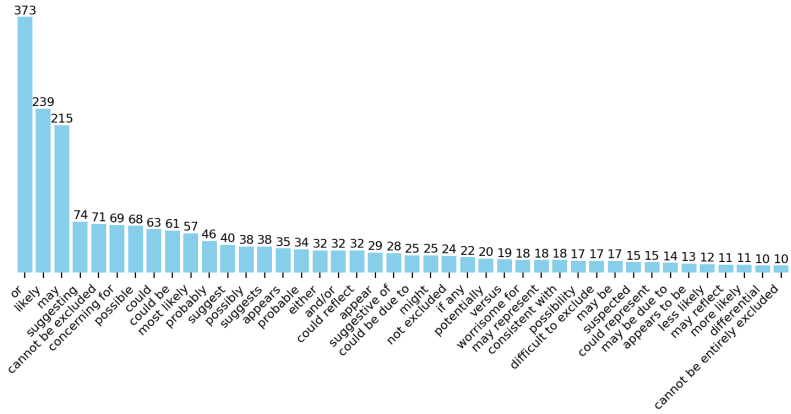


Figure 8.5: **Our vocabulary of 42 common hedging phrases**, including how many times each phrase was extracted for tentative finding-sentence pairs in LUNGUAGE.

the variance of the item's performance around its average skill in a single match, set to 25/6 per default.

When we use an LLM to compare two hedging phrases, we are essentially playing a match where one hedging phrase is the winner. After each match, TrueSkill updates both μ and σ using Bayesian inference: if an item performs better than expected, its μ increases, with the magnitude of this change depending on σ, and vice versa if it performs worse than expected. This mechanism is formalized in TrueSkill's update equations [Herbrich et al., 2006], which ensure that σ naturally shrinks as the system becomes more confident in its estimate, stabilizing each item's rating. Since the winner of a match is picked as the phrase that is closest to "finding is certainly present", a higher μ will be obtained for phrases closer to that part of the likelihood spectrum. In our case, we input all pairwise comparisons (across 861 pairs of hedging phrases, each repeated 10 times, with 4 LLMs) one after the other, in a random order. Since the final ranking is order-dependent, we repeat this 10 times with different random orders, averaging the final μ and σ across runs.

Prompting details The full prompt that was used to perform comparisons between pairs of hedging phrases is shown in Listing 8.4. We executed this prompt for each pair of hedging phrases, repeating 10 times with different example sentences (*sentence_1* and *sentence_2*), and with each of the four LLMs: *gemini-2.5-flash*, *gpt-4o*, *claude-sonnet-4*, *medgemma-27b-text-it*. For each LLM, we set the maximum number of tokens to 100, *temperature* to 1, *top_p* to 1. This time, we assigned zero thinking budget to Gemini, to pull all the models to the same level of reasoning capabilities.

Listing 8.4: Full prompt for hedging phrase comparison

```
SYSTEM: You are a radiologist who is ranking sentences expressing uncertainty.

TASK: You will be given two sentences from radiology reports. Each sentence contains a
     placeholder "<finding>", which represents a medical observation (e.g.,
     consolidation, effusion, nodule). Each sentence includes a phrase that expresses
     the degree of certainty about the presence or absence of the finding.

Assume there is a certainty spectrum ranging from:
    "<finding> is certainly absent"
    to
    "<finding> is certainly present"
Your task is to identify which sentence is **closer to "<finding> is certainly present
     "** on this scale, using the context of the sentence. In other words, your task is
      to identify which sentence expresses a higher degree of certainty that the
     finding is present.

Respond with **only** the chosen sentence (sentence_1 or sentence_2).

Here are some examples.

---

Example 1:
INPUT:
{
  "sentence_1": "interstitial markings are prominent, suggest possible mild <finding>.",
  "sentence_2": "allowing for low inspiratory volumes, the <finding> is probably
       unchanged."
}
OUTPUT:
"sentence_2"

---

Example 2:
INPUT:
{
  "sentence_1": "given the clinical presentation, <finding> must be suspected.",
  "sentence_2": "although this could represent severe <finding>, the possibility of
       supervening pneumonia or even developing ards must be considered."
}
OUTPUT:
"sentence_1"

---

Example 3:
INPUT:
{
  "sentence_1": "this could be either pneumonia in the left upper lobe or fissural <
       finding>.",
  "sentence_2": "the presence of a minimal left <finding> cannot be excluded, given
       blunting of the left costophrenic sinus.",
}
OUTPUT:
"sentence_1"

---

INPUT:
{
  "sentence_1": {sentence_1},
  "sentence_2": {sentence_2}
}
```

Table 8.4: Pairwise agreement between LLMs across 8610 hedging phrase comparisons.

Model	Gemini	GPT-4o	Claude	MedGemma
Gemini	1.000	0.861	0.868	0.866
GPT-4o	0.861	1.000	0.861	0.862
Claude	0.868	0.861	1.000	0.870
MedGemma	0.866	0.862	0.870	1.000

```
Which of the two sentences ("sentence_1" or "sentence_2") indicates that <finding> is
     more certainly present? Respond with your choice **only**.

OUTPUT:
```

Reference ranking To assess the robustness of our reference ranking (shown in Figure 8.3), we constructed an alternative ranking using only five comparisons per phrase pair. Compared to the original ranking in Figure 8.3, this reduced version showed an average absolute difference of 0.36 in μ and 0.76 in rank, with a Spearman correlation of 0.996 between both rankings. These results indicate that ten comparisons per pair provide a sufficiently stable and reliable ranking. We also compute pairwise agreement between the four LLMs across all 8610 comparisons, which is shown in Table 8.4. The average agreement scores are 0.865 for Gemini, 0.861 for GPT-4o, 0.866 for Claude, and 0.866 for MedGemma. Fleiss' Kappa between all four models is 0.860 and Krippendorff's Alpha is 0.722. These values indicate strong but imperfect agreement across models, highlighting the need to integrate the judgments of all four LLMs when constructing the final ranking, as no single model can be assumed to provide the correct outcome in every case.

To further explore the variation across LLMs, we apply the TrueSkill algorithm to each individual LLM's set of comparisons (8610 each), repeating 10 times for different random orderings of the matches. We then compute pairwise Spearman rank correlations between the obtained rankings. The full correlation plot is show in Figure 8.6. Note that correlations are high for within-LLM comparisons, indicating that averaging across 10 ordering seeds should result in a stable ranking. Correlations are lower for across-LLM comparisons, once again indicating that relying on a single LLM to build our reference ranking would not be sufficient. As stated before, our final reference ranking (shown in Figure 8.3) is obtained by applying the TrueSkill algorithm to the full set of comparisons, repeating 10 times across ordering seeds and averaging the final μ and σ across runs. Figure 8.7 shows the Spearman rank correlation between each individual LLM's ranking across 10 seeds and the final reference ranking. Note that these correlations are generally higher than the pairwise correlations between individual LLMs from Figure 8.6.

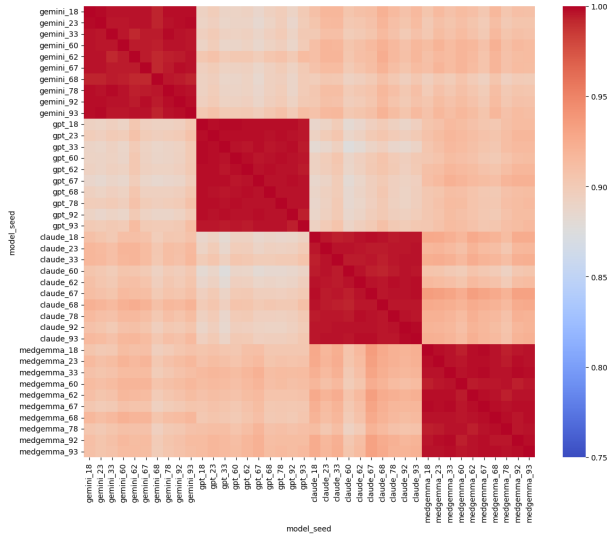


Figure 8.6: **Pairwise Spearman rank correlations** between TrueSkill rankings obtained for individual LLMs, across 10 seeds.

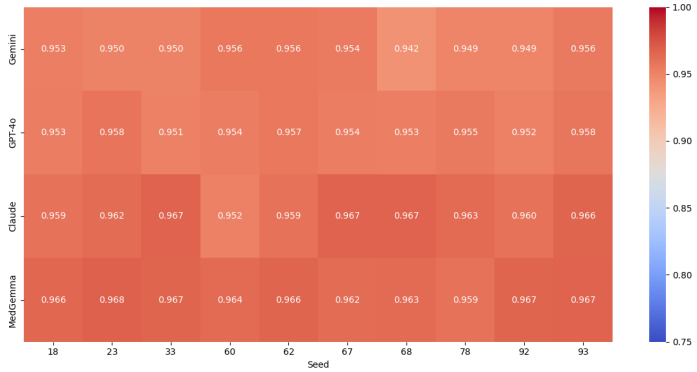


Figure 8.7: **Spearman rank correlations with the reference ranking** for TrueSkill rankings obtained for individual LLMs, across 10 seeds.

Expert evaluation Each participant was presented with 50 pairs of hedging phrases, with five example sentences per phrase. They were asked to choose the phrase that expresses a higher degree of certainty that the finding is present; an example is shown in Listing 8.5. The full survey, including detailed instructions for the evaluators, can be found on our Github repository, in the file `survey_uncertainty.pdf`. Table 8.5 shows the occupation and experience of the six experts involved in our expert evaluation study. Figure 8.8 shows the pairwise agreement between experts, and between experts and the reference ranking. Agreement is defined as the proportion

Table 8.5: **Expert occupation and experience**

Type	Occupation	Experience
Writer	Radiologist	5 to 10 years
Writer	Radiologist	0 to 5 years
Reader	Internist	30 to 40 years
Reader	Oncologist	5 to 10 years
Reader	GP	30 to 40 years
Reader	GP	0 to 5 years

of phrase pairs (out of 50) for which the expert's relative ordering matches that of the other expert (or of the reference ranking).

Listing 8.5: Example question in expert evaluation study

```
Phrase 1: difficult to exclude.
Example sentences:
- bilateral hilar vascular prominence is re-demonstrated with subtle <finding> in the
     left upper lung likely representing confluence of vasculature though a true nodule
      **difficult to exclude**.
- there is slight blunting of both costophrenic angles, felt most likely be due to
     overlying soft tissues, but a trace <finding> be **difficult to exclude**.
- no large <finding> is seen, although trace effusions are **difficult to exclude**.
- no large <finding> however, trace bilateral <finding>s **difficult to exclude**.
- mild <finding> is **difficult to exclude** in the correct clinical setting.

Phrase 2: appear.
Example sentences:
- the <finding> **appear** clear.
- the <finding> **appear** well inflated.
- the mediastinal and <finding> **appear** unchanged, allowing for differences in
     technique.
- mid <finding> **appear** intact.
- <finding> **appear** grossly intact.
```

8.7.1.3 Fitting each finding-sentence pair into the reference ranking

Algorithm Using Algorithm 2, we can fit any tentative finding-sentence pair into the reference ranking. Here, the draw probability between the target sentence t_{tar} with $(\mu_{tar}, \sigma_{tar})$, and an opponent phrase opp with $(\mu_{opp}, \sigma_{opp})$ is calculated as follows:

$$DrawProb(t_{tar}, opp) = exp\left(\frac{-(\mu_{tar} - \mu_{opp})^2}{2c^2}\right)\sqrt{d}, \tag{8.2}$$

with $c^2 = 2\beta^2 + \sigma_{tar}^2 + \sigma_{opp}^2$, $d = 2\frac{\beta^2}{c^2}$, and $\beta^2 = 25/6$ [Herbrich et al., 2006].

Validation To validate our algorithm and the use of draw probability as an opponent selection strategy, we designed the following experiment. While the true rank of individual sentences in our dataset is unknown, we

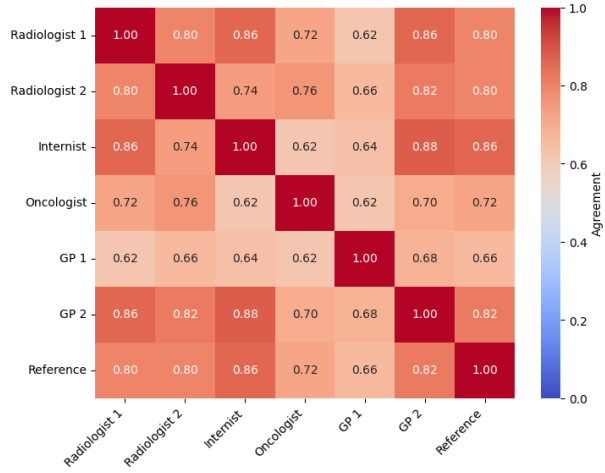


Figure 8.8: **Pairwise agreement between experts and reference ranking**, across 50 hedging phrase pairs.

can assess performance using the hedging phrases in our reference vocabulary, whose ranks are known. Specifically, we simulate a modified reference ranking in which one of the 42 hedging phrases is temporarily excluded, then fit that phrase back into the ranking using a modified version of Algorithm 2. We consider two opponent selection strategies: (i) the *draw probability* strategy described previously, and (ii) a *random* strategy, where an opponent is chosen uniformly at random at each step. We also test multiple LLM configurations: one using *all* LLMs for the first K iterations (as in Algorithm 2) and then picking an LLM randomly, and another using a single LLM (*Gemini*, *GPT-4o*, *Claude*, or *MedGemma*) for all comparisons. Each configuration is repeated with 10 random seeds, which control sentence sampling, LLM selection, and opponent choice in the random setting. At each iteration, we compute the absolute distance between the current estimated rank of the phrase and its true rank in the reference ranking. If the algorithm converges before 100 iterations, the final rank is used for all subsequent iterations. We then average the absolute rank distances across all 42 phrases and 10 seeds. In these experiments, K was set to 10 and N to 5.

