Explanations for Natural Language: From Theory to Practice

Dheeraj Rajagopal

CMU-LTI-22-008
May 2022

Language Technologies Institute
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy
In Language and Information Technologies.

Keywords: explanations, interpretability, natural language processing, semantics

Abstract

Understanding the reasoning process through explanations is spontaneous, ubiquitous and fundamental to our sense of perceiving the world around us. Scientific progress often relies on explanations to facilitate discovery of hypotheses, identify applications and also identify systematic errors and correct them. An in-depth study of explanation thus help shed light on core cognitive issues, such as learning, induction and conceptual representation. Current NLP systems, despite significant advances, are usually treated as black boxes with little to no insight into **how** they reason. Learning with Explanations is an under-explored area in the natural language processing literature due to the lack of a unified theory.

In this work, we propose a theory towards how models can incorporate explanation. Our results in part of this show two promising approaches and corresponding instantiations as to how we can reliably incorporate explanations in NLP systems. We also show that explanations can not only help models in downstream tasks but could help humans improve upon a task.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

A central task in many of the sciences is to explain. A decision making process of any kind involves explaining self or others why and how the decision was arrived at. What are the basic aspects of explanations: facts, beliefs, morals ? An account of what an explanation is still largely debated among intellectuals without consensus. The challenges are largely amplified when AI systems are also involved in the decision making process. A system made decision without any accountability leads to many unforeseen consequences.

Large scale neural network NLP models have made significant progress in several NLP applications, yet they have been largely treated as blackbox [104] models. As their performances scale over time, it is essential to also understand whether these models learn the task or rely on spurious patterns. We summarize the contributions of this thesis as follows:

1.1 Contributions

1. **A Unified Theory:** Explanations in NLP systems have largely lacked a unifying theory. Although there have been several attempts, they lack a coherent connection as to what aspect of explanations they solve. Without a guiding principle, explanations have taken different interpretations across various NLP tasks. In this thesis, we propose the *recursive descent theory* — for practitioners for building NLP systems that are inherently able to explain their decisions.
2. **Data Based Explanations:** In our first instantiation of the recursive descent based explanations, we show how to realize explanations through data by enforcing specific structures. They could either be static structures (chapter 3) or dynamic structures (chapter 4). We propose a framework to decouple explanations from end-task, and show how such an approach would be beneficial.
3. **Model Based Explanations:** In the event of unavailability of external annotations for explanations, in chapter 6 we show that we can elicit explanations from the model by defining the explanation in terms of functions of input data. We observe that it is possible to achieve self-explaining systems without sacrificing end performance.

1.2 Thesis Overview

Thesis Statement : *Explanations are recursive, context dependent and it depends on various stakeholders of the system to define an acceptable explanation. Explanations can be reliably produced by both data and model based deep learning methods, and a resulting system produces decisions that are interpretable to humans.*

1. **Chapter 2 :** This chapter proposes a *recursive descent theory* for explanations, and discusses how to operationalize explanations in deep learning architectures for NLP. We also discuss the generalization goals for explanations and modes of explanation.
2. **Chapter 3 :** In this chapter, we show how to realize explanations through data [141]. We first decouple the explanations and the end-task, and show how explanations can improve end-task performance in the procedural text domain.
3. **Chapter 4 :** While chapter 3 discusses how to achieve explanations through data via a fixed structure, this work [109] introduces data-based explanations where the explanations can have arbitrary graph structure by leveraging large-scale pretrained models.
4. **Chapter 5 :** In comparison to chapters 3,4 where explanations help models improve upon an end-task, we show an instance where explanations can help humans improve on a hard task. We show that human annotators improve on the defeasible reasoning task using graph based explanations.
5. **Chapter 6 :** This chapter discusses a model-based approach for explanations, by eliciting interpretable output from systems that do not have explicitly annotated explanations. We propose the SELFEXPLAIN framework, that can architecturally augment both local and global interpretability to neural network models.
6. **Chapter 7 :** This chapter proposes a *recursive descent theory* for explanations, and discuss different use-cases in NLP applications.
7. **Chapter 8 :** In this chapter, we operationalize recursive descent via using templates. Towards this, we propose a prompt based template filling approach using text-to-text generation models. We present this approach on a cross domain reasoning task.
8. **Chapter 9 :** In this chapter, we conclude with the summary of overall contributions, limitations of the current work and directions for future work.

Chapter 2

Goals of Explanations

A system, like a person, that can't explain itself can't be trusted. Trustworthy NLP is fundamentally limited by our ability to construct models which can be audited. Despite wide interest, no general rigorous ontology has been proposed to define what an explanation *is* and *must* accomplish. First, we propose such a formal treatment of what an explanation *is* that unifies the existing NLP literature and points to next steps. Second, a model's explanations *must* be accessible to the stakeholders using and affected by the model. The secondary affect of our treatment is that it bridges scientific reasoning and machine learning to facilitate building more rigorous reasoning systems.

2.1 Introduction

Explanations offer us deeper insights towards the inner-workings of any system. Humans rely on explanations in many high-stakes situations in diverse scenarios such as law, medicine, engineering, politics and everyday life. Often, a good explanation helps advance learning, taking a step towards "comprehension". AI technologies are becoming increasingly prevalent — they are either used to replace humans or assist humans in complex decision making processes. Yet, a large gap exists in whether they can *explain why they predicted what they predicted.* Although there is widespread agreement that explanations are essential to build trust with humans who interact with NLP systems, there is little agreement on *what is an effective explanation?* and *how to model them for real world applications?*

In this work, we take a step in this direction by proposing a generalized framework towards how to design explanations. First, we establish the goals of an explanation depending on the people who are impacted by it – i.e. *stakeholders*. Next, we identify the factors through which such stakeholders can interact with the system to accomplish their goals for explanation. Taking inspiration from social sciences, we propose that the end goal of explanations in an NLP system should be modeled via their (i) *utility* and (ii) *simplicity*.

Whether an explanation is *accepted* is determined by a utility and a simplification function specified by the various stakeholders of the system.

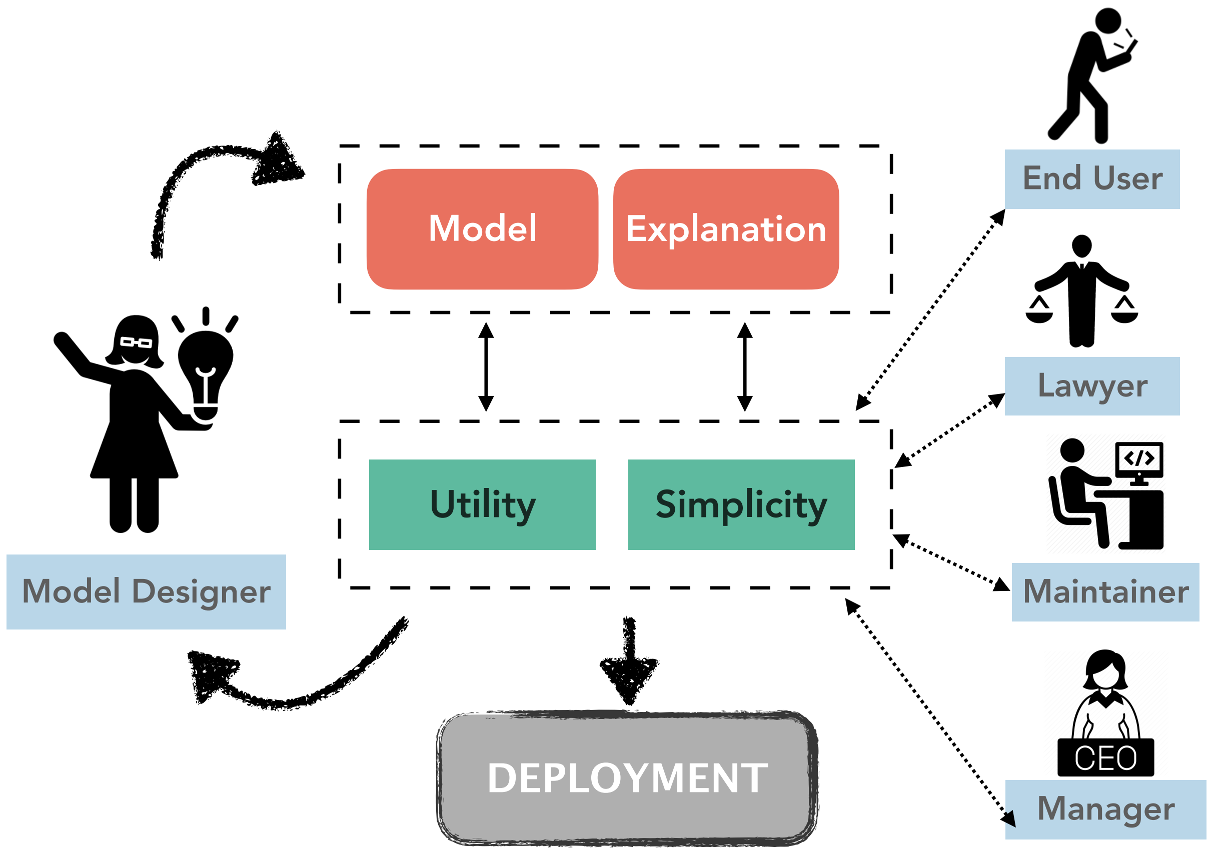


Figure 2.1: An overview of our proposed framework.

2.2 Explanations

To motivate our explanation framework, we first establish the basic definitions.

Definition 1 (System) *NLP software that aims to establish expertise in a task by explaining its predictions.*

Definition 2 (Explanation) *A model's justification for a decision it makes in service of a task.*

Definition 3 (Model) *The procedure that the system was trained on to enable decision making and explanation capacity for a given task. Typically, this system is a trained machine learning method.*

2.2.1 Explanation Stakeholders (S_E)

Even among humans, not all questions are equally good at probing the phenomenon in question to get at a good explanation [69]. Situations like this often raise two important questions:

What is a *good* explanation?
Who is the explanation for?

We argue that explanations are governed by social structures and individuals within these social structures. *Stakeholders* are a collection of humans that are related to the system in a way that the decision of the system affects them and how they relate to the system. For any explanation producing system, it is imperative that the stakeholders are defined clearly and how their decisions impact the explanation design. We define the primary stakeholders as having two primary roles:

Explanation Producer(s): are the architects of a system whose goal is to design a system capable of providing explanations alongside the decision for a given task.

Explanation Consumer(s): are the end-users of the system, who rely on the output of the system to make judgements for specific use-cases.

2.2.2 Utility ($U(t)$)

In economics, utility is a measure of the relative satisfaction from, or desirability of, consumption of various goods and services [171]. In the context of AI applications, optimizing for overall utility is often applied the problem of evaluating factors with respect to multiple performance criteria. Maximizing utility has been widely used in intelligent applications [10, 27, 52, 89].

We extend this idea for explanations, and define the end-goal of the explantion in terms of their utility. The choice of the utility function maybe task-specific, but identifying general utility factors across multiple tasks is a challenging and widely impactful reserach challenge. A good utility function should provide model the ability to collect adequate evidence for each production in the explanation and also be able to evaluate the efficacy of the explanation as a whole. In practice, we can employ a variety of approaches to collect evidences for each of the production from the explanation grammar. For NLP research, some candidates for identifying utility of a production are shown below:

Type	**Dataset**	$U(t)$	$P(t)$	**Research Questions**
Attention	Bahdanau et al. [7]	"Importance" values	Input overlay	1. Faithful? ($U(t)$) 2. Handle complex attentions ? ($P(t)$)
Chained Sentences from Corpus	Inoue et al. [77]; Jhamtani and Clark [84]; Xie et al. [198]	Reasoning Chains (depth > 1)	NL	1. Arbitrary depth ? ($U(t), S(t)$) 2. Multiple Decompositions for same sample ? ($S(t)$)
Connecting Sentence	Camburu et al. [24]; Rajani et al. [142]	Chains (depth=1)	NL	1. Beyond shallow ? ($U(t)$) 2. Repeat the answer in a sentence ? ($S(t)$)
Deductive Proofs	Clark et al. [32]; Saha et al. [150]	Proof structures	logical proofs	1. Generalizes beyond scientific domain ? ($U(t)$)
Qualitative Structures	Rajagopal et al. [141]	Qual. relations + sentences	NL	1. Generalizes beyond procedural domain ? ($U(t)$)
Rationales	DeYoung et al. [43]; Thorne et al. [184]; Yang et al. [199]	Supporting text in input	NL	1. Expl. outside input space ? ($U(t)$) 2. Can handle composition ? ($U(t), S(t)$)
Modular Networks	Andreas et al. [4]; Jiang and Bansal [86]; Gupta et al. [63]	Functional reasoning modules	functional primitives	1. Preserves performance ? ($U(t)$) 2. Can leverage pretraining ? ($U(t)$)

Table 2.1: Overview of explanation learning tasks proposed in the literature (and open research questions) seen through the lens of *recursive descent*. "Input overlay" means that the obtained values are then overlaid on to the input features to simplify the interpretation. "NL" simplification implies that the output is already in the natural language form (which we assume human understandable in general). Research Questions are shortened to fit.

1. **Retrieval from a Knowledge Source :** Every node or a node-edge-node pair can be augmented with a score w.r.t. to its validity based on an external knowledge base [33, 44, 173]. Such a score could be useful for knowledge-intensive explanations where grounding every production rule to a reliable source is essential.

2. **Via auxiliary tasks :** Another way of validating a production from the explanation grammar is using an auxiliary task. For instance, we can augment each proposition in the explanation with a set of Natural Language Inference (NLI) [110] statements. An acceptable explanation production should not only identify associated entailments, but also contradictions and irrelevant (neutral) statements.

We also outline some approaches to score the utility of the overall explanation in addition to scoring the productions individually:

1. **Downstream-Task Performance :** An explanation's utility can also be computed based on whether it a task that can benefit by incorporating the explanations downstream.

2. **Simulation :** If an end-task can be simulated using virtual environment, an explanation can be verified by checking whether they can be used to achieve a given goal in a simulated environment [94, 135]. Such approaches provide strong grounding(§7.3), but the verifiability is limited to the actions and objects in the environment.

3. **Approximate Proofs :** An explanation could also act as a approximate-proof [31, 150, 178], i.e., we should be able to trace a soft proof structure such that it is immediately evident if the model makes a mistake. Such an u formulation has an additional advantage — the ability of being used as feedback to the model itself, potentially leading towards *interactively explaining models*.

A utility function, in theory can employ a variety of evidence methods to support an explanation. But, we want to emphasize that a key aspect of an explanation producing system is that the output explanations should be **finite time verifiable**. An explanation should have the ability to be verified in finite time towards a bounded rational choice [169], for flexibility to work in real-world use-cases. In our grammar formulation, achieving grounded nodes and edges within finite recursive depth can be adapted to impose this time-constraint specification for interpretability as described in Doshi-Velez and Kim [46].

Type	**Approach**	$U(t)$	$P(t)$	**Research Questions**
Influence Functions	Han et al. [68]; Koh and Liang [92]	Influential Training Samples	NL	1. Works at high dimensions? ($S(t)$) 2. Local influences ? ($U(t)$)
Saliency Maps	Simonyan et al. [165]; Sundararajan et al. [175]; Smilkov et al. [167]	Gradients	Input Overlay	1. Gradient reflect importance ? ($U(t)$) 2. Can it be faithful ? ($U(t)$) 3. Additional utilities ? ($U(t)$)
Contextual Decomposition	Singh et al. [166]; Jin et al. [87]	Feature Clustering	Input Overlay	1. Beyond binary classification? ($U(t)$) 2. Additional utilities ? ($U(t)$)
Perturbations	Feng et al. [54]; Ebrahimi et al. [50] Ribeiro et al. [145]	Local Perturbations	Decision Boundary Projection	1. Beyond local ? ($U(t)$) 2. Distribution shifts ? ($U(t), S(t)$) 3. Robustness ? ($U(t)$)
Linguistic Probing	Hewitt and Liang [73]; Voita and Titov [188]	End-Task Probes	Predefined Task(s)	1. If no predefined end task? ($U(t)$) 2. Good probe acc. = competence ? ($U(t), S(t)$)

Table 2.2: Overview of inverse explanation learning tasks proposed in the literature (and open research questions) seen through the lens of *recursive descent*. “Input Overlay” means that the obtained values are then overlaid on to the input features to simplify the interpretation. “NL” simplification implies that the output is already in the natural language form (which we assume human understandable in general). Research Questions are shortened to fit.

2.2.3 Simplification ($P(t)$)

Edge Foundation posed the following question to a group of 191 intellectuals including academics, writers, artists, and intellectuals : *What is your favorite deep, elegant, or beautiful explanation?*

[1]. An overarching pattern consistent among most thinkers is that the most important factor for a good explanation is *simplicity*. Simplicity defines the ability to concisely abstract the fundamental properties of a system [9, 164]. A simplified explanation has the ability to be understood by the maximum amount of stakeholders. Jeffreys [83] argued that "the simpler laws have the greater prior probability". We extend this argument to explanations to say that simplified explanations have the higher prior probability of finding evidence in the real world. The theory of simplicity has also the advantage of being modeled as a machine learning problem [112].

Consider the example of neural network models. These models are often treated as black-box models [104]. They often represent features as continuous vectors, which are often inscrutable. In neural network models, a simplification function $P(t)$ would map the neural network representations in the vector space to human interpretable concepts expressed through natural language. A good explanation should *simplify the explanation* such that all the stakeholders can adequately ground each unit of the explanation. The following instance shows an example where the same explanation can be expressed in multiple ways.

Consider the following example. From top to bottom, the explanation's complexity increases, thereby only accounting for lesser and lesser users as a function of their depth in vocabulary.

Input : Does boiling a medical device with chemicals remove bacteria from it ?
System Output : Yes
Explanation 1: Chemicals *cleans* the medical device and heat kills bacteria
Explanation 2: Chemicals *disinfect* the medical device and heat kills bacteria
Explanation 3: Chemicals *sterilize* the medical device and heat kills bacteria
Explanation 4: Chemicals *depyrogenates* the medical device and heat kills bacteria

The formulation of the simplification function $P(t)$ enables us to tune the right level of complexity to explain depending on the requirements determined by the stakeholders. Additionally, the simplification function allows us to personalize explanations of a single sample to multiple end-users, potentially leading to tailor explanations for diverse set of users.

2.3 Conclusion

In this chapter, we establish some fundamental goals towards building explainable NLP systems. Our approach takes a holistic view of their stakeholders' utility and simplicity requirements. It helps establish how we should view explanations

[1]https://www.edge.org/inthenews/what-is-your-favorite-deep-elegant-or-beautiful-explanation

Part I

Explanations via Data

Chapter 3

Explanations via Data : Static Structure

In this chapter, we first explore data-based explanations and whether such explanations are useful for a downstream task. In this work, wse present QUARTET, a system that constructs such explanations from paragraphs, by modeling the explanation task as a multitask learning problem. QUARTET constructs explanations from the sentences in the procedural text, achieving ~ 18 points better on explanation accuracy compared to several strong baselines on a recent process comprehension benchmark. On an end task on this benchmark, we show a surprising finding that good explanations do not have to come at the expense of end task performance, in fact leading to a 7% F1 improvement over SOTA.

Utility : End-task performance
Simplification : Sentences in source document, and the corresponding qualitative relationships
Explanation Structure : A *fixed* production from the grammar where the nodes are sentences and the edges are qualitative relationships.

3.1 Introduction

Procedural text is common in natural language (in recipes, how-to guides, etc.) and finds many applications such as automatic execution of biology experiments [124], cooking recipes [17] and everyday activities [201]. However, the goal of procedural text understanding in these settings remains a major challenge and requires two key abilities, (i) understanding the dynamics of the world *inside* a procedure by tracking entities and what events happen as the narrative unfolds. (ii) understanding the dynamics of the world *outside* the procedure that can influence the procedure.

