Explain and Improve Natural Language Processing Systems with Human Insights

by

Xinyan Zhao

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Information)
in The University of Michigan
2022

Doctoral Committee:

Associate Professor VG Vinod Vydiswaran, Chair
Professor Qiaozhu Mei
Assistant Professor Xu Shi
Associate Professor Ryan William Stidham

Xinyan Zhao
zhaoxy@umich.edu
ORCID iD: 0000-0001-5937-0688

To Chen;

to my parents

ACKNOWLEDGEMENTS

I write this dissertation with my greatest gratitude. First of all, I would like to express my sincere thanks to Prof. Vinod Vydiswaran who guided me through my PhD years not only as an advisor, but also as a mentor and a friend. Without his support, patience, and incentive research insights and expertise, I would not be able to finish this dissertation. I started working with Vinod in summer 2015 when I was a Master's student. Since then, I have been learning from him how to be an independent and collaborative researcher, how to collaborate with other people, how to present our work, how to guide other students, and how to provide constructive feedback. I could not have wished for a better advisor!

It is also my greatest honor to thank my committee members: Prof. Qiaozhu Mei, Prof. Xu Shi, and Prof. Ryan Stidham, for accepting my invitation and providing their insightful suggestions. Prof. Mei and Prof. Shi recognized my work, which gave me tremendous confidence that I am exploring the right track. They also encouraged and guided me to think bigger and appreciate the philosophical connections among the components in this dissertation. I also appreciate my amazing collaboration experience with Prof. Stidham. He provided many valuable insights from a clinical perspective, including data interpretation, experiment design, model evaluation, and paper writing, which deepened my understanding of the challenges presented in healthcare and the importance of the research that I explore.

Doctoral study is never an easy journey. However, being part of the NLP4Health group, this journey is always relaxing and inspiring. As the first group member, I

have witnessed (in fact, I am still witnessing) our group growing along these years. Many of my research ideas came from our discussions or casual conversations and many of my work are the results of the strengths that I learned from everyone in the group. I would like to thank Deahan Yu, Hyeon Joo, Asher Strayhorn, Jiazhao Li, Kate Weber, Zhuofeng Wu, Jessica Kim, Peijin Han, Kenan Alkiek, Dharani Krishnan Senthil Kumar, Ashley Beals, and Sagnik Ray Choudhury, and Jane Ferraro.

Of course, my doctoral journey cannot miss the support and help that I received from collaborators, colleagues, staff members, and friends. I would like to thank Prof. Tammay Chang, Prof. Daniel Romero, Prof. Tiffany Veinot, Prof. Yun Jiang, Prof. Corey Lester, Yuting Ding, and Haibo Ding for the wonderful collaborations which helped pave my research path. I thank Prof. Paul Resnick and Prof. Steve Oney for showing me how to be a good graduate student instructor. I thank Prof. David Jurgens for being the committee member for my preliminary exam. I thank the UMSI and DHLS staff members, Amy Eaton and Liz Rodriguiz, for navigating me through my defense and responding so many last-minute emails from me! I also want to thank my peers and friends at University of Michigan: Abraham Mhaidli, April Yi Wang, Ashwin Rajadesingan, Brian Hall, Christopher Quarles, Damon Carucci, Fangzhou (Ark) Zhang, Haiyin Liu, Hao Peng, Ihudiya Finda Williams, Jiaqi Ma, Jiaxin Pei, Ju Yeon Jung, Kaiwen Sun, Lei Zhang, Linfeng Li, Liz Marquis, Miao Yu, Qiaoning Zhang, Ruihan Wang, Sangmi Kim, Shiqing (Licia) He, Shiyan Yan, Teng Ye, Tong Guo, Vaishnav Kameswaran, Wei Ai, Xinghui Yan, Xuedong Li, Yan Chen, Yang Liu, Yichi Zhang, Yingzhi Liang, Yixin Zou, Yue Wang, Yulin Yu, Zhe Zhao, Zihan Wu, for all the fun experience that we shared.

Finally, I would like to thank my financée Chen Liang and my parents for their unfailing love and support for every decision I made for my study and career. This dissertation is dedicated to them.

TABLE OF CONTENTS

LIST OF FIGURES

Figure

LIST OF TABLES

Table

ABSTRACT

Human insights play an essential role in artificial intelligence (AI) systems as it increases the coherence between the human mind and system decisions by allowing the systems to be interpretable or aided by human input. However, the use of human insights in AI systems remains mysterious, despite the fact that their predictive power has demonstrated high correspondence regarding system output. This missing bridge between human and AI is caused mainly by the black-box nature of deep learning-based models. It often presents two challenges in applying these systems to critical domains such as healthcare, finance, or law practice. First, the low coherence between humans and AI often leads to systems with high performance yet lack human trust. Interpreting a system decision demands simple, faithful, and consistent explanations for humans to understand. Failing to provide such explanations decreases human trust in the system and results in serious outcomes. The second challenge concerns the large-scale data annotation that is usually expensive and impractical in the above critical domains as it requires specially trained domain expertise.

To advance the AI applications in critical domains, in this dissertation, I explore methods that increase the coherence between humans and AI while maintaining high system correspondence. I look at AI systems from the "pipeline" (instead of "model-only") perspective and focus on designing "grey-box" pipelines that have not only high system correspondence but also high coherence between humans and the pipeline.

I investigate two directions. The first direction investigates if we could improve the human-AI coherence by mimicking human behavior with high-correspondence

black-box models. The second direction investigates if mimicking human insights could further improve system correspondence.

To investigate the first direction, I evaluate how black-box models could mimic human insights as explanations in various formats, including natural language, extractive data snippets, and human heuristics. In Chapter III, I introduce an approach that generates natural language explanations based on human rationales that improve the explanation quality. Furthermore, in Chapter IV, I conduct a study in a clinical scenario where the system predicts the medication use of patients with inflammatory bowel disease (IBD) based on patients' medical records. I show that sentence-based explanations could be well predicted by using heuristics-based weak supervision with minimal annotation or annotation-based supervised learning with large annotation budgets. Lastly, in Chapter V, I examine if black-box models could well mimic human heuristics. Using adverse drug event identification as an example, I show that while human heuristics could be easily implemented into interpretable rule-based or white-box systems, they are often over-simplified. To overcome this limitation, I show that the semantics of heuristics could be well augmented by the black-box model with a light-loaded annotation effort from human-in-the-loop.

To investigate the second direction, I further evaluate if human insights could improve system correspondence. In Chapter III, I show that high-quality language explanations bring system correspondence gain. In Chapter IV, I show that identifying evidence support from patient notes brings substantial performance gain for the inference of IBD medication history. In Chapter V, I show that the semantically augmented human heuristics could be easily applied to support weak supervision and generate strong correspondence.

This dissertation contributes to the field of explainable AI. More specifically, this dissertation provides opportunities for designing AI systems that desire explanations to increase human trust in critical domains by collaborating with human insights.

CHAPTER I

Introduction

1.1 What is explanation in AI?

Explainability in artificial intelligence traces back to the symbolic systems in the 1980s, where systems made decisions by integrating handcrafted rules, and these rules served as explanation sources (*Haugeland* (1989)). However, the definition of "explainability" (or "explanation", "interpretability", and "interpretation") in the machine learning or AI community is never clearly defined, and this is because the purpose of using these terms often varies in different contexts (*Biran and Cotton* (2017)).

Miller (2019) defines the explanation from a perspective in social science where an explanation is an informative product of a social and cognitive process that is transferred between the explainer and explainee so that it could help the explainee understand the cause of an event. The produced explanation can usually help answer what (happened), how (did it happen), and why (did it happen) questions. *Lipton* (2018) confined the concept of "explanation" into "interpretability" where an explanation is the result of model interpretability.

More specific perceptions of explanations are vaguely defined based on different lines of work: explanations could be perceived as generated visualizations from data that could be used as justifications for model predictions (*Simonyan et al.* (2013);

Zeiler and Fergus (2014); *Karpathy et al.* (2015); *Lei et al.* (2016); *Bahdanau et al.* (2014); *Xu et al.* (2015)); it could also be defined as the transparent mechanism of modeling process that allows us to diagnose the "thought process" of the machine that are often described as interpretable models (*Agrawal et al.* (1993); *Ribeiro et al.* (2016); *Letham et al.* (2015); *Rudin et al.* (2013)).

1.2 What is a good explanation?

What is a good explanation, or how to evaluate the interpretability of an interpretable model? Previous studies have argued for different properties that qualify a good explanation. *Thagard* (1989); *Ranney and Thagard* (1988) pointed out that an explanation should have high *coherence* as humans are more likely to accept an explanation if it is consistent with people's prior beliefs; *Cohen et al.* (1990) reported that *simplicity* is an essential property because humans tend to prefer explanations that are straightforward to understand over complicated ones, and *Cohen et al.* (1990) also argued that explanations should have *generality* that can fit more events.

Another set of properties for explanations contains the *truth* and *probability*, the probability of the explanation being true. However, they have been challenged in *Hilton* (1996); *McClure* (2002); *Charness et al.* (2010) as their experiments showed that humans care more about the *pragmatic influences* (e.g. *reference* and *completeness*). *Koh et al.* (2010) also proposed three principles to evaluate an explanation which are *sound*, *complete*, and *not Overwhelming*.

The above properties of explanations focus on relating the explanation to human belief. Recently,*DeYoung et al.* (2019) pointed out that a good explanation should be informative to model predictions, which is being *faithful* that is often referred to as *fidelity*.

1.3 How to evaluate an explanation?

Doshi-Velez and Kim (2017) laid out the taxonomy for evaluating explanations, which includes application-grounded, human-grounded, and functionally-grounded evaluations. Application-grounded evaluation comes with the highest specificity and cost. It conducts experiments with domain experts in a real-world task setting to see if the produced explanation can assist experts in improving the end goal of the task. Human-grounded evaluation reduces the experiment cost by allowing for using designed tasks with less complexity that do not require specialized domain expertise from humans. For example, we can design an experiment to ask a lay human to take actions that would infer the quality of the explanation. The functionally-grounded evaluation is the most cost-efficient as it does not involve human efforts. It only requires the proxy task(s) that automatically demonstrate the quality of the explanation, and the main challenge, though, is to design appropriate proxy task(s). Most of explanations in AI systems are evaluated following these three categories (*Zhou et al.* (2021)).

Application-grounded evaluation metric The most direct way to evaluate an explanation method is to see how much performance improvement the generated explanations could help system users to gain (*Ribeiro et al.* (2016); *Carton* (2019)). There are also metrics implicitly yet objectively evaluating the quality of explanations by looking at how effective the explanations create psychological indications to users. For example, *Schmidt and Biessmann* (2019) defined *information transfer rate* to compare the mutual information created between human labelers and model predictions per time unit when the explanation is presented versus not presented.

Human-grounded evaluation metric Human-ground metrics focus on the subjectivity of human appreciation towards explanations. This kind of evaluation is done

by conducting user interviews. For example, *Hoffman et al.* (2018) measured user *curiosity* and *trust* towards explanation by asking users questions such as "why have you asked for an explanation?" and "do you plan to rely on the explanation?".

Functionally-grounded evaluation metric A standard evaluation metric in this category evaluates the faithfulness of explanations. To achieve this, we ask the model to make predictions based only on produced explanations *DeYoung et al.* (2019). This metric is often referred to as *sufficiency*. Another example metric is called *comprehensiveness*. This metric looks at the change of predicted probability distributions for a class by stripping the explanation from the input. This change indicates how influential an explanation is for making the original decision.

1.4 Human insights in explainable AI

As argued by *Broniatowski et al.* (2021), it is human who needs to interpret an explanation. Therefore it is important to consider human insights in explainable AI systems. According to *Broniatowski et al.* (2021), there are three perspectives to incorporating human insights into the design of AI systems.

Coherence and white-box models The first perspective focuses on *coherence* (*Hammond* (2000)) and white-box models. Because it is human who needs to perceive and comprehend the explanation, *coherence* emphasizes the process of how the result is generated. In this approach, the output results from a set of universally accepted axioms. The design of early rule-based systems is consistent with this line of work. The design of white-box models is similar to this approach as they are transparent, and it is easy for a human to understand how the rules are combined and operated.

Correspondence and black-box models The second perspective is about creating *correspondence* and emphasizing the model performance. Black-box models (e.g.,

neural networks) have shown significant advances in this direction. However, achieving such high performance is usually at the cost of low explainability. The main human insights involved here are large-scale annotation efforts.

Grey-box models The third perspective considers the tacit nature of human expertise (*Nonaka* (1994); *Polanyi* (2009); *Nonaka et al.* (2000)). In this scenario, a fully transparent explanation mechanism (e.g., white-box models) is unavailable, yet the system should not be blindly trusted with a black-box model. Therefore, grey-box models look for communication with human experts so that the "tacit" knowledge of domain expertise can be incorporated into the system's decision-making process. With grey-box models, human experts can increase the interpretability by designing models that provide the rationale for a given decision relative to a set of functional requirements (designed by humans).

1.5 Interpretability is more than interpretable models - Human is key

Applying an explainable AI system is not only about having an interpretable model. Instead, it is a pipeline that includes but is not limited to problem definition, user research, data collection and annotation, data process, model design, model training, model deployment, and even output verification. The decision made at every step could cause cascading effects on the outcome of the following steps.

Human is the key player throughout the entire pipeline. *Broniatowski et al.* (2021) argues that explanations are detailed mental representations that communicate the implementation mechanisms that led to an output. This argument is supported by one of the leading theories in human psychology, Fuzzy-Trace Theory (*Reyna and Brainerd* (1995)). The experiments of the Fuzzy-Trace Theory showed that humans tend to make multiple mental representations with different degrees of precision from

stimuli and rely on the most straightforward representation that can still lead to a distinction for making a decision (*Reyna* (2012)). ***It is the human's mental representation that emphasizes the implementation of AI systems and how they could be applied***.

Therefore, even though that high-performance black-box models have low interpretability, it is still possible to create interpretable pipelines for AI applications. Such pipelines are consistent with the theory of using grey-box models discussed above.

1.6 Research questions and outline

As discussed in Chapter 1.5, the application of AI systems is a pipeline that goes beyond having interpretable models. The strengths of heuristic rules, white-box models, and black-box models should not be designed mutually exclusive but should be working complementarily together. Therefore, in this dissertation, I focus on designing grey-box pipelines that have both the coherence between human and system output and the correspondence between model and the output.

Figure 1.1 illustrates the difference between white-box models (or rule-based systems), black-box models, and the grey-box pipeline. The white-box models (Figure 1.1a) have high coherence between human and the system decision due to the transparent decision-making process. However, the model correspondence is sacrificed because of the simplicity of model complexity. Due to the high model complexity, the black-box models (Figure 1.1b) yield high correspondence towards accurate decisions at the cost of coherence to human interpretation. In Figure 1.1c, I introduce a grey-box pipeline framework that relies on highly correspondent models to mimic human insights and make decisions from the human insights. Therefore, three research questions need to be studied:

R1. Can black-box models mimic human insights with high correspondence?

(a) Workflow of rule-based or white-box models with high coherence

(b) Workflow of black box models with high correspondence

(c) Workflow of grey-box pipeline with high coherence and correspondence

Figure 1.1: Illustration of the difference between white-box models, black-box models, and grey-box pipelines.

R2. Can model-generated human insights lead to accurate decisions?

R3. Do model decisions have high coherence to human interpretation?

To answer these questions, I investigate three types of human insights: natural language explanation, extractive rationale, and heuristic rules.

In Chapter III, I investigate natural language explanations based on extractive human rationale.

Previous studies have reported that large-scale explanation collection creates room for training a generator that generates fluent language explanations conditioned a model prediction (*Camburu et al.* (2018); *Kumar and Talukdar* (2020)). However, I found that the previous approaches overlooked the fact that humans could have different reasoning processes and therefore create alternative explanations. On the other hand, the approaches are also prone to generate spurious explanations. To address these two issues, I introduce a framework to generate high-quality explanations that are highly relevant to human rationale. This contribution is presented in Chapter III. These explanations further help improve the overall task performance. I also conduct detailed analysis to evaluate the quality of the generated explanations.

In Chapter IV, I investigate the use of heuristic rules and sentence-based rationales for document classification.

Despite the promising potential of natural language explanation, one challenge for training a language generator is the demand for large-scale annotation that is often unavailable in many specific domains (e.g., healthcare). Therefore, investigating other reliable methods that require less manual effort is essential. As *Miller* (2019) claimed, explanations are selected as a small subset of all possible causes in the human's cognitive process. This finding indicates that a small set of possible explanations (not necessarily complete) could lead to an accurate prediction. Similarly, the prediction could be explained with a small set of possible causes. This scenario motivates a possible interpretable solution for document classification tasks and named entity extraction tasks. Based on this motivation, I investigate the following research questions for the two tasks.

More specifically, I ask **Can we effectively identify evidence support from documents with human insights, and to what extent can the identified evidence support a document classification task?**

To answer these questions, I study a real-world task in the clinical setting: identify

medication use history for patients with inflammatory bowel disease. Specifically, I investigate if the document-level decision could be inferred based on a small set of identified evidence candidates and if the performance gained from using evidence support could be generalized to different models. I also investigate if evidence support could be further enhanced by utilizing the evidence-level polarity without large-scale human annotation.

In Chatper V, I investigate the possibility of creating language-based rules by combining natural language and heuristics.

Compared to document classification, extracting entities from text requires a different type of effort in human insights. This is because entity extraction is usually performed on a sentence basis, and therefore, the selected explanation should be bounded within phrases or tokens per the above-mentioned "explanation completeness" by *Miller* (2019). The recent success of weak supervision has made it possible that rules designed by human experts could be utilized to create large-scale weak annotations without access to the ground truth. One issue, though, is that designing a large set of high-quality rules that can cover a wide range of language variations is still challenging.

While the rules carefully designed by human experts are usually highly accurate, the rule-based human insights suffer from the large language variation in the text. For example, "patient developed $*$ while on medication" is a valid rule that can identify mentions of adverse drug events, but using this rule needs to consider a specific context window. Usually, these context windows are restricted to up to three tokens for it to be computationally tractable (i.e. the above example has two tokens on the left and three tokens on the right). This restriction would prevent the proposed method from finding the rules that have essentially the same semantic meaning yet require a larger context window to be identified, such as "patient immediately developed serious $*$ while on her medications".

Two address this challenge, I explore the possible connection between natural language and heuristic rules and enrich the semantic matching of a particular rule with natural language. For example, if a domain expert writes down a template rule with natural language, "patient developed [MASK] from medications", then by exploring the semantic relations between the template and the mentioned two rules ("patient developed $*$ while on medication" and "patient immediately developed serious $*$ while on her medications"), both examples could be identified.

CHAPTER II

Review of Explanation Methods in Neural Networks

2.1 Post-hoc and intrinsic explanations

One way to look at explanation methods depends on when the explanations are generated. Is it separated from the modeling process, or is it generated as an intermediate step between input data and model predictions?

Post-hoc explanations Post-hoc methods generate explanations from existing (trained) models. These methods aim at understanding model behaviors rather than affecting the outcome decisions. Therefore, they often appear in cases where system users or developers want to interpret a model decision or examine how reliable the model is.

(i) *Gradients-based explanations* explain a model prediction using the gradients of the outputs w.r.t a particular input feature. This method was first used by *Simonyan et al.* (2013) who created the visualization of the deep image classification model, ConvNets, by computing a gradient-based class saliency. However, such gradients could lead to misleading interpretation by giving zero gradients to import input features due to non-firing activation functions (*Shrikumar et al.* (2016)). To tackle

this issue, further extended approaches are proposed, including DeepLift (*Shrikumar et al.* (2017)) and Layer-wise Relevance Propagation (LRP) (*Binder et al.* (2016)). *Sundararajan et al.* (2017) proposed Integrated Gradients and argued that gradient-based methods should satisfy *sensitivity* and *implementation invariance*, the latter of which is violated by DeepLieft and LRP. Recently, *Sanyal and Ren* (2021) proposed a language model-focused Discretialized Integrated Gradients by relaxing the linear interpolation strategy used by *Sundararajan et al.* (2017).