Figure 8.9 presents the resulting performance across all configurations. Note that the strategy which uses *draw probability* and *all* LLMs performs best, which is indeed the strategy implemented in Algorithm 2. While the *random* strategy performs similarly to *draw probability* in the *single-LLM* setting, it leads to less stable runs, as evidenced by the standard deviation of the distance to the true rank at step 100, averaged across all phrases, which is shown in Figure 8.9 as well.

Algorithm 2 Determine TrueSkill score μ for finding-sentence pair (f_{tar}, t_{tar}) based on reference ranking

Input: Target sentence t_{tar}, reference ranking $\mathcal{R}$, LLMs, K, N, max_steps = 100, patience = 10
Output: TrueSkill score $(\mu_{tar}, \sigma_{tar})$ for t_{tar}
Initialize $\mu_{tar} \leftarrow 25$, $\sigma_{tar} \leftarrow \frac{25}{3}^2$, $step \leftarrow 0$, $s \leftarrow 0$
Initialize opp_counts[phrase] $\leftarrow 0$, $\forall$ phrase $\in \mathcal{R}$
while $step <$ max_steps **and** $s <$ patience **do**
    Select $opp \in \mathcal{R}$ with $max\ DrawProb(t_{tar}, opp)$
    if $opp\_counts[opp] \geq N$ **then** skip
    $opp\_counts[opp] \leftarrow opp\_counts[opp] + 1$
    Randomly select sentence t_{opp} containing opp
    if $step < K$ **then**
        $models \leftarrow$ all LLMs
    else
        $models \leftarrow$ randomly pick 1 LLM
    end if
    Compare t_{tar} vs t_{opp} using $models$
    Update TrueSkill $(\mu_{tar}, \sigma_{tar})$ for t_{tar}
    Recalculate rank r_{tar} of μ_{tar} among $\mathcal{R}$
    If r_{tar} unchanged **then** $s \leftarrow s + 1$ **else** $s \leftarrow 0$
    $step \leftarrow step + 1$
end while

Hyperparameters Algorithm 2 contains hyperparameter K, which controls the number of iterations for which all four LLMs perform the phrase-sentence comparison, and hyperparameter N, which decides the number of times each opponent can be selected for comparison. Following the same experiment setup outlined above, with the *draw probability* strategy and including *all* LLMs, we perform hyperparameter optimization across the hedging phrases in our vocabulary. For K, we ran Algorithm 2 for $K \in [0, 1, 5, 10, 20, 50, 100]$ with $N = 5$. Figure 8.10a shows the average distance to the true rank in the reference ranking, averaged across all phrases. Setting $K = 10$ forms the right balance between cost-efficiency and performance. We performed the same experiment for $N \in [2, 3, 5, 10, 20]$, with $K = 10$, ultimately choosing $N = 5$ (Figure 8.10b).

Results and cost Applying Algorithm 2 to the 2,066 tentative finding-sentence pairs in LUNGUAGE incurred a total cost of $92.16, averaging $0.045/pair. Specifically, we spent $5.59 on Gemini, $39.49 on GPT-4o, and $45.12 on Claude, while MedGemma incurred no cost as it was run locally. On average, 24.61 ranking steps were required per finding-sentence pair, ranging from 10 to 100. The average μ value is 22.62, with a minimum of 0.41 and a maximum of 48.58, compared to a minimum of 7.07 and a maximum of 43.44 in the reference ranking (see Figure 8.3). The average

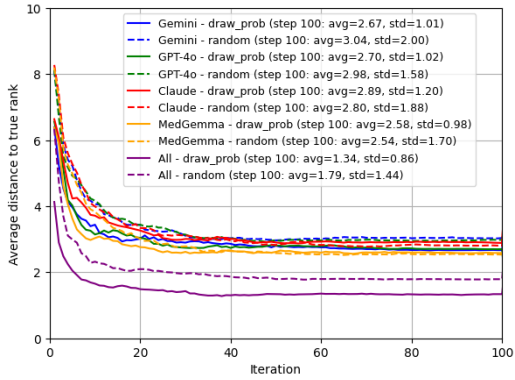
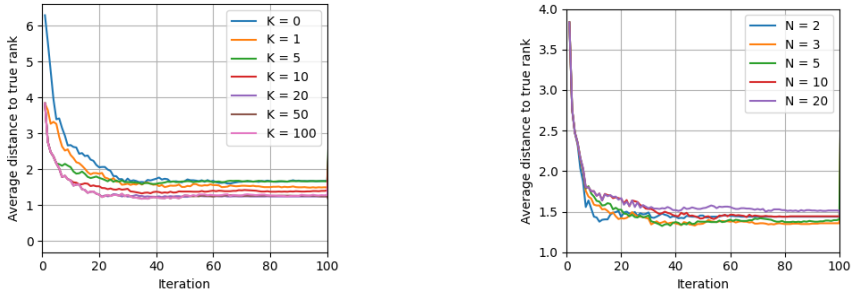


Figure 8.9: **Average distance to the true rank across iterations of variants of Algorithm 2**. We test different variants of the opponent selection strategy–*draw probability* vs. *random*–and LLMs used for comparisons–*all* or *single-LLM* (Gemini, GPT-4o, Claude, or MedGemma).



(a) K (b) N

Figure 8.10: Hyperparameter tuning results for K and N.

rank when fit into the reference ranking is 23.87, with a minimum rank of 1 and a maximum rank of 43.

8.7.1.4 Map to probability scale

We use an expert-informed sigmoid mapping (described in Section 8.3.4) to transform each μ into its probability value p. A histogram of the probabilities in the tentative portion of LUNGUAGE^{++} is shown in Figure 8.11.

This mapping is anchored by the expert-defined probabilities for the phrases *less likely* and *most likely*. Two radiologists received the following instruction:

```
On a scale of certainly absent (0) to certainly present (100), where would you place the
      phrase <phrase>, as it relates to the <finding> in each of the following
     sentences? Keep in mind that your answer may differ based on the context of the
     sentence.
```

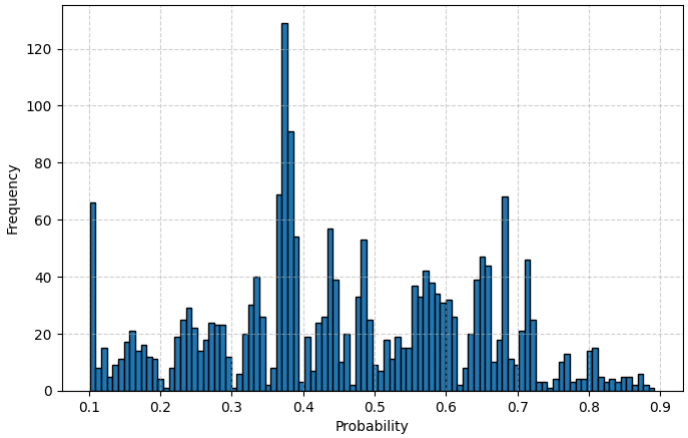


Figure 8.11: **Histogram of probabilities in the tentative portion of LUNGUAGE^{++}.**

They were asked to assign such probabilities for 10 example sentences, for both phrases. An example sentence for *less likely* includes: "Minimal blunting of the right costophrenic sulcus is more suggestive of similar slight atelectatic change, less likely persistent trace <finding>". An example sentence for *most likely* includes: "Streaky predominantly right-sided mid and lower lung opacities are seen, most likely due to <finding>". The full instructions are included in the survey, which can be found on our Github repository, in the file `survey_uncertainty.pdf`.

8.7.2 Implicit uncertainty

8.7.2.1 Diagnostic pathways

We present the diagnostic pathways defined in *Pathway Expansion Framework*. In practice, the pathways encode context primarily along two axes: (i) **imaging view and patient positioning**, which determine which radiographic signs can be expected or meaningfully interpreted (e.g., pleural effusion on an erect PA view typically presents with costophrenic angle blunting, whereas on a supine AP view it more often appears as diffuse haziness over the pleural space); and (ii) **clinical acuity together with morphologic/anatomic subtype**, which jointly modulate how a diagnosis presents (e.g., pneumothorax shows peripheral pleural air with absent pulmonary lung markings, whereas tension pneumothorax additionally shows mediastinal shift; pleural effusion may be non-loculated or loculated). These conditioning factors are encoded in the pathway dictionary. During expansion, the *Pathway Expansion Framework* uses these pathways to recover omitted intermediate findings in a report. Tables 8.6, 8.7, and 8.8

present representative pathway variants for the 14 diagnoses. Figures 8.12, 8.13, 8.14, and 8.15 represent these pathways visually. In the figures, each panel depicts the diagnosis as the root and its subfindings as children, with arrows indicating expert-defined dependencies. For example, in congestive heart failure the pathway links the root to cardiomegaly, pulmonary edema, and dyspnea (congestive heart failure → cardiomegaly + pulmonary edema + dyspnea). A dictionary with all pathway variants is also released on our Github repository, in the file `dx_pathway.csv`.

8.7.2.2 Cascading expansion framework

Blacklist for Finding Deduplication We merge near-duplicate findings into a single canonical form using cosine similarity from a clinical embedding model (e.g., BioLord [Remy et al., 2024]) with a threshold of 0.9. Even with this high threshold, purely lexical similarity can still spuriously merge clinically distinct statements that differ along critical axes (e.g., ***left*** *lower lobe new consolidation* vs. ***right*** *lower lobe new consolidation*). To avoid false merges of distinct concepts, we apply an explicit blacklist with two layers: (i) exact-pair exclusions and (ii) pattern-level rules. We use a manually curated blacklist of mutually incompatible pairs, which is invoked when two candidates belong to the same study and to the same coarse anatomic region but differ along key discriminative axes such as laterality, lobar/segmental location, diagnostic status, or pathophysiologic mechanism. A candidate pair is merged only if it passes the similarity threshold and does not match any blacklist entry; otherwise both findings are preserved. Representative blacklist entries and patterns are summarized in Table 8.9.

Details of Pathway Matching Algorithm 3 describes the pathway matching procedure, which aligns each input tuple $(\textit{finding}, \textit{loc}, \textit{attr}, \textit{view})$ to at most one pathway variant from the pathways summarized in Tables 8.6 to 8.8. The full code can be found on our Github repository.

For each row from LUNGUAGE, we first apply `NormalizeInput` to obtain a normalized tuple (f', ℓ', a', v'), mapping raw report phrases to the LUNGUAGE vocabulary (e.g., "RUL" → "right upper lung," "TB" → "tuberculosis") and mapping view metadata to a standardized `view_information` string (AP, PA, LATERAL, erect, supine). In parallel, we apply `NormalizeDict` to the pathway dictionary $\mathcal{D}$, yielding $\mathcal{D}'$ with normalized findings, locations, attributes, and views for every pathway entry.

Given (f', ℓ', a', v') and $\mathcal{D}'$, the `BestPathwayMatch` routine proceeds in three compatibility steps. First, it performs a *finding-level lexical match*: we restrict candidates to those pathways whose normalized finding term is identical to f'. This ensures that, for example, tuples with finding "fracture" only compete among fracture-related pathways, and tuples with finding "pleural effusion" only among effusion pathways.

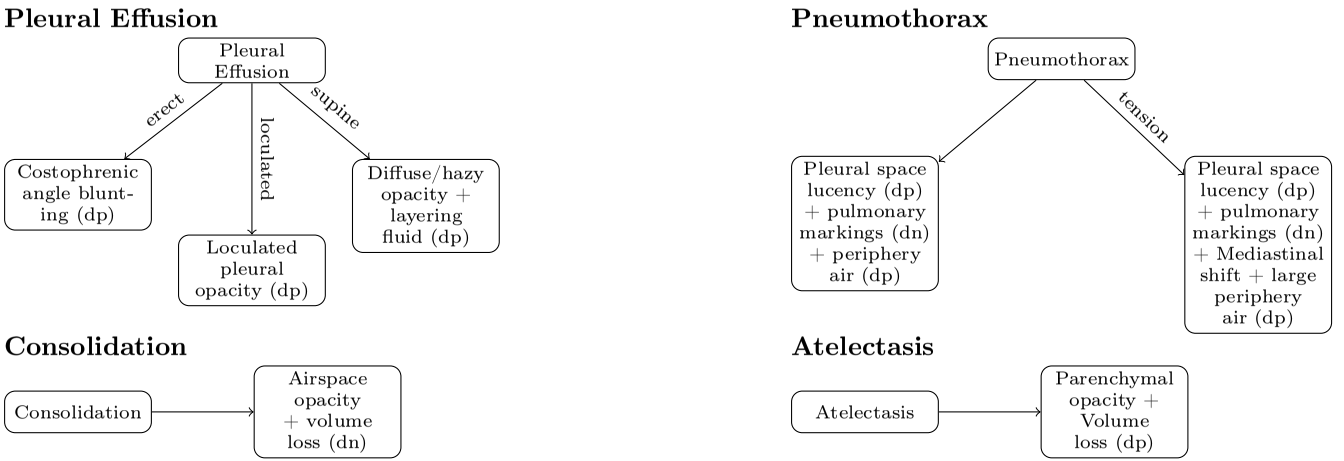


Figure 8.12: **Diagnostic pathway panels** (Part 1 of 4) illustrating exemplar pathways from the hand-crafted dictionary used by the pathway matching algorithm. Each panel is a small directed graph whose root node is a normalized diagnosis (*finding*) and whose children are normalized sub-finding patterns over location, attributes, view, and (optionally) clinical context (e.g., patient symptom). Edges are annotated with the attribute or view modifiers that the matcher uses when checking joint compatibility between an input tuple $(finding, loc, attr, view)$ and a pathway variant. Here, *dp* stands for definitive positive, while *dn* stands for definitive negative. Shown pathways correspond to: Pleural Effusion, Pneumothorax, Consolidation, and Atelectasis. For Pleural Effusion, the branches encode erect versus supine presentations and a loculated variant. For Pneumothorax, the variants capture simple pneumothorax, tension pneumothorax, and hydropneumothorax. Consolidation and Atelectasis map directly to their canonical imaging signatures.