While recent systems for procedural text comprehension have focused on understanding the dynamics of the world *inside* the process, such as tracking entities and answering questions about what events happen, e.g., [18, 71, 181], the extent to which they understand the influences of *outside* events remains unclear. In particular, if a system fully understands a process, it should be able to predict what would happen if it was perturbed in some way due to an event from the *outside* world. Such counterfactual reasoning is particularly challenging because, rather than

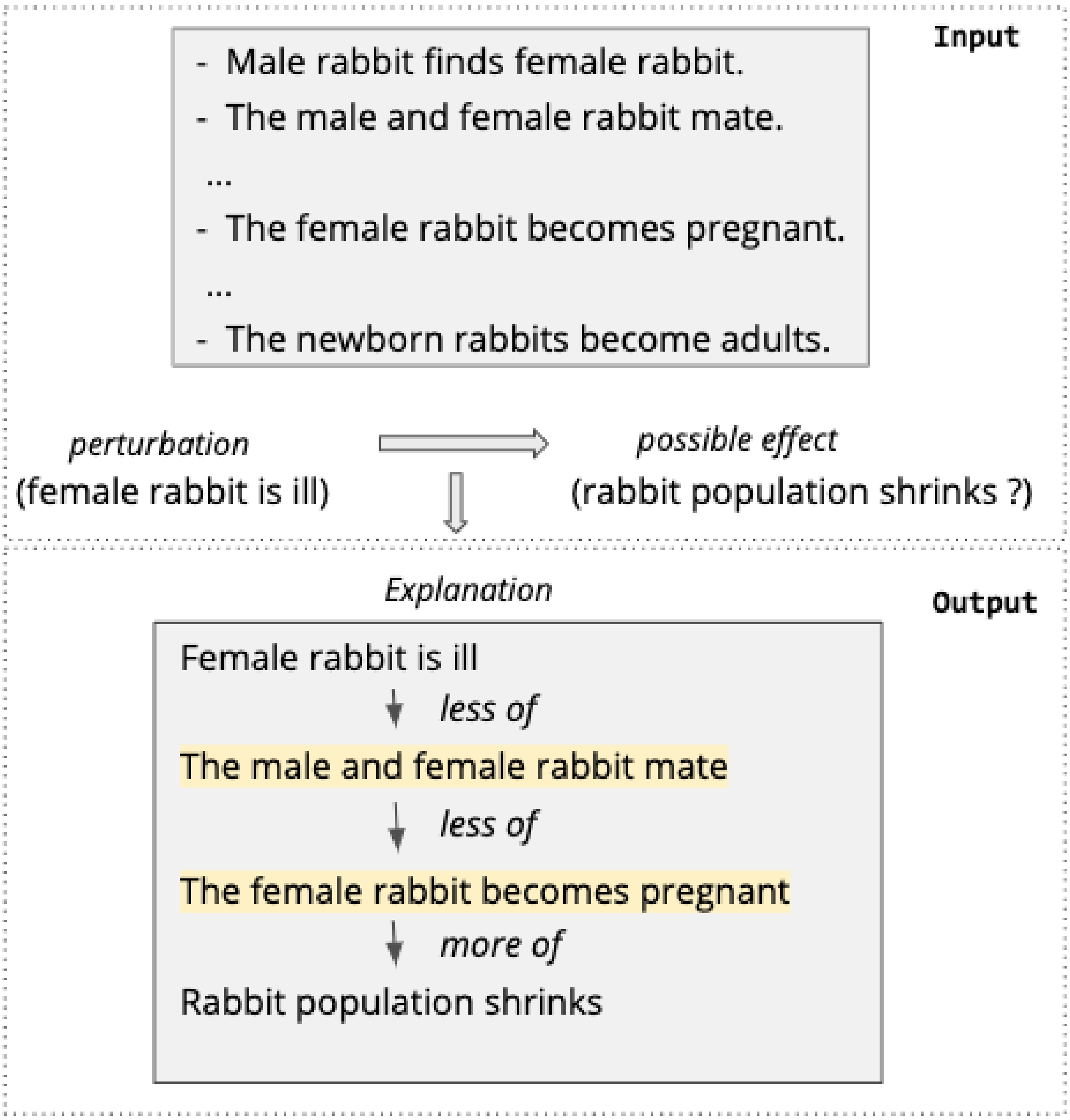

Input
- Male rabbit finds female rabbit.
- The male and female rabbit mate.
...
- The female rabbit becomes pregnant.
...
- The newborn rabbits become adults.
perturbation
(female rabbit is ill)
possible effect
(rabbit population shrinks ?)
Explanation
Output
Female rabbit is ill
less of
The male and female rabbit mate
less of
The female rabbit becomes pregnant
more of
Rabbit population shrinks

Figure 3.1: Given a procedural text, the task is to explain the effect of the perturbation using the input sentences.

asking what happened (described in text), it asks about what **would** happen in an alternative world where the change occurred.

Recently, Tandon et al. [183] introduced the WIQA dataset that contains such problems, requiring prediction of the effect of perturbations in a procedural text. They also presented several strong models on this task. However, it is unclear whether those high scores indicate that the models fully understand the described procedures, i.e., that the models have knowledge of the causal chain from perturbation to effect. To test this, Tandon et al. [183] also proposed an explanation task. While the general problem of synthesizing explanations is hard, they proposed a simplified version in which explanations were instead assembled from sentences in the input paragraph and qualitative indicators (more/less/unchanged). Although they introduced this explanation task and dataset, they did not present a model to address it. We fill this gap by proposing the first solution to this task.

We present a model, QUARTET (QUAlitative Reasoning wiTh ExplanaTions) that takes as input a passage and a perturbation, and its qualitative effect. The output contains the qualitative effect and an explanation structure over the passage. See Figure 3.1 for an example. The explanation structure includes up to two supporting sentences from the procedural text, together with the qualitative effect of the perturbation on the supporting sentences (*more of* or *less of* in Figure 3.1). QUARTET models this qualitative reasoning task as a multitask learning problem to explain the effect of a perturbation.

Our main contributions are:

- We present the first model that explains the effects of perturbations in procedural text. On a recent process comprehension benchmark, QUARTET generates better explanations compared to strong baselines.
- On an end task on this benchmark, we show a finding that good explanations do not have to come at the expense of end task performance, in fact leading to a 7% F1 improvement over SOTA. (refer §3.6). Prior work has found that optimizing for explanation can hurt end-task performance. Ours is a useful datapoint showing that good explanations do not have to come at the expense of end-task performance[1].

ears less protected → (MORE/+) sound enters the ear → (MORE/+) sound hits ear drum → (MORE/+) more sound detected
blood clotting disorder → (LESS/-) blood clots → (LESS/-) scab forms → (MORE/+) less scab formation
breathing exercise → (MORE/+) air enters lungs → (MORE/+) air enters windpipe → (MORE/+) oxygen enters bloodstream
squirrels store food → (MORE/+) squirrels eat more → (MORE/+) squirrels gain weight → (MORE/+) hard survival in winter
less trucks run → (LESS/-) trucks go to refineries → (LESS/-) trucks carry oil → (MORE/+) less fuel in gas stations
coal is expensive → (LESS/-) coal burns → (LESS/-) heat produced from coal → (LESS/-) electricity produced
legible address → (MORE/+) mailman reads address → (MORE/+) mail reaches destination → (MORE/+) on-time delivery
more water to roots → (MORE/+) root attract water → MORE/+) roots suck up water → (LESS/-) plants malnourished
in a quiet place → (LESS/-) sound enters the ear → (LESS/-) sound hits ear drum → (LESS/-) more sound detected
eagle hungry → (MORE/+) eagle swoops down → (MORE/+) eagle catches mouse → (MORE/+) eagle gets more food

Table 3.1: Examples of our model's predictions on the dev. set in the format: "$q_p \rightarrow d_i$ x_i $\rightarrow d_j$ x_j $\rightarrow d_e$ q_e". Supporting sentences x_i, x_j are compressed e.g., "the person has his ears less protected" → "ears less protected"

[1] All the code will be publicly shared upon acceptance

3.2 Related work

Procedural text understanding: Machine reading has seen tremendous progress. With machines reaching human performance in standard QA benchmarks [42, 143], more challenging datasets have been proposed [48] that require background knowledge, commonsense reasoning [180] and visual reasoning [5, 207]. In the context of procedural text understanding which has gained considerable amount of attention recently, [18, 38, 71] address the task of tracking entity states throughout the text. Recently, [183] introduced the WIQA task to predict the effect of *perturbations*.

Understanding the effects of perturbations, specifically, qualitative change, has been studied using formal frameworks in the qualitative reasoning community [57, 191] and counterfactual reasoning in the logic community [99]. The WIQA dataset situates this task in terms of natural language rather than formal reasoning, by treating the task as a mixture of reading comprehension and commonsense reasoning. However, existing models do not explain the effects of perturbations.

Explanations: Despite large-scale QA benchmarks, high scores do not necessarily reflect understanding [119]. Current models may not be robust or exploit annotation artifacts [65]. This makes explanations desirable for interpretation [155].

Attention based explanation has been successfully used in vision tasks such as object detection [131] because pixel information is explainable to humans. These and other token level attention models used in NLP tasks [193] do not provide full-sentence explanations of a model's decisions.

Recently, several datasets with natural language explanations have been introduced, e.g., in natural language inference [24], visual question answering [128], and multi-hop reading comprehension (HotpotQA dataset) [199]. In contrast to these datasets, we explain the effects of perturbations in procedural text. HotpotQA contains explanations based on two sentences from a Wikipedia paragraph. Models on the HotpotQA would not be directly applicable to our task and require substantial modification for the following reasons: (i) HotpotQA models are not trained to predict the qualitative structure (more or less of chosen explanation sentences in Figure 3.1). (ii) HotpotQA involves reasoning over named entities, whereas the current task focuses on common nouns and actions (models that work well on named entities need to be adapted to common nouns and actions [154]). (iii) explanation paragraphs in HotpotQA are not procedural while the current input is procedural in nature with a specific chronological structure.

Another line of work provides more structure and organization to explanations, e.g., using scene graphs in computer vision [59]. For elementary science questions, Jansen et al. [82] uses a science knowledge graph. These approaches rely on a knowledge structure or graph but knowledge graphs are incomplete and costly to construct for every domain [190]. There are trade-offs between unstructured and structured explanations. Unstructured explanations are available abundantly while structured explanations need to be constructed and hence are less scalable [24]. Generating free-form (unstructured) explanations is difficult to evaluate [36, 208], and adding qualitative structure over them is non-trivial. Taking a middle ground between free-form and knowledge graphs based explanations, we infer a qualitative structure over the sentences in the paragraph. This retains the rich interpretability and simpler evaluation of structured explanations as well as leverages the large-scale availability of sentences required for these explanation.

It is an open research problem whether requiring explanation helps or hurts the original task being explained. On the natural language inference task (e-SNLI), Camburu et al. [24] observed that models generate correct explanations *at the expense of* good performance. On the Cos-E task, recently Rajani et al. [142] showed that explanations help the end-task. Our work extends along this line in a new task setting that involves perturbations and enriches natural language explanations with qualitative structure.

3.3 Problem definition

We adopt the problem definition described in Tandon et al. [183], and summarize it here.

Input: 1. Procedural text with steps $x_1 \dots x_K$. Here, x_k denotes step k (i.e., a sentence) in a procedural text comprising K steps.
2. A perturbation q_p to the procedural text and its likely candidate effect q_e.

Output: An explanation structure that explains the effect of the perturbation q_p:

$$q_p \rightarrow d_i x_i \rightarrow d_j x_j \rightarrow d_e q_e$$

- i: step id for the first supporting sentence.
- j: step id for the second supporting sentence.
- $d_i \in \{+ - \bullet\}$: how step id i is affected.
- $d_j \in \{+ - \bullet\}$: how step id j is affected.
- $d_e \in \{+ - \bullet\}$: how q_e is affected.

See Figure 3.1 for an example of the task, and Table 3.1 for examples of explanations.

An explanation consists of up to two (i.e., zero, one or two) supporting sentences i, j along with their qualitative directions d_i, d_j. If there is only one supporting sentence, then $j = i$. If $d_e = \bullet$, then $i =$Ø, $j =$Ø (there is no valid explanation for no-effect).

While there can be potentially many correct explanation paths in a passage, the WIQA dataset consists of only one gold explanation considered best by human annotators. Our task is to predict that particular gold explanation.

Assumptions: In a procedural text, steps $x_1 \dots x_K$ are chronologically ordered and have a forward flowing effect i.e., if $j > i$ then more/increase of x_i will result in more/increase of x_j. Prior work on procedural text makes a similar assumption [38]. Note that this assumption does not hold for cyclic processes, and cyclic processes have already been flattened in WIQA dataset. We make the following observations based on this *forward-flow* assumption.

a1: $i <= j$ (*forward-flow* order)

a2: $d_j = d_i$ (*forward-flow* assumption) [2]

[2]Note that this does not assume all sentences have the same directionality of influence. For example, a paragraph could include both positive and negative influences: "Predators arrive. Thus the rabbit population falls...". Rather, the $d_j = d_i$ assumption is one of narrative coherence: the *more* predators arrive, the *more* the rabbit population falls. That is, within a paragraph, we assume enhancing one step will have enhanced effects (both positive or negative effects) on future steps - a property of a coherently authored paragraph.

a3: For the WIQA task, d_e is the answer label because it is the end node in the explanation structure.

a4: If $d_i = \bullet$ then answer label $= \bullet$ (since q_p does not affect q_e, there is no valid explanation.)

a5: $1 \leq i \leq K$; if $d_i = \bullet$, then i = Ø (see a4)

a6: $i \leq j \leq K$; if $d_e = \bullet$, then j = Ø (see a4)

This assumption reduces the number of predictions, removing d_j and answer label (see a2, a3). Given $x_1 \ldots x_K$, q_p, q_e the model must predict four labels: i, j, d_i, d_e .

3.4 QUARTET model

We can solve the problem as a classification task, predicting four labels: i, j, d_i, d_e. If these predictions are performed independently, it requires several independent classifications and this can cause error propagation: prediction errors that are made in the initial stages cannot be fixed and can propagate into larger errors later on [61].

To avoid this, QUARTET predicts and explains the effect of q_p as a multitask learning problem, where the representation layer is shared across different tasks. We apply the widely used parameter sharing approach, where a single representation layer is followed by task specific output layers [11]. This reduces the risk of overfitting to a single task and allows decisions on i, j, d_i, d_e to influence each other in the hidden layers of the network. We first describe our encoder and then the other layers on top, see Figure 3.2 for the model architecture.

Encoder: To encode $x_1 \ldots x_K$ and question q we use the BERT architecture [42] that has achieved state-of-the-art performance across several NLP tasks [30], where the question $q = q_p \oplus q_e$ ($\oplus$ stands for concatenation). We start with a byte-pair tokenization [156] of the concatenated passage and question ($x_1 \ldots x_K \oplus q$) . Let $[x_k]$ denote the byte-pair tokens of sentence x_k. The text is encoded as `[CLS]` $[x_1]$ $[unused1]$ `[SEP]` $[x_2]$ $[unused2]$ `[SEP]` `..` $[q]$ `[SEP]`. Here, `[CLS]` indicates a special classification token. `[SEP]` and $[unused1..K]$ are special next sentence prediction tokens.

These byte-pair tokens are passed through a 12-layered Transformer network, resulting in a contextualized representation for every byte-pair token. In this contextualized representation, the vector $\mathbf{u} = [\mathbf{u_1}, ...\mathbf{u_K}, \mathbf{u_q}]$ where $\mathbf{u_k}$ denotes the encoding for $[x_k]$, and $\mathbf{u_q}$ denotes question encoding. Let E^l be the embedding size resulting from l^{th} transformer layer. In that l^{th} layer, $[\mathbf{u_1}, ...\mathbf{u_K}] \in \mathbb{R}^{K*E^l}$. The hidden representation of all transformer layers are initialized with weights from a self-supervised pre-training phase, in line with contemporary research that uses pre-trained language models [42].

To compute the final logits, we add a linear layer over the different transformer layers in BERT that are individual winners for individual tasks in our multitask problem. For instance, out of the total 12 transformer layers, lower layers (layer 2) are the best predictors for $[i, j]$ while upper layers (layer 10 and 11) are the best performing predictors for $[d_i, d_e]$. Zhang et al. [208] found that the last layer is not necessarily the best performing layer. Different layers seem to learn complementary information because their fusion helps. Combining different layers by weighted averaging of the layers has been attempted with mixed success [30, 208]. We observed the same

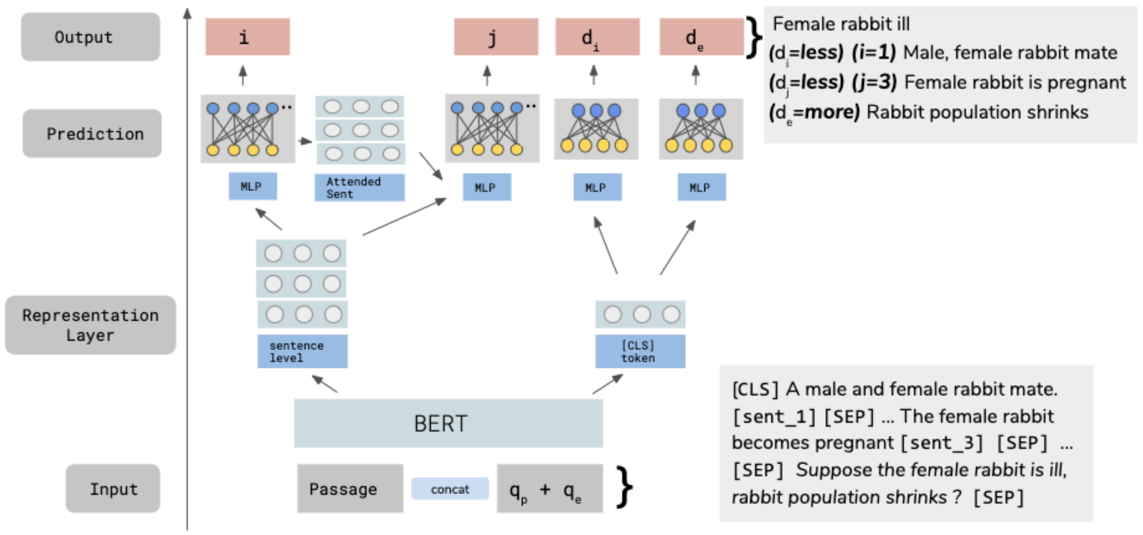

Output
Prediction
Representation Layer
Input
i
j
d_i
d_e
Female rabbit ill
(d_i=less) (i=1) Male, female rabbit mate
(d_j=less) (j=3) Female rabbit is pregnant
(d_e=more) Rabbit population shrinks
MLP
Attended Sent
MLP
MLP
MLP
sentence level
[CLS] token
BERT
Passage
concat
$q_p + q_e$
[CLS] A male and female rabbit mate. [sent_1] [SEP] ... The female rabbit becomes pregnant [sent_3] [SEP] ... [SEP] Suppose the female rabbit is ill, rabbit population shrinks ? [SEP]

Figure 3.2: QUARTET model. *Input*: Concatenated passage and question using standard BERT word-piece tokenization. *Representation Layer*: The input is encoded using BERT transformer. We obtain `[CLS]` and sentence level representations. *Prediction*: From the sentence level representation, we use an MLP to model the distributions for i and j (using attended sentence representation). From `[CLS]` representation, we use MLP for d_i (and d_j, since $d_i = d_j$) and d_e distributions. *Output*: Softmax to predict $\{i, j, d_i, d_j, d_e\}$

trend for simple weighted transformation. However, we found that learning a linear layer over concatenated features from winning layers improves performance. This is probably because there is very different information encoded in a particular dimension across different layers, and the concatenation preserves it better than simple weighted averaging.

Classification tasks: To predict the first supporting sentence x_i, we obtain a softmax distribution $s_i \in \mathbb{R}^K$ over $[\mathbf{u_1}, ...\mathbf{u_K}]$. From the *forward-flow* assumption made in the problem definition section earlier, we know that $i \leq j$, making it possible to model this as a span prediction $x_{i:j}$. Inline with standard span based prediction models [157], we use an attended sentence representation $(s_i \odot [\mathbf{u_1}, ...\mathbf{u_K}]) \oplus ([\mathbf{u_1}, ...\mathbf{u_K}]) \in \mathbb{R}^{K*2E^l}$ to predict a softmax distribution $s_j \in \mathbb{R}^K$ to obtain x_j. Here, $\odot$ denotes element-wise multiplication and $\oplus$ denotes concatenation.

For classification of d_i (and d_j, since $d_i = d_j$), we use the representation of the first token (i.e., `CLS` token $\in \mathbb{R}^{E^l}$) and a linear layer followed by softmax to predict $d_i \in \{+ - \bullet\}$. Classification of d_e is performed in exactly the same manner.

The network is trained end-to-end to minimize the sum of cross-entropy losses for the individual classification tasks i, j, d_i, d_e. At prediction time, we leverage assumptions (a4, a5, a6) to generate consistent predictions.

3.5 Experiments

Dataset: We train and evaluate QUARTET on the recently published WIQA dataset [3] comprising of 30,099 questions from 2107 paragraphs with explanations (23K train, 5K dev, 2.5K test). The perturbations q_p are either linguistic variation (17% examples) of a passage sentence (these are called in-para questions) or require commonsense reasoning to connect to a passage sentence (41% examples) (called, out-of-para questions). Explanations are supported by up to two sentences from the passage: 52.7% length 2, 5.5% length 1, 41.8% length 0. Length zero explanations indicate that $d_e =\bullet$ (called, no-effect questions), and ensure that random guessing on explanations gets low score on the end task.

Metrics: We evaluate on both explainability and the downstream end task (QA). For explainability, we define explanation accuracy as the average accuracy of the four components of the explanation: $acc_{expl} = \frac{1}{4} * \sum_{i \in \{i,j,d_i,d_e\}} acc(i)$ and $acc_{qa} = acc(d_e)$ (by assumption a3). The QA task is measured in terms of accuracy.

Hyperparameters: QUARTET fine-tunes BERT, allowing us to re-use the same hyperparameters as BERT with small adjustments in the recommended range [42]. We use the BERT-base-uncased version with a hidden size of 768. We use the standard adam optimizer with a learning rate 1e-05, weight decay 0.01, and dropout 0.2 across all the layers. All the models are trained on an NVIDIA V-100 GPU.

Models: We measure the performance of the following baselines (two non-neural and three neural).

- RANDOM: Randomly predicts one of the three labels $\{+ - \bullet\}$ to guess $[d_i, d_e]$. Supporting sentences i and j are picked randomly from $|avg_{sent}|$ sentences.
- MAJORITY: Predicts the most frequent label (*no effect i.e.* d_e=• in the case of WIQA dataset.)
- q_eONLY : Inspired by existing works [65], this baseline exploits annotation artifacts (if any) in the explanation dataset by retraining QUARTET using only q_e while hiding the permutation q_p in the question.
- HUMAN upper bound (Krippendorff's alpha inter-annotator values on $[i, j, d_i]$) on explainability reported in [183][4].
- TAGGING: We can reduce our task to a structured prediction task. An explanation i, j, d_i, d_e requires span prediction $x_{i:j}$ and labels on that span. So, for example, the explanation $i = 1, j = 2, d_i =+, d_j =-$ for input $x_1 \cdot x_5$ can be expressed as a tag sequence: `B-CORRECT E-OPPOSITE O O O`. Explanation $i = 2, j = 4, d_i =+$, $d_j =-$ would be expressed as: `O B-CORRECT I-CORRECT E-OPPOSITE O`. When $d_e = \bullet$, then the tag sequence will `O O O O O`. This BIEO tagging scheme has seven labels T = {`B-CORRECT, I-CORRECT, B-OPPOSITE, I-OPPOSITE, E-CORRECT, E-OPPOSITE, O`}.
  Formulating as a sequence tagging task allows us to use any standard sequence tagging model such as CRF as baseline. The decoder invalidates sequences that violate assumptions (a3 - a6).