(ii) *Model-agnostic explanations* generate explanations by treating the original model as a black box. It completely separates the explanation generation from the original model design and training. One way to do this is by learning an interpretable model that mimics the predictions of the original model, and then extracting explanations from the interpretable model (*Baehrens et al.* (2010); *Craven and Shavlik* (1995)). However, such methods only focus on a global interpretation and ignore the local fidelity. Therefore, an alternative is to introduce local perturbations (*Krause et al.* (2016); *Strumbelj and Kononenko* (2010); *Ribeiro et al.* (2016)). *LIME* (*Ribeiro et al.* (2016)) is the leading interpretation perturbation-based framework that can locally interpret a black-box model. To explain a data example, LIME first generates a set of perturbed examples and gets model predictions. Based on the perturbed data instances and the proximity of each instance to the original example, LIME learns a new linear model that approximates the target model behavior and uses the learned weights as explanations. The major advantage of LIME is that (1) this method does not require access to the architecture of black-box models, and (2) the approximation performance of the linear model shows how reliable the interpretation is. However, LIME may be unstable as the approximate is highly related to the perturbation result and the explanation complexity (i.e., number of features to be learned in the linear model) that has to be pre-defined.

(iii) *Shapley value-based explanations* compute the marginal contribution of a

particular feature to the system output, given all the possible feature groups. More specifically, for a given feature, its shapley value is computed as a weighted sum over all possible combinations of other features. Unlike LIME, the shapley value does not assume that explanations are linearly separable, and since shapley value relies on all possible coalitions, it gives stable explanation results. Another difference to LIME is the interpretation of the shapley value. Unlike LIME, which directly estimates the local fidelity of a feature to the model prediction, shapley value measures the contribution of a feature to the difference between the actual model prediction and the mean prediction. Shapley value also has its limitations. Considering all coalitions, it includes unrealistic data instances, and in the real machine learning setting, calculating shapley value becomes computationally intractable as the number of features is usually large. Another branch of work with shapley value is SHAP (*Lundberg and Lee* (2017)), which is based on the shapley value, and assumes that each feature is associated with a shapley value representing the impact of the feature on the model prediction (which is different to the interpretation of shapley value). The authors proposed a series of approximation methods to calculate SHAP, including the model-agnostic KernelSHAP and model-dependent TreeSHAP, DeepSHAP, and MaxSHAP, with faster implementation and computation. The fast approximation allows for a global explanation by scanning the entire data set. However, due to the perturbation process in KernelSHAP, like LIME, SHAP could also lead to misinterpretation (*Slack et al.* (2020)).

Intrinsic explanations Intrinsic explanations are generated when the models make predictions as part of the modeling process. These explanations require models to be intrinsically interpretable.

(i) *Feature weights* from linear regression, logistic regression, decision trees, etc., are considered as intrinsic explanations. In addition, Naive Bayes gives a different

type of feature weight, which is the conditional probability of a certain prediction given a particular feature, $P(y|x_i)$. This conditional probability could be treated as feature importance due to its assumption of conditional independence for features.

(ii) *Attention mechanism* was designed specifically for neural network models. It was first introduced by *Bahdanau et al.* (2014) in machine translation tasks to enable the model to automatically search snippets in the source sentence to predict target tokens. Since then, it has been viewed as a way of model interpretation in later studies.

One option for using attention for interpretation is to distribute attention values over the input tokens (in text) or pixels (in images) via a softmax function. Higher attention values represent the higher importance of the corresponding tokens. (*Xu et al.* (2015); *Choi et al.* (2016); *Martins and Astudillo* (2016); *Xie et al.* (2017); *Mullenbach et al.* (2018); *Vaswani et al.* (2017)). It could also be applied to generate sentence-based attentions where sentences with higher attention values are more important in the model rationalization (*Yang et al.* (2016)).

Although the attention in the above studies could be served as a good way of visualizing the importance distribution inside a model, it is hard to confidently derive clear rationales from them because all features would have a non-zero attention value. In text, the same token could have different attention values simply because of the length of the text. *Lei et al.* (2016) argues that rationale should only be a portion of text which is concise and sufficient. Therefore, the authors proposed a rationale generator that is jointly trained with the encoder that makes predictions based on the extracted rationale tokens. A similar idea was further explored by *Carton et al.* (2018) for determining text toxicity in social media.

2.2 Global and local explanations

Explanations can be generated using global or local interpretation methods. Global methods can be considered as ways of summarizing the model parameters and struc-

tures, while local methods aim to generate explanations based on the local context given an example.

Global explanations Rule-based models are usually created from a global point of view, and models such as linear regression, logistic regression, decision trees, etc., create global explanations (i.e., the feature weights). In SHAP, due to the fast approximation of TreeSHAP, it also makes it possible to compute global interpretations. *Ibrahim et al.* (2019); *Frosst and Hinton* (2017); *Yang et al.* (2018) introduced methods to extract global explanations from black-box models such as neural networks. The global explanation provides summaries of the knowledge learned by models, which helps perform model diagnostics. However, its high-level knowledge distillation provides limited power in explaining instances with high complexity. Recently, *Agarwal et al.* (2021) proposed neural additive models (NAMs) that belong to the family of Generalized Additive Models (GAMs). It takes as input the global features into a set of neural networks, and the networks individually model each feature, and then each network is linearly combined for predictions. NAMs share a similar intuition as the interpretable linear models.

Local explanations Unlike global explanations, local explanations are generated based on the context of a particular instance, such as LIME or attentions. It aims to explain a particular instance by identifying pieces of the instance that faithfully drive the model prediction. Compared to global interpretations, interpreting an instance locally is more preferred in the real-world setting (e.g., patient diagnosis) by generating fine-grained interpretations that are usually hidden from the global view.

2.3 Supervised and unsupervised explanations

Another way to distinguish explanations methods is to see if the explanations are generated under supervised or unsupervised settings.

Supervised explanations Supervised explanations require additional human efforts for annotations. In this setting, ground-truth explanations are annotated in addition to the task annotation itself. Then the model is either trained to generate explanations and predict task output in separate steps or jointly trained to make both predictions. This line of work includes *Camburu et al.* (2018); *Nie et al.* (2019); *DeYoung et al.* (2020, 2019); *Zhang et al.* (2016); *Arous et al.* (2021); *Du et al.* (2019). While supervised explanations create better interpretations (*Strout et al.* (2019)), the main limitation lies in the high annotation cost, which is usually not practical in a critical domain such as healthcare.

Unsupervised explanations Unsupervised explanations do not have any annotation cost as they can be naturally generated from either intrinsically interpretable models or model-agnostic post-hoc methods described above.

2.4 Interpretability and performance-oriented explanations

In addition to the above categorizations of explanation methods, here I provide another point of view, which is by looking at the purpose of the explanation uses.

Interpretability-oriented Interpretability-oriented methods aim at generating explanations for model behaviors. For example, the post-hoc methods (e.g., gradients and LIME) explore existing models and discover important pieces to the predicted outcome. These methods often follow a "predict-then-explain" workflow (*Camburu et al.* (2018)). Therefore, the generated explanations do not affect the model outcome.

Performance-oriented On the contrary, performance-oriented methods utilize the explanation created from human rationale to help models make better predictions. For example, in *Zaidan et al.* (2007), human-annotated explanations are used to train the model to be more confident towards the ground-truth label when they are present than when they are not. *Zhang et al.* (2016) uses annotated sentence-level rationales to improve document classification by creating supervised attention training objectives. Similar work are also explored by *Du et al.* (2019); *Arous et al.* (2021).

Apart from using annotated rationales from data instances, another line of this direction focuses on directly using human explanation to annotate task data with weak supervision (*Bach et al.* (2017); *Fries et al.* (2017); *Ratner et al.* (2017); *Safranchik et al.* (2020)). This method has been proved effective when we only have a limited annotation budget but have access to large-scale unlabeled data.

2.5 Formats of explanations

In this section I summarize the formats of explanations that are commonly used in previous studies.

Extractive explanation As the explanation methods described above, one of the most common explanation formats comes out as subsets of tokens or sentences highlighted by models. Under a supervised setting, it requires human efforts to annotate rationale tokens or sentences that are decision triggers for each data instance. (*Zaidan et al.* (2007); *Marshall et al.* (2016); *Zhang et al.* (2016); *DeYoung et al.* (2020); *Arous et al.* (2021); *Du et al.* (2019); *Lin et al.* (2020)). Additionally, methods like LIME also highlight important phrases or tokens. These explanations locally interpret model decisions. Similarly, global explanation methods also generate a subset of features, which also falls into this category.

Prototypes Prototypes are a group of representative data instances that are similar to each other in terms of some features. Therefore, given a new instance, if it could be classified into a specific prototype group, we can assume that it shares the same label as other instances in the group. Sometimes the prototypes could be defined as groups of data instances that are dissimilar to the new instance, also referred to as counterfactual explanations.

Natural language Recently, *Camburu et al.* (2018) proposed natural language as another type of human insight as the explanation method in addition to task-targeted data annotation. The idea is to pre-train a language generator that generates language explanations based on the input data. In addition to the final task label, training such a language generator requires human annotation in natural language to explain the label decision.

Heuristic rules Early NLP systems started with rule-based methods, the logical forms of human rationale. These rules offer a direct way of transferring human insight into the systems without having to train models on a large amount of annotated data. These logical rules allow system developers and domain experts to make precise decisions on data that are matched by the rules. For example, in clinical de-identification systems, the phrases such as "Dr" and "M.D" are strong indicators to catch "Dr X" and "X M.D" as doctor names. Besides high precision, rule-based human insights often have good readability as they often follow the semantic and syntactic structure of the text.

Recently, logical rules have been well demonstrated in weakly supervised systems (*Bach et al.* (2017); *Fries et al.* (2017); *Ratner et al.* (2017); *Wang et al.* (2019); *Safranchik et al.* (2020); *Boecking et al.* (2020)). The general idea is to use rule-based labeling functions to assign labels to data that matched by the rules. Then a generative model will estimate the noisy label matrix and generate probabilistic labels.

2.6 Human and model-driven explanations

We can also categorize explanation methods by looking at the explanation sources - are they generated intrinsically from the model design or human insights?

Model-driven explanation model-driven explanations highlight the operation mechanism, such as model weights, gradients, attention layers, etc., running inside a model that leads to a model decision. While these explanations are easy to obtain and allow us to evaluate the model mechanism, they do not necessarily provide explanations that are interpretable to humans because they do not guarantee to carry human belief, making them difficult for humans to comprehend or accept. This is defined as "confirmation bias" by *Nickerson* (1998) for scenarios where humans tend to ignore information that is inconsistent with their prior knowledge, which violates the *soundness* that we discussed in Chapter 1.2.

Human-driven explanation human-driven explanations are generated by bringing in human insights in the process. For example, the rule-based systems are completely human-driven. Another way to incorporate human insights is by providing human beliefs as supervision signals to train models. During the model training process, the embedded human beliefs will affect how the model looks at the training data based on what human experts emphasize.

The idea of including human insights as a part of the model training in natural language processing traces back to *Zaidan et al.* (2007) where it was referred to as human rationale. The authors proposed a deterministic method to force the model to provide better predictions when complete rationales are provided. Experiments showed that the rationales brought significant performance improvement on a sentiment classification task. Recently, *Zhang et al.* (2016); *DeYoung et al.* (2020); *Arous et al.* (2021); *Du et al.* (2019) demonstrated that annotating sentence-based rationales could

Studies	Explanation format	Global or local	Supervised or unsupervised	Models
Chapter 3	natural language	local	supervised	black-box
Chapter 4	heuristic rules attention	both	supervised weakly supervised	black-box white-box
Chapter 5	language rule	global	human-in-the-loop	black-box rule-based

Table 2.1: Summaries of explanation types and methods that are explored in this dissertation. All methods are intrinsic and proposed for both interpretation and performance.

significantly improve the prediction accuracy in document classification tasks, and *Bao et al.* (2018) showed that human rationale also provides useful supervision signals in low-resource settings. *Strout et al.* (2019) showed that explanations generated from human supervision outperform the machine-generated explanations in helping human judges make decisions.

2.7 Bridging to this dissertation

In this section, I reviewed different methods for inducing explanations from models or incorporating explanations from human insights into models. This dissertation is most related to the studies that bring human insights in to the models for performance and interpretability. Table 2.1 summarizes the main categorizations of the studies in this dissertation.

In this dissertation, I focus on the argument discussed in Chapter 1.5 that **interpretability is more than having interpretable models**. I design grey-box pipeline frameworks that use black-box models to achieve high correspondence and human insights to create high coherence between human belief and model output.

CHAPTER III

Improving Explanation Quality with Relevant Language Rationale

This chapter is based on *Zhao and Vydiswaran* (2020). I investigate extractive token rationale and natural language explanations (NLE) for a particular natural language understanding task, natural language inference (NLI).

In particular, I introduce a framework that generates natural language explanations that are faithful and highly relevant to human rationale. To evaluate the *coherence* of the framework to human annotators, I use both automatic and manual evaluation metrics to analyze the performance of rationale identification and NLE relevance. In addition, task accuracy and faithfulness are used to evaluate the explanation *correspondence* to task performance, and the rest of used to evaluate the explanation *coherence* to human belief.

3.1 Motivation

As described in Chapter II, language-based rationale provides additional human reasoning in addition to label annotation alone. They have been suggested to potentially improve the performance and interpretability of deep learning-based models – i.e. either augmenting model performance by incorporating NLEs as additional con-

textual features, or explaining model decisions by training an explanation generator. Researchers have parsed NLEs into structured logical forms (*Srivastava et al.* (2017); *Hancock et al.* (2018); *Lee et al.* (2020); *Qin et al.* (2020)) or directly encoded them into a vector-based semantic representation (*Fidler et al.* (2017)). Recent success in language modeling and generation have enabled trained models to explicitly provide human-readable rationales for classification tasks (*Kim et al.* (2018); *Huk Park et al.* (2018); *Camburu et al.* (2018); *Kumar and Talukdar* (2020); *Rajani et al.* (2019)). Similarly, studies such as *Rajani et al.* (2019) have reported significant performance improvements on commonsense reasoning tasks by including NLEs in training a language generation model. However, these trends do not carry over to NLI task in which a premise-hypothesis pair is expected to be classified into *entailment*, *neutral*, or *contradiction*. Previous studies on utilizing NLEs for NLI tasks have reported a drop in overall performance, even with powerful deep learning-based models such as LSTM (*Camburu et al.* (2018)), RoBERTa, and GPT2 (*Kumar and Talukdar* (2020)). In this chapter, I study this discrepancy in more detail and start first by identifing the following two primary issues, described below, with how NLEs have been incorporated for the NLI task so far.

Issue 1: Lack of rationale in NLE Generation Current approaches for explanation generation produce only one specific explanation for each data instance. However, these approaches ignore the variability in human reasoning and alternative explanations. Annotators could assign the same label to a data instance by considering different rationales. For example, given the premise and the hypothesis shown in Figure 3.1a, it is easy to infer that the label should be *contradiction*. However, this label could be explained using two different rationales – indicated by *"sitting"* or by *"car"*. Ignoring this aspect would limit the application of NLE in NLI tasks because trustworthy explanations should be consistent with the appropriate rationale used by

Premise: A man wearing a red uniform and helmet stands on his motorbike.
Hypothesis: A man sitting in a car.
Gold label: contradiction

Hypothesis: A man sitting in a car. **Explain:** A man cannot be both standing and sitting.
Hypothesis: A man sitting in a car. **Explain:** A motorbike is not a car.

(a) Example of different NL explanations using different rationales.

Premise: A man wearing a white shirt is playing the drums.
Hypothesis: A man is playing a musical instrument.
Gold label: entailment

Explain (entailment): Drums is a musical instrument.
Explain (neutral): Drumming does not alway involve playing a musical instrument.
Explain (contradiction): A man playing the drums isn't playing a musical instrument.

(b) Example of correct and incorrect NL explanations generated for a same premise-hypothesis, but favoring different labels.

Figure 3.1: Examples of NLEs in NLI task.

humans to interpret the label.

Issue 2: Inclusion of Spurious Explanations NLEs that are inconsistent with commonsense logic provide little help for model prediction (*Camburu et al.* (2020)). For example, given the premise-hypothesis pair in Figure 3.1b, the explanation regarding the correct label (*entailment*) aligns with the fact that drums are indeed musical instruments. However, while tailored explanations could be generated to favor other labels – *neutral* and *contradiction*, such explanations are themselves factually incorrect. This happens because while deep learning-based text generators are powerful enough to generate readable sentences, they often lack commonsense reasoning ability (*Zhou et al.* (2020). When generating an explanation, such models are prone to output negating text if conditioned to the *contradiction* label, or output text with uncertainty if conditioned to the *neutral* label, without reasoning about the plausibility of the generated text.

In the following sections, I will introduce a framework called LIREx to address

the above two limitations. Then I will investigate the behavior of each component in LIREx and show that LIREx generates flexible, faithful, and relevant NLEs that allow the model to be more robust to spurious explanations and better aligned to human interpretation.

3.2 Related work

NLEs have been studied along two main directions. The first direction focused on how explanations could be treated as contextual features to improve the model performance. Studies in this direction include *Li et al.* (2016); *Srivastava et al.* (2017); *Wang et al.* (2017); *Hancock et al.* (2018); *Qin et al.* (2020); *Lee et al.* (2020), in which the authors used semantic parsers to convert unstructured NLEs into structured feature-like logical forms. These logical forms could further benefit low-resource setting by weakly labeling more unlabeled data. One limitation of such approaches is that the localized contexts usually have limited ability to represent the semantic meaning of text and are often difficult to convert to logical forms when they get too complicated.

The second direction focused on training NLE generators to justify model predictions, usually as a post-hoc exercise. *Kim et al.* (2018) trained textual explanation generators conditioned on the video frames and commands in self-driving cars to describe and justify the operated actions. *Huk Park et al.* (2018) proposed an explanation module for visual question answering and an activity recognition task, in which they first use encoders to jointly predict labels and infer the rationale regions of an image, and then generate text explanations by conditioning on the predicted labels and inferred rationales. *Camburu et al.* (2018) suggested the inclusion of NLEs for NLI task by proposing e-SNLI, an expanded dataset that contains NLE annotations. They also jointly trained the prediction model and the explanation generation model conditioned on the predicted label. However, jointly training the two models led to a non-negligible loss in performance (by about 2 points in F1).

In recent studies, generated NLEs have been combined with original data for label prediction tasks. *Rajani et al.* (2019) proposed CAGE for commonsense reasoning task, where they first trained an explanation generator to predict explanations based on the question and answer choices, and then expanded the original classifier input by combining questions and generated explanations. This strategy achieved a significant improvement over the baseline that only used the original data as input. *Kumar and Talukdar* (2020) attempted a similar approach, NILE, for the NLI task on the e-SNLI dataset but with a modification where, instead of generating one explanation per training instance, they trained three independent generators conditioned on each label (*entailment*, *neutral*, and *contradiction*), respectively. Then the final NLI model takes as input, the premise-hypothesis pair as well as all three generated explanations. They also evaluated the faithfulness of the explanations to demonstrate that the explanations are well correlated with model predictions, but reported a drop in performance on the NLI task.

3.3 Framework

To address the aforementioned issues observed in how NLEs have been incorporated in NLI models, I propose a framework for Language Inference with Relevant Explanations (LIREx). LIREx augments the NLI model with relevant, plausible NLEs produced and selected by a rationale-enabled explanation generator and an instance selector. In this section, I will describe LIREx in detail.

Overall framework The overall workflow of LIREx is shown in Figure 3.2. Given a premise-hypothesis (P-H) pair, a label-aware rationalizer predicts rationales by taking as input a triplet (P, H, x; $x \in$ {entail, neutral, contradict}) and outputs a rationalized P-H pair, (P, H_x). Next, the NLE generator generates explanations (E_x) for each rationalized P-H pair. Then, the explanations are combined with the original

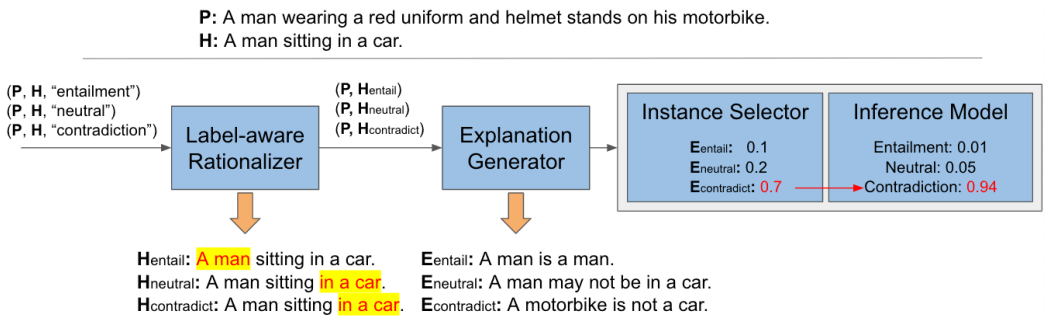

Figure 3.2: The overall workflow of LIREx framework

P-H pair as input to the instance selector and inference model to predict the final label. Each component is described below.