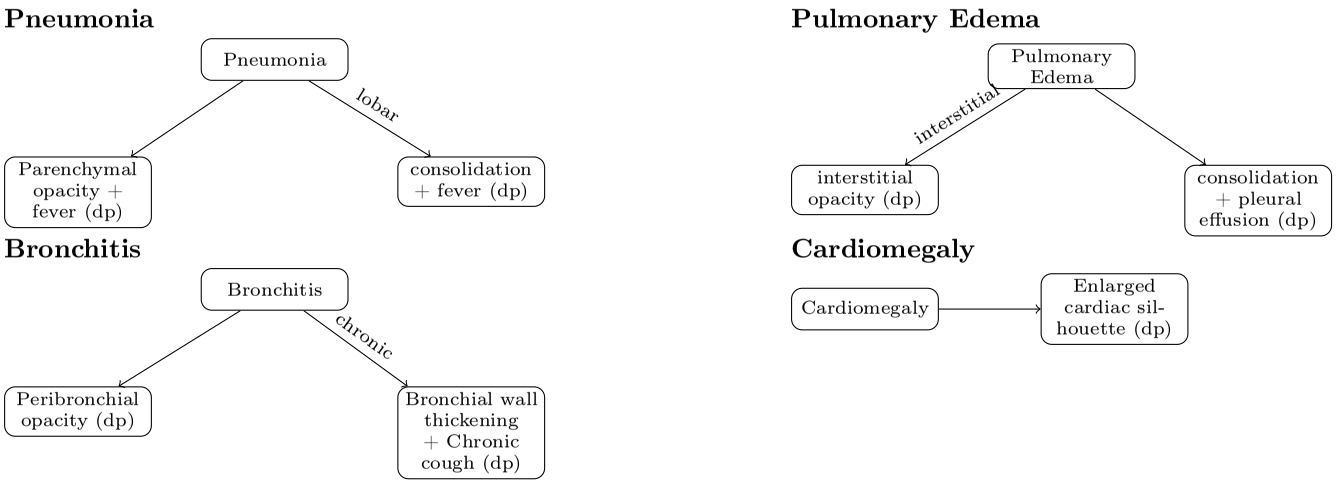


Figure 8.13: **Diagnostic pathway panels** (Part 2 of 4). Shown pathways correspond to: Pneumonia, Pulmonary Edema, Bronchitis, and Cardiomegaly. For Pneumonia and Pulmonary Edema, the pathways encode typical parenchymal and interstitial/alveolar patterns (consolidation) together with key clinical or co-occurring radiographic findings (e.g., fever, pleural effusion). Bronchitis is represented as an airway-centered process, while Cardiomegaly maps directly to its canonical imaging signature.

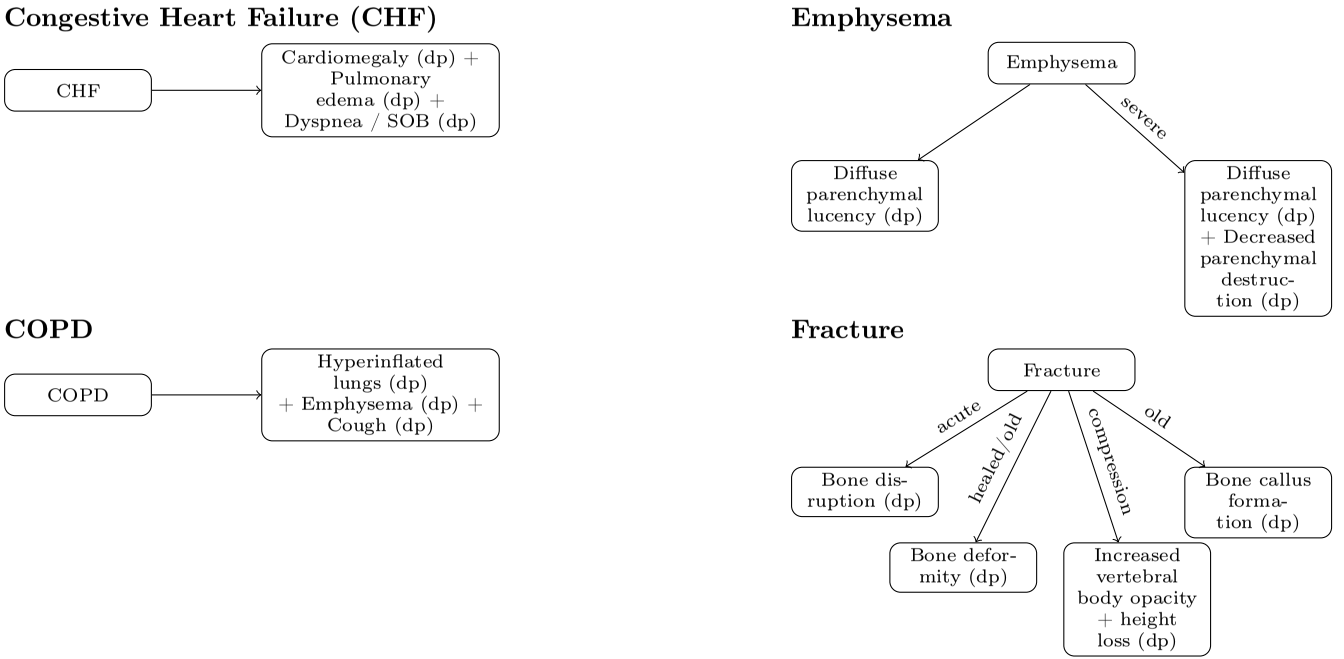


Figure 8.14: **Diagnostic pathway panels** (Part 3 of 4). Shown pathways correspond to CHF, Emphysema, COPD, and Fracture. CHF is modeled as a cardiopulmonary congestion cluster, where cardiomegaly, pulmonary edema, and dyspnea co-occur during exacerbation. Emphysema encodes a severity axis from diffuse parenchymal lucency to severe parenchymal destruction with decreased peripheral vascularity. COPD is represented as a chronic obstructive cluster combining hyperinflated lungs, emphysematous change, and chronic cough. Fracture pathways separate acute cortical breaks, old or healed deformity, and vertebral compression patterns.

Figure 8.15: **Diagnostic pathway panels** (Part 4 of 4). Shown pathways correspond to Tuberculosis and Lung Cancer. Tuberculosis includes an explicit disease-state axis, contrasting active parenchymal opacity with old or inactive upper-lobe scarring that shows calcified nodules and architectural distortion. Lung Cancer is represented by a focal nodular or mass-like opacity that may reflect primary or metastatic disease.

Table 8.6: **Diagnostic pathways** (Part 1 of 3). `view` denotes projection or patient orientation; `ent`, `status`, `loc`, and `attr` indicate the entity, status (`dp` is definitive positive, `dn` is definitive negative), anatomical location, and attributes (e.g., morphology, distribution, measurement). Pathways are ordered sequences joined by "`>`", with required co-occurrence marked by "`&&`". We display one representative pathway out of nine defined for pleural effusion and, in total, 33 pathways spanning 14 diagnoses.

Diagnosis	**Specific diagnosis**	**Diagnostic pathways**
Pleural Effusion	Non-loculated pleural effusion *(one of nine pathways)*	`view: ap, pa, lateral > ent: blunting > status: dp > loc: costophrenic angle && ent: opacity > status: dp > loc: pleural space > attr: hazy, diffuse`
	Loculated pleural effusion	`view: ap, pa, lateral > ent: opacity > status: dp > loc: pleural space > attr: loculated`
Pneumo-thorax	Pneumothorax	`view: ap, pa, lateral > ent: lucency > status: dp > loc: pleural space && ent: marking > status: dn > loc: pulmonary && ent: air > status: dp > loc: lung periphery`
	Tension pneumothorax	`Pneumothorax && ent: shift > status: dp > loc: mediastinal && attr: large amount`
Consolidation		`view: ap, pa, lateral > ent: opacity > status: dp > loc: airspace && ent: volume loss > status: dn`
Atelectasis		`view: ap, pa, lateral > ent: opacity > status: dp && ent: volume loss > status: dp`

Table 8.7: **Diagnostic pathways** (Part 2 of 3).

Diagnosis	**Specific diagnosis**	**Diagnostic pathways**
Pneumonia	Pneumonia	`view: ap, pa, lateral > ent: opacity > status: dp && ent: fever > status: dp`
	Aspiration pneumonia	`Pneumonia > loc: lung > attr: dependent portion`
	Lobar pneumonia	`view: ap, pa, lateral > ent: consolidation > status: dp && ent: fever > status: dp`
Pulmonary Edema	Interstitial pulmonary edema	`view: ap, pa, lateral > ent: opacity > status: dp > loc: interstitial`
	Alveolar pulmonary edema	`view: ap, pa, lateral > ent: consolidation > status: dp && ent: pleural effusion > status: dp`
Bronchitis	Bronchitis	`view: ap, pa, lateral > ent: opacity > status: dp > loc: peribronchial`
	Chronic Bronchitis	`view: ap, pa, lateral > ent: thickening > status: dp > loc: bronchial wall && ent: cough > status: dp`
Cardiomegaly		`view: ap, pa > ent: heart size > status: dp > loc: cardiothoracic > attr: increased`
CHF		`view: ap, pa, lateral > ent: cardiomegaly > status: dp && ent: pulmonary edema > status: dp && ent: dyspnea > status: dp`

Table 8.8: **Diagnostic pathways** (Part 3 of 3).

Diagnosis	Specific diagnosis	Diagnostic pathways
Emphysema	Emphysema	view: ap, pa, lateral > ent: lucency > status: dp > loc: lung parenchyma > attr: diffuse
	Severe emphysema	Emphysema && ent: pulmonary vascularity > status: dp > attr: decreased && ent: destruction > status: dp > loc: lung parenchyma
COPD		view: ap, pa, lateral > ent: hyperinflation > status: dp > loc: lungs && ent: emphysema > status: dp && ent: cough > status: dp
Fracture	Acute fracture	view: ap, pa, lateral > ent: disruption > status: dp > loc: bone
	Chronic/Old fracture	view: ap, pa, lateral > ent: callus formation > status: dp > loc: bone
	Healed fracture	view: ap, pa, lateral > ent: deformity > status: dp > loc: bone
	Spinal/Compression fracture	view: ap, pa, lateral > ent: opacity > status: dp > loc: vertebral body > attr: increased && ent: loss of height > status: dp
Tuberculosis	Active tuberculosis	view: ap, pa, lateral > ent: opacity > status: dp
	Chronic/Non-active tuberculosis	view: ap, pa, lateral > ent: nodules > status: dp > loc: bilateral upper lung > attr: calcified && ent: architectural distortion > status: dp > loc: bilateral upper lung
Lung Cancer		view: ap, pa, lateral > ent: opacity > status: dp > attr: nodular

Second, we enforce *location and attribute compatibility*. For example, location compatibility distinguishes skeletal fractures from device fractures even when the normalized finding f' is "fracture". Attribute terms then select more specific variants: "tension" triggers the tension pneumothorax pathway, "loculated" the loculated effusion pathway, "compression" the compression fracture pathway, and acuity modifiers such as "acute," "old," or "healed" select the corresponding fracture branch.

Finally, we check *view compatibility* using `view_information`, retaining only variants that are observable under the given view (e.g., erect versus supine presentations of pleural effusion, as discussed earlier). By construction, these compatibility rules are defined so that at most one pathway variant remains for any normalized input tuple. If a unique compatible variant exists, `BestPathwayMatch` returns it as $c^\star$; otherwise it returns *none*.

Status Conflict Resolution During pathway expansion we observed status conflicts for the same finding, and we also identified inconsistencies within original reports. In Table 8.10, we categorize conflicts by *source* and *type* and count their occurrence in LUNGUAGE. Sources are `original_vs_expansion` (an expanded node contradicts an explicitly stated node), `original_vs_original` (two original nodes disagree), and `expansion_vs_expansion` (two expanded nodes disagree). Types are `polarity` (`dp` vs. `dn`), `certainty_positive` (`dp` vs. `tp`), `certainty_negative` (`dn` vs. `tn`), `duplicate_pos` (`tn` vs. `tn`) and `duplicate_neg` (`tp` vs. `tp`).

Conflict detection operates at the granularity of (`entity`, `location`) within each report with the same `study_id`. Across the expanded LUNGUAGE dataset (19,216 rows), expansion-related conflicts were rare overall, totaling 616 cases (**3.2%**): 165 (**0.9%**) `original_vs_expansion` and 451 (**2.3%**) `expansion_vs_expansion`. Within `original_vs_expansion` (n=165), conflicts due to polarity of definitive findings dominated, with `polarity` accounting for 107 (64.8%) conflicts, `certainty_pos` for 34 (20.6 %), and `certainty_neg` for 24 (14.5%). Within `expansion_vs_expansion` (n=451), conflicts were dominated by duplicates of positively tentative findings: `polarity` accounted for 103 (22.8%) conflicts, `certainty_pos` for 133 (29.5%), `certainty_neg` for 16 (3.5%), `duplicate_pos` for 181 (40.1%), and `duplicate_neg` for 18 (4.0%). In addition, some inconsistencies were already present in the original reports: `original_vs_original` totaled 158 cases (**0.8%**). These conflicts were dominated by positive certainty conflicts: `polarity` accounted for 10 (6.3%) conflicts, `certainty_pos` for 74 (46.8%), `certainty_neg` for 2 (1.3%), `duplicate_pos` for 70 (44.3%), and `duplicate_neg` for 2 (1.3%). Overall conflicts across all sources summed to 774 (**4.0%** of the initial expanded LUNGUAGE dataset).