[3]WIQA dataset link: `http://data.allenai.org/wiqa/`
[4]https://allenai.org/data/wiqa

To make the encoder strong and yet comparable to our model, we use exactly the same BERT encoder as QUARTET. For each sentence representation u_k, we predict a tag $\in T$. A CRF over these local predictions additionally provides global consistency. The model is trained end-to-end by minimizing the negative log likelihood from the CRF layer.

- BERT-NO-EXPL: State-of-the-art BERT model [183] that only predicts the final answer d_e, but cannot predict the explanation.
- BERT-W/-EXPL: A standard BERT based approach to the explanation task that predicts the explanation structure. This model minimizes only the cross-entropy loss of the final answer d_e, predicting an explanation that provides the best answer accuracy.
- DATAAUG: This baseline is adapted from Asai and Hajishirzi [6], where a RoBERTa model is augmented with symbolic knowledge and uses an additional consistency-based regularizer. Compared to our model, this approach uses a more robustly pre-trained BERT (RoBERTa) with data-augmentation optimized for QA Accuracy.
- QUARTET: our model described in §3.4 that optimizes for the best explanation structure.

3.5.1 Explanation accuracy

QUARTET is also the best model on explanation accuracy. Table 3.2 shows the performance on $[i, j, d_i, d_e]$. QUARTET also outperforms baselines on every component of the explanation. QUARTET performs better at predicting i than j. This trend correlates with human performance-picking on the second supporting sentence is harder because in a procedural text neighboring steps can have similar effects.

We found that the explanation dataset does not contain substantial annotation artifacts for the q_eONLY model to leverage (q_eONLY < MAJORITY)

Table 3.1 presents canonical examples of QUARTET dev predictions.

	acc_i	acc_j	acc_{d_i}	acc_{d_e}	acc_{expl}
RANDOM	12.50	12.50	33.33	33.33	22.91
q_eONLY	32.77	32.77	33.50	44.82	36.00
MAJORITY	41.80	41.80	41.80	41.80	41.80
TAGGING	42.26	37.03	56.74	58.34	48.59
BERT-W/-EXPL	38.66	38.66	69.20	75.06	55.40
QUARTET	**69.24**	**65.97**	**75.92**	**82.07**	**73.30**
HUMAN	75.90	66.10	88.20	96.30	81.63

Table 3.2: Accuracy of the explanation structure (i, j, d_i, d_e). Overall explanation accuracy is acc_{expl}. (Note that BERT-NO-EXPL and DATAAUG do not produce explanations).

We also tried a simple bag of words and embedding vector based alignment between q_p and x_i in order to pick the most similar x_i. These baselines perform worse than random, showing that aligning q_p and x_i involves commonsense reasoning that the these models cannot address.

3.6 Downstream Task

In this section, we investigate whether a good explanation structure leads to better end-task performance. QUARTET advocates explanations as a first class citizen from which an answer can be derived.

3.6.1 Accuracy on a QA task

We compare against the existing SOTA on WIQA no-explanation task. Table 3.3 shows that QUARTET improves over the previous SOTA BERT-NO-EXPL by 7%, achieving a new SOTA results. Both these models are trained on the same dataset[5]. The major difference between BERT-NO-EXPL and QUARTET is that BERT-NO-EXPL solves only the QA task, whereas QUARTET solves explanations, and the answer to the QA task is derived from the explanation. Multi-tasking (i.e., explaining the answer) provides the gains to QUARTET.

	QA accuracy
RANDOM	33.33
MAJORITY	41.80
q_eONLY	44.82
TAGGING	58.34
BERT-NO-EXPL	75.19
BERT-W/-EXPL	75.06
DATAAUG	78.50
QUARTET	**82.07**
HUMAN	96.30

Table 3.3: QUARTET improves accuracy on the QA (end task) by 7% points.

All the models get strong improvements over RANDOM and MAJORITY. The least performing model is TAGGING. The space of possible sequences of correct labels is large, and we believe that the current training data is sparse, so a larger training data might help. QUARTET avoids this sparsity problem because rather than a sequence it learns on four separate explanation components.

Table 3.4 presents the accuracy based on question types. QUARTET achieves large gains over BERT-NO-EXPL on the most challenging out-of-para questions. This suggests that QUARTET improves the alignment of q_p and x_i that involves some commonsense reasoning.

3.6.2 Correlation between QA and Explanation

QUARTET not only improves QA accuracy but also the explanation accuracy. We find that QA accuracy (acc_{d_e} in Table 3.2) is positively correlated (Pearson coeff. 0.98) with explanation

[5]We used the same code and parameters as provided by the authors of WIQA-BERT. The WIQA with-explanations dataset has about 20% fewer examples than WIQA without-explanations dataset [http://data.allenai.org/wiqa/] This is because the authors removed about 20% instances with incorrect explanations (e.g., where turkers didn't have an agreement). So we trained both QUARTET and WIQA-BERT on exactly the same vetted dataset. This helped to increase the score of WIQA-BERT by 1.5 points.

Model	in-para	out-of para	no-effect	overall
RANDOM	33.33	33.33	33.33	33.33
MAJORITY	00.00	00.00	100.0	41.80
q_eONLY	20.38	20.85	78.41	44.82
BERT-NO-EXPL	71.40	53.56	90.04	75.19
BERT-W/-EXPL	72.83	58.54	92.03	75.06
QUARTET	**73.49**	**65.65**	**95.30**	**82.07**

Table 3.4: QUARTET improves accuracy over SOTA BERT-NO-EXPL across question types.

accuracy (acc_{expl}). This shows that if a model is optimized for explanations, it leads to better performance on end-task. Thus, with this result we establish that (at least on our task) models can make better predictions when forced to generate a sensible *explanation structure*. An educational psychology study [49] hypothesizes that student performance improves when they are asked to explain while learning. However, their hypothesis is not conclusively validated due to lack of evidence. Results in Table 3.2 hint that, at least on our task, machines that learn to explain, ace the end task.

3.7 Error analysis

We analyze our model's errors (marked in red) over the dev set, and observe the following phenomena.

1. Multiple explanations: As mentioned in Section 3.3, more than one explanations can be correct. 22% of the incorrect explanations were reasonable, suggesting that overall explanation accuracy scores might under-estimate the explanation quality. The following example illustrates that while `gathering firewood` is appropriate when `fire is needed for survival`, one can argue that `going to wilderness` is less precise but possibly correct.

Gold: need fire for survival → `(MORE/+)` gather firewood → `(MORE/+)` build fire for warmth → `(MORE/+)` extensive camping trip
Pred: need fire for survival → `(MORE/+)` go to wilderness → `(MORE/+)` build fire for warmth → `(MORE/+)` extensive camping trip

2. i, j errors: Fig. 3.3 shows that predicted and gold distributions of i and j are similar. Here, sentence id $= -1$ indicates no effect. The model has learned from the data to never predict $j < i$ without any hard constraints.

The model is generally good at predicting i, j and in many cases when the model errs, the explanation seems plausible. Perhaps for the same underlying reason, human upper bound is not high on i (75.9%) and on j (66.1%). We show an example where i, j are incorrectly predicted (in red), but sound plausible.

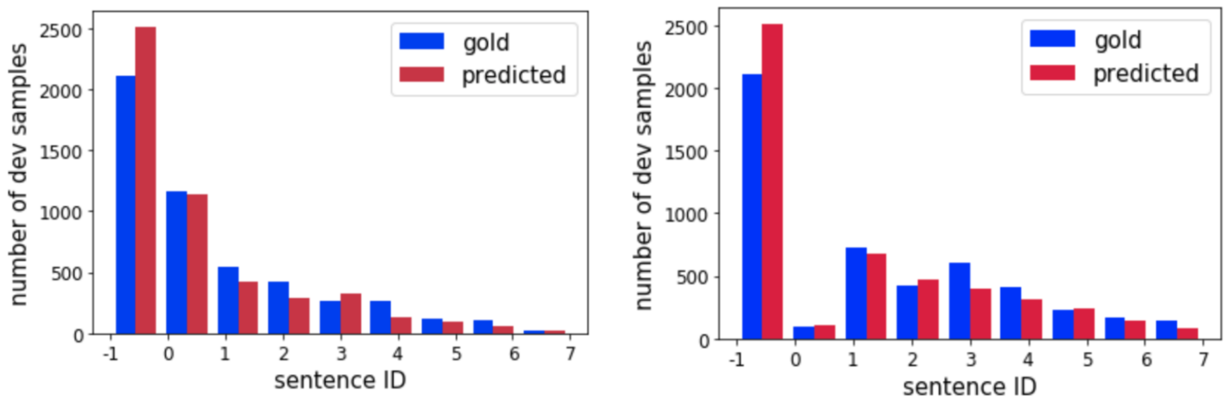


Figure 3.3: Gold vs. predicted distribution of i & j resp.

Gold: ear is not clogged by infection $\rightarrow$ `(OPP/-)` sound hits ear $\rightarrow$ `(OPP/-)` electrical impulse reaches brain $\rightarrow$ `(OPP/-)` more sound detected

Pred: ear is not clogged by infection $\rightarrow$ `(OPP/-)` sound hits ear $\rightarrow$ `(OPP/-)` drum converts sound to electrical impulse $\rightarrow$ `(OPP/-)` more sound detected

3. d_i, d_e errors: When the model incorrectly predicts d_i, a major source of error is when '$-$' is misclassified. 70% of the '$-$' mistakes, should have been classified as '$+$'. A similar trend is observed for d_e but the misclassification of '$-$ is less skewed. Table 3.5 shows the confusion matrix of d_i and of d_e in $\{ + - \bullet \}$.

	•	+	−
•	1972	91	47
+	295	883	358
−	226	492	639

	•	+	−
•	1972	89	49
+	261	909	295
−	252	346	830

Table 3.5: Confusion matrix for d_i (left) and d_e overall (right). (gold is on x-axis, predicted on y-axis.)

The following example shows an instance where '$-$' is misclassified as '$+$'. It implies that there is more scope for improvement here.

Gold: less seeds fall to the ground $\rightarrow$ `(OPP/-)` seed falls to the ground $\rightarrow$ `(OPP/-)` seeds germinate $\rightarrow$ `(MORE/+)` fewer plants

Pred: less seeds fall to the ground $\rightarrow$ `(OPP/-)` seed falls to the ground $\rightarrow$ `(OPP/-)` seeds germinate $\rightarrow$ `(OPP/-)` fewer plants

4. in-para vs. out-of-para: The model performs better on in-para questions (typically, linguistic variations) than out-of-para questions (typically, commonsense reasoning). Also see empirical evidence of this in Table 3.4.

The model is challenged by questions involving commonsense reasoning, especially to connect q_p with x_i in out-of-para questions. For example, in the following passage, the model incorrectly predicts • (no effect) because it fails to draw a connection between `sleep` and `noise`:

> Pack up your camping gear, food. Drive to your campsite. Set up your tent. Start a fire in the fire pit. Cook your food in the fire. Put the fire out when you are finished. Go to sleep. Wake up ...
>
> q_p: less noise from outside
> q_e: you will have more energy

Analogous to i and j, the model also makes more errors between labels '+' and '−' in out-of-para questions compared to in-para questions (39.4% vs 29.7%) – see Table 3.6.

	•	+	−
+	29	295	78
−	49	130	259

	•	+	−
+	266	588	280
−	177	362	380

Table 3.6: Confusion matrix d_i for in-para & out-of-para

[183] discuss that some in-para questions may involve commonsense reasoning similar to out-of-para questions. The following is an example of an in-para question where the model fails to predict d_i correctly because it cannot find the connection between `protected ears` and `amount of sound entering`.

> Gold: ears less protected → `(MORE/+)` sound enters ear → `(MORE/+)` sound hits ear drum → `(MORE/+)` more sound detected
> Pred: ears less protected → `(OPP/-)` sound enters the ear → `(OPP/-)` sound hits ear drum → `(MORE/+)` more sound detected

5. Injecting background knowledge: To study whether additional background knowledge can improve the model, we revisit the out-of-para question that the model failed on. The model fails to draw a connection between `sleep` and `noise`, leading to an incorrect (no effect) '•' prediction.

By adding the following relevant background knowledge sentence to the paragraph "`sleep requires quietness and less noise`", the model was able to correctly change probability mass from d_e = '•' to '+'. This shows that providing commonsense through Web paragraphs and sentences is a useful direction.

> Pack up your camping gear, food ... ***Sleeping requires quietness and less noise.*** Go to sleep. Wake up ...
>
> q_p: less noise from outside
> q_e: you will have more energy

3.8 Assumptions and Generality

QUARTET makes two simplifying assumptions: (1) explanations are assembled from the provided sentences (question + context), rather than generated, and (2) explanations are chains of qualitative, causal influences, describing how an end-state is influenced by a perturbation. Although these (helpfully) bound this work, the scope of our solution is still quite general: Assumption (1) is a common approach in other work on multihop explanation (e.g., HotpotQA), where authoritative sentences support an answer. In our case, we are the first to apply the same idea to chains of influences. Assumption (2) bounds QUARTET to explaining the effects of qualitative, causal influences. However, this still covers a large class of problems, given the importance of causal and qualitative reasoning in AI. The WIQA dataset provides the first large-scale dataset that exemplifies this class: given a qualitative influence, assemble a causal chain of events leading to a qualitative outcome. Thus QUARTET offers a general solution within this class, as well as a specific demonstration on a particular dataset.

3.9 Conclusion

Explaining the effects of a perturbation is critical, and we have presented the first system that can do this reliably. QUARTET not only predicts meaningful explanations, but also achieves a new state-of-the-art on the end-task itself, leading to an interesting finding that models can make better predictions when forced to explain. Our work opens up new directions for future research: 1) Can additional background context from the Web improve explainable reasoning? 2) Can such structured explanations be applied to other NLP tasks? We look forward to future progress in this area.

Chapter 4

Explanations via Data : Dynamic Structure

In this chapter, we explore the idea of dynamically expanding explanation structure. In this chapter, we propose a method to iteratively build an explanation graph of relevant consequences explicitly in a structured situational graph (*st* graph) using natural language queries over a finetuned language model ($\mathcal{M}$). Across multiple domains, CURIE generates *st* graphs that humans find relevant and meaningful in eliciting the consequences of a new situation. We show that *st* graphs generated by CURIE improve a situational reasoning end task (WIQA-QA) by 3 points on accuracy by simply augmenting their input with our generated situational graphs, especially for a hard subset that requires background knowledge and multi-hop reasoning.

Utility : End task performance on a downstream task - supervised and zero-shot
Translation : Situational explanations through natural language concepts
Explanation Structure : Dynamic recursive tree, ability to expand at any node

4.1 Introduction

A long-standing challenge in reasoning is to model the consequences of a novel situation in a context. Consider these questions - *Would it rain more if we plant more trees?*, or *What would help water to boil faster?* - answering these questions requires comprehending the complex events such as *plant growth* and *water boiling*, where much of the information remains implicit (by Grice's maxim of quantity [62]), thus requiring inference.

Tasks that require situational reasoning are increasingly observed by machines deployed in the real world - unexpected situations are common, and machines are expected to gracefully handle them. It is also essential for tasks such as qualitative reasoning [176, 182], physical commonsense reasoning [15, 152], and defeasible inference [148]. Unlike humans, machines are not adept at such reasoning.

Prior systems that address situational reasoning take as input a context providing background information, a situation (*st*), *and* an ending, and predict the reachability from *st* to that ending either in a classification setting (e.g., Tandon et al. [182] grounds the path on at most two sentences in the context) or recently, in a story-generation setting [136], where the goal is to generate an

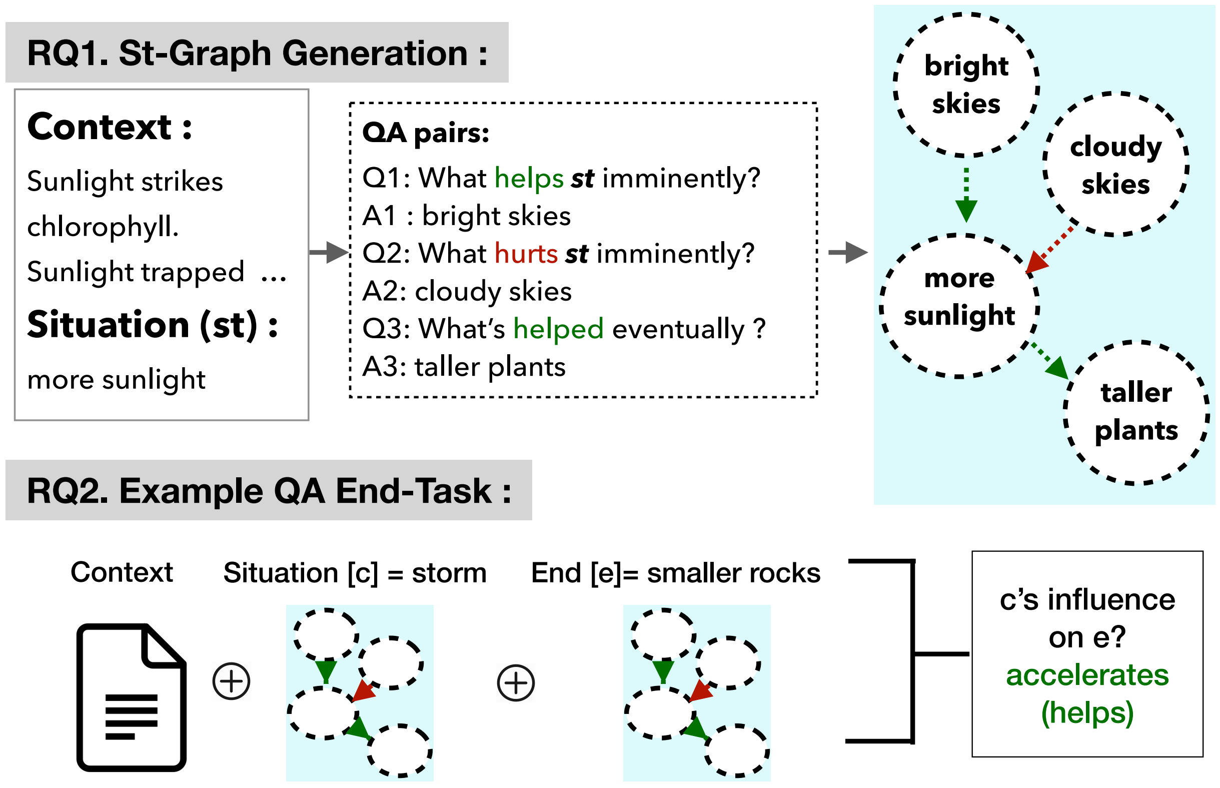


Figure 4.1: RQ1: CURIE generates situational graphs through iterative queries to a model, making the model's knowledge of influences explicit (above; positive, and negative influence) iteratively. RQ2: Such graphs can improve situational reasoning QA when added to the QA input (below, where the context is a passage about erosion).

alternate ending when the original ending and a counterfactual situation are given. However, generating effects of situations in real-world scenarios, where the ending is typically unknown is still an open challenge. We also might need *st*-reasoning capabilities across multiple domains (beyond stories).

Further, multiple types of consequences to a situation might have to be generated (e.g, positive and negative impacts or eventual and immediate impacts), which requires outputs in a structured form.

To address these limitations, we propose CURIE- a generation framework that generalizes multiple reasoning tasks under a general situational reasoning framework.

The task is illustrated in Figure 4.1: given some context and just a situation *st* (short phrase), our framework generates a *situational reasoning graph* (*st*-graph). At its core, CURIE constructs a reasoning graph based on the contextual knowledge that supports the following kinds of reasoning:

1. If *st* occurs, what will happen imminently/ eventually?
2. If *st* occurs, which imminent/ eventual effect will not happen?
3. What will support/ prevent the *st*?

As shown in Figure 4.1, our approach to this task is to iteratively compile the answers to

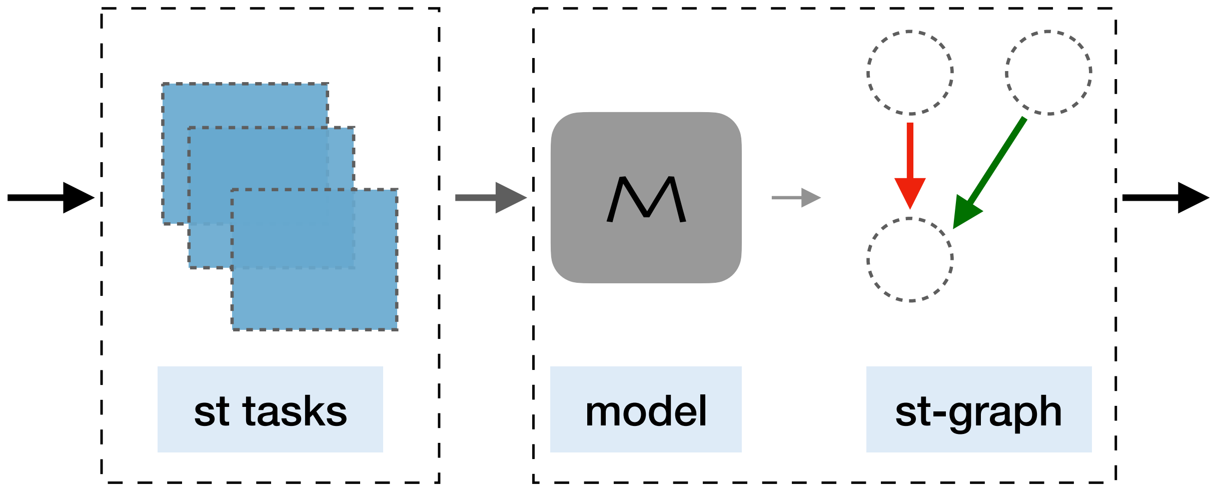


Figure 4.2: CURIE framework consists of two components: (i) a formulation that adapts datasets that allow *st*-reasoning for pretraining (ii) a method to iteratively build structured *st*-graphs using natural language queries over a fine-tuned language model ($\mathcal{M}$).

questions 1,2,3 to construct the *st*-graph. Compared to a free-form text output obtained from an out-of-the-box seq-to-seq model, our approach gives more control and flexibility over the graph generation process, including arbitrarily reasoning for any particular node in the graph. Downstream tasks that require reasoning about situations can compose natural language queries to construct a *st*-reasoning graph that can be simply augmented to their input. In this chapter, we ask the following two research questions:

RQ1 Given a specific context and situation, can we iteratively generate a situational reasoning graph of potential effects?