Label-aware Rationalizer - R(·) As described in *Camburu et al.* (2018), NLEs are created based on the rationales highlighted by annotators. We simulate this process by using a rationalizer to provide the relevant rationales for the NLE generator. This operation models how the explanations are generated for human interpretation.

We formulate this step as a token-level binary classification task where 1 indicates a rationale token and 0 indicates a background token. The rationale classification is only performed on the hypothesis because we consider the premise as background context for the NLI task, where the task is to justify if the hypothesis is an entailment, contradiction, or neutral statement with respect to the premise. Therefore, we hypothesize that the rationales in the hypothesis are sufficient to predict the correct label. We first construct the input sequence as S^p=<s>*Label*<s>*Premise*<s> and S^h= <s>*Hypothesis*<s>, where <s> is a special token that separates the components.[1] We append the label information to the premise to inform the rationalizer to highlight label-related rationales. Then we use a RoBERTa$_{base}$ model (*Liu et al.* (2019)) to extract hidden representations for S^p and S^h, denoted as $\mathbf{H}^p = [..., \mathbf{h}^p_i, ...]$ and $\mathbf{H}^h = [..., \mathbf{h}^h_j, ...]$, respectively. Since the rationales in hypothesis depends on the semantic meaning of

[1]RoBERTa includes two special tokens, <s> and </s>. For simplicity, we use <s> to denote both of them.

the premise, we use cross attention to embed premise into hypothesis, defined as

$$a_{ij} = \frac{\exp(\mathbf{h}_i^{hT}\text{Tanh}(\mathbf{W}_1^T\mathbf{h}_j^p))}{\sum_{k=0}^{L^p}\exp(\mathbf{h}_i^{hT}\text{Tanh}(\mathbf{W}_1^T\mathbf{h}_k^p))} \tag{3.1}$$

$$\hat{\mathbf{h}}_\mathbf{i}^\mathbf{h} = \text{concat}(\mathbf{h}_i^h, \text{Pool}(\mathbf{H}^p), \sum_k a_{i,k}\mathbf{h}_j^p) \tag{3.2}$$

where a_{ij} denotes the attention score of the j^{th} token in S^p to the i^{th} token in S^h, L^p denotes the sequence length of S^p, and $\mathbf{W}_1$ is trainable parameter matrix. Then the new representation of the ith token in S^h is created by concatenating its original state representation, maxpooling representation over $\mathbf{H}^p$, and the corresponding sum of attentional representation from $\mathbf{H}^p$. At last, we use a softmax layer with a linear transformation to model the probability of the i^{th} token in S^h being a rationale token: $\text{P}(y_i^h|S^p, S^h) = \text{softmax}(\mathbf{W}_2\hat{\mathbf{h}}_i^h)$, where $\mathbf{W}_2$ is a trainable parameter matrix for linear transformation on $\hat{\mathbf{h}}_i^h$.

NLE Generator - G(·) We model NLE generation as a text generation task, in which we leverage GPT2 (*Radford et al.* (2019)), a language model trained on large-scale language corpus. We choose GPT2_{medium} so that we could have an end-to-end comparison with the previous study that uses the same architecture.

In the previous study (*Kumar and Talukdar* (2020)), the authors finetuned GPT2 independently for each label. Specifically, they trained three GPT2 models separately, which are $\text{G}_x(\text{P}, \text{H}, \text{E})$, $x \in \{\text{entail, neutral, contradict}\}$. Each G_x is trained only with the P-H pairs annotated as x. As described in the Introduction section, such setup is (a) insensitive to the variety of human interpretation toward data, and (b) results in spurious explanations that further harm the label inference task. Additionally, this generation strategy requires training n GPT2 models for n labels, which is still expensive even with fine-tuning. To solve the above issues, we train a single GPT2 model, $\text{G}(\text{P}, \text{H}^*, \text{E})$, where H^* is rationalized hypothesis. For example, for the P-H

pair in Figure 3.1b, we construct the input sequence, S^g, as:

Premise: `P`
Hypothesis: A man is [playing] a [musical] [instrument].
Explanation: `E`

where `P` and `E` represent the premise and NLE text in training data. To inform the generator about the rationales in hypothesis, we highlight rationale tokens by surrounding them with the square brackets '[]'. The generator is fine-tuned by modeling the text input as a whole. To generate an NLE, we simply remove `E` from S^g and then use the rest as a text input prompt for the generator.

Unlike the approach in *Kumar and Talukdar* (2020) where the label information is appended to the text, we hide the label from the generator to force the model to generate rationale-enabled NLEs. This is consistent with our goal to simulate diverse human interpretation, and prevents the model from generating spurious label-based explanations.

During training and evaluating the explanation generator, we use the rationale tokens and NLEs provided by human annotators. After the generator is trained, we generate three new NLEs for each instance based on the rationales by including each label in S^p, independently. Now each P-H pair is provided with three explanations and we remove the original gold explanations in training data. This is to prevent the instance selector model in the next step from overfitting on the training examples.

Instance Selector and Inference - S$(\cdot)$ and Infer$(\cdot)$ When a P-H pair and the generated explanations are fed into an inference model, the model benefits from the addition of the explanations when they are correct (cf. Figure 3.1b). On the other hand, incorrect explanations lead to large uncertainty during the inference process. So, we first select a single plausible explanation for the final inference. To achieve this, we develop a simple strategy assuming that when the labels are correct, the NLEs generated based on the corresponding enabled rationales are the correct explanations.

This allows us to only estimate which NLE is generated by the gold label-enabled rationale. To further simplify this task, if we assume that the gold label-enabled explanation is more likely to be plausible than the other two explanations, we could identify the gold label-enabled explanation by accurately predicting the correct label for the standard NLI task. In other words, a good prediction on the NLE selection task can be achieved by just training a standard NLI classification model.

Training instance selector model We initialize the selector $\mathbf{S}(\cdot)$ with a $\text{RoBERTa}_{\text{base}}$ model and use the representation of the first token, $\mathbf{h}_0$, as the sequence representation. On top of this, an output layer of linear transformation and activation, $\text{Tanh}(\mathbf{U}_1\mathbf{h}_0)\mathbf{U}_2$, is applied for prediction. $\mathbf{U}_1$ and $\mathbf{U}_2$ are parameter matrices. We train $\mathbf{S}(\cdot)$ as a standard supervised learning task where premise and hypothesis are concatenated as a single input sequence, and the model is trained to predict label $Y \in \{\text{entailment, neutral, contradiction}\}$. We pre-train the instance selector and use the label prediction probability distribution as the estimator to find the most likely explanation corresponding to the true label. To improve model robustness, during training, we sample the candidate explanations based on the probability distribution, instead of picking just the most likely explanation. This allows the inference model to better tolerate less plausible explanations. During test phase, we select the explanation with the highest probability.

Training inference model Once the explanation instance is selected, we train the inference model, **Infer**(premise, hypothesis, explanation), with the same model architecture as the selector. Taking insights from the field of weak supervision (*Fries et al.* (2017); *Bach et al.* (2017); *Ratner et al.* (2020)) where weakly labeled data is used for training models, we treat the selected explanations as weakly selected instances. Instead of using the standard cross-entropy loss (that requires gold label) as training objective, we use a probability-oriented training objective, soft cross-entropy

loss, to improve the model robustness towards noisy input:

$$\mathrm{CE}_{soft}(\mathbf{p}, \hat{\mathbf{p}}) = \sum_{l \in \{\mathrm{e,n,c}\}} \hat{\mathbf{p}}_l \log \mathbf{p}_l \tag{3.3}$$

where $\mathbf{p}$ and $\hat{\mathbf{p}}$ are predicted and the "target" probabilities for each label, respectively. In our experiments, we use the estimated probabilities from the instance selector as our "target" probability for training inference model.

3.4 Experiments

In this section, I will first introduce the two data sets used, one of which is for supervised experiment setting and another one is considered as an out-of-domain data set. Then I perform a series of experiments to investigate the performance of LIREx. First, I show that LIREx brings performance improvement on both data sets. Then, I conduct further analysis to show that the rationalizer can correctly identify informative tokens despite automatic metric evaluation gives low score. Thrid, I examine the NLE generator to show that the generator can robustly generate NLEs by pertubing rationale tokens. Furthermore, I show that LIREx is able to generate more plausible NLEs. Last, I conduct faithfulness and relevance evaluation to show that the NLEs in LIREx is not only faithful but highly relevant to the human rationale.

Here are the different baseline and model variations that I used throughout the experiments. I use NILE as the backbone of all baseline models (*Kumar and Talukdar* (2020)).

Baseline the baseline models are denoted as $\mathrm{NILE}_{\mathrm{base}}$ and $\mathrm{LIREx}_{\mathrm{base}}$, which use the same RoBERTa model that takes input only with the P-H pair. $\mathrm{NILE}_{\mathrm{base}}$ is the original reported performance and $\mathrm{LIREx}_{\mathrm{base}}$ is the reproduced version made for fair comparison.

Explanation only These models use only the generated explanation(s) for prediction. The difference is that NILE model, denoted as $\text{NILE}_{\text{expl}}$, uses all three explanations, while the LIREx model, denoted as $\text{LIREx}_{\text{expl\_*}}$ uses the single explanation selected by the instance selector. The LIREx model has the following two variations:

- $\text{LIREx}_{\text{expl\_max}}$ selects the explanation with the highest probability.
- $\text{LIREx}_{\text{expl\_prob}}$ samples an explanation candidate based on the probability distribution.

Explanation and data NILE models use the P-H pair as well as all explanations as input by concatenating them into one input sequence, e.g. "p<s>h<s>e_1<s>e_2<s>e_3", where all components are separated by a special token <s>. Two versions are reported

- NILE_{all} is as described above.
- $\text{NILE}_{\text{all\_extra}}$ uses extra negative samples that are created for valid (P, H, E) triplets by sampling from other explanations.

LIREx use the pre-trained instance selector to first select an explanation candidate, and then concatenate the explanation to the P-H pair for input to the inference model, which also has two versions:

- $\text{LIREx}_{\text{all\_max}}$ selects the explanation with the highest probability.
- $\text{LIREx}_{\text{all\_prob}}$ samples an explanation candidate based on the probability distribution.

3.4.1 Data sets

SNLI (*Bowman et al.* (2015)) is a balanced collection of P-H annotated pairs with labels from {*entailment*, *neutral*, *contradiction*}. It consists of about 550K, 10K and

10K examples for train, development, and test set, respectively. *Camburu et al.* (2018) recently expanded this data set to e-SNLI in which each data instance is also annotated with explanations. Based on the gold labels for each P-H pair, annotators were asked to highlight the rationale tokens, and provide NLEs based on the rationale. Previous studies (*Camburu et al.* (2018); *Kumar and Talukdar* (2020)) removed over 17K non-informative training examples (where the explanations contain the entire premise or hypothesis) from their analysis. In our work, to maintain the original training data with minimal changes, we hold out these non-informative training instances only when training the explanation generator, but use the full training data for the remaining steps.

MultiNLI (*Williams et al.* (2017)) differs from the SNLI data set in that it covers a range of genres of spoken and written text. It contains 433K P-H pairs annotated the same way as SNLI. The evaluation set is divided into Dev-match set (10K) and Dev-mismatch set (10K) – the former is derived from same five domains as the training data and the latter is derived from five other domains.

3.4.2 Task performance evaluation

In-domain performance The model performance on SNLI data is summarized in Table 3.1. I re-implemented the NILE baseline ($\text{NILE}_{\text{base}}$) as $\text{LIREx}_{\text{base}}$ and achieved slight improvement (of 0.06) over published results. The rest of the table shows model improvements compared to the baselines accordingly, with the relative improvements against the baselines summarized in the "vs.baseline" column.

As shown in the table, the best model ($\text{LIREx}_{\text{all\_prob}}$), is able to achieve an absolute performance gain of 0.32 accuracy points. In addition, I also provide other variants of our model, namely, $\text{LIREx}_{\text{expl\_max}}$, $\text{LIREx}_{\text{expl\_prob}}$ and $\text{LIREx}_{\text{all\_max}}$. All the provided models show that models that use the instance selector achieve better performance.

Model	Dev	Test	vs.Baseline	+Data
$\text{SemBERT}_{\text{large}}$	92.0	91.6	-	no
$\text{SemBERT}_{\text{wwm}}$	92.2	91.9	-	no
$\text{NILE}_{\text{base}}$	91.86	91.49	-	no
$\text{NILE}_{\text{expl}}$	88.49	88.11	↓3.38	no
NILE_{all}	91.74	91.12	↓0.37	no
$\text{NILE}_{\text{all\_extra}}$	91.29	90.73	↓0.76	yes
$\text{LIREx}_{\text{base}}$	92.15±.05	91.55±.04	-	no
$\text{LIREx}_{\text{expl\_max}}$	89.95±.05	89.73±.04	↓1.82	no
$\text{LIREx}_{\text{expl\_prob}}$	90.10±.05	90.03±.05	↓1.52	no
$\text{LIREx}_{\text{all\_max}}$	92.15±.04	91.73±.03	↑0.18	no
$\text{LIREx}_{\text{all\_prob}}$	**92.22±.03**	**91.87±.03**	↑**0.32***	no

Table 3.1: Accuracy performance of LIREx on SNLI data (average of five random runs). "*" denotes that the best model is statistically significant (at significance level of 0.05 against baseline and 0.01 against NILE). "+ Data" denotes if additional training data was created. SemBERT (*Zhang et al.* (2019)) is included to refer to the best-reported performance.

Further, the sampling-based selection strategy (as in $\text{LIREx}_{*\_\text{prob}}$) performs better than the greedy selection using highest value (as in $\text{LIREx}_{*\_\text{max}}$).

Out-of-domain performance To test how well LIREx can generalize to a different data set, I directly apply the model trained on SNLI to an out-of-domain data set, MultiNLI. As shown in Table 3.2, without any fine-tuning, the model achieves significantly better performance compared to NILE. When compared to the corresponding baselines, the model performance dropped by 0.27 on dev-matched and improved slightly by 0.06 on dev-mismatched. Performance of NILE models, in contrast, dropped significantly on MultiNLI. In addition, compared to the baseline, $\text{LIREx}_{\text{all\_prob}}$, the final model performs similarly between dev-matched and dev-mismatched, indicating that the inclusion of explanations as supervision improves the overall generalizability of the model.

Model	Dev-Matched		Dev-MisMatched	
	Acc	**vs.baseline**	**Acc**	**vs.baseline**
$\text{NILE}_{\text{base}}$	79.29	-	79.29	-
$\text{NILE}_{\text{expl}}$	61.33	↓17.96	61.98	↓17.31
NILE_{all}	77.07	↓2.22	77.22	↓2.07
$\text{NILE}_{\text{all\_extra}}$	72.91	↓6.38	73.04	↓6.25
$\text{LIREx}_{\text{base}}$	**80.12**	-	79.73	-
$\text{LIREx}_{\text{expl\_max}}$	65.53	↓17.59	65.19	↓14.54
$\text{LIREx}_{\text{expl\_prob}}$	65.57	↓17.63	65.32	↓14.68
$\text{LIREx}_{\text{all\_max}}$	79.71	↓0.41	79.50	↓0.23
$\text{LIREx}_{\text{all\_prob}}$	79.85	↓**0.27**	**79.79**	↑**0.06**

Table 3.2: Transfer performance of LIREx on the out-of-domain MultiNLI data (average of five random runs) without fine-tuning.

3.4.3 NLE qualitative anlaysis

Rationalizer analysis As described earlier, the explanations are generated based on rationales in the hypothesis. Heuristically, we could evaluate the rationalizer simply by looking at the F1 score to see how well the predictions match the true rationales. However, this evaluation strategy alone is insufficient, because annotators may include additional neighboring tokens as rationales. For example, for the hypothesis "A man sitting in a car" in Figure 3.1a, "sitting in a car", "in a car", "a car", and "car" are all reasonably correct rationales as they all contain the most important rationale token "car". If an annotator provides "in a car" as rationale and the rationalizer predicts only "car", the automated F1 metric will be low even though the main rationale was correctly identified. So, in addition to the F1-based evaluation, we also conducted a manual verification of 100 randomly sampled test examples, and report the instance-level accuracy in Table 3.3. The manual verification was conducted by two annotators. The annotators were presented with P-H pairs, predicted rationales, and gold rationales, and were asked a "Yes/No" question: Do predicted rationales contain the key information from the gold rationales? Examples annotated as "Yes"

Model	Dev			Test			Human
	P	R	F1	P	R	F1	Eval
Rationale	59.40	65.22	62.17	59.21	64.89	61.92	90.53

Table 3.3: F1-based performance of the rationalizer and instance-level accuracy of human evaluation from two annotators over 100 randomly sampled test examples with an inter-rater agreement of 0.89.

were treated as correct predictions. The difference between automated and human evaluation shows that, although the rationalizer did not identify the exact human-provided rationales, it did identify the most important rationales (e.g., "car") with high accuracy.

An **unexpected, yet preferred** behavior of the rationalizer is that, when there are no obvious rationales towards the pre-appended label information, the model tends to identify rationales that relate to the correct label. For example, for the hypothesis (a) in Table 3.4, we devised three "alternate" hypotheses to analyze how the rationale predictions change with different hypotheses. We progressively modified a specific component in the original hypothesis – e.g. added *"on stage"* in (b), replaced *"man"* with *"woman"* in (c), and included both these changes and replaced *"musical instrument"* with *"guitar"* in (d). We found that when rationales of a particular label is absent, the rationalizer is prone to output rationales of the correct label (as contradiction rationales in hypothesis (a) and neutral rationales in hypothesis (c)).

Furthermore, when rationales of more than one label exist, the model is capable of identifying at least some correct rationales regarding a label. For example in Hypothesis (b), "playing musical instrument" is an entailment-related rationale, while "playing musical instrument on stage" is a neutral-related rationale. For hypothesis (d), we are able to correctly catch rationales of all labels ("playing" is an entailment, "stage" is neutral and "guitar" is a contradiction).

Generator analysis

Premise: A man wearing shirt is playing the drums.		
Hypothesis	**Label**	**Rationales**
(a) A man is playing a musical instrument	E	musical instrument
	N	musical instrument
	C	musical instrument
(b) A man is playing a musical instrument on stage	E	playing, musical instrument
	N	musical instrument, stage
	C	playing, musical instrument
(c) A woman is playing a musical instrument	E	musical instrument
	N	woman, musical instrument
	C	woman, musical instrument
(d) A woman is playing guitar on stage	E	playing
	N	woman, stage
	C	woman, guitar

Table 3.4: Examples of the rationalizer behavior. All hypotheses share the given premise. Rationales are predicted by appending a corresponding label information to the premise. The true labels of the P-H pairs are highlighted in green.

The plausibility of NLEs I conducted detailed analyses to show (a) why label information should be removed from generation prompts and (b) robustness of the generator towards rationale variants.

Removing label information from generation prompts prevents the generator from producing too many spurious explanations. In Table 3.5, we present an example to compare the explanations generated when label information is either included or excluded. For the provided P-H, all three models use GPT2 to generate explanations for each label. **NILE** uses the plain premise and hypothesis combined with label information as generation prompts. **LIREx-w-label** uses the same information as **NILE**, and also includes an indication of rationale tokens in the format described in the NLE Generator section. Finally, in **LIREx-w/o-label**, the label information is removed. As shown in the example, all three models are able to generate reasonable

explanations. However, when label information in appended to the input data, spurious explanations are presented (e.g. for E and N labels in **NILE**, and the E label in **LIREx-w-label**). This is because the addition of label serves as a very strong constraint for the generator, which then leads to an explanation that is constructed using the label information as guidance. Even with rationales are provided, the issue could only be slightly alleviated (eg. for N label in **LIREx-w-label**).