Resolution follows a deterministic policy: (1) if a group contains both original and expansion rows, retain the originals and discard the expansions (treat the report text as the clinical source of truth); (2) if the remaining

rows show a pure polarity clash with only `dp` and `dn`, drop the group; (3) otherwise select the row with the highest `prob`. When multiple rows tie on `prob`, break ties by a status priority that reflects clinical certainty, `dp` > `tp` > `tn` > `dn`. After resolution, **18,810** rows remained (**97.9%** of 19,216), implying **406** removals (**2.1%**); all remaining inconsistencies were eliminated, preserving logical consistency and clinical validity. Algorithm 4 details the procedure.

Table 8.9: **Manually curated blacklist** of substring pairs that prevent merges during high-precision deduplication.

Category	**Non-merge pairs (substring-level)**
Laterality and anatomical direction	(left, right), (left-sided, right-sided), (left-sided, right), (left, right-sided), (upper, lower), (mid, lower), (upper, mid), (upper, middle), (middle, lower), (anterior, posterior), (anterior, lateral), (posterior, lateral), (superior, inferior), (apical, basal), (central, peripheral), (proximal, distal), (medial, lateral), (ventral, dorsal)
Cardiopulmonary contextual conflicts	(cardio, pulmonary), (mediastinal, pleural), (pericardial, pleural)
Disease mechanism divergence	(effusion, pneumothorax), (effusion, edema), (effusion, atelectasis), (effusion, consolidation), (effusion, pneumonia), (consolidation, opacification), (consolidation, atelectasis), (consolidation, edema), (atelectasis, pneumonia), (atelectasis, aeration), (atelectasis, opacity), (pneumonia, edema)
Mass vs. airspace process	(mass, consolidation), (mass, opacity), (mass, atelectasis)

Algorithm 3 Pathway Matching

Input: (f, ℓ, a, v) from LUNGUAGE, pathway dict $\mathcal{D}$ *// f: finding, ℓ: location, a: attributes, v: view*
Output: $c^\star$ or *none*
function SELECTVARIANT($f, \ell, a, v, \mathcal{D}$)
  $(f', \ell', a', v') \leftarrow$ NormalizeInput(f, ℓ, a, v) *// normalize row-level finding, location, attributes, and view*
  $\mathcal{D}' \leftarrow$ NormalizeDict$(\mathcal{D})$ *// normalize findings, locations, attributes, and views in all pathways*
  $c^\star \leftarrow$ BestPathwayMatch$(\mathcal{D}', f', \ell', a', v')$
  return $c^\star$
end function
function BESTPATHWAYMATCH($\mathcal{D}', f', \ell', a', v'$)
  $C \leftarrow \{\, c \in \mathcal{D}' : \text{Find}(c) = f' \,\}$ *// finding-level lexical match*
  if $C = \emptyset$ **then**
    return *none*
  end if
  $C \leftarrow \{\, c \in C : \text{LocOK}(c, \ell') \wedge \text{AttrOK}(c, a') \wedge \text{ViewOK}(c, v') \,\}$ *// joint compatibility on location, attributes, and view*
  if $|C| = 1$ **then**
    return the sole element of C
  else
    return *none* *// ambiguous or incompatible (should not occur since pathways are independent)*
  end if
end function

Table 8.10: **Pathway expansion conflicts** by source and type. *ori* and *exp* denote original and expansion.

Conflict	**Type**	exp ↔ exp	ori ↔ exp	ori ↔ ori
`dn` ↔ `tn`	`certainty_neg`	16	0	2
`dp` ↔ `tp`	`certainty_pos`	133	34	74
`tn` ↔ `tn`	`duplicate_neg`	18	0	2
`tp` ↔ `tp`	`duplicate_pos`	181	24	70
`dp` ↔ `dn`	`polarity`	103	107	10
		451	165	158

Algorithm 4 Conflict Resolution

Input: Expanded dataset D with columns: `study_id`, `entity`, `location`, `status` $\in$ {`dp`, `tp`, `tn`, `dn`}, `prob`, `source` $\in$ {`original`, `expansion`}
Output: `resolved_df`

keys $\leftarrow$ [`study_id`, `entity`, `location`]
`RES` $\leftarrow$ [ ]
for all group $G \subset D$ by `keys` **do**
  if $|G| < 2$ **then continue**
  end if
  `orig` $\leftarrow$ { $r \in G \mid r.$`source` $=$ `original` }
  `expd` $\leftarrow$ { $r \in G \mid r.$`source` $=$ `expansion` }
  — Case 1: original vs. expansion —
  if `orig` $\neq \varnothing$ **and** `expd` $\neq \varnothing$ **then**
  Rule1: keep originals only; discard expansions
    append all rows in `orig` to `RES`
    continue
  end if
  — Case 2: only originals or only expansions —
  $R \leftarrow$ `orig` if `orig` $\neq \varnothing$ else `expd`
  $has\_dp \leftarrow (\exists r \in R :\ r.$`status` $=$ `dp`$)$
  $has\_dn \leftarrow (\exists r \in R :\ r.$`status` $=$ `dn`$)$
  $has\_tp \leftarrow (\exists r \in R :\ r.$`status` $=$ `tp`$)$
  $has\_tn \leftarrow (\exists r \in R :\ r.$`status` $=$ `tn`$)$
  Rule2: drop if pure polarity clash
  if has_dp **and** has_dn **and not**$(has\_tp$ **or** $has\_tn)$ **then**
    continue
  end if
  Rule 3: pick highest probability; dp > tp > tn > dn
  $p^* \leftarrow \max\{\, r.$`prob` $: r \in R \,\}$
  if $\exists r \in R : r.$`prob` $= p^* \wedge r.$`status` $=$ `dp` **then**
    append any such r to `RES`
  else if $\exists r \in R : r.$`prob` $= p^* \wedge r.$`status` $=$ `tp` **then**
    append any such r to `RES`
  else if $\exists r \in R : r.$`prob` $= p^* \wedge r.$`status` $=$ `tn` **then**
    append any such r to `RES`
  else
    append any $r \in R$ with $r.$`prob` $= p^*$ to `RES`
  end if
end for
`resolved_df` $\leftarrow$ Concat(`RES`)
return `resolved_df`

9 Conclusions and Future Work

We conclude this work by answering the research questions posed in Chapter 3. Along the way, we highlight how the results presented across the previous five chapters address the three major challenges for AI in healthcare introduced in Chapter 1. We then comment on potential future research directions.

9.1 Conclusions

RQ1: Can neural networks mimic the causal and probabilistic properties of a Bayesian network? (Chapter 4)

Our results in Chapter 4 illustrated how a neural network can indeed serve as an understudy to a BN, both approximating the probabilistic behavior and being informed by the causal properties of its BN counterpart.

However, this approach will not scale well to larger BNs. As the number of independence relations grows (which already happens for medium-sized networks with up to 10 variables) the training process will become increasingly inefficient. This scalability issue is exacerbated by the loss of modularity: whereas a BN models each conditional distribution separately, the neural understudy must learn all conditional distributions within a single monolithic architecture. Finally, unlike a BN, which embeds expert knowledge explicitly in its structure, the neural understudy only incorporates such knowledge implicitly through training constraints, limiting its interpretability and thereby its ability to address challenge 1.

We conclude that extending BNs with the ability to incorporate unstructured text provides a more promising path forward than attempting to adapt neural models to mimic BNs, as the latter approach risks losing the very strengths that make BNs valuable. This is indeed the path that was followed in subsequent chapters.

RQ2: How can we automate clinical reasoning across tabular and textual evidence? (Chapter 5)

Our results in Chapter 5 demonstrated the strengths of BNs for formalizing the clinical reasoning process, as well as the clear added value of incorporating unstructured text rather than relying solely on structured features.

To extend the BN with neural text representations, we first explored a generative BN-text model in which a text node is included directly in the network, and its conditional distribution is modeled as a multivariate Gaussian over pre-trained text embeddings, conditioned on its parent values. In practice, however, this parametric assumption proved too restrictive: the high dimensionality of the embeddings, combined with the limited number of samples for most parent configurations, led to unreliable parameter estimation.

We therefore proposed a second, discriminative BN-text model that uses a set of neural text classifiers to model the conditional distribution of any node in the BN that might be mentioned in the text. Although this approach yielded improved performance over the generative variant, it comes with two important limitations. First, it requires inverting the causal structure of the problem, thereby losing the intuitive cause-effect relationship between diagnoses and symptoms that underpins a BN. Second, it requires training a separate classifier for each parent configuration of each node, which limits scalability for nodes with many parents.

Despite these challenges, we showed that BNs remain a natural and powerful candidate for addressing both challenge 1 and challenge 2: they offer an interpretable, expert-informed framework for representing and reasoning about the probabilistic dependencies central to clinical decision making. By augmenting BNs with neural text representations, we enable Bayesian inference over textual evidence in addition to tabular evidence, thereby taking an important step toward addressing challenge 3.

RQ3: How can we overcome the lack of clinical benchmarks which explicitly link tabular data and text? (Chapter 6)

In Chapter 6, we built a self-contained simulated dataset of medical records comprising both structured tabular data and unstructured text. The resulting SimSUM dataset provides a controlled research environment for develop-

ing and prototyping novel methods aimed at integrating tabular and textual information, thereby directly supporting research that addresses challenge 3.

By open-sourcing SimSUM, we offer the research community a flexible and accessible sandbox for experimentation. Unlike real-world clinical datasets – which are constrained by strict privacy requirements, require special licenses to access, and are only allowed to be processed by local LLMs – SimSUM is free of such restrictions. Moreover, users have insight into the ground-truth data-generating process, including both the BN structure that governs the tabular data and the LLM-based prompts used to generate the clinical text. This transparency enables more rigorous and fine-grained evaluation of multi-modal methods than is possible with real-world data.

RQ4: How can we perform patient-level information extraction that consistently integrates tabular and textual evidence? (Chapter 7)

In Chapter 7, we introduced our patient-level information extraction method, which does not extract clinical concepts from unstructured text alone, but additionally leverages the structured tabular features already present in the patient's EHR. It does so in a modular fashion. Tabular features are modeled with a BN, which provides prior probabilities for each clinical concept given all available structured evidence. In parallel, clinical concepts are extracted from the text using a neural classifier built on pre-trained embeddings.

We then integrated these two sources of evidence in an interpretable and fully probabilistic manner, using virtual evidence augmented with our novel consistency node. Virtual evidence injects the text classifier's predictions into the BN, while the consistency node explicitly combines predictions from both models. Together, they enable the BN to (i) supply background information that may be missing from the text but useful for more accurate concept extraction, and (ii) adjust the text classifier's predictions when they conflict with tabular evidence.

We can compare this approach with the generative and discriminative BN-text models introduced in Chapter 5. There, the textual evidence was directly integrated into the conditional probability distributions that made up the BN, which allowed textual evidence to be directly included in the Bayesian inference procedure. In the present method, the BN and the text classifier are decoupled, and their predictions are fused modularly. This separation makes it straightforward for an end user to inspect each model's output. It also resolves two limitations of the discriminative BN–text model: (i) the need to learn a separate conditional text classifier for every combination of parent states for each variable mentioned in the text, and (ii) the implicit absorption of parent–child relationships by these conditional text

classifiers, which obscured the explicit dependencies encoded in the BN.

In summary, Chapter 7 addressed challenge 3 in the context of clinical information extraction. By enabling the construction of structured tabular datasets suitable for interpretable downstream models, it also contributes to challenge 1. Our modular design allows end users to examine the predictions of both the BN and the text classifier, further supporting challenge 1. Finally, because our method is uncertainty-aware and fuses predictions probabilistically, it aligns with the goals of challenge 2.

RQ5: How can we model clinical uncertainty while extracting information from medical reports? (Chapter 8)

In Chapter 8, we argued that medical reports contain two types of uncertainty: *explicit uncertainty*, which reflects doubt about the presence or absence of findings; and *implicit uncertainty*, which arises when clinicians omit part of their reasoning when writing reports, recording only key findings or diagnoses. We modeled explicit uncertainty by analyzing the language of the reports, specifically the *hedging phrases* that signal diagnostic doubt. We used LLMs to build a reference ranking of hedging phrases, mapping each finding in the report to a probability value based on its relative position in this reference. We handled implicit uncertainty by expanding the structured findings mentioned in the reports with characteristic sub-findings that are not explicitly recorded. This expansion relies on expert-defined diagnostic pathways, which systematically fill in missing information for common diagnoses while respecting the explicit uncertainty associated with each finding.

We addressed both types of uncertainty in the context of structuring radiology reports, building a tabular dataset of clinical concepts from unstructured text. In summary, Chapter 8 again tackled challenge 3 in the domain of clinical information extraction, while also addressing challenge 2 by modeling and propagating uncertainty throughout the extraction process.

9.2 Future work

Clinical reasoning with virtual evidence In Chapter 5, we used generative and discriminative BN-text models to incorporate textual evidence into the BN for clinical reasoning. Later, Chapter 7 introduced the virtual evidence and consistency node approaches, which overcome several limitations of those earlier models, as discussed previously. The virtual evidence method, in particular, appears especially promising for integrating text into BN-based clinical reasoning. Testing this idea with the SimSUM dataset would be a natural next step: rather than treating symptoms as prediction targets for CIE (as in Chapter 7), we would instead predict diagnoses (as

in Chapter 5), while still injecting the text classifier's predictions into the symptom nodes via virtual evidence.

Future use of SimSUM While we demonstrate the utility of SimSUM in Chapter 7, where it supports the development of new methods for incorporating background knowledge into CIE, we envision a much broader range of future applications. These include benchmarking approaches for clinical reasoning that integrate both tabular and textual evidence, as discussed in the previous paragraph.