RQ2 Can the *st*-graphs generated by CURIE improve performance at a downstream task?

In response, we make the following contributions:

(i.) We present CURIE, the first domain-agnostic situational reasoning framework that takes as input some context and an *st* and iteratively generates a situational reasoning graph (§4.2). We show that our framework is effective at situational reasoning across three datasets, as validated by human evaluation and automated metrics.

(ii.) We show that *st* graphs generated by CURIE improve a *st*-reasoning task (WIQA-QA) by 3 points on accuracy by simply augmenting their input with our generated situational graphs, especially for a hard subset that requires background knowledge and multi-hop reasoning (§4.4). (Table 4.2).

4.2 CURIE for Situational Reasoning

CURIE provides both a general framework for situational reasoning and a method for constructing *st*-reasoning graphs from pretrained language models. The overall architecture of CURIE is shown

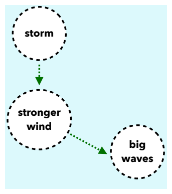
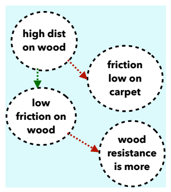
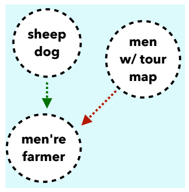
Dataset	Original formulation	*st* formulation	*st* graph
WIQA	*context*: Wind creates waves.. Waves wash on beaches... *ques*: If there is storm, how will it affect bigger waves? *chain*: storm → stronger wind → bigger waves *answer*: bigger waves	Given *context* and *st*: there is a storm Q1: What does *st* help *imminently* ? A1: stronger wind Q2: What does *st* help *eventually* ? A2: bigger waves	
QUAREL	*context*: Car rolls further on wood than on thick carpet *ques*: what has more resistance? (a) wood (b) the carpet *simplified logical form of context, ques*: distance is higher on wood → (a) friction is higher in carpet (or) (b) friction is higher in wood *answer*: (b) the carpet	Given *context* and *st*: distance is higher on wood Q1: What does *st* entail *imminently* ? A1: friction is lower in wood Q2: What does *st* contradict *imminently* ? A2: friction is lower in carpet Q3: What does *st* entail *eventually* ? A3: wood has more resistance	 high dist on wood friction low on carpet low friction on wood wood resistance is more
DEFEAS	*context*: Two men and a dog are standing among the green hills. *hypothesis*: The men are farmers. *evidence type*: strengthener *answer*: the dog is a sheep dog	Given *context* and *st*: dog is a sheep dog Q1: What does *st* strengthen *imminently* ? A1: The men are farmers *st*: men are studying tour maps Q2: What does *st* weaken *imminently*? A2: The men are farmers	

Table 4.1: The datasets used by CURIE and how we re-purpose them for *st* reasoning graph generation task. As explained in §4.2.1, the green edges set depicts relation (r) (entail, strengthen, helps) and red edges depict one of (contradict, weaken, hurts). The { *imminent*, *eventual* } effects (c) are used to support multihop reasoning. DEFEAS = DEFEASIBLE, *chain* refers to reasoning chain. Some examples are cut to fit. The key insight is that an *st*-graph can be decomposed into a series of QA pairs, enabling us to leverage seq-to-seq approaches for *st*-reasoning.

in Figure 4.2. CURIE framework consists of two components: (i) *st-reasoning task formulation :* a formulation that adapts datasets that allow situational reasoning (ii) *st-graph construction :* a method to fine-tune language model $\mathcal{M}$ to generate the consequences of a situation and iteratively construct structured situational graphs (shown in figure 4.1). In this section, we present (i) our task formulation (§4.2.1), (ii) adapting existing datasets for CURIE task formulation (§4.2.2), (iii) the learning procedure (§4.2.3), and (iv) the *st*-graph generation via inference (§4.2.4).

4.2.1 Task Formulation

We describe the general task formulation for adapting pretraining language models to the *st*-reasoning task. Given a context $T = \{s_1, s_2, \ldots, s_N\}$ with N sentences, and a situation *st*, our goal is to generate an *st*-graph G in this changed world.

An *st*-graph $G(V, E)$ is an unweighted directed acyclic graph. A vertex $v \in V$ is an event or a state such that it describes a change to the original conditions in T. Each edge $e_{ij} \in E$ is labeled with an relationship r_{ij}, that indicates whether v_i *positively* or *negatively* influences v_j. Positive influences are represented via green edges comprising one of $\{$*entails, strengthens, helps*$\}$ and negative influences represented via red edges that depict one of $\{$*contradicts, weakens, hurts*$\}$. Our relation set is general and can accommodate various *st*-reasoning tasks. Given two nodes $v_i, v_k \in V$, if a path from v_i to v_k has more than one edge, we describe the effect c as *eventual* and a direct effect as *imminent*.

We obtain the training data for st-graph generation task by decomposing an *st*-graph into a set of question-answer pairs. Each question comprises of the context T, a *st*-vertex v_s, a relation r, and the nature of the effect c. The output is an answer to the question, that corresponds to the target node v_t. An example is shown in Figure 4.1. Compared to an end-to-end approach to graph generation, our approach gives more flexibility over the generation process, enabling reasoning for any chosen node in the graph.

4.2.2 Generalizing Existing Datasets

Despite theoretical advances, lack of large-scale general situational reasoning datasets presents a challenge to train seq-to-seq language models. In this section, we describe how we generalize existing diverse datasets towards *st*-reasoning towards finetuning a language model $\mathcal{M}$. If a reasoning task allows a context, a *st*-situation and can describe the influence of *st* in terms of green and/or red edges, it can be seamlessly adapted to CURIE framework. Due to lack of existing datasets that directly support our task formulation, adapt the following three diverse datasets - WIQA, QUAREL and DEFEASIBLE for CURIE.

WIQA: WIQA task studies the effect of a perturbation in a procedural text [182]. The context T in WIQA is a procedural text describing a physical process, and *st* is a perturbation i.e., an external situation deviating from T, and the effect of *st* is either helps or hurts. An example of WIQA to *st*-formulation is shown in Table 4.1.

QUAREL: QUAREL dataset [176] contains qualitative story questions where T is a narrative, and the *st* is a qualitative statement. T and *st* are also expressed in a simpler, logical form, which we make use of because it clearly highlights the reasoning challenge. The effect of *st* is either entails or contradicts (example in Table 4.1).

DEFEASIBLE: The DEFEASIBLE reasoning task [148] studies inference in the presence of a counterfactual. The context T is given by a premise which describes a everyday context, and the *st* is an observed evidence which either strengthens or weakens the hypothesis. We adapt the original

Research question	**Training dataset**	**Test dataset**	**Task**	**Metrics**
Can we generate good *st* graphs? (§4.3)	WIQA-*st*	WIQA-*st*	generation	ROUGE, BLEU
	QUAREL-*st*	QUAREL-*st*	generation	ROUGE, BLEU
	DEFEASIBLE-*st*	DEFEASIBLE-*st*	generation	ROUGE, BLEU
Can we improve downstream tasks? (§4.4.1, §4.4.2)	WIQA-*st*, WIQA-QA	WIQA-QA	finetuned QA	accuracy
	QUAREL-*st*	QUARTZ-QA	zero shot	accuracy

Table 4.2: Overview of experiments

Dataset	train	dev	test
WIQA	119.2k	34.8k	34.8k
QUAREL	4.6k	1.3k	652
DEFEASIBLE	200k	14.9k	15.4k

Table 4.3: Dataset wise statistics, we maintain the splits

abductive setup as shown in Table 4.1. In addition to commonsense situations, DEFEASIBLE-*st* also comprises of social situations, thereby contributing to the diversity of our datasets.

4.2.3 Learning to Generate *st*-graphs

To reiterate our task formulation (§4.2.1), for a given context and *st*, we first specify a set of questions and the resulting output for the questions is then compiled to form a *st*-graph.

The training data thus consists of tuples $(\mathbf{x}_i, \mathbf{y}_i)$, with $\mathbf{x}_i = (T, st\ , r, c)_i$ where T denotes the context, *st* the situation, r denotes the edge (green or red), c signifies the nature of the effect (imminent or eventual), and $\mathbf{y}_i$ is the output (a short sentence or a phrase depicting the effect). The output of N_Q such questions is compiled into a graph $G = \{\mathbf{y_i}\}_{1:N_Q}$ (as shown in Figure 4.1).

We use a pretrained language model $\mathcal{M}$ to estimate the probability of generating an answer $\mathbf{y}_i$ for an input $\mathbf{x}_i$. We first transform the tuple $\mathbf{x}_i = \langle x_i^1, x_i^2, \ldots, x_i^N \rangle$ into a single query sequence of tokens by concatenating its components i.e. we set $\mathbf{x}_i = \texttt{concat}(T, st\ , r, c)$, where `concat` refers to string concatenation. Let the sequence of tokens representing the target event be $\mathbf{y}_i = \langle y_i^1, y_i^2, \ldots, y_i^M \rangle$, where N and M are the lengths of the query and the target event sequences We model the conditional probability $p_\theta(\mathbf{y}_i \mid \mathbf{x}_i)$ as a series of conditional next token distributions parameterized by θ: as $p_\theta(\mathbf{y}_i \mid \mathbf{x}_i) = \prod_{k=1}^{M} p_\theta(y_i^k \mid \mathbf{x}_i, y_i^1, .., y_i^{k-1})$.

4.2.4 Inference to Decode *st*-graphs

The auto-regressive factorization of the language model p_θ allows us to efficiently generate target event influences for a given test input $\mathbf{x}_j$. The process of decoding begins by sampling the first token $y_j^1 \sim p_\theta(y \mid \mathbf{x}_j)$. The next token is then drawn by sampling $y_j^2 \sim p_\theta(y \mid \mathbf{x}_j, y_j^1)$. The process is repeated until a specified *end-symbol* token is drawn at the K^{th} step. We use nucleus

Algorithm 1 ITERATIVEGRAPHGEN (IGEN): generating *st* graphs with CURIE

```
Given: CURIE language model $\mathcal{M}$.
Given: Context passage $T$, a situation st, a set $R = \{(r_i, c_i)\}_{i=1}^{N_Q}$ made of $N_Q$ $(r, c)$ tuples.
Result: st graph $G$ where the $i^{th}$ node will be generated with the relation $r_i$ and the effect type $c_i$.
Init: $G \leftarrow \emptyset$
for $i \leftarrow 1, 2, \ldots, N_Q$ do
    // Create a query
    $\mathbf{x}_i = \text{concat}(T, st, r_i, c_i)$ /* Sample a node from the language model $\mathcal{M}$ */
    $\mathbf{y}_i \sim \mathcal{M}(\mathbf{x}_i)$ /* Add the sampled node and the edge to the graph */
    $G = G \cup (r_i, c_i, \mathbf{y}_i)$
end
return $G$
```

sampling [76] in practice. The tokens $\langle y_j^1, y_j^2, \ldots, y_j^{K-1} \rangle$ are then returned as the generated answer. To generate the final *st*-reasoning graph G, we combine all the generated answers $\{\mathbf{y}_i\}_{1:N_Q}$ that had the same context and *st* pair (T, st) over all (r, c) combinations. We can then use generated answer $st' \in \{\mathbf{y}_i\}_{1:N_Q}$, as a new input to $\mathcal{M}$ as (T, st') to recursively expand the *st*-graph to arbitrary depth and structures (Algorithm 1). One such instance of using CURIE *st* graphs for a downstream QA task is shown in §4.4.

4.3 RQ1: Establishing Baselines for *st*-graph Generation

This section reports on the quality of the generated *st* reasoning graphs and establishes strong baseline scores for *st*-graph generation. We use the datasets described in section §4.2.2 for our experiments.

4.3.1 Baseline Language Models

To reiterate, CURIE is composed of (i) task formulation component and (ii) graph construction component, that uses a language model $\mathcal{M}$ to construct the *st*-graph. We want to emphasize that any language model architecture can be a candidate for $\mathcal{M}$. Since our *st*-task formulation is novel, we establish strong baselines for the choice of language model. Our experiments include large-scale language models (LSTM and pretrained transformer) with varying parameter size and pre-training, along with corresponding ablation studies. Our $\mathcal{M}$ choices are as follows:

LSTM Seq-to-Seq: We train an LSTM [74] based sequence to sequence model [8] which uses global attention described in [106]. We initialize the embedding layer with pre-trained 300 dimensional Glove [129][1]. We use 2 layers of LSTM encoder and decoder with a hidden size of 500. The encoder is bidirectional.

[1]https://github.com/OpenNMT/OpenNMT-py

Model ($\mathcal{M}$)	BLEU	ROUGE
WIQA-*st*		
LSTM Seq-to-Seq	7.51	18.71
GPT $\sim$(w/o T)	7.82	19.30
GPT-2 $\sim$(w/o T)	10.01	20.93
GPT	9.95	19.64
GPT-2	**16.23**	**29.65**
QUAREL-*st*		
LSTM Seq-to-Seq	13.05	24.76
GPT $\sim$(w/o T)	20.20	36.64
GPT-2 $\sim$(w/o T)	26.98	41.14
GPT	25.48	42.87
GPT-2	**35.20**	**50.57**
DEFEASIBLE-*st*		
LSTM Seq-to-Seq	7.84	17.50
GPT $\sim$(w/o T)	9.91	20.63
GPT-2 $\sim$(w/o T)	9.17	9.43
GPT	10.49	**21.79**
GPT-2	**10.52**	21.19

Table 4.4: Generation results for CURIE with baselines for language model $\mathcal{M}$. We find that context is essential for performance (w/o T). We provide these baseline scores as a reference for future research.

GPT: We use the original design of GPT [137] with 12 layers, 768-dimensional hidden states, and 12 attention heads.

GPT-2: We use the medium (355M) variant of GPT-2 [138] with 24 layers, 1024 hidden size, 16 attention heads.

For both GPT and GPT-2, we initialize the model with the pre-trained weights and use the implementation provided by Wolf et al. [196].

4.3.2 Automated Evaluation

To evaluate our generated *st*-graphs, we compare them with the gold-standard reference graphs. To compare the two graphs, we first flatten both the reference graph and the *st*-graph as text sequences and then compute the overlap between them. We use the standard evaluation metrics BLEU [127], and ROUGE [102] [2]. Our results indicate that the task of *st* generation is challenging, and suggests that incorporating *st*-reasoning specific inductive biases might be beneficial. At the same time, Table 4.4 shows that even strong models like GPT-2 struggle on the *st*-graph generation task, leaving a lot of room for model improvements in the future.

We also show ablation results for the model with respect to the context T (§4.2.1), by fine-tuning without the context. We find that context is essential for performance for both GPT and GPT-2 (indicated with w/o T in Table 4.4). Further, we note that the gains achieved by adding context are higher for GPT-2, hinting that larger models can more effectively utilize the context.

4.3.3 Human Evaluation

Task	GPT-2 (w/o T)	GPT-2	No Preference
Relevance	23.05	46.11	30.83
Reference	11.67	31.94	56.39

Table 4.5: Results of human evaluation. The numbers show the percentage(%) of times a particular option was selected for each metric.

In addition to automated evaluation, we perform human evaluation on the ablation (GPT-2-w/o T and GPT-2 models) to assess the quality of generations, and the importance of grounding generations in context. Three human judges annotated 120 unique samples for *relevance* and *reference*, described next. Both models (with and without context) produced grammatically fluent outputs without any noticeable differences.

Relevance: The annotators are provided with the input of a procedural text T, the *st*, and the relational questions. The output events generated by GPT-2 (w/o T) and GPT-2 are also provided in random order. The annotators were asked, "Which system (A or B) is more accurate relative to the background information given in the context?" They could also pick option C (no preference).

[2]We use Sharma et al. [159] for calculating the overlap. https://github.com/Maluuba/nlg-eval

Error Class	Description	%	Question	Reference	Predicted
Polarity	The predicted polarity was wrong but event was correct	5%	What does 'oil fields over-used' help at eventually ?	there is not oil refined	more oil is refined
Linguistic Variability	The output was a linguistic variant of the reference	20%	What does 'fewer rabbits will become pregnant' hurts at imminently ?	more rabbits	more babies
Related Event	The output was related but different reference expected	17%	What does you inhale more air from the outside hurts at imminently ?	there will be less oxygen in your blood	you develop more blood clo- -ts in your veins
Wrong	The output was was completely unrelated	30%	What does 'less nutrients for plants' hurt at eventually ?	more plants	more wine being produced
Erroneous Reference	The gold annotations were erroneous	2%	What does 'less rabbit rabbit mating' hurt at imminently?	less rabbits	more babies

Table 4.6: Examples of error categories. Error analysis is only shown for the incorrect outputs.

Comparison with true event (reference): We measure how accurately each system-generated event reflects the reference (true) event. Here, the annotators saw only the reference sentence and the outputs of two systems (A and B) in a randomized order. We asked the annotators, "Which system's output is closest in meaning to the reference?" The annotators could pick the options A, B, or C (no preference).

For relevance and reference comparison tasks (Table 4.5), we present the percentage of the count of human judges for each of the three categories. The table illustrates that `GPT-2` performs better than `GPT-2` (w/o T) on both the metrics. Particularly, `GPT-2` not only performs better than `GPT-2` (w/o T) but also much better than the "No Preference" option in the relevance metric. This means that `GPT-2` generates target events that logically follow the passage and source events. The reference and relevance task scores together show that `GPT-2` does not generate target events that are exactly similar to the reference target events, but they are correct in the context of the passage and the source event. This can happen due to linguistic variation in the generation, as well as the ability of the source event to influence multiple target events in the context of the passage. We study this in more detail in the error analysis presented below.

4.3.4 Error Analysis

Table 4.6 shows the error analysis on 100 random samples from the validation set. We found that for about 26% of samples, the generated event influence had an exact match with the reference, and about 30% of the samples had no overlap with the reference (category *Wrong* in Table 4.6). We found that for 20% of the cases, the generated target event was correct but was expressed differently compared to the reference text (*Linguistic Variability* class in Table 4.6). Furthermore, we observed that in 17% of cases, the generated target event was not the same as the reference target event, but was relevant to the passage and the question, as shown in the *Related Event* category in Table 4.6. In 5% of the samples (*Polarity*), the model generates events with opposite polarity compared to the reference. A small fraction (2%) of samples had incorrect gold annotations.

4.3.5 Consistency Analysis

Finally, we measure if the generated *st*-graphs are consistent. Consider a path of length two in the generated *st*-graph (say, A $\rightarrow$ B $\rightarrow$ C). A consistent graph would have identical answers to *what does A help eventually* i.e., "C", and *what does B help imminently* i.e., "C".

To analyze consistency, we manually evaluated 50 random generated length-two paths, selected from WIQA-*st* development set. We observed that 58% of the samples had consistent output w.r.t to the generated output. We also measure consistency w.r.t. the gold standard, and observe that the system output is about 48% consistent. Despite being trained on independent samples, our *st*-graphs show reasonable consistency and improving consistency further is an interesting future research direction.

4.3.6 Discussion

In summary, our task formulation allows adapting pretrained language models for generating *st*-graphs that humans find meaningful and relevant. Automated metrics show the utility of using large-scale models and grounding the *st*-graph generation in context. We establish multiple baselines with varying levels of parameter size and pretraining to guide future research.

4.4 RQ2: CURIE for Downstream Tasks

In this section, we describe the approach for augmenting *st* graphs for downstream reasoning tasks. We first identify the choice of tasks (*st*-tasks) for domain adaptive pretraining [66] and obtain CURIE language model $\mathcal{M}$. The downstream task then provides input context, *st* and (relation, type) tuples of interest, and obtains the *st*-graphs (see Algorithm 1). We describe one such instantiation in the section §4.4.1.

4.4.1 CURIE augmented WIQA-QA

We examine the utility of CURIE-generated graphs in the WIQA-QA [182] downstream question answering benchmark. Input to this task is a context supplied in form of a passage T, a starting event c, an ending event e, and the output is a label {*helps*, *hurts*, or *no_effect*} depicting how the ending e is influenced by the event c.

We hypothesize that CURIE can augment c and e with their influences, giving a more comprehensive picture of the scenario compared to the context alone. We use CURIE trained on WIQA-*st* to augment the event influences in each sample in the QA task as additional context.

More concretely, we obtain the influence graphs for c and e by defining $R_{fwd} = \{$(*helps*, *imminent*), (*hurts*, *imminent*) $\}$ and $R_{rev} = \{$ (*helped by*, *imminent*), (*hurt by*, *imminent*)$\}$, and using algorithm 1 as follows:

$$G(c) = \texttt{IGEN}(T, c, R_{fwd})$$
$$G(e) = \texttt{IGEN}(T, e, R_{rev})$$

Query Type	BERT + CURIE	BERT
1-hop	**78.78**	71.60
2-hop	**63.49**	62.50
3-hop	**68.28**	59.50
Exogenous	**64.04**	56.13
In-para	73.58	**79.68**
Out-of-para	**90.84**	89.38
Overall	**76.92**	73.80

Table 4.7: QA accuracy by number of hops, and question type. BERT refers to the original BERT results reported in Tandon et al. [182], and BERT + CURIE are the results obtained by augmenting the QA dataset with the influences generated by CURIE.

We hypothesize that WIQA-*st* graphs are able to generate reasoning chains that connect c to e, even if e is not an immediate consequence of c. Following Tandon et al. [182], we encode the input sequence $\texttt{concat}(T, c, e)$ using the BERT encoder E [42], and use the [CLS] token representation ($\hat{\mathbf{h}}_i$) as our sequence representation.