However, the issue is resolved when labels are excluded from the generator because the generator is not forced to generate a label-aligned explanation, which could be spurious. Instead, it utilizes only the rationales and the original context to construct the explanations. As discussed earlier, when the rationale of a specified label does not exist, the rationalizer tends to provide rationales related to the true label. So we obtain three reasonable explanations for the E, N, and C labels in **LIREx-w/o-label**(E,N,C). Further, this would greatly increase the chance of a plausible explanation being selected as an explanation candidate by the instance selector.

Robustness of generator towards varied rationales The example in Table 3.6 shows different explanations generated by using one hypothesis token at-a-time as rationale. Out of the seven generated explanations, only two are invalid. In reality, however, it is rare to have only non-informative stopwords as rationales. This observation strengthens our argument that including rationales in the training of the explanation generator could improve the robustness of the model.

Effect of Spurious Explanation As presented in Tables 3.5 and 3.6, **LIREx** is able to consistently generate plausible NLEs with rationale-enabled explanations while **NILE** tends to generate spurious NLEs due to the inclusion of label information in explanation generation. To show how the spurious explanations could affect model performance, we train our model with the best NLE from the selector for each data instance, and then use a randomly selected NLE during evaluation. The results are

Premise: Families waiting in line at an amusement park for their turn to ride. **Hypothesis:** People are waiting to see a movie.			
L	**NILE**	**LIREx-w-label**	**LIREx-w/o-label**
E	People are waiting to see a movie at an amusement park.	**Rationales:** people, waiting, see, movie	
		Families are people. If people are waiting in line, it means that they are waiting to see a movie.	People waiting in line are not waiting to see a movie.
N	Not all rides at amusement parks are a movie.	**Rationales:** movie	
		You cannot infer it's for a movie.	Amusement park and movie are not the same.
C	People cannot be waiting to ride an amusement park and a movie at the same time.	**Rationales:** waiting, see, movie	
		People are either waiting to ride an amusement park, or waiting to see a movie.	People are either waiting in line for a ride or waiting to see a movie, but not both.

Table 3.5: Example of NLEs generated when label information is included and excluded. For each label, we present the generated explanation from each system. The highlighted green "C" is the correct label for the P-H pair.

Premise: Families waiting in line at an amusement park for their turn to ride. **Hypothesis:** People are waiting to see a movie.		
Rationale	**Explanation**	**Valid**
People	Families implies more than one person	√
are	are waiting and waiting are different	×
waiting	You can not infer they are waiting	×
to	Rides does not imply to see a movie	√
see	To ride and to see are different	√
a	One cannot ride and see simultaneously.	√
movie	Just because families are waiting in line at amusement park doesn't mean they are waiting to see a movie	√

Table 3.6: Example of explanations generated using different rationales – one hypothesis token at-a-time as rationale.

Model	**SNLI-Dev**		**SNLI-Test**		**MNLI-M**		**SNLI-Mis**	
	best	rand	best	rand	best	rand	best	rand
LIREx	92.15	91.84	91.80	91.55	79.51	79.51	79.52	79.33
LIREx$_{\text{NILE}}$	91.59	85.97	91.46	85.58	79.48	72.01	79.50	71.97

Table 3.7: Effects of spurious explanations on model performance. LIREx$_{\text{NILE}}$ uses the same LIREx architecture but with the explanations generated from NILE.

presented in Table 3.7. We observe that since **LIREx** is trained with rationale-enabled NLEs, it suffers only a small performance drop when presented with a randomly selected NLE. On the other hand, if we randomly select an NLE generated by **NILE**, the performance drops significantly compared to when choosing just the best NLE. This shows that (a) **NILE** has a tendency to generate more spurious explanations, and (b) if a spurious explanation is used for training the model, the performance drops significantly. On the other hand, **LIREx** does not use labels when training the generator, and hence, produces fewer spurious explanations, so even a randomly-selected explanation is still relevant.

	SNLI-Dev	SNLI-Test	MNLI-M	MNLI-Mis
D+E	92.22	91.87	79.85	79.79
D	90.95	91.07	77.10	76.88
E	62.40	62.21	43.35	43.93

Table 3.8: Faithfulness analysis of LIREx on both SNLI and MultiNLI data. "D+E" uses both P-H pairs and selected explanations as inputs for inference model, "D" uses only the P-H pair, and "E" uses only the selected explanations.

3.4.4 Faithful and relevance evaluation

Faithfulness It is argued by *DeYoung et al.* (2019) that a rationale-augmented classifier may not necessarily rely on the rationales but on the original data. Therefore, they propose to measure the faithfulness of the rationales by measuring the *comprehensiveness* (removing rationales from input) and *sufficiency* (using only the rationales as input). Since the **LIREx** inference model uses the generated explanations instead of rationales as input, following *Kumar and Talukdar* (2020), we probe the model by removing explanations and using just the explanations to measure comprehensiveness and sufficiency. As shown in Table 3.8, when compared to the complete input, removal of explanation from the input reduces the performance on all data sets. Just using explanations leads to a significantly larger drop in performance, which is expected because an explanation is more meaningful when combined with the appropriate context (P-H pair) rather than by itself. These two observations show that the model depends on both the P-H pairs and explanations to make predictions, and that the explanations do demonstrate faithfulness. However, it is not clear why the effect on comprehensiveness is not as significant as that on sufficiency.

Relevance Evaluation Finally, we postulate that trustworthy explanations should be consistent with the appropriate rationale used to interpret the label. Given a specific example that contains different rationales leading a same label, the generator should be able to generate different yet reasonable explanations for each kind of

	SNLI-Dev	SNLI-Test	MNLI-M	MNLI-Mis
NILE	84	84	-	-
LIREx	99	97	95	95

Table 3.9: Manual evaluation of the relevance score over 100 randomly sampled data from each data set by two annotators with the inter-rater agreement of 0.95. NILE evaluations of MultiNLI corpus are missing because we do not have ground truth rationales from human annotators.

rationale (cf. Figure 3.1a). We analyze the generated NLEs based on their **relevance** to human interpretation. From each data set, we randomly sampled 100 examples and ask two annotators to judge the relevance of the generated explanations. Each annotator was provided with context information (premise, hypothesis, rationale, and explanation), and asked to label them as 1 if they agree that the information about the rationales is contained in the explanation, or 0 otherwise. Since **NILE** does not use rationale to generate explanations, we use human-provided rationales in the dataset as the reference target. For **LIREx**, we used predicted rationales as reference targets. As shown in Table 3.9, **LIREx** is able to maintain a high relevance score between explanations and predicted rationales, even when transferred to the out-of-domain data sets. This shows that the rationale-enabled explanations in **LIREx** are more aligned with human interpretation of the rationales.

3.5 Conclusion and open questions

This work demonstrates the possibility of mimicking human insights in the NLE format and shows that highly relevance NLE explanations improves the correspondence of the framework. In this chapter, I first identified two flaws in the current strategy of using NLEs for the NLI task. To overcome these limitations, I propose a novel framework, LIREx, that incorporates a rationale-enabled explanation generator and an instance selector to augment NLI models with only relevant, plausible NLEs. The proposed framework achieves significant improvements over a strong baseline by 0.32

accuracy points on the SNLI data set. It is comparable to the current state-of-the-art performance on the task. When evaluated over an out-of-domain MultiNLI data set, the proposed approach demonstrated significantly better performance than previously published results without fine-tuning. Furthermore, I conduct extensive qualitative analysis to evaluate each component of the LIREx framework, which showed that LIREx generates flexible, faithful, and relevant NLEs that allow the model to be more robust to spurious explanations and better aligned to human interpretation.

This study also opens new challenges and questions, such as how LIREx could be generalized into other domains. As a pipeline in the general domain, LIREx requires large-scale annotations to train a reliable NLE generator, making it difficult to be transferred into domains where large annotated data sets are unavailable. In the medical domain, there are large-scale unlabeled patient notes, yet specially trained expertise is expensive to acquire for annotation. Therefore, domain experts usually prefer choosing different human insights that require less annotation, such as rationale tokens or sentences (preserved as potential evidence in patient notes) and heuristic rules.

Taking the above challenges into consideration, in the next two chapters, I will focus especially on addressing challenges in the medical domain and discuss how we could utilize other types of human insights to design grey-box pipelines.

CHAPTER IV

Improving Document Classification via Sentence-based Evidence Support

In this chapter, I investigate (1) mimicking human insights with both heuristic rules and black-box models and (2) improving system performance with sentence rationale for identifying the medication status of patients with inflammatory bowel disease in clinical notes.

In particular, I introduce a framework that effectively turns human insights into a sentence-based evidence identification model from documents. Then I use the identified evidence to support document-level decisions. The evaluation metrics included in this study are task-designed performance metrics on both document and sentence levels. Document-level performance evaluates the *correspondence* of predicted sentence explanations and the sentence-label performance evaluates the *coherence* of predicted sentence evidence.

4.1 Motivation

Compared to sentence classification, document classification presents a unique challenge to the state-of-the-art models, which is the length of context. Despite the recent success of deep learning-based methods, it is often impractical for models to

take an entire document as input because the document usually exceeds the model's memory limit. On the other hand, in critical domains such as the clinical domain, clinicians need to identify evidence in patient notes to support any clinical decisions, which requires the model to not only make accurate predictions but also make decisions based on valid evidence support. Therefore, in this chapter, I conduct a clinical case study to investigate a pipeline approach that makes document-level predictions by utilizing sentence-based evidence support.

Medication status is a crucial component of a patient's medical records. For example, the "Current Medications" section in a patient note contains medications that are currently being used for patients; the "Prior Medications" section has medications that have been stopped for various reasons (e.g., allergies, patients showing no improvement, and finance changes). Being able to successfully access the status of these medications is critical because it provides information on the patient's medical history and guides future treatment. However, identifying the medication status is time-consuming and challenging work because inconsistent status cues could be distributed across the entire note. It is essential that the decision-making is based on all the cues.

Recent studies (*Zhang et al.* (2016); *DeYoung et al.* (2020); *Arous et al.* (2021); *Du et al.* (2019)) demonstrated the possibility of combining the interpretability and strength of deep learning-based models for making predictions on patient note level. However, there are two critical questions that have not been answered.

The trade-off between annotation cost and performance gain Identifying evidence requires additional human efforts. For example, in addition to the document label, the annotators must also identify the evidence segments that support the document label. Such identification could be of different granularity, including plain text identification and evidence-level label annotation, the former of which is considered to be cheaper than the latter. How much performance gain can each evidence support

Scenario	Supervision	Annotation	Deep learning
1	weak	no	optional
2	full	yes	no
3	full	yes	yes

Table 4.1: Three scenarios that are investigated in this work.

method bring is to be discussed.

In the scenarios where training deep learning models is impractical, is evidence support still important for patient note predictions? There are often cases where supervised annotations and training deep learning models are not available. In such a case, could we still achieve the same performance gain using computationally less expensive methods?

In this study, to build a strong evidence-based document classifier that predicts patients' medication use history based on the evidence support extracted from their health records, I investigate different scenarios to create evidence-level support with different data availability and model complexity, as presented in Table 4.1.

4.2 Related work

The recent success of deep learning models has significantly advanced document classification tasks. *Liu et al.* (2017); *Chen* (2015); *Johnson and Zhang* (2014) developed methods with convolutional neural networks and achieved significant progress on long-text classification tasks. Later, *Adhikari et al.* (2019b) showed that recurrent neural networks also yield strong performance. Recently, in the current trend of using pre-trained language models, state-of-the-art performance in document classification tasks has been achieved by fine-tuning language models that are pre-trained on large-scale text corpora, such as BERT (*Devlin et al.* (2018)), RoBERTa (*Liu et al.* (2019)), or XLNet (*Yang et al.* (2019)), etc. However, despite the powerful semantic representation

learned from pre-training, one of the main challenges with BERT-like models is that they cannot handle long text. To tackle this issue, *Adhikari et al.* (2019a) proposed DocBERT by distilling BERT into a simpler neural network, and *Beltagy et al.* (2020) proposed Longformer by re-designing the attention mechanism in BERT to make it scale linearly with text sequence.

Another perspective of document classification focuses on modeling the structure of the documents. *Yang et al.* (2016) proposed a hierarchical attention network that distributes attention values over sentences and words in a document. *Zhang et al.* (2016) developed a rationale-augmented convolutional network that uses annotated rationale sentences as additional supervision to train a model to generate attention values. *DeYoung et al.* (2020) further showed that document classification often relies on only a small amount of "evidence". Therefore, using the evidence along would also lead to correct predictions. Considering the document structure like these methods, the attention values could also be viewed as a way to interpret model decisions.

4.3 Framework

In this section, I introduce the pipeline framework to extract evidence and make predictions on the document level.

Overview The overall workflow of the approach is presented in Figure 4.1. As step (a), a patient note is first pre-processed with section header extraction and sentence segmentation. Then in step (b), we extract a set of evidence candidates from the note and create evidence candidate instances. Each candidate instance consists of a (h, m, t) triplet, denoting section header, medication mention, and sentence text. In step (c), each candidate instance is predicted with an evidence-level label by the evidence classifier. Last, in step (d), the note classifier makes the final prediction by aggregating the information collected and predicted on the evidence level. Now I will

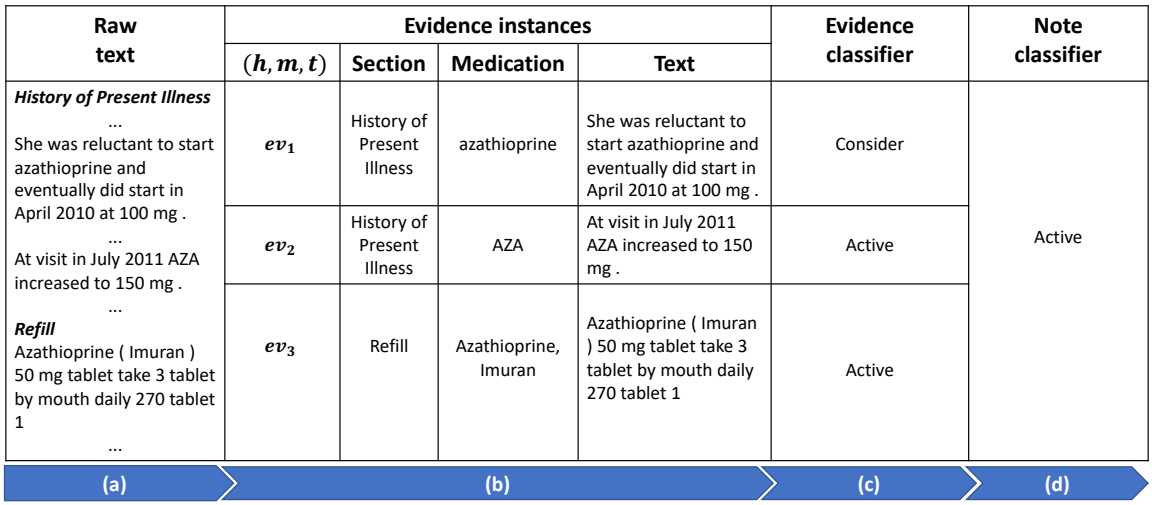

Figure 4.1: Overview of the medication classification framework.

introduce each step in detail.

Pre-processing Patient notes are pre-processed with medspaCy library (*Eyre et al.* (2021)). Each note is tokenized into sentences, and each sentence is associated with the section header to which it belongs. The section headers (e.g. *History of Present Illness* and *Refill* in Figure 4.1) are extracted via keyword-matching with SecTag (*Denny et al.* (2009, 2008)).

Evidence extraction In this step, we extract evidence candidates containing mentions of the medications that are related to IBD treatments. Table 4.2 summarizes the eight medication groups considered in this study, and each medication group is expanded with a list of common synonyms.

Evidence classifier The evidence classifier performs classification on extracted evidence. For an evidence instance extracted from a note, the evidence classifier predicts the corresponding medication status based on the contents of the evidence itself without using any global information from the note. As described above, each evidence instance is represented as a (h, m, t) triplet, and the task for the evidence

Group	Medications
IFX	Remicade, Infliximab, Inflectra
ADA	Humira, Adalimumab
CZP	Cimzia, Certolizumab pegol
GOL	Simponi, Golimumab
UST	Stelara, Ustekinumab
VDZ	Entyvio, Vedolizumab, Vedo
Thio	Imuran, Azathiprine, AZA, 6-Mercaptopurine, 6-MP
MTX	Methotrexate, Trexall
ASA	Mesalamine, Pentasa, Lialdia, Apriso, Asacol, Delzico, Canasa, Rowasa

Table 4.2: Summary of IBD medications.

classifier is to predict the medication status of the medication m based on the context information, namely, section header and evidence text.

As introduced earlier, in this work, I investigate three different scenarios (Table 4.1) considering different data availability and model complexity. In each scenario, training data is created based on different human efforts.

In scenario 1, the cost of human annotation is too high to be practical, and only large-scale unlabeled data instances are available. In such case, I apply weak supervision (WS) methods to create a weakly annotated data set by directly utilizing heuristic rule-based domain knowledge (*Ratner et al.* (2017, 2018)).

First, I design a set of heuristic triggers where each trigger serves as a weak labeler that votes for a label if the instance is matched by the rule. Two types of trigger rules are developed, denoted as $ContextTriggers = [c_1, c_2, ..., c_n]$ and $SecHeadTrigger = t$, where c_i and t are designed contextual patterns and the section header, respectively, that help indicate specific status for a medication. For example, the context pattern "currently on Humira" indicates that Humira (ADA group) is currently being actively used by the patient; similarly, if the evidence appears under the section header *Prior Medication*, then we can decide the current status of Humira is prior use. Table 4.3 present some examples of the trigger. In this work, 17, 23, and 17 rules are created for "Active", "Prior", and "Consider" classes, respectively.

Feature	Examples		Status
ContextTrigger	currently on $[m]$	$\rightarrow$	Active
	had difficulty with $[m]$	$\rightarrow$	Prior
	the option of $[m]$	$\rightarrow$	Consider
SecHeadTrigger	Current Medication	$\rightarrow$	Active
	Prior Medication	$\rightarrow$	Prior
	Allergy	$\rightarrow$	Prior

Table 4.3: Examples of ContextTrigger and SecHeadTrigger

Once the trigger rules are created, they are applied to a set of unlabeled data instances to generate a label matrix, Λ, where $\Lambda_{i,j}$ is the vote of the jth trigger on the ith instance. Then, without knowing the ground-truth labels, a generative model distribution could be defined as

$$\mathcal{P}(\Lambda_i, y_i) = \frac{e^{\sum_j^n \theta_j \phi_j(\Lambda_{ij}, y_i)}}{\sum_{\Lambda', y'} e^{\sum_j^n \theta_j \phi_j(\Lambda', y')}} \tag{4.1}$$

The model θ could be estimated by minimizing the negative log marginal likelihood of the label matrix (*Ratner et al.* (2018)).

$$\theta = \arg\min_{\theta} -\sum_i \log \sum_y \mathcal{P}_\theta(\Lambda_i, y) \tag{4.2}$$

In scenario 2, a data annotation budget is available, but complex deep learning methods is not available. In this case, I investigate two linear yet strong text classification models for supervised training, namely, Support Vector Machine (SVM) and Naive Bayes (NB). To train these two models, I first include the output of the previously designed heuristic rules (in scenario 1) as features. Furthermore, I also expand the feature set by including N-grams within a window size, w, around the medication mention.

In scenario 3, the same data annotation budget is available, and deep

Document instance									
History of Present Illness			Current Medication			...	Assessment and Plan		
Active	Prior	Consider	Active	Prior	Consider	...	Active	Prior	Consider
0.21	0.78	0.01	0.95	0.03	0.02	...	0.61	0.34	0.05

Figure 4.2: Example of document feature representation for SVM and NB.

learning models are also accessible. In this setting, I choose BERT (*Devlin et al.* (2018), a powerful neural network model pre-trained on large text corpora, as the main backbone of the text classifier. To convert the data into the input format preferred by BERT, each evidence instance is converted into the following format: $[CLS]h[SEP]w_1, w_2 \dots [MED] \dots w_n[SEP]$ where h is section header, $w_1, w_2 \dots [MED] \dots w_n$ is the evidence sentence and $[MED]$ is a special marker denoting the mask of the medication mention. I use $[MED]$ instead of the actual medication name to enforce the model to learn from the evidence context and prevent the model from over-fitting towards some particular medication names. Following the classification strategy in BERT, we extract the representation of the token $[CLS]$ and then apply a fully connected neural network layer to perform classification.