Beyond CIE and clinical reasoning, SimSUM is also well suited for research in causal inference. This field has a clear need for synthetic benchmarks with a known data-generating mechanism – something real-world datasets cannot provide – so that recovered causal effects can be evaluated against ground truth. Indeed, SimSUM has already been used to benchmark causal effect estimation methods in the presence of textual confounding [Arno et al., 2024, Arno and Demeester, 2025], for which it proved a good fit. The same advantages extend to causal discovery methods, which aim to recover causal relationships directly from data [Glymour et al., 2019]. SimSUM serves as a particularly useful testbed for these methods: not only is the ground-truth BN underlying the tabular data known, but the dataset also poses realistic challenges by incorporating non-categorical variables, diverse distribution types, and free-text clinical notes.

Validation on real data So far, all our methods for integrating text into BNs have been evaluated on simulated data, primarily to maintain a controlled research environment. While this provides an appropriate and necessary testbed for initial development, it can create an overly optimistic picture of the methods' real-world performance. Actual clinical tabular data are far more challenging: they contain missing values, errors, and substantial coding variability across clinicians and institutions [Aung et al., 2021]. Likewise, real clinical notes can be very long, filled with hospital- or clinician-specific jargon, and organized into numerous subsections that vary by institution [Tayefi et al., 2021]. In contrast, the simulated notes used in Chapters 5-7 are relatively clean and consistently structured. Moreover, real clinical data have an inherent temporal dimension, as a patient's status evolves over time. Although it is sometimes reasonable to abstract away these dynamics in diagnostic settings (such as our SimSUM use case of respiratory disease diagnosis in primary care) they are often crucial in other contexts, particularly in in-hospital environments like intensive care.

Several concrete avenues exist for validating our approach using real-world data. One option would be to validate the BN developed for diagnosing respiratory diseases in primary care settings. This would require access to primary care data; however, to our knowledge, no publicly available datasets exist in this domain, particularly those containing de-identified

patient consultation notes. Consequently, we would need to establish a dedicated study in collaboration with primary care practices in Flanders to collect such data. Alternatively, we could explore the semi-public MIMIC-III and MIMIC-IV datasets [Johnson et al., 2016, 2023], which encompass both structured and unstructured patient information. Nevertheless, since these data originate from hospital rather than primary care settings, our existing BN would not be directly applicable. This approach would necessitate developing a new use case and constructing a novel BN in collaboration with domain specialists. The most promising avenue may be leveraging the MIMIC-CXR subset [Johnson et al., 2019b]. In this scenario, we could engage radiologists to expand the diagnostic pathways presented in Chapter 8 into comprehensive BNs for selected diagnoses, training these models on the structured data available in the Lunguage dataset [Moon et al., 2025]. The complete radiology reports would serve as textual evidence, with the potential for incorporating chest X-ray images as a third data modality.

BN DAG development A further simplification in our experiments is that, although the BN parameters were learned from data, the DAG structure was assumed to be known and set to the ground-truth structure. In real applications, however, the ground-truth DAG is not available, and expert input is required to construct one. Although challenging, such expert-designed DAGs have been developed successfully in prior work [McLachlan et al., 2020b, Edye et al., 2021]. To ensure that the resulting structure reflects clinical reality as closely as possible, input from a panel of experts is typically needed. The intended use case must also be sufficiently focused and well-defined, so that all relevant variables can be modeled. Success in this process depends heavily on providing experts with clear guidance and training on DAG construction, for which the guidelines of Kyrimi et al. [2020] can be particularly helpful.

Validation in clinical practice Beyond validation on real data, any CDSS must ultimately be evaluated in clinical practice. Integrating such a system into the clinical workflow is a substantial non-technical challenge. Even if the three major technical challenges introduced in Chapter 1 are fully addressed – resulting in a system that is technically well-designed – this by no means guarantees adoption in real clinical settings. Practical implementation introduces numerous obstacles, including data acquisition, privacy concerns (e.g., pseudonymization of text data), regulatory requirements, financial cost, and the willingness of clinicians to adapt their workflow, among many others [Sterckx and Van Biesen, 2024, Aung et al., 2021, Henzler et al., 2025].

Crucially, a CDSS must be rigorously evaluated within the clinical workflow in which it will be deployed [Wiens et al., 2019, Lenharo, 2024]. It must be demonstrated that decisions made with the system lead to better patient

outcomes than decisions made without it. Furthermore, just as strong performance in a research setting does not guarantee strong performance in a clinical setting, evidence from a single study in one environment cannot be assumed to generalize to others, given differences in workflows, patient populations, institutional cultures, and resource availability.

Finally, deploying a CDSS is not a one-off achievement. It requires ongoing monitoring and iterative refinement to ensure that it continues to meet its intended goals and genuinely improves patient care [Sterckx and Van Biesen, 2024].

Extension to other modalities Although the methods described in Chapter 7 were designed to integrate tabular data and text, they can be readily extended to include additional modalities. Both the virtual evidence approach and the consistency node flexibly accept probabilistic inputs without imposing constraints on the underlying methodology used to generate these probabilities. Consequently, replacing the text classifier with an image classifier enables the incorporation of information derived from 2D or 3D imaging data, and this principle extends equally to other biosignal types or video data. In particular, we envision the use case of jointly extracting information from chest X-ray images, radiology reports, and tabular data. Here, complementary features extracted from both the chest X-ray and the corresponding radiology report could be integrated into the BN using multiple virtual evidence nodes, thereby enabling clinical reasoning that spans three distinct modalities: tabular, textual, and visual data.

From BN to diagnostic pathway Building a full BN may become infeasible in settings where the diagnostic landscape is too large or where modeling all relevant factors is overly complex for the given use case. In such cases, a more lightweight alternative may be preferable, like the diagnostic pathways introduced in Chapter 8. In this approach, radiologists are asked to specify the key findings they expect to observe at least 80% of the time when encountering a particular diagnosis. Like a BN, a diagnostic pathway is a DAG that encodes cause-effect relations: given a diagnosis, we expect to observe a characteristic set of symptoms or findings. Unlike a BN, however, these pathways impose much looser probabilistic requirements; experts are not required to specify full conditional independence assumptions. As a result, diagnostic pathways do not define a joint probability distribution, a defining feature of BNs.

Despite these limitations, diagnostic pathways offer a practical alternative when constructing a full BN is too ambitious. While they cannot answer probabilistic queries, they can still be used to fill gaps in clinical reasoning by inferring likely but unreported findings. For this reason, the use of diagnostic pathways may be worth exploring in contexts beyond radiology reports.

Probability quantification from natural language When probabilities cannot be learned from structured tabular data – whether because such data are unavailable, or because no meaningful BN structure can be defined – we must obtain these probabilities through alternative means. Chapter 8 demonstrated the potential of estimating uncertainty directly from the natural language used to describe a finding. There, we combined these language-derived probabilities with diagnostic pathway DAGs to expand reported findings in an uncertainty-aware manner.

Building on this idea, an interesting future research direction is to use the linguistically derived probability of a finding as virtual evidence in a BN. This may offer advantages over relying solely on the predicted probabilities of a neural text classifier, as done in Chapter 7. Neural networks are well known to suffer from calibration issues, meaning their output probabilities do not always correspond to true event likelihoods. By contrast, probabilities derived from clinicians' own reporting language are more directly grounded in expert judgment. A hybrid approach of combining both the classifier's predicted probabilities and the linguistically inferred probabilities as virtual evidence could prove even more powerful.

References

Ankur Ankan and Abinash Panda. pgmpy: Probabilistic Graphical Models using Python. In *Proceedings of the 14th Python in Science Conference*, pages 6 – 11, 2015.

Anthropic. Claude sonnet. `https://www.anthropic.com/transparency/model-report`, 2025.

Henri Arno and Thomas Demeester. Personalized treatment effect estimation from unstructured data. *arXiv preprint arXiv:2507.20993*, 2025.

Henri Arno, Paloma Rabaey, and Thomas Demeester. From text to treatment effects: A meta-learning approach to handling text-based confounding. In *Causal Representation Learning Workshop at NeurIPS 2024*, 2024.

P. Arora, D. Boyne, J. J. Slater, A. Gupta, D. R. Brenner, and M. J. Druzdzel. Bayesian Networks for Risk Prediction Using Real-World Data: A Tool for Precision Medicine. *Value Health*, 22(4):439–445, 2019a.

Paul Arora, Devon Boyne, Justin J. Slater, Alind Gupta, Darren R. Brenner, and Marek J. Druzdzel. Bayesian networks for risk prediction using real-world data: A tool for precision medicine. *Value in Health*, 22(4), 2019b.

Yuri YM Aung, David CS Wong, and Daniel SW Ting. The promise of artificial intelligence: a review of the opportunities and challenges of artificial intelligence in healthcare. *British medical bulletin*, 139(1):4–15, 2021.

Shruthi Bannur, Kenza Bouzid, Daniel C Castro, Anton Schwaighofer, Anja Thieme, Sam Bond-Taylor, Maximilian Ilse, Fernando Pérez-García, Valentina Salvatelli, Harshita Sharma, et al. Maira-2: Grounded radiology report generation. *arXiv preprint arXiv:2406.04449*, 2024.

Pietro Barbiero, Francesco Giannini, Gabriele Ciravegna, Michelangelo Diligenti, and Giuseppe Marra. Relational concept bottleneck models. *Advances in Neural Information Processing Systems*, 37:77663–77685, 2024.

Leonard Berlin. Pitfalls of the vague radiology report. *American Journal of Roentgenology*, 174(6):1511–1518, 2000.

Eta S Berner and Tonya J La Lande. Overview of clinical decision support systems. In *Clinical decision support systems: Theory and practice*, pages 1–17. Springer, 2016.

Bingli. Bingli: Ai buided diagnosis, disease management, and care navigation, 2025. URL `https://www.bingli.eu/en`. Accessed: 2025-11-25.

Philippe Brouillard, Sébastien Lachapelle, Alexandre Lacoste, Simon Lacoste-Julien, and Alexandre Drouin. Differentiable causal discovery from interventional data. In *Advances in Neural Information Processing Systems*, volume 33, pages 21865–21877. Curran Associates, Inc., 2020.

Michael A Bruno, Jonelle Petscavage-Thomas, and Hani H Abujudeh. Communicating uncertainty in the radiology report. *American Journal of Roentgenology*, 209(5):1006–1008, 2017.

Andrew L Callen, Sara M Dupont, Adi Price, Ben Laguna, David McCoy, Bao Do, Jason Talbott, Marc Kohli, and Jared Narvid. Between always and never: evaluating uncertainty in radiology reports using natural language processing. *Journal of Digital Imaging*, 33(5):1194–1201, 2020.

Danton S Char, Nigam H Shah, and David Magnus. Implementing machine learning in health care—addressing ethical challenges. *The New England journal of medicine*, 378(11):981, 2018.

B. Chin-Yee and R. Upshur. Clinical judgement in the era of big data and predictive analytics. *J Eval Clin Pract*, 24(3):638–645, 2018.

Edward Choi, Mohammad Taha Bahadori, Le Song, Walter F Stewart, and Jimeng Sun. Gram: graph-based attention model for healthcare representation learning. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 787–795, New York, NY, USA, 2017. Association for Computing Machinery.

Edward Choi, Cao Xiao, Walter Stewart, and Jimeng Sun. Mime: Multilevel medical embedding of electronic health records for predictive healthcare. *Advances in neural information processing systems*, 31:4552–4562, 2018.

Edward Choi, Zhen Xu, Yujia Li, Michael Dusenberry, Gerardo Flores, Emily Xue, and Andrew Dai. Learning the graphical structure of electronic health records with graph convolutional transformer. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 606–613, 2020.

Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, et al. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*, 2025.

Corilus. Careconnect ai assistant, 2025. URL https://ai.corilus.be/. Accessed: 2025-11-25.

Jesse Davis and Mark Goadrich. The relationship between precision-recall and roc curves. In *Proceedings of the 23rd International Conference on Machine Learning*, page 233–240, 2006.

Alexander Decruyenaere, Heidelinde Dehaene, Paloma Rabaey, Johan Decruyenaere, Christiaan Polet, Thomas Demeester, and Stijn Vansteelandt. Debiasing synthetic data generated by deep generative models. *Advances in Neural Information Processing Systems*, 37:41539–41576, Dec 2024a.

Alexander Decruyenaere, Heidelinde Dehaene, Paloma Rabaey, Christiaan Polet, Johan Decruyenaere, Stijn Vansteelandt, and Thomas Demeester. The real deal behind the artificial appeal: Inferential utility of tabular synthetic data. In Negar Kiyavash and Joris M. Mooij, editors, *Proceedings of the Fortieth Conference on Uncertainty in Artificial Intelligence*, volume 244 of *Proceedings of Machine Learning Research*, pages 966–996. PMLR, 15–19 Jul 2024b. URL https://proceedings.mlr.press/v244/decruyenaere24a.html.

JJ Deeks, J Dinnes, and HC Williams. Sensitivity and specificity of skinvision are likely to have been overestimated. *Journal of the European Academy of Dermatology and Venereology*, 34(10):e582–e583, 2020.

Karel D'Oosterlinck, François Remy, Johannes Deleu, Thomas Demeester, Chris Develder, Klim Zaporojets, Aneiss Ghodsi, Simon Ellershaw, Jack Collins, and Christopher Potts. BioDEX: Large-scale biomedical adverse drug event extraction for real-world pharmacovigilance. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 13425–13454, December 2023.

Ernesto Ocampo Edye, Juan Francisco Kurucz, Lucas Lois, Agustín Paredes, Francisco Piria, Josefina Rodríguez, and Silvia Herrera Delgado. Applying Bayesian networks to help physicians diagnose respiratory diseases in the context of covid-19 pandemic. In *2021 IEEE URUCON*, pages 368–371, 2021.