We then use the same encoder E to encode the generated effects $\texttt{concat}(G(c), G(e))$, and use the [CLS] token to get a representation for augmented c and e ($\hat{\mathbf{h}}_a$). Following the encoded inputs, we compute the final loss as: $\mathbf{l}_i = \texttt{MLP}_1(\hat{\mathbf{h}}_i)$, and $\mathbf{l}_a = \texttt{MLP}_1(\hat{\mathbf{h}}_a)$ and $\mathcal{L} = \alpha \times \mathcal{L}_i + \beta \times \mathcal{L}_a$,

where $\mathbf{l}_i, \mathbf{l}_a$ represent the logits from $\hat{\mathbf{h}}_i$ and $\hat{\mathbf{h}}_a$ respectively, and $\mathcal{L}_i$ and $\mathcal{L}_a$ are their corresponding cross-entropy losses. α and β are hyperparameters that decide the contribution of the generated influence graphs and the procedural text to the loss. We set $\alpha = 1$ and $\beta = 0.9$ across experiments.

QA Evaluation Results Table 4.7 shows the accuracy of our method vs. the vanilla BERT model by question type and number of hops between cf and e. We also observe from Table 4.7 that augmenting the context with generated influences from CURIE leads to considerable gains over BERT based model, with the largest improvement seen in 3-hop questions (questions where the e and c are at a distance of three reasoning hops in the influence graphs). The strong performance on the 3-hop question supports our hypothesis that generated influences might be able to connect two event influences that are farther apart in the reasoning chain. We also show in Table 4.7 that augmenting with CURIE improves performance on the difficult exogenous category of questions, which requires background knowledge.

In summary, the evaluation highlights the value of CURIE as a framework for improving performance on downstream tasks that require counterfactual reasoning and serves as an evaluation of the ability of CURIE to reason about *st*-scenarios.

4.4.2 Zero-shot Evaluation

In addition to supervised augmentation, we also evaluate CURIE-$\mathcal{M}$ in a zero-shot setting. Towards this, we perform a zero-shot evaluation on QUARTZ [177], a dataset for qualitative counterfactual reasoning.

Each sample in QUARTZ consists of a question $\mathbf{q}_i$ = *If the top of the mountain gets hotter, the ice on the summit will:*, context $\mathbf{k}_i$ = *ice melts at higher temperatures*, the task is to pick the right answer from two options $\mathbf{a}_i^1$ = *increase*, and $\mathbf{a}_i^2$ = *decrease*. Since this task is setup as a qualitative binary classification task, CURIE cannot be directly adopted to augment the QA pairs like described in Algorithm 1.

For the zero-shot setting, we use CURIE-$\mathcal{M}$ fine-tuned on QUAREL-*st* as our language model. For an unseen test sample $(\mathbf{q}_i, \mathbf{a}_i^1, \mathbf{a}_i^2, \mathbf{k}_i)$, we select $\mathbf{a}_i^1$ as the correct answer if $p_\theta(\mathbf{a}_i^1 \mid \mathbf{x}_i) > p_\theta(\mathbf{a}_i^2 \mid \mathbf{x}_i)$, and select $\mathbf{a}_i^2$ otherwise (here p_θ stands for QUAREL-*st*). Our zero-shot CURIE-$\mathcal{M}$ achieves a 54% accuracy compared to supervised BERT model which achieves 54.7% accuracy.

These results suggest that CURIE performs competitively at tasks while having no access to any supervision.

4.4.3 Discussion

In summary, we show substantial gains when a generated st-graph is fed as an additional input to the QA model. Our approach forces the model to reason about influences within a context, and then ask questions, which proves to be better than asking the questions directly.

4.5 Related Work

Closed-domain *st* reasoning : In NLP, a large body of work has focused on what-if questions where the input is a context, *st*, and an ending, and the task is to predict the reachability from *st* to the ending. The most common approach [141, 176, 182] is a classification setting where the path is defined as more or less (qualitative intensities) over the sentences in the input context (a paragraph or procedural text with ordered steps). Such models do not generalize across domains because it is difficult to deal with changing vocabularies across domains. In contrast, our framework combines such diverse *st*-reasoning tasks under a general framework.

Open-domain *st* reasoning : Very recently, there has been interest in *st* reasoning from a retrieval setting [103] and a more common generation setting, attributed partially to the rise of neural generation models [202]. Qin et al. [136] presents generation models to generate the path from a counterfactual to an ending in a story. Another recent dataset [148] proposes defeasible inference in which an inference (X is a bird, therefore X flies) may be weakened or overturned in light of new evidence (X is a penguin), and their dataset and task is to distinguish and generate two types of new evidence – intensifiers and attenuators. We make use of this dataset by reformulating their abductive reasoning setup into a deductive setup (see §4.2.2 for details).

Current systems make some simplifying assumptions, e.g. that the ending is known. Multiple *st* (e.g., more sunlight, more pollution) can happen at the same time, and these systems can only handle one situation at a time. Finally, all of these systems assume that the *st* happens once in a context. Our framework strengthens this line of work by dropping that assumption of an ending being given, during deductive *st* reasoning. In principle, our formulation is general enough to allow for multiple *st* and recursive reasoning as more situations unfold. Most importantly, our framework is the first to allow for *st* reasoning across diverse datasets, within a realistic setting where only the context and *st* are known.

4.6 Conclusion

We present CURIE, a situational reasoning that: (i) is effective at generating *st*-reasoning graphs, validated by automated metrics and human evaluations, (ii) improves performance on two downstream tasks by simply augmenting their input with the generated *st* graphs. Further, our framework supports recursively querying for any node in the *st*-graph. For future work, we aim to design advanced models that seeks consistency, and another line of research to study recursive *st*-reasoning as a bridge between dialog and reasoning.

Chapter 5

Explanations for Humans

Defeasible reasoning is a mode of reasoning where conclusions can be overturned by taking into account new evidence. A commonly used method in cognitive science and logic literature is to handcraft argumentation supporting inference graphs. While humans find inference graphs very useful for reasoning, constructing them at scale is difficult. In this chapter, we automatically generate such inference graphs through transfer learning from a related NLP task that shares the kind of reasoning that inference graphs support. Through automated metrics and human evaluation, we find that our method generates meaningful graphs for the defeasible inference task. Human accuracy on this task improves by 20% by consulting the generated graphs. Our findings open up exciting new research avenues for cases where machine reasoning can help human reasoning.

5.1 Introduction

Defeasible inference [148] is a mode of reasoning in which given a premise $\mathbf{P}$ (Rob went for a hike), a hypothesis $\mathbf{H}$ (Rob saw an elephant, it was pink) may be weakened or overturned in light of new evidence i.e., an update $\mathbf{U}$ (Rob often has hallucinations). Given the non-monotonic nature of this reasoning, humans find it challenging to master this task [123]. This problem has been widely studied in classical AI through logic [79, 113], and in cognitive science through argumentative models [132]. A prominent approach is to support defeasible inference through argumentations by constructing an *inference graph* [133].

Despite their prominence [12], argumentative models are not scalable because an inference graph needs to be handcrafted for every example. Recently, Rudinger et al. [148] proposed two auxiliary tasks related to defeasible inference: (i) an NLI task to predict whether an update $\mathbf{U}$ would weaken or strengthen a hypothesis $\mathbf{H}$, and (ii) a generative task to generate an update $\mathbf{U}$ given a premise $\mathbf{P}$ and a hypothesis $\mathbf{H}$. However, this only addresses a part of the problem because their inference is still not supported by the line of reasoning that a human typically uses to solve this task, namely mediators (e.g., hallucinations can be deceptive) and contextualizers (some elephants can have mutated gene which makes them look different) that are inherently embedded in an inference graph, limiting their utility for humans (figure 5.1).

In this paper, we adopt the concept of an inference graph for defeasible reasoning from cognitive science and provide a computational model to make their generation scalable. Training such a model would require a large amount of annotated inference graphs, which will be too expensive to obtain. Instead, our solution is to draw a parallel to a related reasoning task in NLP [182], where the reasoning is supported by a graph that we find has similarities with the kind of reasoning that an inference graph supports. We train a model that can learn from the NLP task and effectively transfer it to generate inference graphs. Such transfer learning is made possible due to the powerful seq-to-seq neural language models that did not exist before.

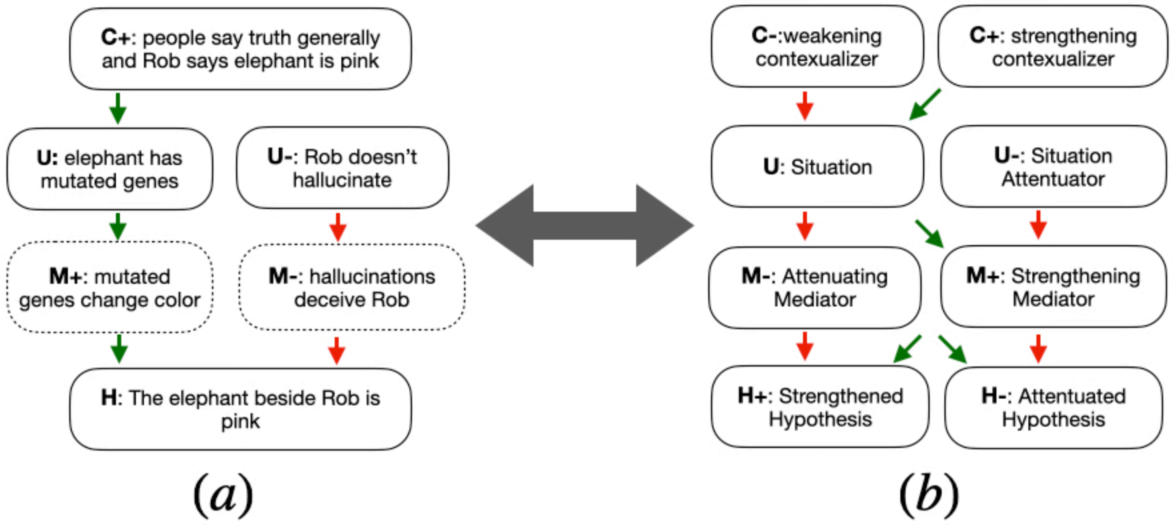

C+: people say truth generally and Rob says elephant is pink
U: elephant has mutated genes
U-: Rob doesn't hallucinate
M+: mutated genes change color
M-: hallucinations deceive Rob
H: The elephant beside Rob is pink
(a)
C-:weakening contexualizer
C+: strengthening contexualizer
U: Situation
U-: Situation Attentuator
M-: Attenuating Mediator
M+: Strengthening Mediator
H+: Strengthened Hypothesis
H-: Attentuated Hypothesis
(b)

Figure 5.1: (**a**) An example of an Inference Graph adapted from Pollock [133] and (**b**) Structure of an Influence Graph adapted from WIQA [182] dataset. The adapted influence graph incorporates the contextualizers, mediators, hypotheses and situations, making them useful for defeasible reasoning.

The contributions of this paper are the answers to the following two research questions:

- Can we automate the construction of the argumentation supporting inference graphs? In §5.2, we show that we can effectively construct meaningful graphs using transfer learning.
- Can our generated graphs help improve human performance? In §5.3, we show that humans leverage generated graphs to improve their performance on a previously reported benchmark.

5.2 Generating argumentation supporting Inference Graphs

We start by drawing parallels to a counterfactual reasoning task in NLP - the WIQA [182] task. WIQA consists of a set of procedural passages, each accompanied by a human-curated *influence graph*. The influence graph captures the causal influences between the events in the context of the process described by the passage. We draw a connection between inference graphs [133] and influence graphs [182] by drawing parallels between their reasoning structures. In essence, each inference graph from Pollock [132] can be instantiated via an influence graph from Tandon et al. [182] by interpreting the nodes in both the graphs as follows (Figure 5.1):

i. **Contextualizers (C):** these nodes set the context around a situation and connect to the $\mathbf{P}$ in some way.

ii. **Updates (U):** these nodes are new situations that emerge which might overturn an inference.

iii. **Hypothesis (H):** Hypothesis nodes describes the outcome/conclusion of the situation.

iv. **Mediators (M):** Mediators are nodes that help bridge the knowledge gap between a situation and a hypothesis node by explaining their connection explicitly.

Figure 5.1 presents an example to highlight the similarities between the two graphs by labeling an example node adapted from [133], and the structure of the influence graph from [182] with the four node types that we defined above. A green edge indicates that the source node has a positive influence on the target node, and a red edge indicates a negative influence. Further, each node can either act as a *strengthener* (**+**) or a *weakener* (**-**) for the hypothesis. Consequently, these graphs can support similar type of reasoning e.g., the effect of $\mathbf{U}$ on $\mathbf{H}$ and how this can change in light of external influences ($\mathbf{C}$) is captured by graph paths $\mathbf{C+}$ to $\mathbf{U}$ and from $\mathbf{U}$ via a mediator node (**M+**/**M-**) to $\mathbf{H}$. Inspired by these similarities, we hypothesize that influence graphs can be used to supplement defeasible reasoning.

5.2.1 Influence Graphs Generation

To obtain an influence graph for each defeasible query, we perform a zero-shot transfer from WIQA [182], a corpus of 2100 (passage, influence graphs) pairs.

Training : We treat influence graph generation as a sequence-to-sequence mapping task. We leverage WIQA to derive parallel data $\{(\mathbf{seq}^i_{ip}, \mathbf{seq}^i_{op})\}_{i=1}^{N}$ for the task. Let $(\mathbf{T}_i, \mathbf{G}_i)$ be a sample in WIQA, where $\mathbf{T}_i$ is the passage text (e.g. describing how viruses spread), and $\mathbf{G}_i$ is the corresponding influence graph (e.g., Figure 5.2). To create tokens of the input sequence $\mathbf{seq}^i_{ip}$, the model trains best with explicit markers:

$$\mathbf{seq}^i_{ip} = \text{Premise: } \mathbf{T}_i \mid \text{Update: } \mathbf{U}_i \mid \text{less/ more: } \mathbf{H}_i \tag{5.1}$$

where $\mathbf{T}_i$ is the passage text (e.g. steps describing how viruses spread) and $\mathbf{U}_i$ and $\mathbf{H}_i$ are nodes of $\mathbf{G}_i$ (these are phrases as shown in Figure 5.2).

The output $\mathbf{seq}^i_{op}$ is set to a DOT-string representation of the corresponding influence graph $\mathbf{G}_i$, as such a representation was shown to be effective at extracting high-quality graphs [108] from free-form text using language models (examples in the appendix). Thus, each passage-graph pair $(\mathbf{T}_i, \mathbf{G}_i)$ from WIQA is mapped to an input-output pair $\mathcal{D} = (\mathbf{seq}^i_{ip}, \mathbf{seq}^i_{op})$. We use this corpus to fine-tune an autoregressive language model $\mathcal{M}$ for graph generation. Essentially, the fine-tuned $\mathcal{M}$ allows us to efficiently sample an influence graph for a given input sequence $\mathbf{seq}^j_{ip}$ by drawing samples from $\mathbf{G}_j \sim P_\theta(y \mid \mathbf{seq}^j_{ip})$ using greedy sampling, where θ denotes the parameters of the language model.

Zero-shot Transfer to Defeasible Inference : We use the model $\mathcal{L}$ trained on WIQA to generate inference graphs on the defeasible inference dataset by Rudinger et al. [148]. We obtain an

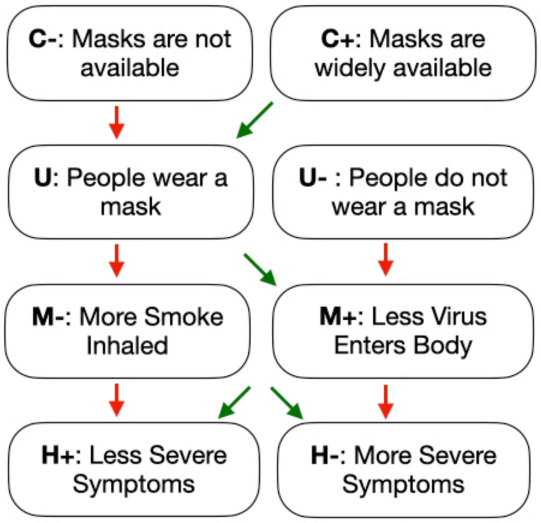

C-: Masks are not available
C+: Masks are widely available
U: People wear a mask
U- : People do not wear a mask
M-: More Smoke Inhaled
M+: Less Virus Enters Body
H+: Less Severe Symptoms
H-: More Severe Symptoms

Figure 5.2: An example of an influence graph similar to ones in WIQA that we train on.

influence graph for each defeasible input (**P**, **H**, **U**) by converting it to an input sequence that can be fed to $\mathcal{M}$ by filling the template (5.1). This conversion from (**P**, **H**, **U**) to template (5.1) is done by setting the premise **P** as the context passage **T**, the update **U** as the node **U**, and the attenuated and strengthened outcomes are simulated by prefixing the hypothesis **H** with the tokens *Less* and *More* respectively. This input is then passed to the $\mathcal{M}$ to generate an influence graph.

Results on Influence Graph Generation We use T5-11B [140] fine-tuned on $\mathcal{D}$ derived from WIQA (§5.2.1) as our graph generation language model ($\mathcal{M}$). All the graphs generated by our model were in valid DOT format. We use the standard generation metrics BLEU [127] and ROUGE [102] to evaluate $\mathcal{M}$ on the test split of WIQA. Each node N_i in the reference graph is compared with the corresponding generated node $\hat{N}_i$ using $\texttt{BLEU}(N_i, \hat{N}_i)$ (Node-BLEU). Further, node-edge-node pairs (neighbors) (N_i, N_j) and $(\hat{N}_i, \hat{N}_j)$ are compared using Rel-BLEU = $\texttt{HM}(\texttt{BLEU}(N_i, \hat{N}_i), \texttt{BLEU}(N_j, \hat{N}_j))$ where HM is the harmonic mean. These metrics are averaged over the graph (i.e., across the nodes and the edges), and further averaged across the corpus. We perform these experiments across two different language models: GPT-2-MEDIUM [138] and T5-11B. Finally, we calculate the overlap in the edge structures of the reference and generated graphs match as Edge-MATCH%. We report the numbers in Table 5.1, and include a random baseline for reference. A random baseline will correctly generate the nodes **S**, **H**+, and **H**- as they are part of the query ($\frac{3}{8}$ nodes). As neither of these nodes are connected to another, the random baseline will likely not generate any node pair correctly (Rel-BLEU $\sim$ 0). Since two unique graph structures are possible [182], a random baseline would get Edge-match $\sim$ 50%. Table 5.1 shows that our T5-based model is able to generate syntactically valid (high edge-match) and semantically meaningful graphs. Additionally, we find that our generated graphs are helpful to humans on a downstream task, as described next.

Model	Random	GPT-2-MEDIUM	T5-11B
Node-BLEU	37.5	46.05	**50.94**
Rel-BLEU	0.0	19.34	**33.01**
Edge-match%	50.0	92.86	**97.63**

Table 5.1: Results on automated metrics showing that our T5-11B model is able to generate very accurate graph structure and meaningful nodes that sufficiently match the reference nodes.

5.3 Do generated graphs help humans at defeasible reasoning?

Human Evaluation Rudinger et al. [148] performed a human evaluation on 2000 defeasible queries, where given (**P**, **H**, **U**), the task was to label the nature of the effect of **U** on **H** as *Intensifies* or *Attenuates*. Three human judges labeled each query, and the majority label was then compared with the ground-truth to ascertain the accuracy. In their setup, human judges were collectively right on 1745 samples (correct pool) and wrong on 255 samples (wrong pool). We create a challenging pool of 510 queries for the human judges by combining the 255 queries in the wrong pool with 255 queries sampled from the correct pool, giving a baseline accuracy of 50% for this eval pool. Each query in this pool is supplemented with a generated influence graph (§5.2). We found that our generated influence graphs showed high-levels of redundancy in contextualizers and mediators, with about 46% of the generated influence graphs repeating these nodes. We found that humans find it simpler to follow positive chains of influence, so to reduce their cognitive load, we post-process each influence graph to only retain the strengthening contextualizer (Figure 5.1), the situation (**U**), the strengthening mediator (**M+**), and the hypothesis (**H**).

In order to establish comparable gains, we replicate the evaluation setup of Rudinger et al. [148] by using use the same Amazon Mechanical Turk template and the instruction set, and the same pool of 230 qualified annotators that Rudinger et al. [148] selected based on a paid qualification test, in which the workers were asked to answer SNLI queries of varying levels of difficulty. We paid slightly above $15 per hour for the tasks.

For each query, in addition to answering the defeasible question, three judges were asked to evaluate the augmented influence graphs on two aspects:

i) **Is the influence graph useful?** The judges were allowed to select from the following:
   (a) *helpful*: the graph was crucial in helping towards answering the question
   (b) *relevant but not helpful*: the graph had the right topic (relevant to the question) but did not help in answering the question.
   (c) *irrelevant or misleading*: the graph was irrelevant to the question or misled the human judge to a wrong answer.

ii) **Why is the influence graph useful?** The judges were given an option to highlight the most useful aspect of the generated influence graph. They were allowed to tag one or more of the following aspects as the most helpful: i) Extraneous node, ii) Mediating node, and iii) Structure of the graph.

We summarize the key findings below.

Finding 1: influence graphs are helpful and relevant As Table 5.2 shows, a large majority of the human judges found the influence graphs to be helpful or relevant. We calculate the inter-annotator agreement for this question using majority-agreement $= \frac{1}{N}\sum_{i=1}^{N} \texttt{ma}_i$ where $\texttt{ma}_i$ indicates a majority agreement for the i^{th} sample (i.e., at least 2 out of 3 judges agreed on the label for the sample). The majority-agreement (`ma`) on these labels was 0.83. The judges marked about 25% of the graphs as relevant but not helpful. The graphs in such cases were on topic but not helpful in answering the query, thereby distinguishing the cases when the graph was crucial in reaching the correct answer. Finally, we note that the graphs provided as hints could have been helpful in two ways: by helping the human annotators arrive at the answer, or by reinforcing their mental picture that helped them in making the right decision. Future research in this direction is needed to study these aspects in depth.

Helpful	47.25
Relevant but not helpful	25.09
Irrelevant or misleading	10.58
No majority agreement	17.05

Table 5.2: Helpfulness of the augmentations.