Note classifier The note classifier makes the final prediction by leveraging the evidence support predicted by the evidence classifier. I consider three different strong classification models as note classifiers, including SVM, NB, and LSTM (denoted as BERT-LSTM).

To train **SVM** and **NB** models, for each patient note, I accumulate predicted probability distributions from the evidence classifier on the evidence candidates. Then the accumulated probabilities are converted into feature vectors grouped by section headers, as shown in Figure 4.2

BERT-LSTM is built based on the BERT evidence classifier. Specifically, once the evidence classifier (BERT model) is trained, it is used as the evidence encoder

Data	Train	Val	Test	Unlabeled	Total
Note	218	55	924	1000	2197
Evidence instance	1755	381			
Note instance	586	137	2324	2535	5582

Table 4.4: Summary of data statistics

to generate sentence representations. Then the LSTM model is fine-tuned for note classification. This idea is also similarly discussed by *Adhikari et al.* (2019a). To investigate the use of the attention mechanism for interpretability, I also create a variant model, **BERT-LSTM-Attention**, that includes an attention layer over the LSTM. The attention mechanism distributes attention values over evidence candidates.

4.4 Experiments

In this section, I first introduce the data set that is used in this study. Then I show that the use of evidence brings substantial performance gain. I also demonstrate the effectiveness of the *ContextTrigger* and *SecHeadTrigger* via an ablation study. Lastly, I conduct analysis to showcase the explainability of the framework.

4.4.1 Data set

Note-level annotation The data set is collected from a large tertiary care health system that includes outpatient gastroenterology consultation notes from 2015-2018, which contains 2,743 unique medical notes. Each note is separately annotated with respect to one particular medication group for IBD disease. As presented in Table 4.2, nine medication groups are considered, namely, IFX, ADA, CZP, GOL, UST, VDZ, Thio, MTX, and ASA. Three labels are used for annotations: Active (i.e., the medication is currently being used by the patient), Prior (i.e., the medication was active but has been stopped), and Consider (i.e., the medication was considered or

planned for use in the future). 546 notes that do not fall in these three categories with any medication groups are discarded. 2197 notes are left in total.

Evidence-level annotation 218 (10%) and 55 (3%) notes are randomly sampled as training and validation data, and they are manually annotated with the sentence-based evidence. 1000 notes from the test set are held out as extra unlabeled data when the evidence classifier is trained with weak supervision. Table 4.4 presents the detailed statistics of the data set. The evidence instances are used for training evidence classifiers, and the note instances are used for training note classifiers.

4.4.2 Performance evaluation

The results of the evidence classifier Table 4.5 shows the overall performance of the evidence classifier on validation data. As described above, I consider different methods for three scenarios, WS[1] for scenario 1, NB[2] and SVM[3] for scenario 2, and BERT[4] for scenario 3.

As shown in Table 4.5, WS yields the lowest performance among the four models. However, it is also worth pointing out that this performance is achieved without using any ground-truth label in training data. The next two models are NB and SVM, which perform comparably similar. The best performance is reported by finetuning BERT, which significantly outperforms other methods. Due to the superb performance of BERT, we use BERT as the evidence classifier for the following note-level classification.

[1]Weak supervision is trained with Snorkel MeTaL (*Ratner et al.* (2018))

[2]NB is trained with grid search over (1) window size, w, used as context span to extract n-gram features from, (2) number of features used for training, (3) N-gram size, and (4) the smoothing parameter in computing the probability of a specific feature appearing in a particular class. As a result, the final NB model extracts unigram and bigram features within the window size 8. The top 4,500 features (ranked by term frequency) are selected, and the best smoothing parameter is set as 0.4

[3]SVM is trained with the same strategy as NB, which results in the same window size and feature number, and the regularization parameter is 0.6

[4]BERT is initialized with the embedding provided by *Peng et al.* (2019) who trained the BERT embedding using large-scale clinical corpora including the MIMIC-III data set (*Johnson et al.* (2016)) and PubMed abstracts (*Fiorini et al.* (2018)).

Evidence classification	Val	
	Micro F1	Macro F1
WS	0.8609	0.7869
NB	0.8766	0.8381
SVM	0.8740	0.8212
BERT	**0.9160**	**0.8936**

Table 4.5: Overall performance result of evidence classifiers on validation data. The results are the average of five runs with randomly selected seeds. Note that the WS is trained with sentences extracted from 1000 extra notes (in addition to the training instances), and all data examples are treated as unlabeled data.

Note classification	Val		Test	
	Micro F1	Macro F1	Micro F1	Macro F1
NB	0.8613	0.8649	0.8605	0.8508
SVM	**0.8713**	0.8753	0.8693	0.8604
BERT-LSTM	0.8686	**0.8778**	**0.8906**	0.8789
BERT-LSTM-Attention	0.8686	0.8767	0.89	**0.8837**

Table 4.6: Overall performance result of note classifiers on validation and test data.

The results of the document classifier Table 4.6 presents the result of note classifiers on validation and test data. According to Table 4.5, BERT yields the best evidence classification performance. Therefore, both NB and SVM note classifiers use the prediction of the BERT evidence classifier. [5].

I follow the same training strategy (e.g., 5-fold cross-validation and grid search for NB and SVM, and an average of 5 random runs for DocBERT) as the evidence classifier to tune the three models. The best NB uses 0.6 for the smoothing parameter, and SVM uses 0.25 for the regularization parameter. BERT-LSTM and its variant with attention use the same random seeds as the BERT evidence classifier.

[5]Due to the randomness during the training of neural networks, the predicted probability distributions could vary drastically with different random setups, which makes it questionable to use BERT evidence prediction for interpretation. To alleviate this issue, I used MC dropout to estimate the distribution (*Gal and Ghahramani* (2016))

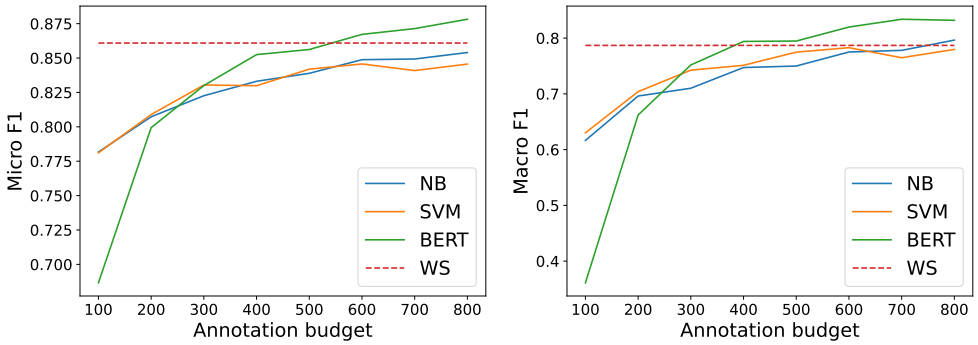

Figure 4.3: Performance of different methods with limited annotation budget

4.4.3 Evidence classifier with limited annotation budget

As presented in Table 4.5, supervised methods (i.e., NB, SVM, and BERT) outperform weak supervision that does not rely on human annotation. However, large-scale data annotation in the health domain is often not available. Therefore, it is crucial to effectively and efficiently utilize human domain knowledge to build the model with reasonably strong performance.

In Figure 4.3, I present the performance of each method with different annotation budgets of up to 800 randomly sampled evidence instances. As shown in the figure, when the budget is extremely small (e.g., < 300), BERT gives the worst performance due to the lack of training data. As the budget grows, BERT starts showing significant performance gain due to its superiority in modeling text semantics. This figure also demonstrates that, when only a limited budget is available for creating annotations, WS presents a unique advantage as it does not rely on data ground-truth and could scale well with carefully designed heuristic rules.

4.4.4 Effectiveness of evidence support

To demonstrate the effectiveness of using evidence support in our approach, for each note classifier, I compare its performance by utilizing different levels of evidence granularity to train the model: (1) Full note: I use the full note text to train the

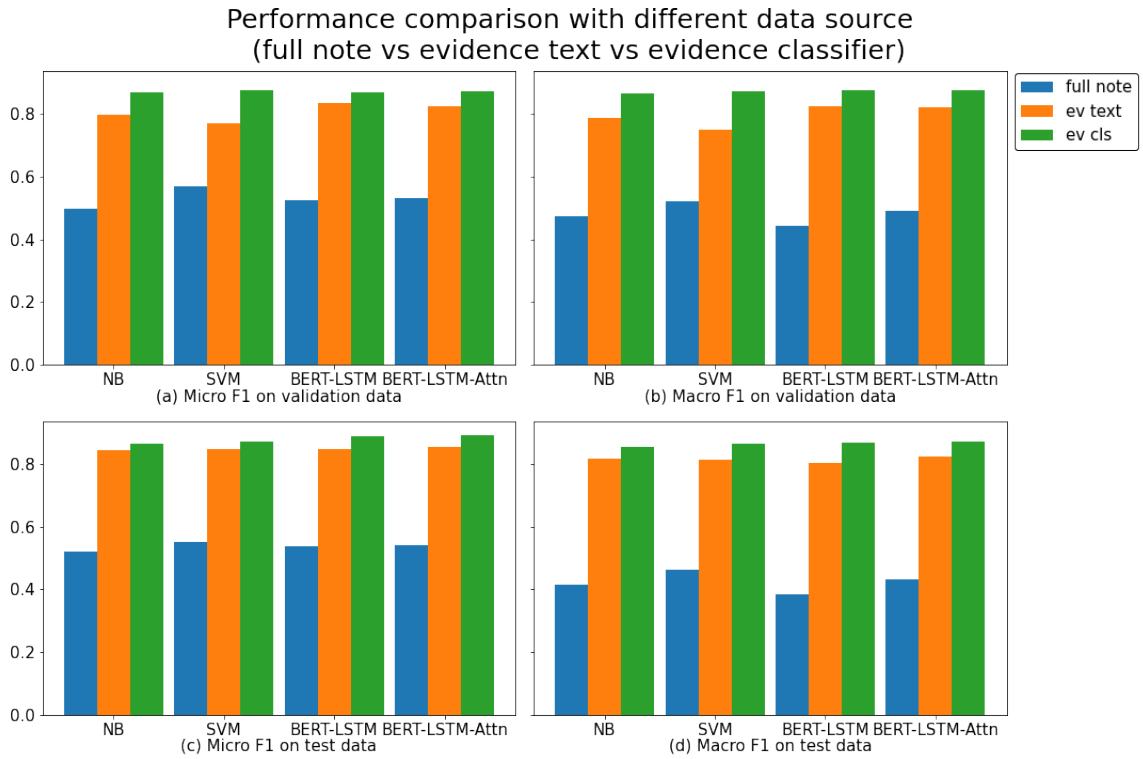

Figure 4.4: Performance comparison with different granularity of evidence support.

models without using any evidence support, (2) Evidence text: I use only the extracted evidence text to train the models, and the evidence labels are not included in the input, and (3) Evidence classier: I use the predictions of the evidence classifier to support the training of the note classifier, which are the model presented in Table 4.6.

As shown in Figure 4.4, across all of the four note classifiers, using evidence text brings substantial performance gain on both Micro and Macro F1 scores on validation and test data. Furthermore, with the support of label propensity from the evidence classifier, the performance is further improved for all models.

4.4.5 Interpretability analysis

While the experiments show that using evidence can provide substantial performance improvements, I also want to highlight the importance of interpretability in the model predictions as explaining model decisions is desired in the clinical domain. In

Class	Medication	Evidence	Predicted evidence label distribution			Score
			Active	Prior	Consider	
Active	Thio	[CLS] hpi [SEP] she was reluctant to start [MED] and eventually did start in april 2010 at 100 mg . [SEP]	0.2037	0.015	0.7807	0.0784
		[CLS] hpi [SEP] at visit in july 2011 [MED] increased to 150 mg . [SEP]	0.9980	0.0012	0.0008	0.1131
		[CLS] refill [SEP] [MED] ([MED]) 50 mg tablet take 3 tablet by mouth daily 270 tablet 1 [SEP]	0.9968	0.0019	0.0013	0.8085
Prior	IFX	[CLS] hpi [SEP] she was started on [MED] in 2009 and around that time , developed a massive pe that required surgical thrombectomy . [SEP]	0.1773	0.8214	0.0013	0.3346
		[CLS] hpi [SEP] [MED] had to be discontinued because of infusion reactions , and she was switched to humira in 2009 . [SEP]	0.0472	0.9521	0.0007	0.6654
Consider	CZP	[CLS] assessment and plan [SEP] alternatively , could do nurse injection of [MED] . [SEP]	0.0866	0.0041	0.9093	0.3225
		[CLS] assessment and plan [SEP] if he is not eligible ; however , we may have to do our best to maximize the benefit of [MED] with nurse injection . [SEP]	0.6828	0.0040	0.3132	0.0840
		[CLS] assessment and plan [SEP] if this does not work out , i think this seems [MED] is a reasonable secondary choice , but only with nurse injection , not with him injecting himself as this failed with humira . [SEP]	0.0991	0.0032	0.8977	0.2799
		[CLS] assessment and plan [SEP] he could try [MED] as a nurse injected option since he has great difficulty injecting himself ; however , i am concerned he may not succeed with this as he has previously had difficulty with remicade and this would be his third anti-tnf . [SEP]	0.0163	0.0038	0.9799	0.3136

Table 4.7: Examples of model interpretability.

Table 4.7, I showcase the interpretability of our approach with three examples using BERT-LSTM-Attention. Each example comes from a particular category. For instance, the document instance is labeled as Active for the Thio medication. Three pieces of evidence are extracted from the original document text, listed by the order of their occurrence. As shown in the table, the last evidence is assigned with a larger attention value than the other two, indicating its dominance in the prediction. Intuitively, the last piece of evidence acquires a large score because it occurs under the "refill" section in the document, which is a strong indicator stating that the medication is currently active.

Similarly, in the other two examples, our model is also able to find the evidence that is more supportive (e.g., evidence 2 for the Prior example and evidence 1, 3, and 4 for the Consider example.)

4.5 Conclusion and open questions

In this chapter, I presented an interpretable approach for identifying medication status for patients with IBD. By manually annotating the evidence extracted from 200 patient notes and evaluating over 2000 patient notes, the proposed approach yields substantial performance gain across a set of strong baselines, demonstrating the effectiveness of using evidence support. On the other hand, I also investigated how different methods for identifying evidence support would vary based on different annotation budgets. I showed that heuristic rules could help build a fairly strong evidence classifier when the budget is limited, but this advantage started diminishing as more annotations become available for supervised learning. The pipeline also provides reasonable interpretations in the context using deep learning models that could help clinicians verify the model decisions based on evidence-label prediction and note-level attention values. This study showed how to design grey-box pipelines with limited annotation efforts and that heuristic rules have great potential for transferring human insights into machine learning models.

Promising aspects aside, one major challenge with using heuristic rules is that they are often oversimplified and are limited by linguistic variations. Therefore, an open question coming out of this study is how to design heuristic rules that can handle the complexity of natural language. In other words, is there a better way to implement heuristic rules?

CHAPTER V

Enhancing Heuristic Rules with Language Representation for Information Extraction

In this chapter, I investigate a new demonstration of rule-based human insights, namely language-augmented rule or language rule. I investigate if the rule-based human insights could be better learned by going beyond the oversimplified pattern matching and enhancing its semantic matching ability.

In particular, I introduce a framework in the setting of adverse drug event (ADE) detection in which we can effectively train a semantic rule matcher from scratch with only a small annotation budget via human-in-the-loop. The main evaluation metrics included in this study are task-deigned performance metrics on both rule level and task level. The rule-level performance evaluates the *coherence* of semantically matched ADEs, and the task-level performance evaluates the *correspondence* of the trained rules to system performance.

5.1 Motivation

In previous chapters, I investigated human insights with different formats in different tasks. In Chapter III, I demonstrated that human insights could be well preserved in language generation and that the generated language explanations could

help improve model performance in language understanding tasks. At the end of the chapter, I also pointed out that, despite the potential of language explanation, it may suffer from the high annotation cost in domains such as healthcare.

Next, I started investigating methods that require less annotation efforts to mimic human insights. In Chapter IV, I examined if evidence support could be effectively identified from documents and be used to support models for document-level predictions. Via a real-world task in the clinical domain (i.e., identifying medication use history for IBD patients), I showed that (1) keyword matching-based evidence support that requires minimal human annotation efforts leads to substantial performance gains compared to using the full note text, (2) with the extra human effort that only requires annotation over 1,800 evidence candidates extracted from 200 patient notes, a strong evidence classifier could be trained to yield accurate evidence-level label propensity that further improves the document-level performance, and (3) with limited annotation budget, human-designed heuristic rules could help build a strong model for evidence support without having to access to the ground-truth.

It has been shown that heuristic rules derived from human insights could result in high-precision annotations. However, despite the findings from previous chapters, there are two major limitations: (1) Natural language generation requires large-scale training data, which is expensive and impractical in many domains. (2) Human heuristics are usually precise and could scale well; however, the main disadvantage is that they are not good at matching context with high complexity caused by language variations.

As the last study of this dissertation, I explore a way to combine the merits of both natural language and the logical development of heuristic rules, which are referred as language-augmented rules. I describe the difference between standard heuristic rules and language-augmented rules in the following scenario.

Suppose we need to extract adverse drug events (ADE) from patient notes. ADE

Sentence	isADE?
Patient developed rashes from Linezolid.	Yes
Patient developed medication-induced rashes.	Yes
Patient developed rashes.	No
Patient developed rashes after eating penuts.	No
There is no sign of drug rash	No

Table 5.1: Examples of ADEs

refers to any unwanted, uncomfortable, or dangerous effects that patients may experience from taking particular medication(s). Table 5.1 presents some examples of valid and invalid ADEs. All of the four examples use "rash(es)" as candidates.

Standard heuristic rules According to Chapter IV, the following rules could be designed to create annotations. Alternatively, *Zhao et al.* (2021) also demonstrated the possibility that these rules are semantically related. Therefore, each rule could be automatically learned from propagation by defining only a few seeding rules.

- `developed` $*$ `from` $\rightarrow$ `ADE`
- `medication` $-$ `induced` $*$ $\rightarrow$ `ADE`
- `developed` $*$ $\rightarrow$ `None`
- `developed` $*$ `after eating` $\rightarrow$ `None`
- `no sign of` $*$ $\rightarrow$ `None`

However, these rules suffer from two major drawbacks. The first drawback is that they do not scale well for the ADEs that appear in context with higher complexity because the complexities of the rules are pre-defined (i.e. context window size). Therefore, this method will not perform well on the following examples:

- *After taking Linezolid, patient developed serious rashes*
- *Patient developed rashes, which is like induced by her medication*
- *there is no significant sign indicating drug rash*

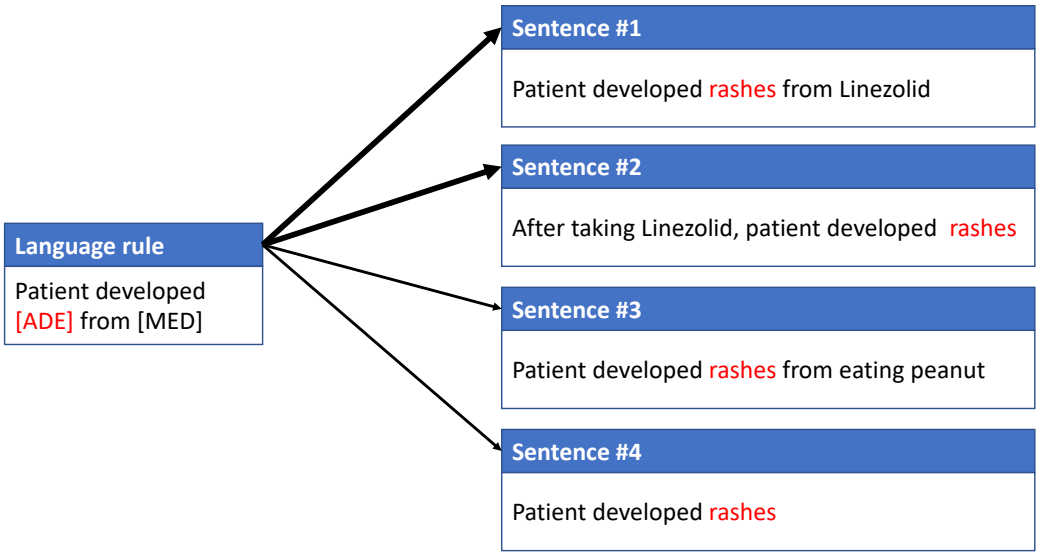

Figure 5.1: Example of language-based rule

The second drawback is that these rules do not align very well with the definition of ADEs. Taking "`developed * after taking` $\rightarrow$ `ADE`" as an example, there is no restriction about the text span (e.g., in the sentence "*developed difficulty of breathing from his pneumonia*", "*difficulty of breathing*" would be matched, which is not ADE.)