European Union. Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data (General Data Protection Regulation). https://eur-lex.europa.eu/eli/reg/2016/679/oj, 2016.

European Union. Regulation (eu) 2024/1689 of the european parliament and of the council of 13 june 2024 laying down harmonised rules on artificial intelligence and amending regulations (ec) no 300/2008, (eu) no 167/2013, (eu) no 168/2013, (eu) 2018/858, (eu) 2018/1139 and (eu) 2019/2144

and directives 2014/90/eu, (eu) 2016/797 and (eu) 2020/1828 (artificial intelligence act) (text with eea relevance). `http://data.europa.eu/eli/reg/2024/1689/oj`, Jul 2024.

Xiaojuan Feng and Christopher KI Williams. Training bayesian networks for image segmentation. In *Mathematical Modeling and Estimation Techniques in Computer Vision*, volume 3457, pages 82–92. SPIE, 1998.

Elizabeth Ford, John A Carroll, Helen E Smith, Donia Scott, and Jackie A Cassell. Extracting information from the text of electronic medical records to improve case detection: a systematic review. *J Am Med Inform Assoc*, 23(5):1007–1015, 2016. ISSN 1067-5027.

Víctor Garcia Satorras and Max Welling. Neural enhanced belief propagation on factor graphs. In *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*, volume 130 of *Proceedings of Machine Learning Research*, pages 685–693. PMLR, 2021.

Julia Gärtner, Pascal O Berberat, Martina Kadmon, and Sigrid Harendza. Implicit expression of uncertainty–suggestion of an empirically derived framework. *BMC Medical Education*, 20(1):83, 2020.

Atticus Geiger, Zhengxuan Wu, Hanson Lu, Josh Rozner, Elisa Kreiss, Thomas Icard, Noah Goodman, and Christopher Potts. Inducing causal structure for interpretable neural networks. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 7324–7338. PMLR, 2022.

Clark Glymour, Kun Zhang, and Peter Spirtes. Review of causal discovery methods based on graphical models. *Frontiers in genetics*, 10:524, 2019.

Lawrence R Goodman. *Felson's principles of chest roentgenology, a programmed text*. Elsevier Health Sciences, 2014.

L. D. Gruppen. Clinical reasoning: Defining it, teaching it, assessing it, studying it. *West J Emerg Med*, 18(1):4–7, 2017.

Jiaqi Guan, Runzhe Li, Sheng Yu, and Xuegong Zhang. A method for generating synthetic electronic medical record text. *IEEE/ACM transactions on computational biology and bioinformatics*, 18(1):173–182, 2019.

Richard B Gunderman. Biases in radiologic reasoning. *American Journal of Roentgenology*, 192(3):561–564, 2009.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. In *International conference on machine learning*, pages 1321–1330, Sydney, Australia, 2017. PMLR, JMLR.

Paul Hager, Friederike Jungmann, Robbie Holland, Kunal Bhagat, Inga Hubrecht, Manuel Knauer, Jakob Vielhauer, Marcus Makowski, Rickmer Braren, Georgios Kaissis, et al. Evaluation and mitigation of the limitations of large language models in clinical decision-making. *Nature medicine*, 30(9):2613–2622, 2024.

Udo Hahn and Michel Oleynik. Medical information extraction in the age of deep learning. *Yearbook of medical informatics*, 29(01):208–220, 2020.

Michael Harradon, Jeff Druce, and Brian E. Ruttenberg. Causal learning and explanation of deep neural networks via autoencoded activations, 2018. URL `http://arxiv.org/abs/1802.00541`.

Iryna Hartsock and Ghulam Rasool. Vision-language models for medical report generation and visual question answering: A review. *Frontiers in artificial intelligence*, 7:1430984, 2024.

Dennis Henzler, Sebastian Schmidt, Ayca Koçar, Sophie Herdegen, Georg L Lindinger, Menno T Maris, Marieke AR Bak, Dick L Willems, Hanno L Tan, Michael Lauerer, et al. Healthcare professionals' perspectives on artificial intelligence in patient care: a systematic review of hindering and facilitating factors on different levels. *BMC Health Services Research*, 25 (1):633, 2025.

Ralf Herbrich, Tom Minka, and Thore Graepel. Trueskill™: a bayesian skill rating system. *Advances in neural information processing systems*, 19, 2006.

Mikel Hernandez, Gorka Epelde, Ane Alberdi, Rodrigo Cilla, and Debbie Rankin. Synthetic data generation for tabular health records: A systematic review. *Neurocomputing*, 493:28–45, 2022.

Adele Hill, Dylan Morrissey, and William Marsh. What characteristics of clinical decision support system implementations lead to adoption for regular use? a scoping review. *BMJ Health & Care Informatics*, 31(1): e101046, 2024.

Geoffrey Hinton. Deep Learning—A Technology With the Potential to Transform Health Care. *Journal of the American Medical Association (JAMA)*, 320(11):1101–1102, 2018.

Jonathan L Hobby, BD Tom, C Todd, PW Bearcroft, and Adrian K Dixon. Communication of doubt and certainty in radiological reports. *The British Journal of Radiology*, 73(873):999–1001, 2000.

Kexin Huang, Jaan Altosaar, and Rajesh Ranganath. Clinicalbert: Modeling clinical notes and predicting hospital readmission. *arXiv preprint arXiv:1904.05342*, 2019.

Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*, 2024.

Mahmoud Ibrahim, Yasmina Al Khalil, Sina Amirrajab, Chang Sun, Marcel Breeuwer, Josien Pluim, Bart Elen, Gökhan Ertaylan, and Michel Dumontier. Generative ai for synthetic data across multiple medical modalities: A systematic review of recent developments and challenges. *Computers in biology and medicine*, 189:109834, 2025.

Jeremy Irvin, Pranav Rajpurkar, Michael Ko, Yifan Yu, Silviana Ciurea-Ilcus, Chris Chute, Henrik Marklund, Behzad Haghgoo, Robyn Ball, Katie Shpanskaya, et al. Chexpert: A large chest radiograph dataset with uncertainty labels and expert comparison. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 590–597, 2019.

Ayush Jain, David Way, Vishakha Gupta, Yi Gao, Guilherme de Oliveira Marinho, Jay Hartford, Rory Sayres, Kimberly Kanada, Clara Eng, Kunal Nagpal, et al. Development and assessment of an artificial intelligence-based tool for skin condition diagnosis by primary care physicians and nurse practitioners in teledermatology practices. *JAMA network open*, 4(4):e217249–e217249, 2021a.

Saahil Jain, Ashwin Agrawal, Adriel Saporta, Steven QH Truong, Du Nguyen Duong, Tan Bui, Pierre Chambon, Yuhao Zhang, Matthew P Lungren, Andrew Y Ng, et al. Radgraph: Extracting clinical entities and relations from radiology reports. *arXiv preprint arXiv:2106.14463*, 2021b.

Alistair Johnson, Matt Lungren, Yifan Peng, Zhiyong Lu, Roger Mark, Seth Berkowitz, and Steven Horng. Mimic-cxr-jpg-chest radiographs with structured labels. *PhysioNet*, 101:215–220, 2019a.

Alistair EW Johnson, Tom J Pollard, Lu Shen, Li-wei H Lehman, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G Mark. Mimic-iii, a freely accessible critical care database. *Scientific data*, 3(1):1–9, 2016.

Alistair EW Johnson, Tom J Pollard, Seth J Berkowitz, Nathaniel R Greenbaum, Matthew P Lungren, Chih-ying Deng, Roger G Mark, and Steven Horng. Mimic-cxr, a de-identified publicly available database of chest radiographs with free-text reports. *Scientific data*, 6(1):317, 2019b.

Alistair EW Johnson, Lucas Bulgarelli, Lu Shen, Alvin Gayles, Ayad Shammout, Steven Horng, Tom J Pollard, Sicheng Hao, Benjamin Moody, Brian Gow, et al. Mimic-iv, a freely accessible electronic health record dataset. *Scientific data*, 10(1):1, 2023.

Kevin B Johnson, Wei-Qi Wei, Dilhan Weeraratne, Mark E Frisse, Karl Misulis, Kyu Rhee, Juan Zhao, and Jane L Snowdon. Precision medicine, ai, and the future of personalized health care. *Clinical and translational science*, 14(1):86–93, 2021.

Nan Rosemary Ke, Olexa Bilaniuk, Anirudh Goyal, Stefan Bauer, Hugo Larochelle, Bernhard Schölkopf, Michael C. Mozer, Chris Pal, and Yoshua Bengio. Learning neural causal models from unknown interventions, 2019. URL `https://arxiv.org/abs/1910.01075`.

Jenna Kefeli and Nicholas Tatonetti. Tcga-reports: A machine-readable pathology report resource for benchmarking text-based ai models. *Patterns*, 5(3), 2024.

Sameer Khanna, Adam Dejl, Kibo Yoon, Steven QH Truong, Hanh Duong, Agustina Saenz, and Pranav Rajpurkar. Radgraph2: Modeling disease progression in radiology reports via hierarchical information extraction. In *Machine learning for healthcare conference*, pages 381–402. PMLR, 2023.

Mervyn King and John Kay. *Radical uncertainty: Decision-making for an unknowable future*. Hachette UK, 2020.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR*, 2015.

J Kips, J Papeleu, A Shen, S Mylle, E Genouw, I Hoorens, E Verhaeghe, and L Brochez. Artificial intelligence based smartphone app for skin cancer detection: a prospective diagnostic accuracy study. *EJC Skin Cancer*, 3, 2025.

Pang Wei Koh, Thao Nguyen, Yew Siang Tang, Stephen Mussmann, Emma Pierson, Been Kim, and Percy Liang. Concept bottleneck models. In *International conference on machine learning*, pages 5338–5348. PMLR, 2020.

D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques*. Adaptive computation and machine learning. MIT Press, 2009. ISBN 9780262013192.

Guilan Kong, Dong-Ling Xu, and Jian-Bo Yang. Clinical decision support systems: a review on knowledge representation and inference under uncertainties. *International Journal of Computational Intelligence Systems*, 1(2):159–167, 2008.

Klaus Krippendorff. *Content analysis: An introduction to its methodology*. Sage publications, 2018.

Jonathan Kuck, Shuvam Chakraborty, Hao Tang, Rachel Luo, Jiaming Song, Ashish Sabharwal, and Stefano Ermon. Belief propagation neural networks. In *Advances in Neural Information Processing Systems*, volume 33, pages 667–678. Curran Associates, Inc., 2020.

Johan Kwisthout. Motivating explanations in bayesian networks using map-independence. *International Journal of Approximate Reasoning*, 153:18–28, 2023.

Yeonsu Kwon, Jiho Kim, Gyubok Lee, Seongsu Bae, Daeun Kyung, Wonchul Cha, Tom Pollard, Alistair Johnson, and Edward Choi. Ehrcon: Dataset for checking consistency between unstructured notes and structured tables in electronic health records. In *Advances in Neural Information Processing Systems*, volume 37, 2024.

Trent Kyono, Yao Zhang, and Mihaela van der Schaar. Castle: Regularization via auxiliary causal graph discovery. In *Advances in Neural Information Processing Systems*, volume 33, pages 1501–1512. Curran Associates, Inc., 2020.

Evangelia Kyrimi, Mariana Raniere Neves, Scott McLachlan, Martin Neil, William Marsh, and Norman Fenton. Medical idioms for clinical bayesian network development. *Journal of Biomedical Informatics*, 108, 2020.

Evangelia Kyrimi, Kudakwashe Dube, Norman Fenton, Ali Fahmi, Mariana Raniere Neves, William Marsh, and Scott McLachlan. Bayesian networks in healthcare: What is preventing their adoption? *Artif Intell Med*, 116:102079, 2021a.

Evangelia Kyrimi, Kudakwashe Dube, Norman Fenton, Ali Fahmi, Mariana Raniere Neves, William Marsh, and Scott McLachlan. Bayesian networks in healthcare: What is preventing their adoption? *Artificial Intelligence in Medicine*, 116:102079, 2021b.

Evangelia Kyrimi, Kudakwashe Dube, Norman Fenton, Ali Fahmi, Mariana Raniere Neves, William Marsh, and Scott McLachlan. Bayesian networks in healthcare: What is preventing their adoption? *Artificial Intelligence in Medicine*, 116, 2021c.

Evangelia Kyrimi, Scott McLachlan, Kudakwashe Dube, Mariana R Neves, Ali Fahmi, and Norman Fenton. A comprehensive scoping review of bayesian networks in healthcare: Past, present and future. *Artificial Intelligence in Medicine*, 117:102108, 2021d.

Evangelia Kyrimi, Scott McLachlan, Kudakwashe Dube, Mariana R. Neves, Ali Fahmi, and Norman Fenton. A comprehensive scoping review of bayesian networks in healthcare: Past, present and future. *Artificial Intelligence in Medicine*, 117, 2021e.

Evangelia Kyrimi, Scott McLachlan, Kudakwashe Dube, Mariana R Neves, Ali Fahmi, and Norman Fenton. A comprehensive scoping review of bayesian networks in healthcare: Past, present and future. *Artificial Intelligence in Medicine*, 117:102108, 2021f.

Evangelia Kyrimi, Scott McLachlan, Kudakwashe Dube, Mariana R. Neves, Ali Fahmi, and Norman Fenton. A comprehensive scoping review of Bayesian networks in healthcare: Past, present and future. *Artif Intell Med*, 117:102108, 2021g. ISSN 0933-3657.

Yanis Labrak, Adrien Bazoge, Emmanuel Morin, Pierre-Antoine Gourraud, Mickael Rouvier, and Richard Dufour. BioMistral: A collection of open-source pretrained large language models for medical domains. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 5848–5864, Bangkok, Thailand, August 2024. Association for Computational Linguistics.