Finding 2: Mediators are the most helpful for defeasible queries For every sample, we asked the human judges to mark which parts of the graph was the most helpful. The judges could select more than one aspect of the graph if they found multiple useful aspects. Table 5.3 shows the percentage of human judges that selected the particular graph aspect as most helpful. We observe that 49.48% of the judges who found the graphs useful indicated the mediator node as the most helpful. This indicates that while there may be other events that impact $\mathbf{U}$ and $\mathbf{H}$, the mediating events are the most informative in determining the type of link between them.

Aspect	% marked useful
Mediator	49.48
Extraneous	32.03
Structure	12.81
None helpful	5.68

Table 5.3: Most useful aspects of an influence graph.

Finding 3: Machine generated influence graphs help humans in defeasible reasoning Table 5.4 shows that performance improves across all three tasks when the defeasible query is augmented with an influence graph. On our challenging set of 510 queries, the overall accuracy jumps nearly 20 points from 0.50 to 0.698. Figure 5.3 highlights that 113 queries that were previously given the wrong answers were marked correctly when augmented with the influence graphs.

Dataset	Human [148]	Human (ours)
SNLI	0.461 ± 0.11	0.553 ± 0.11
SOCIAL	0.628 ± 0.07	0.814 ± 0.06
ATOMIC	0.418 ± 0.06	0.657 ± 0.06
overall	0.500 ± 0.04	$\mathbf{0.698} \pm \mathbf{0.04}$

Table 5.4: Human performance (accuracy) on the three tasks with and without generated influence graphs along with Wilson's score intervals for $\alpha = 95\%$. We tested the statistical significance of these results using the McNemar's test [115] and found the results to be statistically highly significant ($p < 1e - 6$).

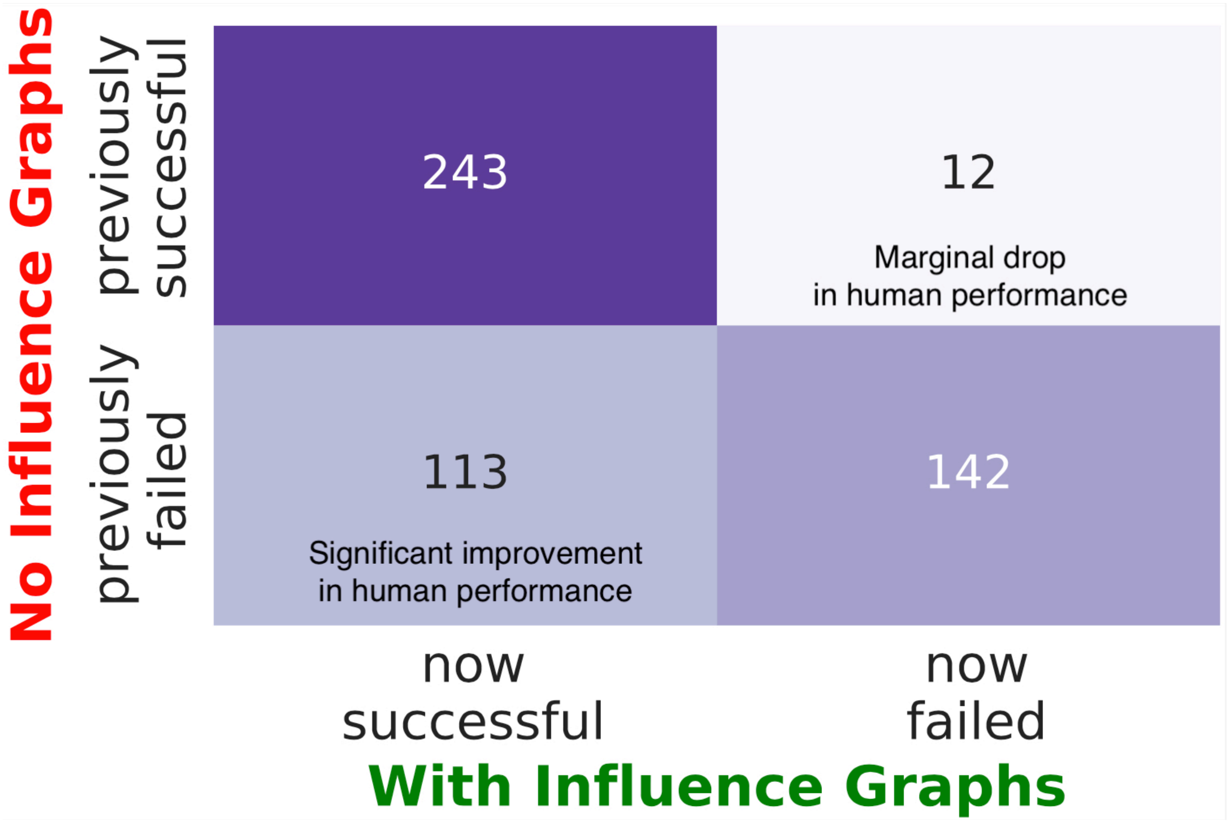


Figure 5.3: Human performance before and after the human judges were provided with the influence graph.

5.4 Discussion and Conclusion

Our work takes the idea of using inference graphs for defeasible inference and scales up its usability by automatically generating and augmenting them to a downstream defeasible task that both humans and machines are known to find difficult. We identify that the contextualizer and mediator nodes are crucial to defeasible inference, and show that our generated graphs generate these critical nodes effectively. Humans perform significantly better (20% absolute improvement) across diverse defeasible datasets and overwhelmingly attribute their success to the mediator nodes – giving insights into what helps and why. In this case study, we show that machines can fill the gaps in human knowledge when for defeasible reasoning. While we establish that humans are helped by these graphs, a further investigation on how (and if) the graphs reinforced their beliefs, and what additional information in the graphs was beneficial to their understanding is essential. Furthermore, a deeper understanding of the trade-offs (time spent in answering these questions with and without the graphs) also forms important future work.

Part II

Explanations via Models

Chapter 6

Explanation via Models: Self-Explaining Models

This chapter proposes methods to achieve interpretability via designing model architectures that are fundamentally amenable to interpretability. The challenge is that such models need to maintain their performance. Predominant approaches for it in NLP are limited to local feature attribution based on attention scores which are often less useful and even deceiving to end users. In this chapter, we introduce **SELFEXPLAIN**, a novel inherently interpretable self-explaining framework for NLP that explains a classifier's predictions using phrase-oriented concepts providing both local (input level) and global (training data level) interpretations. SELFEXPLAIN augments existing neural classifiers by adding (1) a *globally interpretable layer* that identifies the most influential concepts in the training set for a given sample and (2) a *locally interpretable layer* that quantifies the contribution of each local input concept by computing a relevance score relative to the predicted label. Experiments across five text-classification datasets show that SELFEXPLAIN provides improved interpretability without sacrificing performance. Most importantly, explanations from SELFEXPLAINare perceived as more trustworthy, understandable, and useful by human judges as compared to existing strong and widely-used baselines.

Utility : End task performance, interpretability through concepts in the training set and the given sample in the same architecture
Simplification : Natural language concepts via phrases under a parse tree (compared to word-level)
Explanation Structure : Single recursive depth with globally interpretable influential training phrases, and most relevant phrases from input sample

6.1 Introduction

Neural network models are often opaque: they provide limited insight into interpretations of model decisions and are typically treated as "black boxes" [104]. There has been ample evidence that such models overfit to spurious artifacts [64, 96, 114] and amplify biases in data [174, 209]. This underscores the need to understand model decision making.

Prior work in interpretability for neural text classification predominantly follows two approaches [147]: (i) *post-hoc explanation methods* that explain predictions for previously trained models using model internals, and (ii) *inherently interpretable models* whose interpretability is built-in and optimized jointly with the end task. While post-hoc methods are often the only option for already-trained models, inherently interpretable models may provide greater transparency since explanation capability is embedded directly within the model [26, 46, 90, 116, 147].

In natural language applications, feature attribution based on attention scores [40] has been the predominant method for developing inherently interpretable neural classifiers. Such methods interpret model decisions *locally* by explaining the classifier's decision as a function of relevance of features in input samples. While these methods enable interpretations of black-box classifiers, their interpretations are shown to be unreliable [134, 158] and even unfaithful [81, 192]. Moreover, with natural language being highly structured and compositional, explaining the role of higher-level combinational concepts like phrasal structures goes beyond individual low-level feature attributions, and remains an open challenge.

An alternative class of inherently interpretable classifiers explains model predictions *locally* using human-understandable high-level concepts such as prototypes [28, 116] and interpretable classes [93, 95]. They were recently proposed for computer vision applications, but despite their promise have not yet been adopted in NLP. A known limitation of such approaches is that they often cannot provide *global* explanations (explaining their decisions as a function of influential training data), which limits their generality.

In this work, we propose SELFEXPLAIN–a self-explaining model framework for interpretable text classifiers that combines the global and local aspects of interpretability via human-interpretable high-level language concepts, producing a holistic picture of a classifier's decisions. SELFEXPLAIN incorporates two neural network modules: (i) *Globally Interpretable Layer*, a differentiable layer that uses maximum inner product search (MIPS) to retrieve the most influential concepts from the training data for a given input sample, and (ii) *Locally Interpretable Layer*, a layer that quantifies the relevance of each concept to the final label distribution of a sample through activation differences. We show that these layers can be integrated into transformer classifiers, converting them into inherently interpretable architectures. The interpretability of the classifier is enforced through regularization, and the entire model is end-to-end differentiable. To the best of our knowledge, SELFEXPLAINis the first inherently interpretable neural text classification approach to provide both global and local interpretability in a single framework.

Our experiments on three text classification tasks spanning five datasets with pretrained transformer models show that incorporating these interpretable layers facilitates richer interpretation while maintaining end-task performance. The explanations from SELFEXPLAINare perceived by human annotators as more trustable, understandable, and useful compared to strong baseline interpretability methods. Our contributions are:

(i) We propose SELFEXPLAIN a novel *self-explaining* framework for neural text classifiers that embeds both local and global aspects of interpretability. (ii) We introduce the *Globally Interpretable Layer* and *Locally Interpretable Layer*, two novel end-to-end differentiable modules that explain predictions both locally (through relevance of each concept) and globally (through influential concepts from the training data). (iii) SELFEXPLAINs explanations improves over low-level feature attributions and enables explanations through higher-level language phrasal structures.

6.2 Related Work

Interpretability by Probing: Heat maps based on attention [7] are one of the commonly used interpretability tools for many downstream tasks such as machine translation [107], summarization [149] and reading comprehension Hermann et al. [72]. It is widely debated whether such attention mechanisms are reliable [81, 134, 192, 210]. An alternative to attention is to use gradients [167, 175]. A recent finding [54] suggests gradient outputs may be counter-intuitive. Other post-hoc interpretability methods such as Singh et al. [166] and Jin et al. [87] try to decompose relevant and irrelevant aspects from hidden state vectors and obtain a relevance score. These models focus on local interpretability, while methods such as Koh and Liang [92] and Han et al. [68] focus on retrieving the most influential training samples for global interpretations.

Interpretability through Rationales: Another recent line of work explores collecting *rationales* (snippets of text from input that can sufficiently predict the output label)Another recent line of work explores collecting *rationales* (snippets of text from input that can sufficiently predict the output label). Commonly, they are acquired through expert annotations [205]. Current neural networks are trained on large-scale data, often costly to obtain. Some of the work along these lines include collecting explanations along with the end-task. Some of the work along these lines include collecting explanations along with the end-task label for commonsense reasoning [142] and natural language inference [24]. Although very desirable, the costs associated with obtaining human-annotated rationales for interpretability severely limits its applicability.

Interpretability by Design: Card et al. [25] relies on interpreting a given sample as a weighted sum of the training samples while Croce et al. [35] identifies influential training samples using a kernel-based transformation function. Jiang and Bansal [86] produce interpretations of a given sample through modular architectures, where model decisions are explained through outputs of intermediate modules. Our work resembles Melis and Jaakkola [116], in that we jointly train to optimize for interpretability and end-task performance, and we enforce interpretability as a regularization parameter. Additionally, we also provide the most influential training samples and show that the end-performance is not sacrificed for interpretability.

6.3 SELFEXPLAIN framework

Let $\mathcal{M}$ be a neural C-class classification model that maps $\mathcal{X} \rightarrow \mathcal{Y}$, where $\mathcal{X}$ are the inputs and $\mathcal{Y}$ are the outputs. SELFEXPLAIN builds into $\mathcal{M}$, and it provides a set of explanations $\mathcal{Z}$ via high-level "concepts" that explain the classifier's predictions. We first define interpretable concepts in §6.3.1. We then describe how these concepts are incorporated into a concept-aware encoder in §6.3.2. In §6.3.3, we define our Local Interpretability Layer (`LIL`), which provides local explanations by assigning relevance scores to the constituent concepts of the input. In §6.3.4, we define our Global Interpretability Layer (`GIL`), which provides global explanations by retrieving influential concepts from the training data. Finally, in §6.3.5, we describe the end-to-end training procedure and optimization objectives.

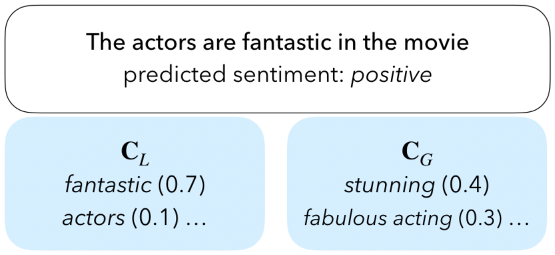

The actors are fantastic in the movie
predicted sentiment: *positive*
$\mathbf{C}_L$
fantastic (0.7)
actors (0.1) ...
$\mathbf{C}_G$
stunning (0.4)
fabulous acting (0.3) ...

Figure 6.1: A sample of interpretable concepts from SELFEXPLAIN for a binary sentiment analysis task. $\mathcal{C}_L$ denotes the most relevant local concepts from the sample while $\mathcal{C}_G$ denotes the most influential phrases from training set that contributed to the predicted label.

6.3.1 Defining human-interpretable concepts

Since natural language is highly compositional [121], it is essential that interpreting a text sequence goes beyond individual words. Let $\mathcal{Z}$ be a set of basic units for interpretability which we call *concepts* that are interpretable by humans. In principle, concepts can be words, phrases, sentences, paragraphs or abstract entities. In this work, we focus on phrases as our concepts. Assume a grammar $\mathbf{G} = \{N, \Sigma, \theta_p\}$, that takes a sentence $\mathbf{x}$ and outputs a parse tree $\mathbf{y}$, where N represents the set of non-terminals, Σ represents the set of terminals and θ_p represents the production rules. Given any sequence $\mathbf{x} = \{w_i\}_{1:T}$, we decompose the sequence into its component non-terminals $N(\mathbf{x}) = \{nt_j\}_{1:J}$, where J denotes the number of non-terminal phrases in x.

Given an input sample $\mathbf{x}$, $\mathcal{M}$ is trained to produce two types of explanations: (i) global explanations from the training data $\mathcal{X}_{train}$ and (ii) local explanations, which are phrases in $\mathbf{x}$. We show an example in Figure 6.1. Global explanations are achieved by identifying the most influential concepts $\mathcal{C}_G$ from the "concept store" $\mathbf{Q}$, which is constructed to contain all concepts from the training set $\mathcal{X}_{train}$ by extracting phrases under each non-terminal in a syntax tree for every data sample (detailed in §6.3.4). Local interpretability is achieved by decomposing the input sample $\mathbf{x}$ into its constituent phrases under each non-terminal in its syntax tree. Then each concept is assigned a score that quantifies its contribution to the sample's label distribution for a given task; $\mathcal{M}$ then outputs the most relevant local concepts $\mathcal{C}_L$.

6.3.2 Concept-Aware Encoder E

We obtain the encoded representation of our input sequence $\mathbf{x} = \{w_i\}_{1:T}$ from a pretrained transformer model [105, 187, 200] by extracting the final layer output as $\{\mathbf{h}_i\}_{1:T}$. Additionally, we compute representations of concepts, $\{\mathbf{u}_j\}_{1:J}$. For each non-terminal nt_j in $\mathbf{x}$, we represent it as the mean of its constituent word representations $\mathbf{u}_j = \frac{\sum_{w_i \in nt_j} \mathbf{h}_i}{len(nt_j)}$ where $len(nt_j)$ represents the number of words in the phrase nt_j. To represent the root node ($\mathbb{S}$) of the syntax tree, $nt_{\mathbb{S}}$, we use the pooled representation (`[CLS]` token representation) of the pretrained transformer as $\mathbf{u}_{\mathbb{S}}$ for brevity.[1] Following traditional neural classifier setup, the output of the classification layer l_Y

[1]We experimented with different pooling strategies (mean pooling, sum pooling and pooled `[CLS]` token representation) and all of them performed similarly. We chose to use the pooled `[CLS]` token for the final model as

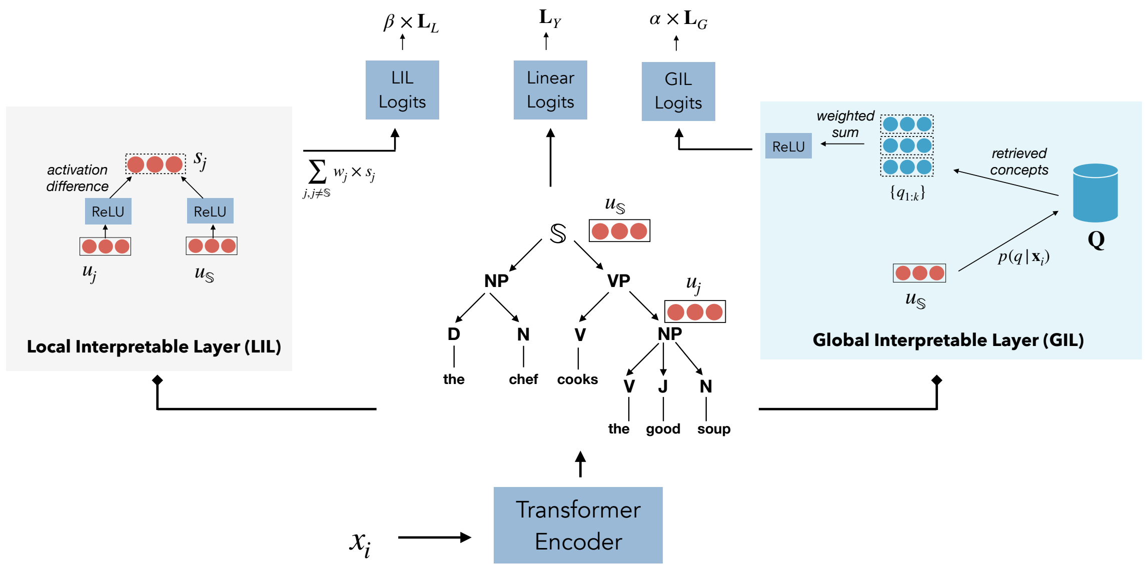


Figure 6.2: Model Architecture: Our architecture comprises a base encoder that encodes the input and its relative non-terminals. `GIL` then uses MIPS to retrieve the most influential concepts that *globally* explain the sample, while `LIL` computes a relevance score for each nt_j that quantifies its relevance to predict the label. The model interpretability is enforced through regularization (example parse tree inspired from Zanzotto et al. [206]).

is computed as follows:

$$l_Y = \texttt{softmax}(\mathbf{W}_y \times g(\mathbf{u}_{\mathbb{S}}) + \mathbf{b}_y)$$
$$P_C = \arg\max(l_Y)$$

where g is a $relu$ activation layer, $\mathbf{W}_y \in \mathbb{R}^{D \times C}$, and P_C denotes the index of the predicted class.

6.3.3 Local Interpretability Layer (`LIL`)

For local interpretability, we compute a local relevance score for all input concepts $\{nt_j\}_{1:J}$ from the sample $\mathbf{x}$. Approaches that assign relative importance scores to input features through activation differences [122, 160] are widely adopted for interpretability in computer vision applications. Motivated by this, we adopt a similar approach to NLP applications where we learn the attribution of each concept to the final label distribution via their activation differences. Each non-terminal nt_j is assigned a score that quantifies the contribution of each nt_j to the label in comparison to the contribution of the root node $nt_{\mathbb{S}}$. The most contributing phrases $\mathcal{C}_L$ is used to locally explain the model decisions.

Given the encoder $\mathbf{E}$, `LIL` computes the contribution solely from nt_j to the final prediction. We first build a representation of the input without contribution of phrase nt_j and use it to score the labels:

$$t_j = g(\mathbf{u}_j) - g(\mathbf{u}_{\mathbb{S}})$$
$$s_j = \texttt{softmax}(\mathbf{W}_v \times t_j + \mathbf{b}_v)$$

this is the most commonly used method for representing the entire input.

where g is a $relu$ activation function, $t_j \in \mathbb{R}^D$, $s_j \in \mathbb{R}^C$, $\mathbf{W}_v \in \mathbb{R}^{D \times C}$. Here, s_j signifies a label distribution without the contribution nt_j. Using this, the relevance score of each nt_j for the final prediction is given by the difference between the classifier score for the predicted label based on the entire input and the label score based on the input without nt_j:

$$\mathbf{r}_j = (l_Y)_i|_{i=P_C} - (s_j)_i|_{i=P_C}$$

where $\mathbf{r}_j$ is the relevance score of the concept nt_j.

6.3.4 Global Interpretability layer (`GIL`)

The Global Interpretability Layer `GIL` aims to interpret each data sample $\mathbf{x}$ by providing a set of K concepts from the training data which most influenced the model's predictions. Such an approach is advantageous as we can now understand how important concepts from the training set influenced the model decision to predict the label of a new input, providing more granularity than methods that use entire samples from the training data for post-hoc interpretability [68, 92].