Language-augmented heuristic rules Instead of matching text span using the rules described above, an alternative is taking advantage of the language representation by semantically embedding the rules into language, such as the following examples:

- `patient developed [ADE] from [MED].`
- `patient developed [MED] – induced [ADE]`
- `no sign of [ADE] is observed.`

In the above examples, [ADE] and [MED] are special tokens denoting adverse drug effects and medications, respectively. Assuming that we could acquire similar token representations on [ADE], entities that appear in context with higher complexity could also be semantically matched, as illustrated in Figure 5.1.

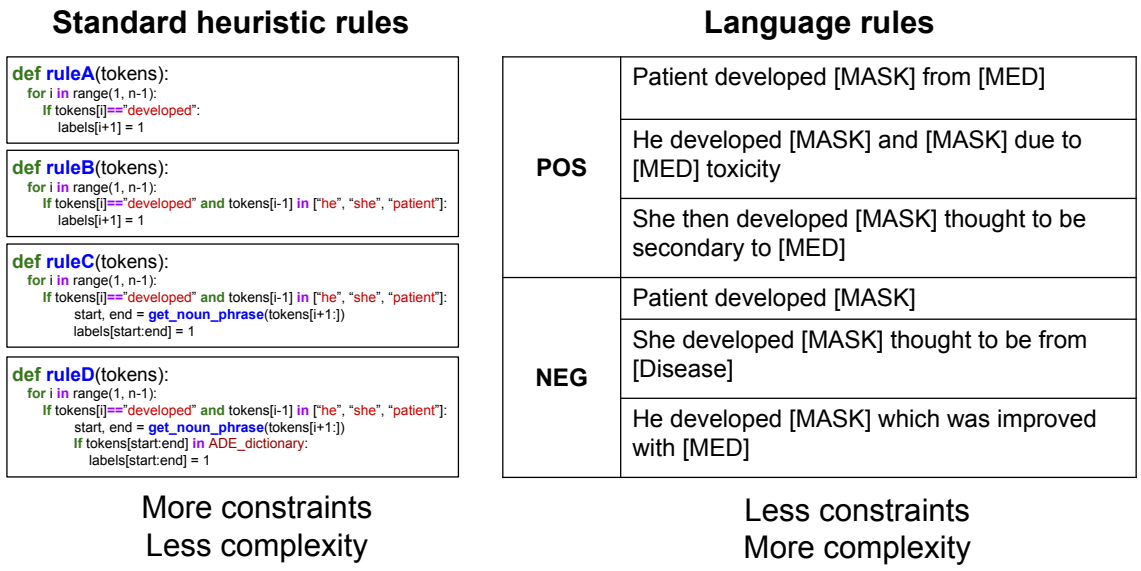

Figure 5.2: Illustration of the difference between standard heuristic rules and language-augmented rules

Figure 5.2 also illustrates the difference between the standard heuristic rules and language-augmented rules using the example `patient developed` $*$. Four standard rules are implemented into programming, and "ruleA" has the least constraints for language variations while being the most fuzzy one among the four. The "ruleD" is the most accurate one by including more language constraints. Yet, it handles the least language complexity. Unlike the standard implementations, the language rules allow for handling high language complexity with less implementation constraints, and it is easier to define a rule with language than the programming-based style.

5.2 Related work

The use of heuristic rules for information extraction has a long history, especially in the pre-deep learning era. Domain dictionaries and pattern-matching are two types of widely used heuristics in rule-based systems or machine learning models that rely on feature engineering (*Nadeau and Sekine* (2007); *Leaman and Gonzalez* (2008)). Recently, most studies focus on deep learning-based methods that have

minimal reliance on heuristic rules (*Collobert et al.* (2011); *Huang et al.* (2015)) and introduce domain-specific knowledge into the models via the pre-training process of language models (*Peng et al.* (2019); *Beltagy et al.* (2019); *Chiu et al.* (2016); *Peters et al.* (2017); *Chen et al.* (2019); *Alsentzer et al.* (2019)). However, recent studies also show that in critical domains (e.g., healthcare, social media, etc.), creating large-scale annotations to train deep learning models with reliable performance is not always available (*Shang et al.* (2018); *Lin et al.* (2020); *Bach et al.* (2017); *Fries et al.* (2017); *Ratner et al.* (2020); *Safranchik et al.* (2020)).

To address the issue with limited data annotation, many weak supervision approaches have been proposed to efficiently create weakly labeled data sets with labeling functions that are developed from human heuristics (*Bach et al.* (2017); *Fries et al.* (2017); *Ratner et al.* (2020); *Safranchik et al.* (2020)). These methods allow domain experts to quickly create annotations with their domain knowledge. One limitation is that while these labeling functions are often highly accurate, they suffer from the low coverage due to language variations. Therefore, *Zhao et al.* (2021); *Li et al.* (2021) proposed to automatically learn new rules by exploring the lexical and contextual cues in unlabeled data sources. To further improve the quality of the automatically learned rules, *Zhang et al.* (2022) proposed an interactive framework to manually select reliable rules by creating additional selection criteria with language prompt and human-in-the-loop.

Another line of work shares a similar mindset to the studies described in Chapter III, which utilizes annotated explanations. *Lin et al.* (2020) created large-scale explanation patterns that serve as annotation triggers and found that these triggers could help improve model performance when training data is limited. However, explanation annotation in this way is also expensive, as described in Chapter III and often requires separate annotation efforts when the task changes.

5.3 Framework

To achieve the goals of language-augmented heuristic rules, I propose a framework, **ActiveRuleMatcher**, to convert the traditional static pattern-matching in standard heuristic rules into an interactive training paradigm with human-in-the-loop.

Overview The overall framework of **ActiveRuleMatcher** is presented is presented in Figure 5.3. First, the domain expert starts with a target standard heuristic rule. However, instead of directly applying the standard rule, the expert writes down a few positive and negative example templates with richer context for the rule. Second, the target rule is used to retrieve actual instances from the data set, and these instances are expected to contain positive patterns. Then, to further improve the performance of the retrieval model, the domain expert manually annotates the top-ranked instances, and these annotated instances, along with the rule extensions, are used to retrain the retrieval model. Once the domain expert is satisfied with the retrieval model or there are no more instances to be retrieved, we move to the last step to train the rule matcher that extracts entities matched by the target rule. Now I will introduce each step in detail.

Target rule extension The framework aims at improving the semantic matching ability of standard heuristic rules. Therefore, we start with a specific rule in the standard form, such as `patient developed` $*$ $\rightarrow$ `ADE`, which would match any following tokens (or noun phrases with more grammatical restrictions) as an ADE candidate. On one hand, obviously, this oversimplified rule is highly error-prone despite the fact that it would catch the correct ADE mentions from data (per Figure 5.1) because the phrase "patient developed" are not semantically related to any symptoms and medications; on the other hand, it will also miss (1) the ADE mentions that start with `he developed` $*$, `she developed` $*$, and `patient then developed` $*$, etc., or (2) the

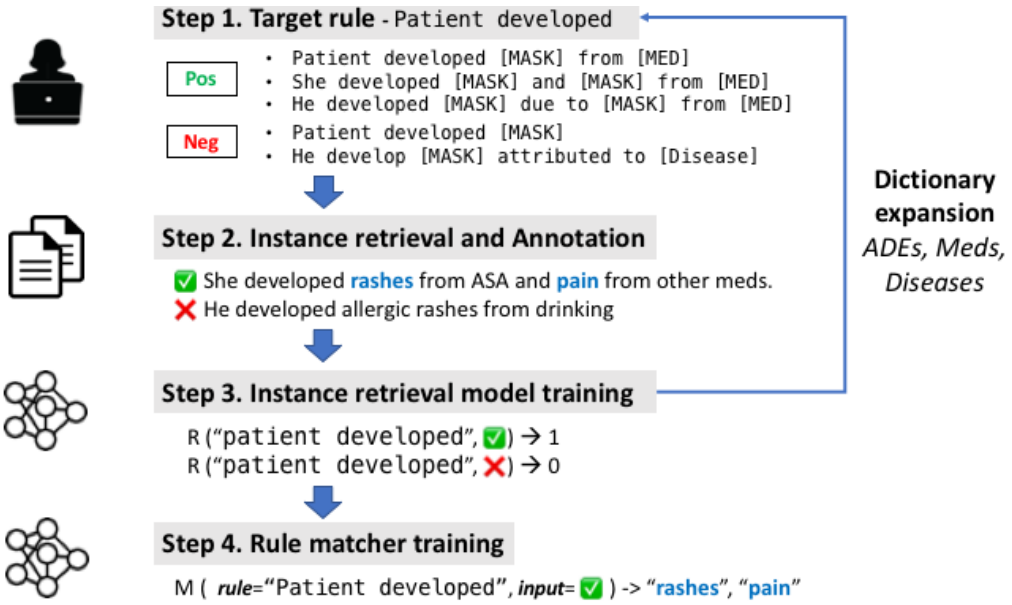

Figure 5.3: Overview of the ActiveRuleMatcher framework

cases where multiple ADE entities are mentioned with conjunctions such as "she then developed *rashes* and *back pain* from Linezolid". To overcome these limitations, we simply ask the domain expert to write down a few very simple positive and negative example templates with richer context using some special markers. For example, "patient developed [MASK] from [MED]" specifies that the rule is looking for the phrases (i.e. [MASK]) that semantically come from the use of a medication (i.e. [MED]). These templates will help the models to learn semantic representations of the rules that are semantically closer to ADE mentions. They will also serve as a way of data augmentations to enhance the model with limited training data. Table 5.2 presents some example extension rules that we used for `patient then developed` $*$.

Instance retrieval and annotation In this step, we use the target rule to retrieve from an unlabeled data set the instances with patterns that semantically match the positive rule extensions created by the domain expert in the previous step. Once the retrieval model returns the instances, we ask the domain expert to annotate the top-ranked k retrieval results with 1 being a valid result and 0 being invalid. These

	Extensions
POS	patient developed [MASK] from [MED] he developed [MASK] and [MASK] due to [MED] toxicity she then developed [MASK] thought to be secondary to [MED]
NEG	patient developed [MASK] she developed [MASK] thought to be from [Disease] he developed [MASK] which was improved with [MED]

Table 5.2: Examples of extension rules for `patient developed` $*$

annotations have three use purposes: (1) they serve as the training data to retrain and improve the retrieval model in the next iteration; (2) they will also be the training examples for the rule matcher in the step 4; (3) the annotation process would expand the vocabulary size of the dictionaries for ADE, medications, and disease names, which could be used to create data augmentations from the rule extension templates.

Instance retrieval model training To train the retrieval model $R(\cdot)$, we formulate the retrieval problem as a binary classification task to predict if the current data contains the pattern that match the target rule. We use BERT *Devlin et al.* (2018) with a fully connected classification layer as our model and the use the format "[CLS] target rule [SEP] instance sentence [SEP]" as the model input. For example, "[CLS] patient developed [SEP] she developed rashes from asa and pain from other meds [SEP]" should predicted as 1 and "[CLS] patient developed [SEP] he developed allergic rashes from drinking [SEP]" should be predicted as 0.

Rule matcher training In this step, we ask the model to identify entities that match the provided language rule for a particular data example. Motivated by *Li et al.* (2019), we formulate this task as a question answering (QA) problem where the question is the provided language rule and the answer is the entities that the rule should match in the text. The training data comprises the annotations from Step 2

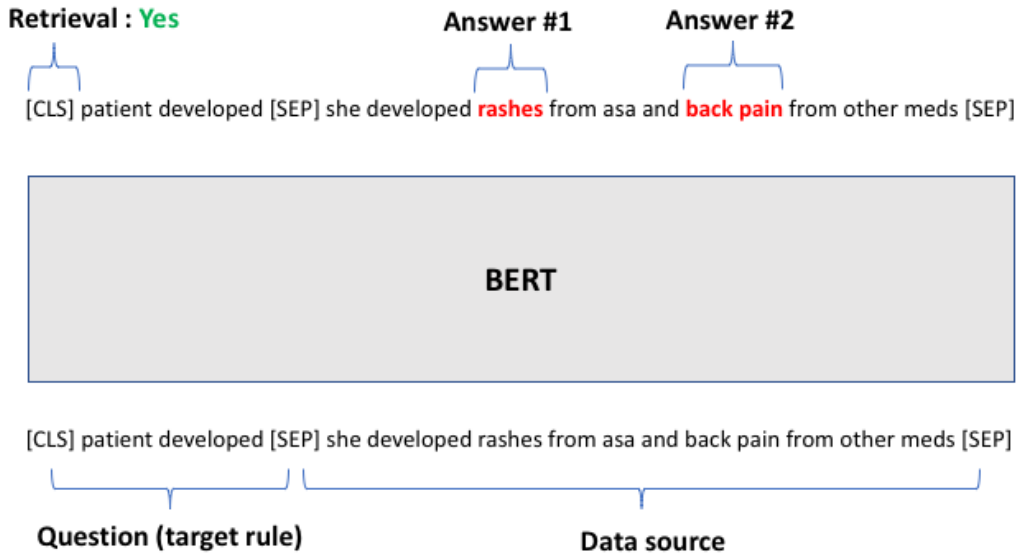

Figure 5.4: Unified retrieval and rule matching model

and the data augmentations created from the human-created rule extension templates. For each rule template, we create an augmentation by randomly sampling entities from the dictionaries that are expanded during human-in-the-loop.

Unified retrieval and rule matching model The retrieval task (Step 3) and the rule matching task (Step 4) share a great deal of similarity since the retrieval task looks at the semantic pattern similarity between the target rule and text, and the rule matcher extracts the entities that match the target rule. Therefore, in this framework, we merge the two tasks into a unified model architecture as presented in Figure 5.4. The classifier for retrieval looks at the global semantic representation of the special token "[CLS]" from BERT, while the rule matcher looks at the local representations of each text token.

5.4 Experiments

5.4.1 Data set

N2C2 2018 data set has been made available as part of the Adverse Drug Event identification task of the national NLP Clinical Challenge (n2c2), 2018 **?**. The goal is to identify nine medical categories of medication-related entities from patient notes. In this study, I focus on the ADE detection as it is the most challenging category due to low volume annotation and the need of domain knowledge. The original data set contains 303 clinical notes as the training set and 202 notes as the test set. In this study, I randomly sample 100 notes (20%) from the original training data as the validation set and the rest (80%) as unlabeled data. Out of the 80% unlabeled data (243 notes), 100 notes were randomly sampled as the unlabeled source for training rule retrieval and matcher. The 202 test set remains unchanged.

Annotation for rule evaluation In addition to the original annotation provided in the N2C2 dataset, for each rule, we individually created annotations which is a subset of the original annotation. For example, in the sentence "patient became *hypotensive* in the setting of likely beta - blocker *toxicity*", both *hypotensive* and *toxicity* are ADE mentions related to "beta - blocker". However, only "hypotensive" semantically fits the rule "`patient became`" and "toxicity" should be matched separately by another rule "`medication toxicity`" that matches the tokens such as "toxicity" or "intoxication" that are caused by medications.

5.4.2 Instance retrieval evaluation

The purpose of the instance retrieval is to efficiently fetch data instances that contain the pattern represented by the target rule so that the human annotator can annotate the ADE mentions that the target rule should match. We use the well-known Normalized Discounted Cumulative Gain (NDCG) metric to evaluate the retrieval

Rules	Retrieved instance	
	Pos	Neg
medication - induced	30	10
patient developed	34	114
medication toxicity	9	1
in setting of medication	16	76
patient became	5	46
secondary to	25	71
medication was held due to	32	12

Table 5.3: Total number of retrieved instances from unlabeled data for each rule.

performance. For each rule, we iteratively retrieve instances up to three iterations with human-in-the-loop, and in each iteration, up to 50 returned instances are annotated. Table 5.3 summarizes the total number of positive and negative instances that are retrieved from the unlabeled data. These instances are used for training the rule matcher, which I will talk more about in detail later.

The performance of the instance retrieval is presented in Table 5.4. For most of the rules, the retrieval model is able to return positive instances with very good performance, such as "patient developed" and "secondary to". For the rules "medication - induced" and "medication was held due to", the evaluation scores are more likely to be biased by the higher number of positive instances. However, if we look at the top ranking positions, the model is still able to find valid examples.

I also want to argue that having negative data returned is not an entirely undesired model behavior as we also expect to have negative samples for the rule matcher training in the next step.

5.4.3 Rule matcher evaluation

To show that the rule matcher performs significantly better than designing standard heuristic rules, I evaluate seven individual rules, each of them focusing on a

Rules	NDCG					#Eval instance	
	@3	@5	@10	@15	@20	Pos	Neg
medication - induced	1.0	0.869	0.915	0.934	0.880	17	3
patient developed	1.0	0.854	0.766	0.640	0.532	10	58
medication toxicity	1.0	0.854	-	-	-	4	1
in setting of medication	0.704	0.655	0.573	0.444	0.370	5	44
patient became	1.0	0.723	0.470	0.364	0.303	3	21
secondary to	1.0	1.0	1.0	1.0	0.931	26	29
medication was held due to	1.0	0.869	0.915	0.891	0.802	14	5

Table 5.4: Evaluation of instance retrieval

different pattern. I considered four strategies of standard heuristic rules, namely, "standard.first", "standard.np", "standard.or", and "standard.key". "standard.first" matches only the first token that follows the rule (e.g. "patient developed *rashes*") or the first token on the left (e.g. "*hypotension* in the setting of metoprolol use"); "standard.np" matches the noun phrase that follows the rule (e.g. "drug - induced *eosinophilic lung disease*") or precedes the rule (e.g. "*acute renal failure* secondary to use of nsaid"); "standard.or" is the logical operation that considers the union of the previous two; and "standard.key" does keyword matching with "*toxicity*" and "*intoxication*". Evaluating these different strategies, it is not hard to see the challenge of designing well-performing heuristic rules for extracting ADEs due to the high complexity of language variance in clinical notes.

The performance of the language rule is highlighted in Table 5.5. The experiments show that the rule-matcher brings substantial performance gain compared to the standard rules, which mainly benefited from the training process where the model learned to semantically match ADE mentions and reduced the impact of language complexity.

Rules	Type	Token-wise			Span-wise		
		Precision	**Recall**	**F1**	**Precision**	**Recall**	**F1**
medication - induced	standard.first	0.586	0.654	0.618	0.517	0.714	0.6
	standard.np	0.667	0.539	0.596	0.563	0.429	0.487
	standard.or	0.588	0.769	0.667	0.517	0.714	0.6
	language	**0.955**	**0.808**	**0.875**	**0.947**	**0.857**	**0.9**
patient developed	standard.first	0.167	0.279	0.209	0.088	0.25	0.130
	standard.np	0.262	0.361	0.303	0.127	0.194	0.154
	standard.or	0.191	0.410	0.260	0.088	0.25	0.130
	language	**0.717**	**0.623**	**0.667**	**0.645**	**0.556**	**0.597**
medication toxicity	standard.key	0.833	1.0	0.909	0.833	1.0	0.909
	language	**0.909**	**1.0**	**0.952**	**0.909**	**1.0**	**0.952**
in setting of medication	standard.first	0.148	0.351	0.208	0.057	0.294	0.095
	standard.np	0.196	0.270	0.227	0.034	0.176	0.057
	standard.or	0.146	0.378	0.211	0.036	0.294	0.063
	language	**0.737**	**0.378**	**0.500**	**0.583**	**0.412**	**0.483**
patient became	standard.first	0.192	0.250	0.217	0.115	0.333	0.171
	standard.np	0.214	0.300	0.250	0.143	0.444	0.216
	standard.or	0.184	0.350	0.241	0.105	0.444	0.170
	language	**0.750**	**0.150**	**0.250**	**0.5**	**0.222**	**0.308**
secondary to	standard.first	0.071	0.171	0.101	0.041	0.167	0.066
	standard.np	0.161	0.122	0.139	0.031	0.125	0.049
	standard.or	0.069	0.171	0.099	0.023	0.167	0.041
	language	**0.800**	**0.878**	**0.837**	**0.679**	**0.792**	**0.731**
medication was held due to	standard.first	0.533	0.205	0.296	0.267	0.182	0.261
	standard.np	0.692	0.231	0.346	0.400	0.182	0.250
	standard.or	0.556	0.256	0.351	0.267	0.182	0.261
	language	**0.786**	**0.846**	**0.815**	**0.630**	**0.773**	**0.694**

Table 5.5: Evaluation of rule matcher

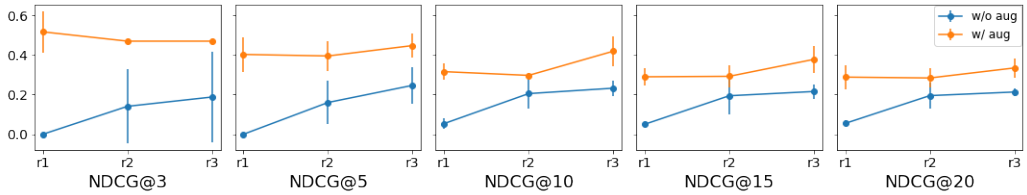

Figure 5.5: Instance retrieval performance for the rule `patient developed` for three iterations

5.4.4 Effects of data augmentation and incremental training for instance retrieval

The data augmentation created from the rule extensions is used for training both instance retrieval and rule matcher. First, I examine the effect of using data augmentation in instance retrieval.