S. L. Lauritzen and D. J. Spiegelhalter. Local Computation with Probabilities on Graphical Structures and their Application to Expert Systems (with discussion). *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 50(2):157–224, 1988.

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

Cindy S Lee, Paul G Nagy, Sallie J Weaver, and David E Newman-Toker. Cognitive and system factors contributing to diagnostic errors in radiology. *American Journal of Roentgenology*, 201(3):611–617, 2013.

Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. Biobert: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36(4):1234–1240, 2020.

Scott H Lee. Natural language generation for electronic health records. *NPJ digital medicine*, 1(1):63, 2018.

Eric Lehman and Alistair Johnson. Clinical-t5: Large language models built using mimic clinical text. *PhysioNet*, 2023.

Mariana Lenharo. The testing of ai in medicine is a mess. here's how it should be done. *Nature*, 632(8026):722–724, 2024.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33:9459–9474, 2020.

Yikuan Li, Shishir Rao, José Roberto Ayala Solares, Abdelaali Hassaine, Rema Ramakrishnan, Dexter Canoy, Yajie Zhu, Kazem Rahimi, and Gholamreza Salimi-Khorshidi. Behrt: transformer for electronic health records. *Scientific reports*, 10(1):7155, 2020.

Asiyah Yu Lin, Sivaram Arabandi, Thomas Beale, William D Duncan, Amanda Hicks, William R Hogan, Mark Jensen, Ross Koppel, Catalina Martínez-Costa, Øystein Nytrø, et al. Improving the quality and utility of electronic health record data through ontologies. *Standards*, 3(3): 316–340, 2023.

Sicen Liu, Xiaolong Wang, Yongshuai Hou, Ge Li, Hui Wang, Hui Xu, Yang Xiang, and Buzhou Tang. Multimodal data matters: language model pre-training over structured and unstructured electronic health records. *IEEE Journal of Biomedical and Health Informatics*, 27(1):504–514, 2022a.

Sicen Liu, Xiaolong Wang, Yongshuai Hou, Ge Li, Hui Wang, Hui Xu, Yang Xiang, and Buzhou Tang. Multimodal data matters: language model pre-training over structured and unstructured electronic health records. *IEEE Journal of Biomedical and Health Informatics*, 27(1):504–514, 2022b.

Yuan Liu, Ayush Jain, Clara Eng, David H Way, Kang Lee, Peggy Bui, Kimberly Kanada, Guilherme de Oliveira Marinho, Jessica Gallegos, Sara Gabriele, et al. A deep learning system for differential diagnosis of skin diseases. *Nature medicine*, 26(6):900–908, 2020.

Scott M Lundberg, Gabriel Erion, Hugh Chen, Alex DeGrave, Jordan M Prutkin, Bala Nair, Ronit Katz, Jonathan Himmelfarb, Nisha Bansal, and Su-In Lee. From local explanations to global understanding with explainable ai for trees. *Nature machine intelligence*, 2(1):56–67, 2020.

Fenglong Ma, Quanzeng You, Houping Xiao, Radha Chitta, Jing Zhou, and Jing Gao. Kame: Knowledge-based attention model for diagnosis prediction in healthcare. In *Proceedings of the 27th ACM international conference on information and knowledge management*, pages 743–752, New York, NY, USA, 2018. Association for Computing Machinery.

Yuchen Ma, Dennis Frauen, Jonas Schweisthal, and Stefan Feuerriegel. Llm-driven treatment effect estimation under inference time text confounding. *arXiv preprint arXiv:2507.02843*, 2025.

David J. C. MacKay. A practical bayesian framework for backpropagation networks. *Neural Computation*, 4(3):448–472, 1992.

Robin Manhaeve, Sebastijan Dumancic, Angelika Kimmig, Thomas Demeester, and Luc De Raedt. DeepProbLog: Neural probabilistic logic programming. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 31, 2018a.

Robin Manhaeve, Sebastijan Dumancic, Angelika Kimmig, Thomas Demeester, and Luc De Raedt. Deepproblog: Neural probabilistic logic programming. *Advances in neural information processing systems*, 31: 103504, 2018b.

Robin Manhaeve, Sebastijan Dumančić, Angelika Kimmig, Thomas Demeester, and Luc De Raedt. Deepproblog: Neural probabilistic logic programming. In *Advances in Neural Information Processing Systems*, volume 31, 2018c.

Subramani Mani and Gregory F Cooper. Causal discovery from medical textual data. In *Proceedings of the AMIA Symposium*, page 542. American Medical Informatics Association, 2000.

Giuseppe Marra, Sebastijan Dumančić, Robin Manhaeve, and Luc De Raedt. From statistical relational to neurosymbolic artificial intelligence: A survey. *Artif Intell*, 328:104062, 2024. ISSN 0004-3702.

Scott McLachlan, Kudakwashe Dube, Graham A Hitman, Norman E Fenton, and Evangelia Kyrimi. Bayesian networks in healthcare: Distribution by medical condition. *Artif Intell Med*, 107:101912, 2020a. ISSN 0933-3657.

Scott McLachlan, Kudakwashe Dube, Graham A Hitman, Norman E Fenton, and Evangelia Kyrimi. Bayesian networks in healthcare: Distribution by medical condition. *Artificial Intelligence in Medicine*, 107:101912, 2020b.

Ashley ND Meyer, Traber D Giardina, Lubna Khawaja, and Hardeep Singh. Patient and clinician experiences of uncertainty in the diagnostic process: current understanding and future directions. *Patient Education and Counseling*, 104(11):2606–2615, 2021.

Tom M Mitchell. The need for biases in learning generalizations. In *Department of Computer Science, Laboratory for Computer Science Research*, 1980.

Stefania Montani and Manuel Striani. Artificial intelligence in clinical decision support: a focused literature survey. *Yearbook of Medical Informatics*, 28(1):120–127, 2019.

Jong Hak Moon, Geon Choi, Paloma Rabaey, Min Gwan Kim, Hyuk Gi Hong, Jung-Oh Lee, Hangyul Yoon, Eun Woo Doe, Jiyoun Kim, Harshita Sharma, Daniel C. Castro, Javier Alvarez-Valle, and Edward Choi. Lunguage: A benchmark for structured and sequential chest x-ray interpretation, 2025. URL `https://arxiv.org/abs/2505.21190`.

Nelson Morgan and Herve A Bourlard. Neural networks for statistical recognition of continuous speech. *Proceedings of the IEEE*, 83(5):742–772, 1995.

Ali Mrad, Véronique Delcroix, Mohamed Amine Maalej, Sylvain Piechowiak, and Mohamed Abid. Uncertain evidence in bayesian networks : Presentation and comparison on a simple example. In *Communications in Computer and Information Science*, volume 299, pages 39–48, 07 2012. ISBN 978-3-642-31717-0. doi: 10.1007/978-3-642-31718-7_5.

Ghulam Mujtaba, Liyana Shuib, Norisma Idris, Wai Lam Hoo, Ram Gopal Raj, Kamran Khowaja, Khairunisa Shaikh, and Henry Friday Nweke. Clinical text classification research trends: Systematic literature review and open issues. *Expert Syst Appl*, 116:494–520, 2019. ISSN 0957-4174.

Blake Murdoch. Privacy and artificial intelligence: challenges for protecting health information in a new era. *BMC medical ethics*, 22(1):122, 2021.

Tuan Nguyen, Thanh Huynh, Minh Hieu Phan, Quoc Viet Hung Nguyen, and Phi Le Nguyen. Carer-clinical reasoning-enhanced representation for temporal health risk prediction. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 10392–10407, 2024.

G. R. Norman, S. D. Monteiro, J. Sherbino, J. S. Ilgen, H. G. Schmidt, and S. Mamede. The causes of errors in clinical reasoning: Cognitive biases, knowledge deficits, and dual process thinking. *Acad Med*, 92(1):23–30, 2017.

Mahmud Omar, Reem Agbareia, Alon Gorenshtein, and Girish N Nadkarni. Large language models chase zebras: Salient cues overrule base rates in clinical diagnosis. *Available at SSRN 5988435 (Preprint)*.

Agnieszka Oniśko, Marek J Druzdzel, and Hanna Wasyluk. Learning bayesian network parameters from small data sets: Application of noisy-or gates. *International Journal of Approximate Reasoning*, 27(2):165–182, 2001.

OpenAI. Models – GPT-4o. `https://platform.openai.com/docs/models/gpt-4o`, 2024. Online; accessed 12 August 2024.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, and et al. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*, volume 35, pages 27730–27744, 2022.

Rong Pan, Yun Peng, and Zhongli Ding. Belief update in bayesian networks using uncertain evidence. In *2006 18th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'06)*, pages 441–444, 2006. doi: 10.1109/ICTAI.2006.39.

Álvaro Parafita and Jordi Vitrià. Causal inference with deep causal graphs, 2020. URL `https://arxiv.org/abs/2006.08380`.

Sungjin Park, Seongsu Bae, Jiho Kim, Tackeun Kim, and Edward Choi. Graph-text multi-modal pre-training for medical representation learning. In *Conference on Health, Inference, and Learning*, pages 261–281. PMLR, 2022.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, volume 32, pages 8024–8035. Curran Associates, Inc., 2019.

Nick Pawlowski, Daniel Coelho de Castro, and Ben Glocker. Deep structural causal models for tractable counterfactual inference. In *Advances in Neural Information Processing Systems*, volume 33, pages 857–869. Curran Associates, Inc., 2020.

J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference.* Morgan Kaufman, San Mateo, CA, 1988a.

Judea Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference.* Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1988b.

Judea Pearl. *Causality.* Cambridge university press, 2009.

N. Peiffer-Smadja, T.M. Rawson, R. Ahmad, A. Buchard, and et al. Machine learning for clinical decision support in infectious diseases: A narrative review of current applications. *Clin Microbiol Infect*, 26(5):584–595, 2020a. ISSN 1198-743X.

N. Peiffer-Smadja, T.M. Rawson, R. Ahmad, A. Buchard, P. Georgiou, F.-X. Lescure, G. Birgand, and A.H. Holmes. Machine learning for clinical decision support in infectious diseases: a narrative review of current applications. *Clinical Microbiology and Infection*, 26(5):584–595, 2020b.

Sarah J Price, Sal A Stapley, Elizabeth Shephard, Kevin Barraclough, and William T Hamilton. Is omission of free text records a possible source of data loss and bias in clinical practice research datalink studies? A case–control study. *BMJ Open*, 6(5):e011664, 2016. ISSN 2044-6055.

Tine Proesmans, Christophe Mortelmans, Ruth Van Haelst, Frederik Verbrugge, Pieter Vandervoort, and Bert Vaes. Mobile phone–based use of the photoplethysmography technique to detect atrial fibrillation in primary care: diagnostic accuracy study of the fibricheck app. *JMIR mHealth and uHealth*, 7(3):e12284, 2019.

Thomas P Quinn, Stephan Jacobs, Manisha Senadeera, Vuong Le, and Simon Coghlan. The three ghosts of medical ai: Can the black-box present deliver? *Artificial intelligence in medicine*, 124:102158, 2022.

Paloma Rabaey, Henri Arno, Stefan Heytens, and Thomas Demeester. Simsum – simulated benchmark with structured and unstructured medical records, 2024a. URL `https://arxiv.org/abs/2409.08936`.

Paloma Rabaey, Johannes Deleu, Stefan Heytens, and Thomas Demeester. Clinical reasoning over tabular data and text with bayesian networks. In *International Conference on Artificial Intelligence in Medicine*, pages 229–250. Springer, 2024b.

Paloma Rabaey, Johannes Deleu, Stefan Heytens, and Thomas Demeester. Clinical reasoning over tabular data and text with bayesian networks. In *International Conference on Artificial Intelligence in Medicine*, pages 229–250. Springer, 2024c.

Laila Rasmy, Yang Xiang, Ziqian Xie, Cui Tao, and Degui Zhi. Med-bert: pretrained contextualized embeddings on large-scale structured electronic health records for disease prediction. *NPJ digital medicine*, 4(1):86, 2021.

Bruce I Reiner. Quantitative analysis of uncertainty in medical reporting: creating a standardized and objective methodology. *Journal of digital imaging*, 31(2):145–149, 2018.

François Remy, Kris Demuynck, and Thomas Demeester. BioLORD-2023: semantic textual representations fusing large language models and clinical knowledge graph insights. *Journal of the American Medical Informatics Association*, page ocae029, 02 2024.

Raanan Y. Rohekar, Shami Nisimov, Yaniv Gurwicz, Guy Koren, and Gal Novik. Constructing deep neural networks by bayesian network structure learning. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, page 3051–3062, Red Hook, NY, USA, 2018. Curran Associates Inc.

Maya Rotmensch, Yoni Halpern, Abdulhakim Tlimat, Steven Horng, and David Sontag. Learning a health knowledge graph from electronic medical records. *Scientific reports*, 7(1):5994, 2017.

Cynthia Rudin. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature machine intelligence*, 1(5):206–215, 2019.

Kaspar Rufibach. Use of brier score to assess binary predictions. *Journal of Clinical Epidemiology*, 63(8):938–939, 2010. ISSN 0895-4356. doi: https://doi.org/10.1016/j.jclinepi.2009.11.009. URL `https://www.sciencedirect.com/science/article/pii/S0895435609003631`.

Stuart J. Russell and Peter Norvig. *Artificial Intelligence: a modern approach*. Pearson, 3 edition, 2009.

Zahra Sadeghi, Roohallah Alizadehsani, Mehmet Akif Cifci, Samina Kausar, Rizwan Rehman, Priyakshi Mahanta, Pranjal Kumar Bora, Ammar Almasri, Rami S Alkhawaldeh, Sadiq Hussain, et al. A review of explainable artificial intelligence in healthcare. *Computers and Electrical Engineering*, 118:109370, 2024.