We first build a Concept Store Q which holds all the concepts from the training data. Given the neural classifier model $\mathcal{M}$, we represent each concept candidate from the training data, q_k as a mean pooled representation of its constituent words $q_k = \frac{\sum_{w \in q_k} e(w)}{len(q_k)} \in \mathbb{R}^D$, where e represents the embedding layer of $\mathcal{M}$ and $len(q_k)$ represents the number of words in q_k. The concept store Q is represented by a set of $\{q\}_{1:N_Q}$, which are N_Q number of concepts from the training data. As the model $\mathcal{M}$ is finetuned for a downstream task, the representations q_k are constantly updated. Typically, we re-index all candidate representations q_k after every fixed number of training steps.

For any input $\mathbf{x}$, `GIL` produces a set of K concepts $(q_1, q_2, .., q_K)$ from Q that are most influential as defined by the cosine similarity function:

$$d(\mathbf{x}, Q) = \frac{\mathbf{x} \cdot q}{\|\mathbf{x}\| \|q\|} \quad \forall q \in Q$$

Taking $\mathbf{u}_{\mathbb{S}}$ as input, `GIL` uses dense inner product search to retrieve the top-K influential concepts $\mathcal{C}_G$ for the sample. Differentiable approaches through Maximum Inner Product Search (MIPS) has been shown to be effective in Question-Answering settings [44, 67] to leverage retrieved knowledge for reasoning [2]. Motivated by this, we repurpose this retrieval approach to identify the influential concepts from the training data and learn it end-to-end via backpropagation. Our inner product model for `GIL` is defined as follows: $p(q|\mathbf{x}_i) = \frac{exp\ d(\mathbf{u}_{\mathbb{S}}, q)}{\sum_{q'} exp\ d(\mathbf{u}_{\mathbb{S}}, q')}$

6.3.5 Training

SELFEXPLAINis trained to maximize the conditional log-likelihood of predicting the class at all the final layers: linear (for label prediction), `LIL` , and `GIL` . Regularizing models with explanation specific losses have been shown to improve inherently interpretable models [116] for local interpretability. We extend this idea for both global and local interpretable output for our classifier model. For our training, we regularize the loss through `GIL` and `LIL` layers by optimizing their output for the end-task as well. For the `GIL` layer, we aggregate the scores over all the retrieved $q_{1:K}$ as a weighted sum, followed by an activation layer, linear layer and softmax

[2]MIPS can often be efficiently scaled using approximate algorithms [161]

to compute the log-likelihood loss as follows:

$$l_G = \texttt{softmax}(\mathbf{W}_u \times g(\sum_{k=1}^{K} \mathbf{w}_k \times q_k) + \mathbf{b}_u)$$

and $\mathcal{L}_G = -\sum_{c=1}^{C} y_c \log(l_G)$ where the global interpretable concepts are denoted by $\mathcal{C}_G = q_{1:K}$, $\mathbf{W}_u \in \mathbb{R}^{D \times C}$, $\mathbf{w}_k \in \mathbb{R}$ and g represents $relu$ activation, and l_G represents the logits for the `GIL` layer.

For the `LIL` layer, we compute a weighted aggregated representation over s_j and compute the log-likelihood loss as follows:

$$l_L = \sum_{j, j \neq \mathbb{S}} \mathbf{w}_{sj} \times s_j, \ \mathbf{w}_{sj} \in \mathbb{R}$$

and $\mathcal{L}_L = -\sum_{c=1}^{C} y_c \log(l_L)$. To train the model, we optimize for the following joint loss,

$$\mathcal{L} = \alpha \times \mathcal{L}_G + \beta \times \mathcal{L}_L + \mathcal{L}_Y$$

where $\mathcal{L}_Y = -\sum_{c=1}^{C} y_c \, log(l_Y),$. Here, α and β are regularization hyper-parameters. All loss components use cross-entropy loss based on task label y_c.

6.4 Experiments

6.4.1 Datasets

We evaluate our framework on five classification datasets: (i) SST-2 Sentiment Classification task [168]: the task is to predict the sentiment of movie review sentences as a binary classification task. (ii) SST-5 : a fine-grained sentiment classification task that uses the same dataset as before, but modifies it into a finer-grained 5-class classification task. (iii) TREC-6 : a question classification task proposed by Li and Roth [101], where each question should be classified into one of 6 question types. (iv) TREC-50: a fine-grained version of the same TREC-6 question classification task with 50 classes (v) subj: subjective/objective binary classification dataset [126]. The dataset statistics are shown in Table 6.1.

Dataset	**C**	**L**	**Train**	**Test**
SST-2	2	19	68,222	1,821
SST-5	5	18	10,754	1,101
TREC-6	6	10	5,451	500
TREC-50	50	10	5,451	499
subj	2	23	8,000	1,000

Table 6.1: Dataset statistics, where **C** is the number of classes and **L** is the average sentence length

6.4.2 Experimental Settings

For our SELFEXPLAINexperiments, we consider two transformer encoder configurations as our base models: (1) RoBERTa encoder [105]—a robustly optimized version of BERT [42]. (2) XL-Net encoder [200]—a large-scale transformer model based on Transformer-XL [37] architecture

	SST-2	SST-5	TREC-6	TREC-50	subj
XLNet-Base Classifier					
XLNet	93.4	53.8	**96.6**	82.8	96.2
SELFEXPLAINXLNet (K=5)	**94.6**	**55.2**	96.4	**83.0**	**96.4**
SELFEXPLAINXLNet (K=10)	94.4	55.2	96.4	82.8	96.4
RoBERTa-Base Classifier					
RoBERTa	94.8	53.5	97.0	89.0	96.2
SELFEXPLAINRoBERTa (K=5)	**95.1**	**54.3**	**97.6**	**89.4**	**96.3**
SELFEXPLAINRoBERTa (K=10)	95.1	54.1	97.6	89.2	96.3

Table 6.2: Performance comparison of models with and without `GIL` and `LIL` layers. All experiments used the same encoder configurations. We use the development set for SST-2 results (test set of SST-2 is part of GLUE Benchmark) and test sets for - SST-5, TREC-6, TREC-50 and subj. $\alpha, \beta = 0.1$ for all the above settings

and a permutation language modeling objective. We incorporate SELFEXPLAINnto RoBERTa and XLNet, and use the above encoders without the `GIL` and `LIL` layers as the baselines. We generate parse trees [91] for the input and follow same pre-processing steps as the original encoder configurations for rest.

We also maintain the hyperparameters and weights from the pre-training of the encoders. The architecture with `GIL` and `LIL` modules are fine-tuned for specific datasets described in §6.4.1. For the number of global influential concepts k, we consider two settings $k = 5, 10$. We also perform hyperparameter tuning on $\alpha, \beta = \{0.01, 0.1, 0.5, 1.0\}$ and select our best model configuration for our experimental results. All our models trained on an NVIDIA V-100 GPU.

6.4.3 Results

We study the effect of adding the layers `GIL` and `LIL` to the encoder configurations and present our results in Table 6.2. We compare the performance of our SELFEXPLAINversions of RoBERTa and XLNet with and without the interpretable layers added. From the table, we observe that these layers do not sacrifice end-task performance when integrated with both XLNet and RoBERTa encoders. Across the different classification tasks in our experimental settings, we observe that SELFEXPLAINRoBERTa version consistently shows competitive performance compared to the base models. The SELFEXPLAINXLNet model shows competitive performance on every task except for a marginal drop in TREC-6 dataset. We also observe that the hyperparameter K did not make noticeable difference. We also show ablation analysis for both `GIL` and `LIL` layers in Table 6.3. The results suggest that gains through `GIL` and `LIL` are complementary in nature.

Model	Accuracy
XLNet-Base	93.4
SELFEXPLAINXLNet + LIL	94.3
SELFEXPLAINXLNet + GIL	94.0
SELFEXPLAINXLNet + GIL + LIL	94.6
Roberta-Base	94.8
SELFEXPLAINRoberta + LIL	94.6
SELFEXPLAINRoberta + GIL	94.6
SELFEXPLAINRoberta + GIL + LIL	95.1

Table 6.3: Ablation: SELFEXPLAINXLNet and SELFEXPLAINRoBERTa base models on SST-2

Sample	P_C	Top relevant phrases from LIL	Top influential concepts from GIL
sam mendes segues from oscar winner to oscar - winning potential with a smooth sleight of hand	pos	no sophomore slump, segues	above credibility, spell binding
the iditarod lasts for days - this just felt like it did .	neg	for days	exploitation piece, heart attack
corny, schmaltzy and predictable, but still manages to be kind of heart warming, nonetheless.	pos	corny, schmaltzy, of heart	successfully blended satire, spell binding fun
suffers from the lack of a compelling or comprehensible narrative .	neg	comprehensible, the lack of	empty theatres, tumble weed

Table 6.4: Sample output from the model and its corresponding local and global interpretable outputs SST-2 (P_C stands for predicted class) (some input text cut for brevity).

6.4.4 Interpretability Evaluation

Though performance improvements can assure that the model does not need to be constrained to produce interpretable output, it is essential that any such interpretability is useful to the end-user of the model. Quantitative evaluation of interpretability in empirical terms is challenging [46] and human evaluation is often the most widely used method for it. To this end, we compare interpretable outputs from SELFEXPLAINagainst popular baselines using human judges. We focus our evaluation on the interpretability of our approach across the following three criteria:
(i) **Trustability**: Do the explanations help users trust the model predictions?
(ii) **Usefulness**: Are the explanations useful for understanding model prediction?
(iii) **Understandability**: Are the explanations understandable for a human? [3]

For the human evaluation, 14 graduate students in computer science were selected to be the human judges. Each human judge was presented with 50 samples from the SST-2 validation set of sentiment excerpts [168]. Each judge was provided the evaluation metric with a corresponding description. While administering the evaluation, the methods were anonymized and were asked to rate according to the evaluation criteria alone.

[3]Although *faithfulness* is a common metric to evaluate interpretability, recent work like Jacovi and Goldberg [80] and Wiegreffe et al. [194] have called into question the the binary notion of faithfulness evaluation and highlight that such an evaluation is challenging particularly for self-explaining models. Following their suggestion, we evaluate our approach along multiple human evaluation criteria.

Following Han et al. [68], for the human evaluation we compared our output against two commonly used interpretability methods (i) Influence functions [68] for global interpretability and (ii) Salience detection [165] for local interpretability. We obtained the output from our SELFEXPLAINXLNet model and presented the end-user with two outputs:
(i) **Most relevant local concepts**: We present users with the top ranked phrases based on $\mathbf{r}(nt_j)$ from the `LIL` layer.
(ii) **Top influential global concepts:** We present users with the most influential concepts $q_{1:K}$ ranked by the output of `GIL` layer.
Trustability: Trustability measures how much an interpretability technique helped an end-user to trust the model's label prediction. We evaluate trustability via *mean trust score*, a common human evaluation strategy used by previous work on interpretability [35, 87, 166]. Given an input, explanation, predicted label, and the gold label, judges were instructed to evaluate the explanations as a means to trust the model's prediction. Human judges were asked to assign the trust score on a 1-5 likert scale on each model independently. We the compute a mean of all scores for each model to compute the mean trust.

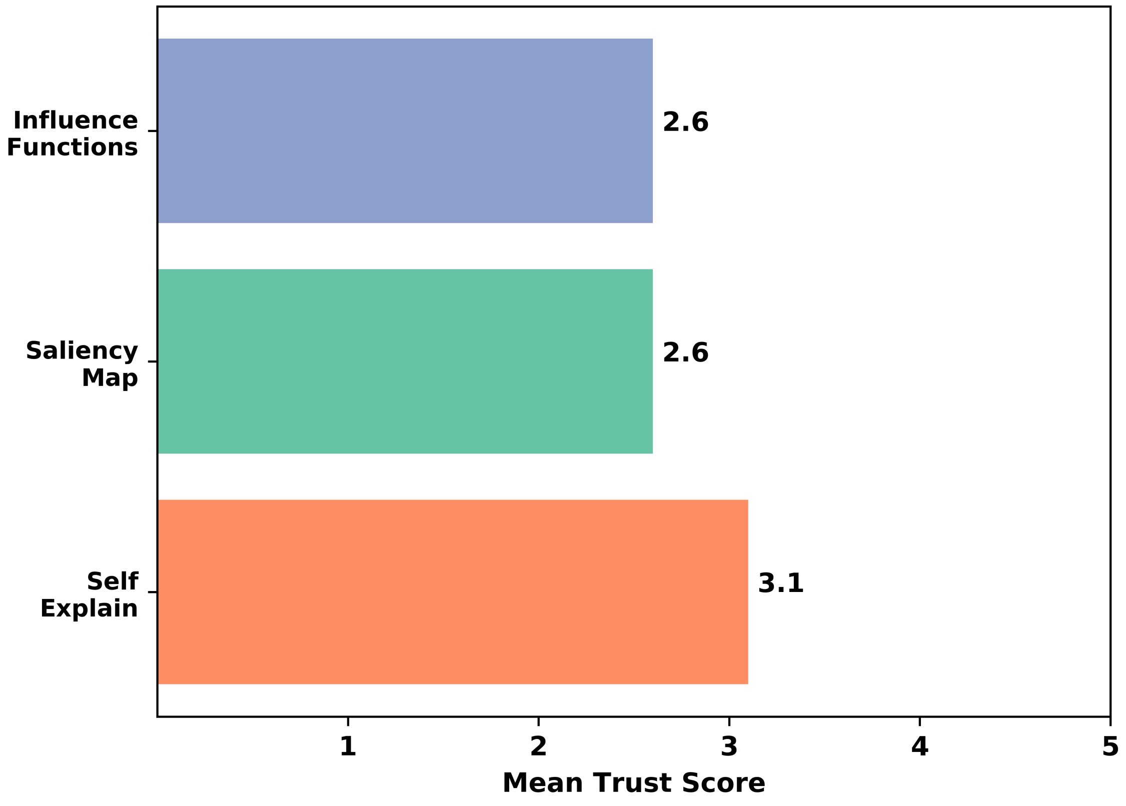


Figure 6.3: Mean Trust Score across models

Figure 6.3 shows the mean-trust score of SELFEXPLAIN in comparison to the baselines. We observe that SELFEXPLAIN scores higher in terms of human annotators' perceived mean trust score compared to the baselines. Our model achieves a mean-trust score of 3.1, an average of 0.5 more than saliency maps and influence functions. Both of the other baselines performed similarly in terms of mean-trust.

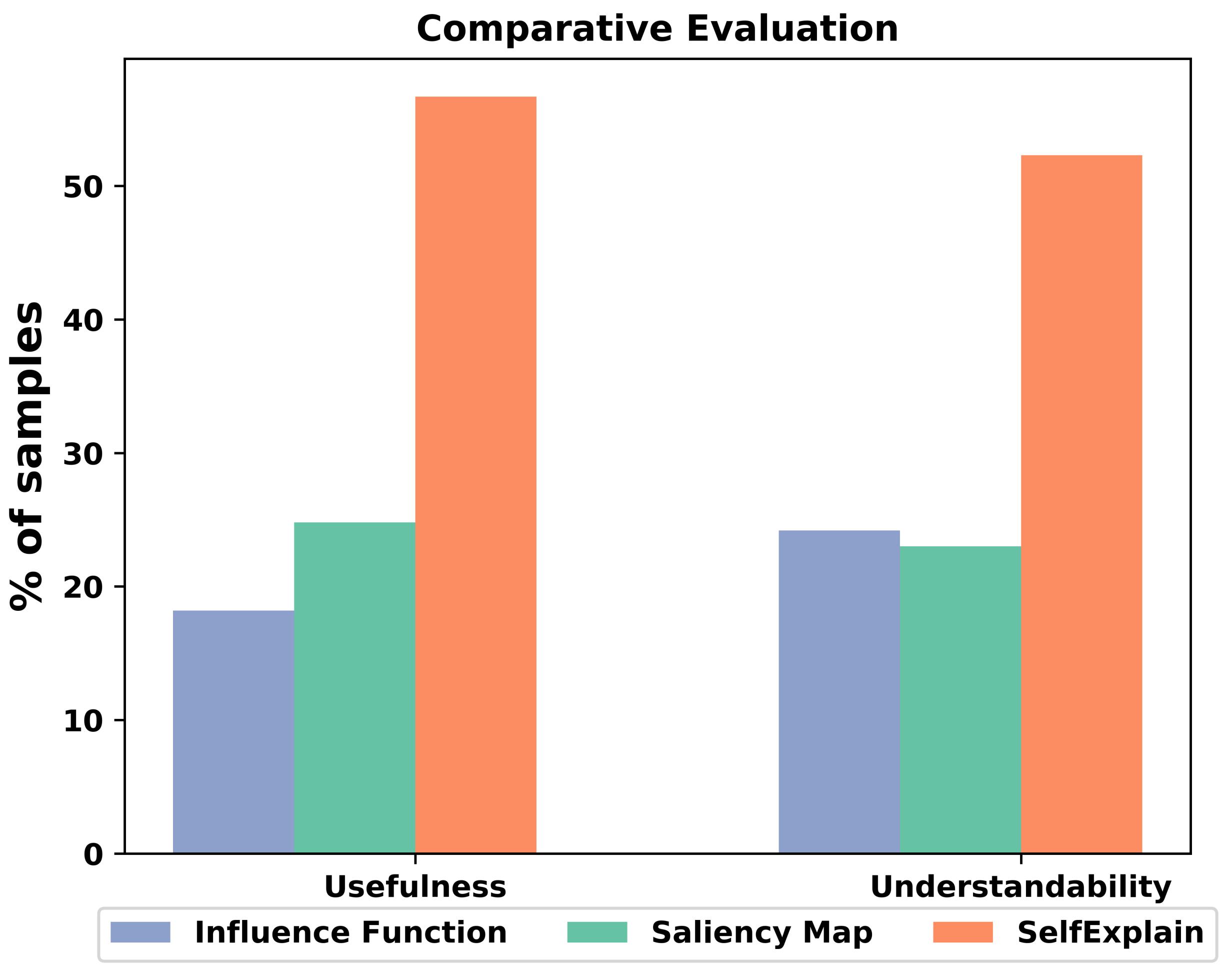


Figure 6.4: Comparative evaluation of *usefulness* and *understandability* of SELFEXPLAINw.r.t baselines

Usefulness: Usefulness is considered to be an important criteria for acceptance of a model [39]. To evaluate usefulness, we ask human judges how "useful" an interpretation was for the given prediction. Human judges were shown the input, gold label, predicted labels and explanations from SELFEXPLAINand the baselines and were asked to rate which system's explanation they perceive as more useful as an explanation to the model prediction. We the compute the ratio of samples that each method was rated as most useful.

Figure 6.4 (left) shows the relative performance of all the models for usefulness. The vertical axis shows the percentage of samples as judged by humans that interpretation method as either the most understandable. SELFEXPLAIN achieves a gain of 32% in terms of perceived usefulness. This evaluation provides further evidence that humans perceive explanations via concepts are perceived more useful.

Understandability: An essential criteria for a transparency in an AI system is the ability of a human to understand its interpretations [47]. Our understandability metric evaluates whether a human judge can understand the explanations presented by the model such that they are equipped to verify the model predictions. For this evaluation, human judges were given the input, gold label, and explanations from different methods (baselines, and SELFEXPLAIN, and were asked to select the explanation that they perceived to be the most understandable.

Figure 6.4 (right) shows the understandability scores of SELFEXPLAIN in comparison to the baselines. SELFEXPLAIN achieves 29% improvement over the best-performing baseline in terms of understandability of the model explanation.

6.5 Analysis

In Table 6.4 we show some qualitative examples from SELFEXPLAIN's explanations. Our qualitative output shows that our model is able to produce reasonable global and local interpretable concepts that are human readable as natural language text.

Sample	Top Contributing Phrases from `LIL`	Top Influential Concepts from `GIL`
it 's a very charming and often affecting journey	often affecting, very charming	scenes of cinematic perfection that steal your heart away, submerged, that extravagantly
it ' s a charming and often affecting journey of people	of people, charming and often affecting	scenes of cinematic perfection that steal your heart away, submerged, that extravagantly

Table 6.5: Sample (from SST-2) of an input perturbation - different local concepts but similar global concepts

Are `LIL` concepts relevant?: For this analysis, we randomly 50 samples from SST2 development set and removed the top most salient phrase ranked by `LIL`. Human judges were asked to predict the label without the most relevant local concept and the accuracy dropped by 7%. We also computed the neural classifier accuracy on the same input and the classifier accuracy dropped by about 14%. This analysis suggests that `LIL` local concepts capture the relevant phrases to a reasonable extent.

Does SELFEXPLAINs explanation help predict model behavior? In this setup, humans are presented with an explanation and an input, and must correctly predict the model's output [46, 98]. For this analysis, we randomly select 16 samples spanning equal number of true positives, true negatives, false positives and false negatives from the development set. Given a few learning examples, three human judges were tasked to predict the model decision with and without the presence of model explanation. We observe that when users were presented with the explanation, their ability to predict model decision improved by an average of 22%, showing that in the presence of SELFEXPLAINs explanations, humans can better understand model behavior.
Do similar examples have similar explanations? Melis and Jaakkola [116] argue that a crucial property that interpretable models need to address is *stability*, where the model should be robust enough that a minimal change in the input should not lead to drastic changes in the observed interpretations. We qualitatively analyze this notion of stability in our method. From our experiments, we identify that similar examples have high overlap of retrieving basis concepts. Table 6.5 shows one such example where a minor modification to the input leads to different phrases ranked by relevance, their global influential concepts remain the same.
`LIL`-`GIL`-Linear layer Agreement: To understand whether our explanations lead to predicting the same label as the model's prediction, we analyze whether the final logits activations on the `GIL` and `LIL` layers agree with the linear layer activations. Towards this, we compute an agreement between label distributions from `GIL` and `LIL` layers to the distribution of the linear layer. Our `LIL`-*linear* F1 is 96.6%, `GIL`-*linear* F1 100% and `GIL`-`LIL`-*linear* F1 agreement is 96.6% for SELFEXPLAINXLNet on the SST-2 dataset. We observe that the agreement between the `GIL` , `LIL` and the linear layer are very high, validating that SELFEXPLAINs layers agree on the same model classification prediction, showing that our interpretability layers `GIL` and `LIL` lead to same predictions.