Using the rule `patient developed` as an example (as shown in Figure 5.5), having the extra augmented training examples significantly improve the retrieval performance. As shown in the figure, I compare the performance of the retrieval model using annotated dataset only versus extra data augmentations. In the augmentation strategy, we randomly create 100 augmented examples for each rule extension by sampling from the vocabularies maintained during the human-in-the-loop process. As a result, the experiments show that the augmentation can significantly alleviate the effect caused by having limited training data.

Furthermore, I found that the retrieval performance of the model could be further improved when additional rules are jointly trained. Targeting with the rule `patient developed`, to show that how jointly training the retrieval for other rule could affect the retrieval performance, I incrementally add five more rules to train following the order `medication − induced`, `in setting of medication`, `patient became`, `secondary to`, and `medicatoin was held due to`. As shown the Figure 5.6, once more rules are added to the training, the performance for the target rule could be significantly improved with better robustness. This effect implies that the retrieval of

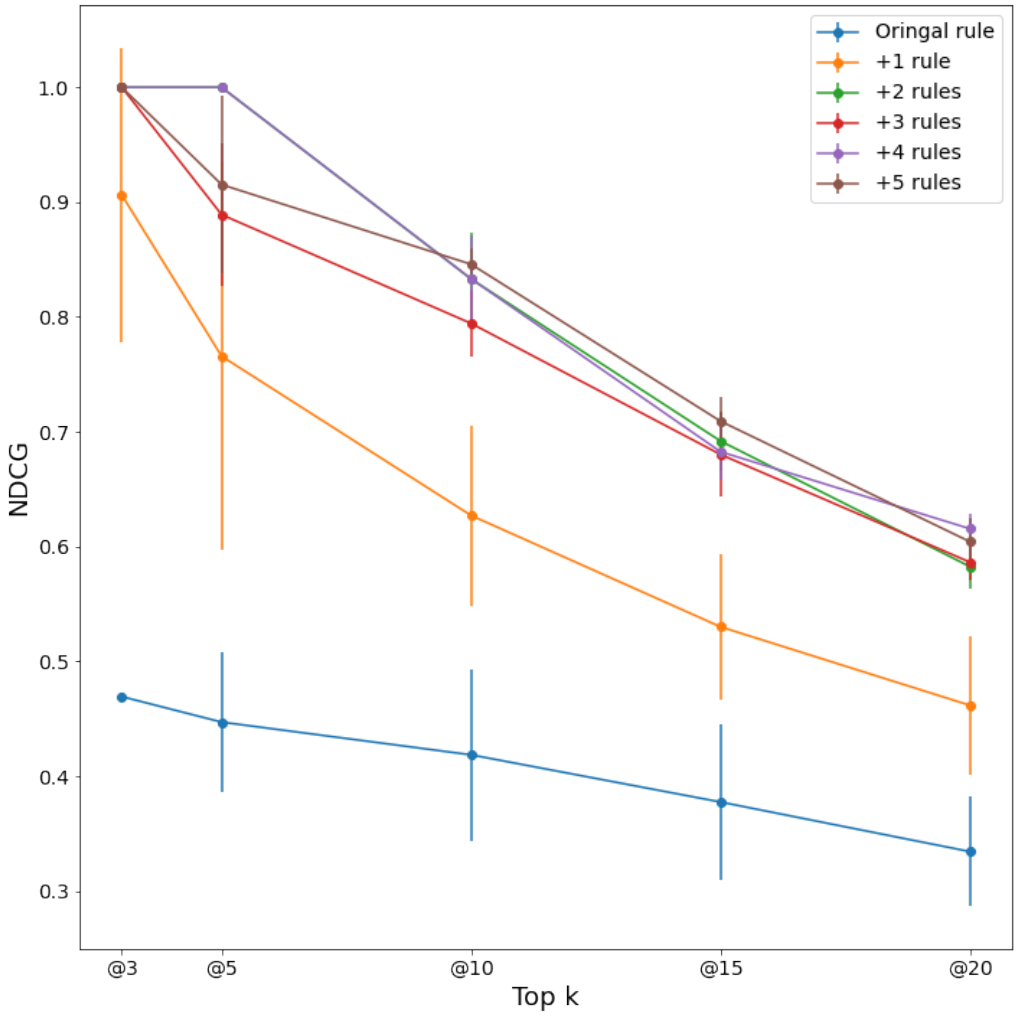

Figure 5.6: Instance retrieval performance for the rule `patient developed` with additional rules being jointly trained

a new rule would start with better performance as the number of previously trained rules increases.

5.4.5 Effects of data augmentation for rule matcher

I also observed a similar effect that the data augmentation brings to the performance of the rule matcher, as presented in Figure 5.7. For each of the seven rules, I evaluated the matching performance when there were 0, 10, 20, and 30 randomly sampled augmented examples created for each rule extension. For all of the seven rules,

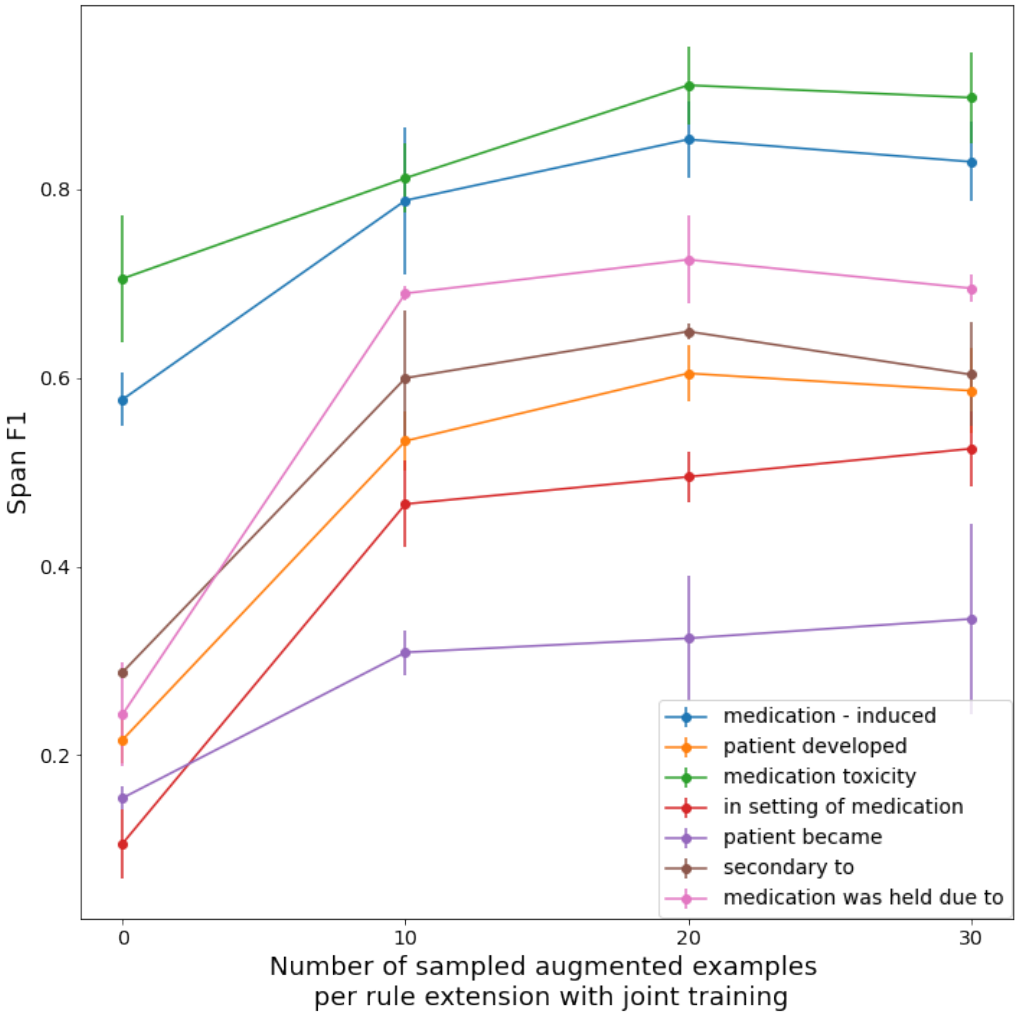

Figure 5.7: Rule matcher performance with seven rules being jointly trained

with additional augmented training examples, the performance was greatly improved, demonstrating that using this efficient data augmentation strategy could significantly reduce the limitation of insufficient training data from the annotation process.

5.4.6 Rule matcher in weak supervision

As introduced in Chapter I and Chapter IV, rules are commonly used in rule-based and weak supervision-based systems that can create high coherence between human insights and system output. However, limited by their oversimplicity, they often suffer from creating low correspondence to the system performance. Therefore, to

Methods	**Precision**	**Recall**	**F1**
Supervised model A	0.6830	0.3349	0.4495
Supervised model B	0.8	0.0769	0.1404
Standard rule			
Majority vote	0.1227	0.5048	0.1974
Weak supervision	0.1637	0.4923	0.2457
Rule matcher			
Majority vote	0.5438	0.4679	0.503
Weak supervise	0.5508	0.4921	0.5198
Standard rule + supervised model A			
Majority vote	0.5074	0.4375	0.4699
Weak supervision	0.4923	0.5128	0.5023
Rule matcher + supervised model A			
Majority vote	0.7027	0.375	0.489
Weak supervision	0.5594	0.5657	**0.5625**

Table 5.6: Evaluation of ADE identification (span level)

evaluate how much performance gain using the rule matcher could yield, I evaluate the rule matcher in both rule-based and weak supervision-based settings as presented in Table 5.6.

To make comprehensive comparisons, I first create two baseline models using fully supervised training, model A and model B. Model A and B share the same BERT-based encoder for named entity recognition *Devlin et al.* (2018). The difference is that model A uses all the sentences in the 100 notes (15, 276 sentences) that are used in the human-in-the-loop of rule matcher. In contrast, model B uses only the

sentences that are annotated during instance retrieval (419 sentences). Different from training the rule matcher, both models are trained with fully annotated sentences in which each token is annotated with either being an ADE token or a background token.

Additionally, to compare the standard rule and the rule matcher, I evaluate them in both a rule-based system with a majority vote and a weak supervision system *Safranchik et al.* (2020) trained on 143 unlabeled notes. By using the rule matcher trained with annotations created from 419 sentences, the system is able to obtain 0.3056 and 0.2741 absolute F1 points on span-level identification via majority vote and weak supervision respectively. Furthermore, taking the trained model A as an additional rule, both majority voting and weak supervision are further improved, and the best performance (0.5625) is reported by the weak supervision system that uses the rule matcher and the model A as weak labelers.

It is worth pointing out that, in these experiments, the fully supervised model is outperformed by the rule matcher-based methods that use significantly less data annotation. The main reason for this promising performance is that the rule-matcher not only contains the high coherence between human insights but also that the training process significantly improves its ability to handle higher orders of language complexity.

5.5 Conclusion and open questions

In this chapter, I presented a framework for increasing the semantic awareness of standard heuristic rules. Via a human-in-the-loop paradigm, I showed that the rule matching could be significantly improved by individually annotating a small group of data for each rule. The effect of creating such enhanced rule matchers is well demonstrated in both rule-based and weak supervision-based systems with substantial performance gain. The framework not only improves the final system performance but also increases the interpretability by following the design principle of "grey-box" pipelines, where each component maintains the coherence to human insights and

correspondence to system performance.

This study opens new questions such as how to incorporate more human insights from the stage of data annotation and at what training paradigm would human insights be most helpful.

CHAPTER VI

Conclusions and Open Questions

In this dissertation, I focus on designing approaches to build grey-box pipelines with high correspondence while maintaining the coherence to human insights. I investigated two directions that are in line with the explainability and interpretability of AI systems: interpretation via human-mimicking and performance improvement via human insights. These two directions focus on two main principles for designing the grey-box pipeline, which is *coherence* and *correspondence*, respectively.

Coherence To explore the possibility of creating coherence between human and AI systems, I focus on the area of natural language processing and investigated three formats of human insights that are common used as explanation source for interpreting AI decisions: natural language, extractive data snippet (tokens or sentences), and heuristic rules. In addition, due to the issue of over-simplicity of standard heuristic rules, I proposed a novel paradigm for enhancing the semantic matching ability of human heuristics via a language-based rule matching framework. To summarize the results:

I first pointed out that natural language explanations tend to be spurious and lack rationale without human involvement. Such explanations would not only harm system explainability but also jeopardize the system's decision. To overcome these issues, I showed that having a rationalizer component to identify rationale tokens from data and

then generating language explanation based on these tokens can significantly improve the quality of the explanation in terms of plausibility, faithfulness, and relevance to human insights.

Second, I explored using human heuristics and extractive sentence-based rationales in a document classification system. This study was conducted in a real-world clinical setting with the challenge of limited annotation budgets. I showed that having a component (i.e., evidence classifier) to mimic human behavior in extracting evidence support and generating decision propensities can significantly improve the system performance on patient notes. Experiments showed that the heuristic rules could well correspond to the expected human behavior in predicting evidence propensities via the weak supervision paradigm, especially when the annotation availability is minimal. The evidence classifier could be further improved when it is trained with more complex neural network models with larger data annotation.

Third, I explored the possibility of improving the standard heuristic rules by providing a more complex semantic matching ability. Using the challenging task, ADE identification, as a demonstration, I showed that the proposed language rule matcher is significantly stronger at semantically identifying ADEs than the standard heuristics that rely on simple pattern matching. Also, the annotation expense for creating such a rule matcher is well considered by training a highly correspondent instance retrieval module that is effectively created via human-in-the-loop.

Correspondence Besides human-AI coherence, another important aspect of this dissertation is correspondence since it directly relates to the performance of the final system output. Therefore, in each study, after the explanations were generated and demonstrated with high coherence, I evaluated how well they corresponded to the expected output.

For the natural language explanation, I showed that it brings additional performance gain in both in-domain and out-of-domain data sets when incorporated as part

of the input for the decision system. The sentence-based explanation identified by the evidence classifier resulted in tremendous performance across four different document classification models, including two white-box models and two black-box models. The language rule matcher also significantly surpassed the standard heuristic rules in both rule-based and weakly supervised systems and even outperformed the fully supervised model with considerably less data annotation.

The experiments in this dissertation pave the way for incorporating a human-focused module to increase the human-AI coherence while maintaining the high correspondence to system performance.

However, there are also concerns and new questions raised from this dissertation. For example, what kind of human insights or explanations should we prefer? As shown in this dissertation, different human insights present different strengths and limitations and require different human attention. The natural language contains richer semantics, but it requires large-scale annotation to train a robust language generator; heuristic rules are easy to apply and do not require any annotation, which makes it scale very well, yet it is limited by its over-simplified pattern matching. To alleviate this issue, I proposed the rule matching paradigm to create semantic matching for the rules. However, further work should be done to investigate the different use of each human insight type and how they can support each other.

Another question that is to be further explored in the future work regards to the evaluation of the explanations. In this dissertation, I focused on evaluating the coherence and correspondence performance of the explanations. The coherence evaluation contains the relevance score in between rationale token and generated language explanation, the performance of the evidence classifier, and the rule matching performance. These evaluations are task-specific and are assumed implicitly related to human insights based on the design logic of the systems. In the future, more studies should be conducted to provide a task-agnostic paradigm in addition to the task

specific ones.

The third question I believe is important to the field is what are other alternatives for incorporating human insights into AI systems. In this dissertation, human insights are incorporated during either the stage of data annotation (e.g., NLE annotation, sentence evidence annotation, rule-matching annotation) or rule implementation (e.g., heuristic rules in weak supervision). Other directions to be explored include creating and learning from explanations during human-AI interactions or pre-learning the commonsense knowledge and reasoning from free text.

More generally, to continue researching this field and exploring answers, I encourage the future work diving deep into the connection between AI systems and human perceptions. As claimed in Chapter I, human is the key in AI applications. How the system output is perceived by humans is important for us to understand why (do we need explanations in AI systems), what (explanations should we use), how (do we design interpretable AI systems), and where (should the interpretable AI systems be applied) questions. I look forward to diving into the future!

BIBLIOGRAPHY

BIBLIOGRAPHY

Adhikari, A., A. Ram, R. Tang, and J. Lin (2019a), Docbert: Bert for document classification, *arXiv preprint arXiv:1904.08398*.

Adhikari, A., A. Ram, R. Tang, and J. Lin (2019b), Rethinking complex neural network architectures for document classification, in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 4046–4051.

Agarwal, R., L. Melnick, N. Frosst, X. Zhang, B. Lengerich, R. Caruana, and G. E. Hinton (2021), Neural additive models: Interpretable machine learning with neural nets, *Advances in Neural Information Processing Systems*, *34*, 4699–4711.

Agrawal, R., T. Imieliński, and A. Swami (1993), Mining association rules between sets of items in large databases, in *Proceedings of the 1993 ACM SIGMOD international conference on Management of data*, pp. 207–216.

Alsentzer, E., J. R. Murphy, W. Boag, W.-H. Weng, D. Jin, T. Naumann, and M. McDermott (2019), Publicly available clinical bert embeddings, *arXiv preprint arXiv:1904.03323*.

Arous, I., L. Dolamic, J. Yang, A. Bhardwaj, G. Cuccu, and P. Cudré-Mauroux (2021), Marta: Leveraging human rationales for explainable text classification, in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, pp. 5868–5876.

Bach, S. H., B. He, A. Ratner, and C. Ré (2017), Learning the structure of generative models without labeled data, in *International Conference on Machine Learning*, pp. 273–282, PMLR.

Baehrens, D., T. Schroeter, S. Harmeling, M. Kawanabe, K. Hansen, and K.-R. Müller (2010), How to explain individual classification decisions, *The Journal of Machine Learning Research*, *11*, 1803–1831.

Bahdanau, D., K. Cho, and Y. Bengio (2014), Neural machine translation by jointly learning to align and translate, *arXiv preprint arXiv:1409.0473*.

Bao, Y., S. Chang, M. Yu, and R. Barzilay (2018), Deriving machine attention from human rationales, *arXiv preprint arXiv:1808.09367*.

Beltagy, I., K. Lo, and A. Cohan (2019), Scibert: A pretrained language model for scientific text, in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 3606–3611.

Beltagy, I., M. E. Peters, and A. Cohan (2020), Longformer: The long-document transformer, *arXiv preprint arXiv:2004.05150*.

Binder, A., G. Montavon, S. Lapuschkin, K.-R. Müller, and W. Samek (2016), Layer-wise relevance propagation for neural networks with local renormalization layers, in *International Conference on Artificial Neural Networks*, pp. 63–71, Springer.

Biran, O., and C. Cotton (2017), Explanation and justification in machine learning: A survey, in *IJCAI-17 workshop on explainable AI (XAI)*, vol. 8, pp. 8–13.