Pedro Sanchez, Jeremy P Voisey, Tian Xia, Hannah I Watson, Alison Q O'Neil, and Sotirios A Tsaftaris. Causal machine learning for healthcare and precision medicine. *Royal Society Open Science*, 9(8):220638, 2022.

Tobias Sangers, Suzan Reeder, Sophie Van Der Vet, Sharan Jhingoer, Antien Mooyaart, Daniel M Siegel, Tamar Nijsten, and Marlies Wakkee. Validation of a market-approved artificial intelligence mobile health app for skin cancer screening: a prospective multicenter diagnostic accuracy study. *Dermatology*, 238(4):649–656, 2022.

DR Sarvamangala and Raghavendra V Kulkarni. Convolutional neural networks in medical image understanding: a survey. *Evolutionary intelligence*, 15(1):1–22, 2022.

Thomas Savage, Ashwin Nayak, Robert Gallo, Ekanath Rangan, and Jonathan H Chen. Diagnostic reasoning prompts reveal the potential for large language model interpretability in medicine. *NPJ Digital Medicine*, 7(1):20, 2024.

Mauro Scanagatta, Antonio Salmerón, and Fabio Stella. A survey on bayesian network structure learning from data. *Progress in Artificial Intelligence*, 8(4):425–439, 2019.

Andrew Sellergren, Sahar Kazemzadeh, Tiam Jaroensri, Atilla Kiraly, Madeleine Traverse, Timo Kohlberger, Shawn Xu, Fayaz Jamil, Cían Hughes, Charles Lau, et al. Medgemma technical report. *arXiv preprint arXiv:2507.05201*, 2025.

Maksut Senbekov, Timur Saliev, Zhanar Bukeyeva, Aigul Almabayeva, Marina Zhanaliyeva, Nazym Aitenova, Yerzhan Toishibekov, and Ildar Fakhradiyev. The recent progress and applications of digital technologies in healthcare: a review. *International journal of telemedicine and applications*, 2020(1):8830200, 2020.

Mark P Sendak, William Ratliff, Dina Sarro, Elizabeth Alderton, Joseph Futoma, Michael Gao, Marshall Nichols, Mike Revoir, Faraz Yashar, Corinne Miller, et al. Real-world integration of a sepsis deep learning technology into routine clinical care: implementation study. *JMIR medical informatics*, 8(7):e15182, 2020.

Nigam H Shah, David Entwistle, and Michael A Pfeffer. Creation and adoption of large language models in medicine. *Jama*, 330(9):866–869, 2023.

Atul B Shinagare, Ronilda Lacson, Giles W Boland, Aijia Wang, Stuart G Silverman, William W Mayo-Smith, and Ramin Khorasani. Radiologist preferences, agreement, and variability in phrases used to convey diagnostic certainty in radiology reports. *Journal of the American College of Radiology*, 16(4):458–464, 2019.

Edward H Shortliffe, Randall Davis, Stanton G Axline, Bruce G Buchanan, C Cordell Green, and Stanley N Cohen. Computer-based consultations in clinical therapeutics: explanation and rule acquisition capabilities of the mycin system. *Computers and biomedical research*, 8(4):303–320, 1975.

Ravid Shwartz-Ziv and Amitai Armon. Tabular data: Deep learning is not all you need. *Information Fusion*, 81:84–90, 2022.

Arabella Simpkin and Richard Schwartzstein. Tolerating uncertainty—the next medical revolution? *New England Journal of Medicine*, 375(18), 2016.

Karan Singhal, Shekoofeh Azizi, Tao Tu, S Sara Mahdavi, Jason Wei, Hyung Won Chung, Nathan Scales, Ajay Tanwani, Heather Cole-Lewis, Stephen Pfohl, et al. Large language models encode clinical knowledge. *Nature*, 620(7972):172–180, 2023.

Christel Sirocchi, Alessandro Bogliolo, and Sara Montagna. Medical-informed machine learning: integrating prior knowledge into medical decision systems. *BMC Medical Informatics and Decision Making*, 24(Suppl 4):186, 2024.

SkinVision. Skinvision - onze partners, 2025. URL `https://www.skinvision.com/nl/onze-partners`. Accessed: 2025-11-26.

SP Somashekhar, M-J Sepúlveda, S Puglielli, AD Norden, Edward H Shortliffe, C Rohit Kumar, A Rauthan, N Arun Kumar, P Patil, Kyu Rhee, et al. Watson for oncology and breast cancer treatment recommendations: agreement with an expert multidisciplinary tumor board. *Annals of Oncology*, 29(2):418–423, 2018.

L. Sterckx, G. Vandewiele, I. Dehaene, O. Janssens, and et al. Clinical information extraction for preterm birth risk prediction. *J Biomed Inform*, 110:103544, 2020.

Sigrid Sterckx and Wim Van Biesen. *Geneeskunde in tijden van AI : handvatten voor vandaag en morgen*. Academia Press, 2024. ISBN 9789401443838.

Sharon E Strauss, Paul Glasziou, W. Scott Richardson, and R. Brian Haynes. *Evidence-based medicine. How to practice and teach EBM.*, volume 5. Elsevier, 2018.

Eliza Strickland. Ibm watson, heal thyself: How ibm overpromised and underdelivered on ai health care. *IEEE spectrum*, 56(4):24–31, 2019.

Reed T Sutton, David Pincock, Daniel C Baumgart, Daniel C Sadowski, Richard N Fedorak, and Karen I Kroeker. An overview of clinical decision support systems: benefits, risks, and strategies for success. *NPJ digital medicine*, 3(1):17, 2020.

Maryam Tayefi, Phuong Ngo, Taridzo Chomutare, Hercules Dalianis, Elisa Salvi, Andrius Budrionis, and Fred Godtliebsen. Challenges and opportunities beyond structured data in analysis of electronic health records. *Wiley Interdisciplinary Reviews: Computational Statistics*, 13, 02 2021.

The Guardian. Doctors fear google skin check app will lead to 'tsunami of overdiagnosis', 2025. URL `https://www.theguardian.com/society/2021/may/21/doctors-fear-google-skin-check-app-will-lead-to-tsunami-of-overdiagnosis`. Accessed: 2025-11-26.

Shubo Tian, Qiao Jin, Lana Yeganova, Po-Ting Lai, Qingqing Zhu, Xiuying Chen, Yifan Yang, Qingyu Chen, Won Kim, Donald C Comeau, et al. Opportunities and challenges for chatgpt and large language models in biomedicine and health. *Briefings in Bioinformatics*, 25(1):bbad493, 2024.

Mark Turner, Julia Ive, and Sumithra Velupillai. Linguistic uncertainty in clinical nlp: A taxonomy, dataset and approach. In *International Conference of the Cross-Language Evaluation Forum for European Languages*, pages 129–141. Springer, 2021.

Emily Vardell and Mary Moore. Isabel, a clinical decision support system. *Medical reference services quarterly*, 30(2):158–166, 2011.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Thomas Verma and Judea Pearl. Causal networks: Semantics and expressiveness. In *Uncertainty in Artificial Intelligence*, volume 9 of *Machine Intelligence and Pattern Recognition*, pages 69–76. North-Holland, 1990.

Vice. Google's new dermatology app wasn't designed for people with darker skin, 2025. URL `https://www.vice.com/en/article/googles-new-dermatology-app-wasnt-designed-for-people-with-darker-skin/`. Accessed: 2025-11-26.

Sandra Wachter, Brent Mittelstadt, and Chris Russell. Counterfactual explanations without opening the black box: Automated decisions and the gdpr. *Harv. JL & Tech.*, 31:841, 2017.

Benyou Wang, Qianqian Xie, Jiahuan Pei, Zhihong Chen, Prayag Tiwari, Zhao Li, and Jie Fu. Pre-trained language models in biomedical domain: A systematic survey. *ACM Computing Surveys*, 56(3):1–52, 2023a.

Xiaochen Wang, Junyu Luo, Jiaqi Wang, Ziyi Yin, Suhan Cui, Yuan Zhong, Yaqing Wang, and Fenglong Ma. Hierarchical pretraining on multimodal electronic health records. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing. Conference on Empirical Methods in Natural Language Processing*, volume 2023, page 2839. NIH Public Access, 2023b.

Yanshan Wang, Liwei Wang, Majid Rastegar-Mojarad, Sungrim Moon, Feichen Shen, Naveed Afzal, Sijia Liu, Yuqun Zeng, Saeed Mehrabi, Sunghwan Sohn, et al. Clinical information extraction applications: a literature review. *Journal of biomedical informatics*, 77:34–49, 2018.

Zifeng Wang and Jimeng Sun. Promptehr: Conditional electronic healthcare records generation with prompt learning. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing. Conference on Empirical Methods in Natural Language Processing*, volume 2022, page 2873, 2022.

W Richard Webb and Charles B Higgins. *Thoracic imaging: pulmonary and cardiovascular radiology.* Lippincott Williams & Wilkins, 2011.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

Jenna Wiens, Suchi Saria, Mark Sendak, Marzyeh Ghassemi, Vincent X Liu, Finale Doshi-Velez, Kenneth Jung, Katherine Heller, David Kale, Mohammed Saeed, et al. Do no harm: a roadmap for responsible machine learning for health care. *Nature medicine*, 25(9):1337–1340, 2019.

Andrew Wong, Erkin Otles, John P Donnelly, Andrew Krumm, Jeffrey McCullough, Olivia DeTroyer-Cooley, Justin Pestrue, Marie Phillips, Judy Konye, Carleen Penoza, et al. External validation of a widely implemented proprietary sepsis prediction model in hospitalized patients. *JAMA internal medicine*, 181(8):1065–1070, 2021.

Joy T Wu, Nkechinyere N Agu, Ismini Lourentzou, Arjun Sharma, Joseph A Paguio, Jasper S Yao, Edward C Dee, William Mitchell, Satyananda Kashyap, Andrea Giovannini, et al. Chest imagenome dataset for clinical reasoning. *arXiv preprint arXiv:2108.00316*, 2021a.

Xuehong Wu, Junwen Duan, Yi Pan, and Min Li. Medical knowledge graph: Data sources, construction, reasoning, and applications. *Big data mining and analytics*, 6(2):201–217, 2023.

Yun-Chun Wu, Ming-Chieh Shih, and Yu-Kang Tu. Using normalized entropy to measure uncertainty of rankings for network meta-analyses. *Medical Decision Making*, 41(6):706–713, 2021b.

Derong Xu, Wei Chen, Wenjun Peng, Chao Zhang, Tong Xu, Xiangyu Zhao, Xian Wu, Yefeng Zheng, Yang Wang, and Enhong Chen. Large language models for generative information extraction: A survey. *Frontiers of Computer Science*, 18(6):186357, 2024a.

Derong Xu, Wei Chen, Wenjun Peng, Chao Zhang, Tong Xu, Xiangyu Zhao, Xian Wu, Yefeng Zheng, Yang Wang, and Enhong Chen. Large language models for generative information extraction: A survey. *Frontiers of Computer Science*, 18(6):186357, 2024b.

Keyang Xu, Mike Lam, Jingzhi Pang, Xin Gao, Charlotte Band, Piyush Mathur, Frank Papay, Ashish K Khanna, Jacek B Cywinski, Kamal Maheshwari, et al. Multimodal machine learning for automated icd coding. In *Machine learning for healthcare conference*, pages 197–215. PMLR, 2019.

Ran Xu, Wenqi Shi, Yue Yu, Yuchen Zhuang, Bowen Jin, May Dongmei Wang, Joyce Ho, and Carl Yang. RAM-EHR: Retrieval augmentation meets clinical predictions on electronic health records. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 754–765, Bangkok, Thailand, August 2024c. Association for Computational Linguistics.

Juri Yanase and Evangelos Triantaphyllou. A systematic survey of computer-aided diagnosis in medicine: Past and present developments. *Expert Syst Appl*, 138:112821, 2019. ISSN 0957-4174.

Xi Yang, Aokun Chen, Nima PourNejatian, Hoo Chang Shin, Kaleb E Smith, Christopher Parisien, Colin Compas, Cheryl Martin, Anthony B Costa, Mona G Flores, et al. A large language model for electronic health records. *NPJ digital medicine*, 5(1):194, 2022.

S. Yazdani and M. Hoseini Abardeh. Five decades of research and theorization on clinical reasoning: A critical review. *Adv Med Educ Pract*, 10: 703–716, 2019.

S. Yazdani, M. Hosseinzadeh, and F. Hosseini. Models of clinical reasoning with a focus on general practice: A critical review. *J Adv Med Educ Prof*, 5(4):177–184, 2017.

Muchao Ye, Suhan Cui, Yaqing Wang, Junyu Luo, Cao Xiao, and Fenglong Ma. Medpath: Augmenting health risk prediction via medical knowledge paths. In *Proceedings of the Web Conference 2021*, pages 1397–1409, 2021.

Ye Ye, Fuchiang Tsui, Michael Wagner, Jessi Espino, and Qi Li. Influenza detection from emergency department reports using natural language processing and Bayesian network classifiers. *J Am Med Inform Assoc*, 21(5): 815–823, 2014. ISSN 1067-5027.

D. Zhang, C. Yin, J. Zeng, X. Yuan, and P. Zhang. Combining structured and unstructured data for predictive models: A deep learning approach. *BMC Med Inform Decis Mak*, 20(1):280, 2020.

Mengliang Zhang, Xinyue Hu, Lin Gu, Tatsuya Harada, Kazuma Kobayashi, Ronald Summers, and Yingying Zhu. Cad-chest: Comprehensive annotation of diseases based on mimic-cxr radiology report. *PhysioNet*, 2023.

Haiyan Zhao, Hanjie Chen, Fan Yang, Ninghao Liu, Huiqi Deng, Hengyi Cai, Shuaiqiang Wang, Dawei Yin, and Mengnan Du. Explainability for large language models: A survey. *ACM Transactions on Intelligent Systems and Technology*, 15(2):1–38, 2024.