Boecking, B., W. Neiswanger, E. Xing, and A. Dubrawski (2020), Interactive weak supervision: Learning useful heuristics for data labeling, *arXiv preprint arXiv:2012.06046*.

Bowman, S. R., G. Angeli, C. Potts, and C. D. Manning (2015), A large annotated corpus for learning natural language inference, *arXiv preprint arXiv:1508.05326*.

Broniatowski, D. A., et al. (2021), Psychological foundations of explainability and interpretability in artificial intelligence, *NIST, Tech. Rep.*

Camburu, O.-M., T. Rocktäschel, T. Lukasiewicz, and P. Blunsom (2018), e-snli: Natural language inference with natural language explanations, *arXiv preprint arXiv:1812.01193*.

Camburu, O.-M., B. Shillingford, P. Minervini, T. Lukasiewicz, and P. Blunsom (2020), Make up your mind! adversarial generation of inconsistent natural language explanations, in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 4157–4165.

Carton, S. (2019), The design and evaluation of neural attention mechanisms for explaining text classifiers, Ph.D. thesis.

Carton, S., Q. Mei, and P. Resnick (2018), Extractive adversarial networks: High-recall explanations for identifying personal attacks in social media posts, *arXiv preprint arXiv:1809.01499*.

Charness, G., E. Karni, and D. Levin (2010), On the conjunction fallacy in probability judgment: New experimental evidence regarding linda, *Games and Economic Behavior*, *68*(2), 551–556.

Chen, Q., Y. Peng, and Z. Lu (2019), Biosentvec: creating sentence embeddings for biomedical texts, in *2019 IEEE International Conference on Healthcare Informatics (ICHI)*, pp. 1–5, IEEE.

Chen, Y. (2015), Convolutional neural network for sentence classification, Master's thesis, University of Waterloo.

Chiu, B., G. Crichton, A. Korhonen, and S. Pyysalo (2016), How to train good word embeddings for biomedical nlp, in *Proceedings of the 15th workshop on biomedical natural language processing*, pp. 166–174.

Choi, E., M. T. Bahadori, J. Sun, J. Kulas, A. Schuetz, and W. Stewart (2016), Retain: An interpretable predictive model for healthcare using reverse time attention mechanism, *Advances in neural information processing systems*, *29*.

Cohen, J. D., K. Dunbar, and J. L. McClelland (1990), On the control of automatic processes: a parallel distributed processing account of the stroop effect., *Psychological review*, *97*(3), 332.

Collobert, R., J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa (2011), Natural language processing (almost) from scratch, *Journal of machine learning research*, *12*(ARTICLE), 2493–2537.

Craven, M., and J. Shavlik (1995), Extracting tree-structured representations of trained networks, *Advances in neural information processing systems*, *8*.

Denny, J. C., R. A. Miller, K. B. Johnson, and A. Spickard III (2008), Development and evaluation of a clinical note section header terminology, in *AMIA annual symposium proceedings*, vol. 2008, p. 156, American Medical Informatics Association.

Denny, J. C., A. Spickard III, K. B. Johnson, N. B. Peterson, J. F. Peterson, and R. A. Miller (2009), Evaluation of a method to identify and categorize section headers in clinical documents, *Journal of the American Medical Informatics Association*, *16*(6), 806–815.

Devlin, J., M.-W. Chang, K. Lee, and K. Toutanova (2018), Bert: Pre-training of deep bidirectional transformers for language understanding, *arXiv preprint arXiv:1810.04805*.

DeYoung, J., S. Jain, N. F. Rajani, E. Lehman, C. Xiong, R. Socher, and B. C. Wallace (2019), Eraser: A benchmark to evaluate rationalized nlp models, *arXiv preprint arXiv:1911.03429*.

DeYoung, J., E. Lehman, B. Nye, I. J. Marshall, and B. C. Wallace (2020), Evidence inference 2.0: More data, better models, *arXiv preprint arXiv:2005.04177*.

Doshi-Velez, F., and B. Kim (2017), Towards a rigorous science of interpretable machine learning, *arXiv preprint arXiv:1702.08608*.

Du, M., N. Liu, F. Yang, and X. Hu (2019), Learning credible deep neural networks with rationale regularization, in *2019 IEEE International Conference on Data Mining (ICDM)*, pp. 150–159, IEEE.

Eyre, H., A. B. Chapman, K. S. Peterson, J. Shi, P. R. Alba, M. M. Jones, T. L. Box, S. L. DuVall, and O. V. Patterson (2021), Launching into clinical space with medspacy: a new clinical text processing toolkit in python, *arXiv preprint arXiv:2106.07799*.

Fidler, S., et al. (2017), Teaching machines to describe images with natural language feedback, in *Advances in Neural Information Processing Systems*, pp. 5068–5078.

Fiorini, N., R. Leaman, D. J. Lipman, and Z. Lu (2018), How user intelligence is improving pubmed, *Nature biotechnology*, *36*(10), 937–945.

Fries, J., S. Wu, A. Ratner, and C. Ré (2017), Swellshark: A generative model for biomedical named entity recognition without labeled data, *arXiv preprint arXiv:1704.06360*.

Frosst, N., and G. Hinton (2017), Distilling a neural network into a soft decision tree, *arXiv preprint arXiv:1711.09784*.

Gal, Y., and Z. Ghahramani (2016), Dropout as a bayesian approximation: Representing model uncertainty in deep learning, in *international conference on machine learning*, pp. 1050–1059, PMLR.

Hammond, K. R. (2000), Coherence and correspondence theories in judgment and decision making.

Hancock, B., M. Bringmann, P. Varma, P. Liang, S. Wang, and C. Ré (2018), Training classifiers with natural language explanations, in *Proceedings of the conference. Association for Computational Linguistics. Meeting*, vol. 2018, p. 1884, NIH Public Access.

Haugeland, J. (1989), *Artificial intelligence: The very idea*, MIT press.

Hilton, D. J. (1996), Mental models and causal explanation: Judgements of probable cause and explanatory relevance, *Thinking & Reasoning*, *2*(4), 273–308.

Hoffman, R. R., S. T. Mueller, G. Klein, and J. Litman (2018), Metrics for explainable ai: Challenges and prospects, *arXiv preprint arXiv:1812.04608*.

Huang, Z., W. Xu, and K. Yu (2015), Bidirectional lstm-crf models for sequence tagging, *arXiv preprint arXiv:1508.01991*.

Huk Park, D., L. Anne Hendricks, Z. Akata, A. Rohrbach, B. Schiele, T. Darrell, and M. Rohrbach (2018), Multimodal explanations: Justifying decisions and pointing to the evidence, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 8779–8788.

Ibrahim, M., M. Louie, C. Modarres, and J. Paisley (2019), Global explanations of neural networks: Mapping the landscape of predictions, in *Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society*, pp. 279–287.

Johnson, A. E., et al. (2016), Mimic-iii, a freely accessible critical care database, *Scientific data*, *3*(1), 1–9.

Johnson, R., and T. Zhang (2014), Effective use of word order for text categorization with convolutional neural networks, *arXiv preprint arXiv:1412.1058*.

Karpathy, A., J. Johnson, and L. Fei-Fei (2015), Visualizing and understanding recurrent networks, *arXiv preprint arXiv:1506.02078*.

Kim, J., A. Rohrbach, T. Darrell, J. Canny, and Z. Akata (2018), Textual explanations for self-driving vehicles, in *Proceedings of the European conference on computer vision (ECCV)*, pp. 563–578.

Koh, K. H., A. Basawapatna, V. Bennett, and A. Repenning (2010), Towards the automatic recognition of computational thinking for adaptive visual language learning, in *2010 IEEE symposium on visual languages and human-centric computing*, pp. 59–66, IEEE.

Krause, J., A. Perer, and K. Ng (2016), Interacting with predictions: Visual inspection of black-box machine learning models, in *Proceedings of the 2016 CHI conference on human factors in computing systems*, pp. 5686–5697.

Kumar, S., and P. Talukdar (2020), Nile: Natural language inference with faithful natural language explanations, *arXiv preprint arXiv:2005.12116*.

Leaman, R., and G. Gonzalez (2008), Banner: an executable survey of advances in biomedical named entity recognition, in *Biocomputing 2008*, pp. 652–663, World Scientific.

Lee, D.-H., R. Khanna, B. Y. Lin, J. Chen, S. Lee, Q. Ye, E. Boschee, L. Neves, and X. Ren (2020), Lean-life: A label-efficient annotation framework towards learning from explanation, *arXiv preprint arXiv:2004.07499*.

Lei, T., R. Barzilay, and T. Jaakkola (2016), Rationalizing neural predictions, *arXiv preprint arXiv:1606.04155*.

Letham, B., C. Rudin, T. H. McCormick, and D. Madigan (2015), Interpretable classifiers using rules and bayesian analysis: Building a better stroke prediction model, *The Annals of Applied Statistics*, *9*(3), 1350–1371.

Li, J., A. H. Miller, S. Chopra, M. Ranzato, and J. Weston (2016), Learning through dialogue interactions by asking questions, *arXiv preprint arXiv:1612.04936*.

Li, J., H. Ding, J. Shang, J. McAuley, and Z. Feng (2021), Weakly supervised named entity tagging with learnable logical rules, *arXiv preprint arXiv:2107.02282*.

Li, X., J. Feng, Y. Meng, Q. Han, F. Wu, and J. Li (2019), A unified mrc framework for named entity recognition, *arXiv preprint arXiv:1910.11476*.

Lin, B. Y., D.-H. Lee, M. Shen, R. Moreno, X. Huang, P. Shiralkar, and X. Ren (2020), Triggerner: Learning with entity triggers as explanations for named entity recognition, *arXiv preprint arXiv:2004.07493*.

Lipton, Z. C. (2018), The mythos of model interpretability: In machine learning, the concept of interpretability is both important and slippery., *Queue*, *16*(3), 31–57.

Liu, Y., et al. (2019), Roberta: A robustly optimized bert pretraining approach, *arXiv preprint arXiv:1907.11692*.

Liu, Z., J. Li, Z. Shen, G. Huang, S. Yan, and C. Zhang (2017), Learning efficient convolutional networks through network slimming, in *Proceedings of the IEEE international conference on computer vision*, pp. 2736–2744.

Lundberg, S. M., and S.-I. Lee (2017), A unified approach to interpreting model predictions, *Advances in neural information processing systems*, *30*.

Marshall, I. J., J. Kuiper, and B. C. Wallace (2016), Robotreviewer: evaluation of a system for automatically assessing bias in clinical trials, *Journal of the American Medical Informatics Association*, *23*(1), 193–201.

Martins, A., and R. Astudillo (2016), From softmax to sparsemax: A sparse model of attention and multi-label classification, in *International conference on machine learning*, pp. 1614–1623, PMLR.

McClure, J. (2002), Goal-based explanations of actions and outcomes, *European review of social psychology*, *12*(1), 201–235.

Miller, T. (2019), Explanation in artificial intelligence: Insights from the social sciences, *Artificial intelligence*, *267*, 1–38.

Mullenbach, J., S. Wiegreffe, J. Duke, J. Sun, and J. Eisenstein (2018), Explainable prediction of medical codes from clinical text, *arXiv preprint arXiv:1802.05695*.

Nadeau, D., and S. Sekine (2007), A survey of named entity recognition and classification, *Lingvisticae Investigationes*, *30*(1), 3–26.

Nickerson, R. S. (1998), Confirmation bias: A ubiquitous phenomenon in many guises, *Review of general psychology*, *2*(2), 175–220.

Nie, A., E. D. Bennett, and N. D. Goodman (2019), Learning to explain: Answering why-questions via rephrasing, *arXiv preprint arXiv:1906.01243*.

Nonaka, I. (1994), A dynamic theory of organizational knowledge creation, *Organization science*, *5*(1), 14–37.

Nonaka, I., R. Toyama, and N. Konno (2000), Seci, ba and leadership: a unified model of dynamic knowledge creation, *Long range planning*, *33*(1), 5–34.

Peng, Y., S. Yan, and Z. Lu (2019), Transfer learning in biomedical natural language processing: an evaluation of bert and elmo on ten benchmarking datasets, *arXiv preprint arXiv:1906.05474*.

Peters, M. E., W. Ammar, C. Bhagavatula, and R. Power (2017), Semi-supervised sequence tagging with bidirectional language models, *arXiv preprint arXiv:1705.00108*.

Polanyi, M. (2009), The tacit dimension, in *Knowledge in organizations*, pp. 135–146, Routledge.

Qin, Y., Z. Wang, W. Zhou, J. Yan, Q. Ye, X. Ren, L. Neves, and Z. Liu (2020), Learning from explanations with neural module execution tree, in *International Conference on Learning Representations*.

Radford, A., J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever (2019), Language models are unsupervised multitask learners, *OpenAI Blog*, *1*(8), 9.

Rajani, N. F., B. McCann, C. Xiong, and R. Socher (2019), Explain yourself! leveraging language models for commonsense reasoning, *arXiv preprint arXiv:1906.02361*.

Ranney, M., and P. Thagard (1988), Explanatory coherence and belief revision in naive physics, *Tech. rep.*, PITTSBURGH UNIV PA LEARNING RESEARCH AND DEVELOPMENT CENTER.

Ratner, A., S. H. Bach, H. Ehrenberg, J. Fries, S. Wu, and C. Ré (2017), Snorkel: Rapid training data creation with weak supervision, in *Proceedings of the VLDB Endowment. International Conference on Very Large Data Bases*, vol. 11, p. 269, NIH Public Access.

Ratner, A., B. Hancock, J. Dunnmon, R. Goldman, and C. Ré (2018), Snorkel metal: Weak supervision for multi-task learning, in *Proceedings of the Second Workshop on Data Management for End-To-End Machine Learning*, pp. 1–4.

Ratner, A., S. H. Bach, H. Ehrenberg, J. Fries, S. Wu, and C. Ré (2020), Snorkel: Rapid training data creation with weak supervision, *The VLDB Journal*, *29*(2), 709–730.

Reyna, V. F. (2012), A new intuitionism: Meaning, memory, and development in fuzzy-trace theory., *Judgment and Decision making*.

Reyna, V. F., and C. J. Brainerd (1995), Fuzzy-trace theory: An interim synthesis, *Learning and individual Differences*, *7*(1), 1–75.

Ribeiro, M. T., S. Singh, and C. Guestrin (2016), " why should i trust you?" explaining the predictions of any classifier, in *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 1135–1144.

Rudin, C., B. Letham, and D. B. Madigan (2013), Learning theory analysis for association rules and sequential event prediction.

Safranchik, E., S. Luo, and S. Bach (2020), Weakly supervised sequence tagging from noisy rules, in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, pp. 5570–5578.

Sanyal, S., and X. Ren (2021), Discretized integrated gradients for explaining language models, *arXiv preprint arXiv:2108.13654*.

Schmidt, P., and F. Biessmann (2019), Quantifying interpretability and trust in machine learning systems, *arXiv preprint arXiv:1901.08558*.

Shang, J., L. Liu, X. Ren, X. Gu, T. Ren, and J. Han (2018), Learning named entity tagger using domain-specific dictionary, *arXiv preprint arXiv:1809.03599*.

Shrikumar, A., P. Greenside, A. Shcherbina, and A. Kundaje (2016), Not just a black box: Learning important features through propagating activation differences, *arXiv preprint arXiv:1605.01713*.

Shrikumar, A., P. Greenside, and A. Kundaje (2017), Learning important features through propagating activation differences, in *International conference on machine learning*, pp. 3145–3153, PMLR.

Simonyan, K., A. Vedaldi, and A. Zisserman (2013), Deep inside convolutional networks: Visualising image classification models and saliency maps, *arXiv preprint arXiv:1312.6034*.

Slack, D., S. Hilgard, E. Jia, S. Singh, and H. Lakkaraju (2020), Fooling lime and shap: Adversarial attacks on post hoc explanation methods, in *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, pp. 180–186.

Srivastava, S., I. Labutov, and T. Mitchell (2017), Joint concept learning and semantic parsing from natural language explanations, in *Proceedings of the 2017 conference on empirical methods in natural language processing*, pp. 1527–1536.

Strout, J., Y. Zhang, and R. J. Mooney (2019), Do human rationales improve machine explanations?, *arXiv preprint arXiv:1905.13714*.

Strumbelj, E., and I. Kononenko (2010), An efficient explanation of individual classifications using game theory, *The Journal of Machine Learning Research*, *11*, 1–18.

Sundararajan, M., A. Taly, and Q. Yan (2017), Axiomatic attribution for deep networks, in *International conference on machine learning*, pp. 3319–3328, PMLR.

Thagard, P. (1989), Explanatory coherence, *Behavioral and brain sciences*, *12*(3), 435–467.

Vaswani, A., N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin (2017), Attention is all you need, *Advances in neural information processing systems*, *30*.

Wang, S. I., S. Ginn, P. Liang, and C. D. Manning (2017), Naturalizing a programming language via interactive learning, *arXiv preprint arXiv:1704.06956*.

Wang, Z., Y. Qin, W. Zhou, J. Yan, Q. Ye, L. Neves, Z. Liu, and X. Ren (2019), Learning from explanations with neural execution tree, *arXiv preprint arXiv:1911.01352*.

Williams, A., N. Nangia, and S. R. Bowman (2017), A broad-coverage challenge corpus for sentence understanding through inference, *arXiv preprint arXiv:1704.05426*.

Xie, Q., X. Ma, Z. Dai, and E. Hovy (2017), An interpretable knowledge transfer model for knowledge base completion, *arXiv preprint arXiv:1704.05908*.

Xu, K., J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhudinov, R. Zemel, and Y. Bengio (2015), Show, attend and tell: Neural image caption generation with visual attention, in *International conference on machine learning*, pp. 2048–2057, PMLR.

Yang, C., A. Rangarajan, and S. Ranka (2018), Global model interpretation via recursive partitioning, in *2018 IEEE 20th International Conference on High Performance Computing and Communications; IEEE 16th International Conference on Smart City; IEEE 4th International Conference on Data Science and Systems (HPCC/SmartCity/DSS)*, pp. 1563–1570, IEEE.

Yang, Z., D. Yang, C. Dyer, X. He, A. Smola, and E. Hovy (2016), Hierarchical attention networks for document classification, in *Proceedings of the 2016 conference of the North American chapter of the association for computational linguistics: human language technologies*, pp. 1480–1489.

Yang, Z., Z. Dai, Y. Yang, J. Carbonell, R. R. Salakhutdinov, and Q. V. Le (2019), Xlnet: Generalized autoregressive pretraining for language understanding, *Advances in neural information processing systems*, *32*.

Zaidan, O., J. Eisner, and C. Piatko (2007), Using "annotator rationales" to improve machine learning for text categorization, in *Human language technologies 2007: The conference of the North American chapter of the association for computational linguistics; proceedings of the main conference*, pp. 260–267.

Zeiler, M. D., and R. Fergus (2014), Visualizing and understanding convolutional networks, in *European conference on computer vision*, pp. 818–833, Springer.

Zhang, R., Y. Yu, P. Shetty, L. Song, and C. Zhang (2022), Prboost: Prompt-based rule discovery and boosting for interactive weakly-supervised learning, *arXiv preprint arXiv:2203.09735*.

Zhang, Y., I. Marshall, and B. C. Wallace (2016), Rationale-augmented convolutional neural networks for text classification, in *Proceedings of the Conference on Empirical Methods in Natural Language Processing. Conference on Empirical Methods in Natural Language Processing*, vol. 2016, p. 795, NIH Public Access.

Zhang, Z., Y. Wu, H. Zhao, Z. Li, S. Zhang, X. Zhou, and X. Zhou (2019), Semantics-aware bert for language understanding, *arXiv preprint arXiv:1909.02209*.

Zhao, X., and V. Vydiswaran (2020), Lirex: Augmenting language inference with relevant explanation, *arXiv preprint arXiv:2012.09157*.

Zhao, X., H. Ding, and Z. Feng (2021), Glara: Graph-based labeling rule augmentation for weakly supervised named entity recognition, *arXiv preprint arXiv:2104.06230*.

Zhou, J., A. H. Gandomi, F. Chen, and A. Holzinger (2021), Evaluating the quality of machine learning explanations: A survey on methods and metrics, *Electronics*, *10*(5), 593.

Zhou, X., Y. Zhang, L. Cui, and D. Huang (2020), Evaluating commonsense in pre-trained language models., in *AAAI*, pp. 9733–9740.