6.6 Conclusion

In this chapter, we propose a self-explaining model that provides interpretability at both local and global levels. Our method uses human understandable phrasal concepts for interpretability and it is trainable end-to-end. Through human evaluation, we show that our interpreted output is perceived as more trustworthy, understandable, and useful for explaining model decisions compared to previous techniques for explainability. This opens an exciting research direction for building inherently interpretable models text classification. The next step would be to extend the framework to tasks beyond single sentence level and more advanced reasoning tasks such as QA.

Part III

Recursive Descent for Explanations

Chapter 7

Recursive Descent

We propose the **recursive descent** grammar framework, and in this framework, we adopt the view that explanations are recursive by nature and can be modeled as a grammar. Modeling explanations as recursive structures naturally lends them to be interactive, where any aspect of an explanation can be expanded as required by the stakeholders. Each transition (by a stakeholder) can be captured by a production from an **explanation grammar**. Whether an explanation is *accepted* is determined by a utility and a simplification function specified by the various stakeholders of the system.

7.1 Recursive Descent

In this section, we formalize the recursive descent for explanations. With the stakeholders defined, we now establish the basic building blocks of an explanation.

Definition 4 (Grounding) *connects a symbol (e.g. a node or edge) to an associated meaning so that the symbol or a composition of symbols can be reliably understood by all stakeholders.*

Definition 5 (Grounded Node) *is a basic fragment of reasoning, grounded in a particular domain theory that describes a stand-alone concept in that domain.*

Definition 6 (Node) *signifies a compositional concept or proposition that be recursively expanded and usually terminates when the final output consists only of grounded nodes.*

Definition 7 (Edge) *describes the relationship between two nodes, such that their composition forms a valid reasoning chain.*

Definition 8 (Grounded Edge) *an edge that is understandable by the stakeholders of interest defined for the system.*

Each node (`n`) quantifies a either a complex concept or proposition that acts as a building block of an explanation, depending on the end use-case. Each edge (`e`) describes the relationship, such that each connection represents the *glue* that gives conceptual coherence to their connection. The nature of the edge also determines the *compositional* nature of the explanations. A *valid* explanation is determined by the stakeholders of the system together, and that is established when all the leaf nodes and edges are grounded in the final explanation. Given the basic units, we now define the rules of how an explanation can be materialized.

The structure of an explanation can be unified under an *explanation grammar*. An explanation grammar can be characterized by a weighted context free grammar CFG $G = \{N, \Sigma, \theta_p, \mathbb{S}, A\}$, that takes an input sample for a task $\mathbf{T}$ and outputs an explanation, where N represents the set of non-terminals, Σ represents the set of terminals, θ_p represents the production rules, and $\mathbb{S}$ denotes the specified start symbol. Here, A represents the action states, in our case the action set $A = \{\textit{expand}, \textit{do not expand}\}$. We formally describe our explanation grammar with the following production rules.

$$\begin{aligned} \langle\text{explanation}\rangle &\models \langle\texttt{n}\rangle \\ \langle\texttt{n}\rangle &\models \langle\texttt{n}\rangle\langle\texttt{e}\rangle\langle\texttt{n}\rangle \mid \langle\texttt{n}\rangle \mid \langle n_G\rangle \\ \langle\texttt{e}\rangle &\models \langle\texttt{e}\rangle \mid \langle e_G\rangle \end{aligned}$$

Here, the nonterminals are the set of nodes and edges defined for a given task at hand $N = \{\{\texttt{n}\}, \{\texttt{e}\}\}$ and the terminal symbols $\Sigma = \{n_G, e_G\}$ that represent the grounded symbols. The non-terminal abstractions are defined based on task-specific abstractions.

1. Given this context free grammar G, $\mathcal{T}_G$ denotes the possible left-most derivations (explanations) under the explanation grammar G. When the task at hand is defined, we will define $\mathcal{T}_G$ as $\mathcal{T}$ for brevity.
2. For any derivation $t \in \mathcal{T}$, we define $\textit{acceptability}(t)$ to denote whether the explanation is accepted for the sample.
3. For a given sample s for a task $\mathbf{T}$, we write $\mathcal{T}(s)$ to refer the set $$\{t : t \in \mathcal{T}, \textit{acceptability}(t) = \texttt{true}\}$$
4. We say that an explanation for a sample s is valid, if $\textit{acceptability}(t) = \texttt{true}$

The key idea in recursive descent is to extend our grammar such that we define the criteria for when an explanation will be $\textit{acceptable}$ based on an **utility function** $U(t)$ and a **simplification function** $P(t)$. The utility $U(t)$ defines whether the explanation satisfies the defined end utility and simplification function $P(t)$ defines whether the explanation is human understandable (grounded in concepts that all the stakeholders can reliably understand). We discuss this in depth in sections §2.2.2 and §2.2.3.

$$acc(t) = \begin{cases} \texttt{true}, & \texttt{if } U(t) \geq \tau_1 \texttt{ and} \\ & P(t) \geq \tau_2 \\ \texttt{false}, & \texttt{otherwise} \end{cases}$$

Here, $acc(t)$ denotes $\textit{acceptability}(t)$, $P(t)$ denotes whether the explanation is grounded in nodes and edges that are human understandable (passes threshold τ_2), and the τ_1 signifies whether the given explanation passes the threshold computed by the utility function defined by various stakeholders.

Utility score of a tree t with rules:

$\alpha_1 \to \beta_1, \alpha_2 \to \beta_2, \cdot, \alpha_n \to \beta_n$ is

$$U(t) = \prod_{i=1}^{n} u(\alpha_i \to \beta_i)$$

where $u(\alpha \to \beta)$ is the utility score for the rule $\alpha \to \beta$. Similarly, we can compute an overall distribution for $P(t)$ for an explanation

$$P(t) = \prod_{i=1}^{n} p(\alpha_i \to \beta_i)$$

where $p(\alpha \to \beta)$ denotes the simplification score of the production.

Depending on what criteria the stakeholders determine to accept the explanation, the leaf nodes in an explanation can be grounded in one of cognitive science, philosophy, physics, biology, linguistics or social sciences. For building NLP systems, we choose *bounded rational choice theory* [163] to bound the explanations in rationality. Explanations that satisfy utility measured by rationality are widely used in fields like political sciences and behavioral economics. Such explanations often resort to *bounded rational choice*, that aims to maximize a given *utility* [13], given a specific task or a goal. In a similar spirit, we take the view that explanations in natural language applications should optimize a utility function determined by the stakeholders. In particular, we adopt the idea of "satisficing" from bounded rational choice theory - the strategy of considering the options available for an explanation until you find one that meets or exceeds a predefined threshold for a minimally acceptable outcome for a current task. This adaptation gives us the flexibility of bounding the recursive depth of any explanation given the rational bounds of the involved stakeholders. From a machine learning perspective, bounded rationality can be modeled as a bias-variance tradeoff problem [60], allowing us to formulate the explanation challenge as a machine learning task. An example is shown in Figure **??**, where we present a sample real-world stakeholder situation for a machine learning system in use. A preferable explanation should meet the most essential criteria across all the stakeholders.

Parameterizing $U(t)$ and $P(t)$: To model real world explainable systems, it is essential that we have the ability to translate $U(t)$ and $P(t)$ to machine learning tasks. In practice, we can parameterize $u(\alpha \to \beta)$ and $p(\alpha \to \beta)$ using a machine learning method (typically a neural network) of choice, and learnt via maximum likelihood estimation. The explanation learning challenge then can be cast as an optimization problem:

$$\arg\max_{E} \quad Q(E, \mathcal{M} | S_E, \mathcal{D}, \mathbf{T})$$

where

E - Explanation

Q - Metric Function (usually a combination of $U(t)$ and $P(t)$)

S_E - Stakeholder Requirements

$\mathcal{D}$ - Dataset (that is being explained)

$\mathbf{T}$ - Task

Case Study - Concept Bottleneck Reasoning: For our first example, we consider the concept bottleneck model proposed in Koh et al. [93]. An adapted example is shown in figure 7.1. The task at hand is classifying the bird given in the image. The concept bottleneck has a set of pre-defined intermediate concepts such as { *head color, beak shape,...* }. The utility of the concepts is motivated by what high-level concepts are useful for practitioners towards understanding the reasoning of the model decision process. Such a system can be naturally extended to a recursive reasoning formulation as shown in figure 7.1. A complex concept can be decomposed further into multiple simpler concepts depending on their utility u_i. In this example, the concept of *body color* is further decomposed into various color and pattern features such as { *yellow, black, black-white pattern* } and so on (which can be further decomposed). Each concept can be selectively decomposed based on whether the stakeholders want to explore further.

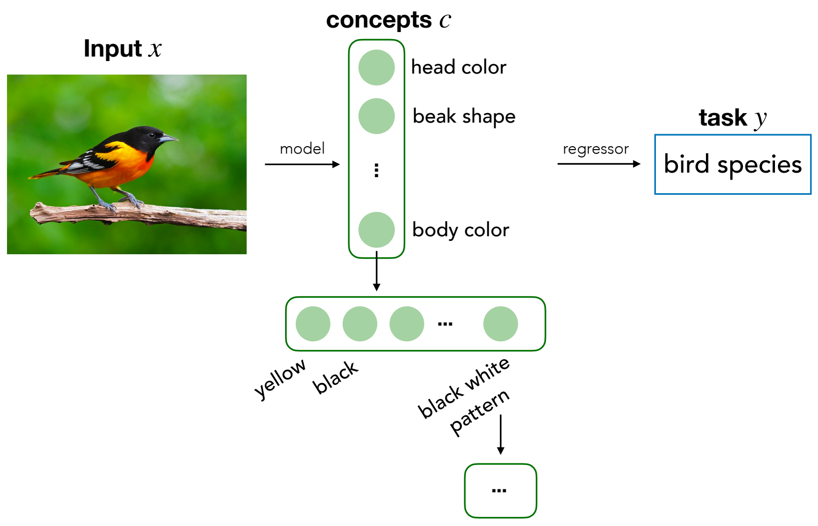


Figure 7.1: A sample recursive formulation of concept bottleneck models [92]

Case Study - Explainable Multihop Reasoning: Consider this example from Clark et al. [31]:

Input : Do nails conduct electricity ?
System Output : Yes
Explanation : Nails are metals

The question we explore here is - this a valid explanation for this question-answer pair? It is dependent on whether the explanation "adequately" explains the model decision. In natural language applications, the challenge is multifold since many reasoning problems cannot be easily decomposed into concepts alone. Consider a sample explanation shown in figure 7.2. Here nodes are scientific propositions, and edges are causal. At the first level of depth, the state elicits a causal mechanism of why nails conduct electricity. At the first level, the explanation assumes a utility based on the underlying causal mechanism. If a stakeholder potentially needs to explain further, such that each proposition in this case can be further validated as shown in the figure. Since the nodes are expressed in natural language itself, a simplification function might not be relevant to this example.

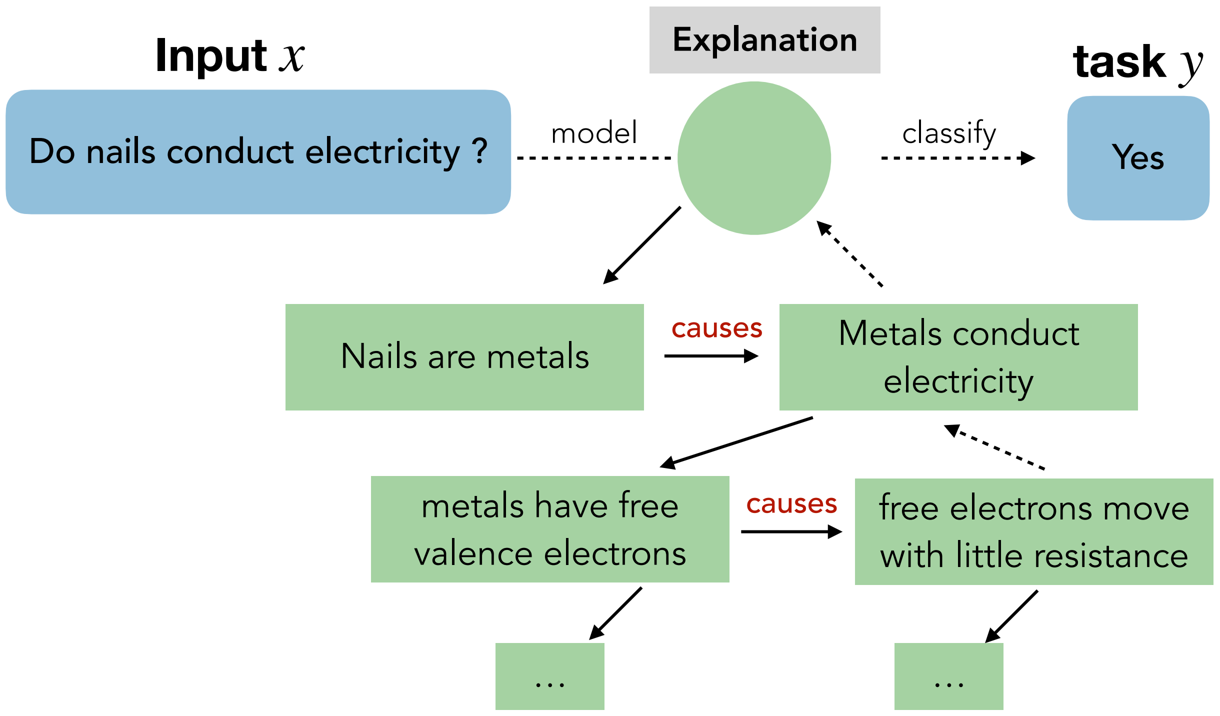

Input x
Explanation
task y
Do nails conduct electricity ?
model
classify
Yes
Nails are metals
causes
Metals conduct electricity
metals have free valence electrons
causes
free electrons move with little resistance
...
...

Figure 7.2: A sample recursive formulation for multihop reasoning for scientific QA. The explanation recursively queries each node at every depth and (could) explain it further by expanding any node.

7.2 Inverse Explanation Learning

Given an observed explanation, can we estimate the parameters of utility function and simplification functions of the explanation? We define this task as the *Inverse Explanation Learning* task, where we are given an utility function and a simplification function for observed explanations, and the task is to estimate their parameters. This is often referred in the literature as *post-hoc* methods for explainability. Although this task is flexible to accommodate models that are trained with and without explanations (as long as the utility and simplification functions are defined correctly), these methods assume that both $U(t)$ and $P(t)$ variables are chosen before-hand. Typically, each node n is a function of input sample, model, and training data ($f(x, \mathcal{M}, \mathcal{D})$) and edges are not involved in the productions. They inherently pose the risk of under-estimating what factors influence the model decision making. Additionally, it is significantly challenging to naturally incorporate human metrics such as reduction of bias, and improving trust into the model. In our opinion, we view post-hoc explanations to be beneficial mostly to Explanation Producer(s), rather than Explanation Consumer(s).

Similar to the explanation task, this can also be learnt through maximum likelihood estimation of the parameters. This can be formalized as optimizing the following function:

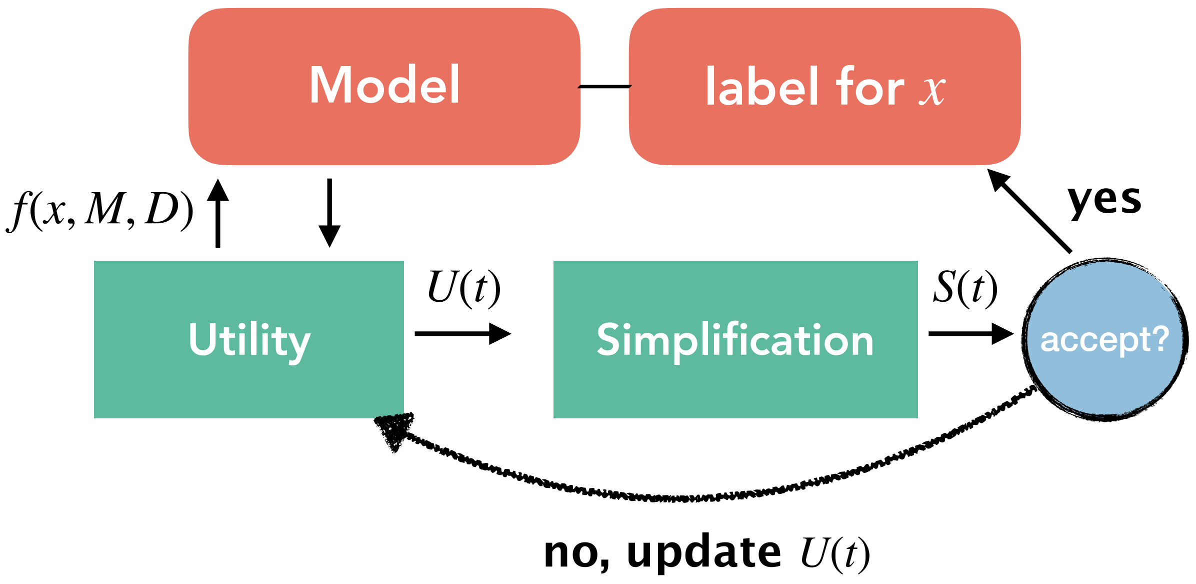


Figure 7.3: Overview of Inverse Explanation framework

$$\arg\max_{E} \quad Q(E|\mathcal{M}, S_E, \mathcal{D}, \mathbf{T})$$

where we assume the features for $u(\alpha \rightarrow \beta)$ and $p(\alpha \rightarrow \beta)$ of model $\mathcal{M}$ are assumed and we estimate the parameters of the features.

An overview of the current methods on inverse explanation framework is shown in Figure 7.3. Consider the example of saliency maps on gradients [165, 167, 175]. Given an input sample x, model $\mathcal{M}$, data $\mathcal{D}$, output label for x, the saliency maps aim to explain the most salient aspects of input that led to the prediction. Here, gradients ($f(x, \mathcal{M}, \mathcal{D})$) are assumed to reflect explanation capacity ($U(t)$). Once we obtain the gradients from the model $\mathcal{M}$, we then overlay each input feature in x to the gradient values (which amounts to $P(t)$). If the current $f(x, \mathcal{M}, \mathcal{D})$ does not adequately explain the prediction, it is then further complemented with additional factors for utility.

7.3 Grounding Assumption

Much of our understanding of world phenomena exists outside the paradigm of text corpora. For an explanation to be complete, each part of the explanation should be grounded in the shared experience of the world. Experiences in the world go far beyond language alone. Humans are adept at understanding the natural language symbols and ground it to their experience [185]. For explanations, we assume that its' basic elements are grounded in the stakeholders' world-view. To the question of whether machines can also achieve such grounding, a deeper treatment (beyond this chapter's scope) might be required and for that, we direct the readers to Bisk et al. [14].

7.4 Explanation Depth

A natural question arises for such a recursive account of explanation: *What is the deepest that an explanation can go, in the absence of a practical utility ?* In this section, we briefly (but not exhaustively) summarize how other fields of science approached this question.

A causal account of explanation is expected to account for all the fundamental causal mechanisms and depending on how it is achieved, there exists two schools of thought — *realist* or *epistemic* sense. A realist [151] posits that all the entities and processes of an explanation exists, thereby bounding the depth to only observable entities. An epistemic interpretation [70] treats entities and processes as tools for constructing an empirical model, and not rooted in exactly describing the reality. According to this, an adequately strong theoretical explanation can also be the bound, beyond only observable entities. Van Fraassen et al. [186] argue against realist interpretations and describe explanations as a series of *Why* questions interpreted via bayesian probability, compared to the logical notions of Hempel and Oppenheim [70].

An alternative view of explanations favor theories that are grounded in how humans perform the process of explanation. A linguistic or communicative theory [1] accommodates the social aspect of explanation where the relationship between individuals in the communication is also accounted for. Such an explanation can be bounded by the limits of the communication itself. Explanations can also be viewed as a purely cognitive activity where it represents the mental representation of the process that aids in understanding. Holland et al. [75] argue that internal representations of processes that occur due to the neural activity can be quantified as *if-then* rules, thereby bounding them to neural activity in human brains.

7.5 Conclusion

In this chapter, we propose a framework to bridge the gap between theory and practice towards building explainable NLP systems. Our unified view of explanation advocates for modeling rigorous explainable systems that are recursive and interactive in nature, and have a holistic view of their stakeholders' utility and simplicity requirements.

Chapter 8

Controllable Reasoning via Templates

In this chapter, we explore an implementation of *recursive descent* via templates. In this chapter, we explore this via free-form text explanations, where a reasoning chain with explanation can be expressed in diverse forms via templates. Towards this we (i) propose a dataset of multihop reasoning template-expansion pairs and (ii) a *prompt-template-filling* approach that uses sequence-to-sequence generation models for expanding multihop reasoning templates. Our experiments show that our approach outperforms baselines both in generation metrics and factuality metrics. We also present a detailed error analysis on our approach's ability to expand multihop reasoning

8.1 Introduction

Multihop reasoning aims to address the task of reasoning across different concepts [199]. Several approaches were proposed recently to address the task of multihop reasoning with explanations [55, 85, 179]. Very recently, Wei et al. [189] propose a chain of thought reasoning paradigm, where the model provides a series of short reasoning chain sentences before generating the final answer for the question. They show that generating the reasoning process before responding to a question greatly improves the reasoning performance. Yet, such methods still falls short of providing the ability to control aspects of reasoning and their corresponding explanations. In this work, we aim to address this challenge, by proposing a dataset and a method towards multihop reasoning systems that are controllable and explainable at the same time.

We formulate the multihop reasoning task as a prompt-based template filling task (*prompt-template-filling*) where a sequence-to-sequence (PLM) model is trained to fill a template that connects concepts with a valid reasoning chain. Figure 8.1 shows an example of our approach, where a single question-answer (QA) pair is represented by two template-expansion pairs. Modeling the reasoning task as a template has distinct advantages such as: (i) expressing the same reasoning sample in multiple ways, and (ii) controlling different aspects of the reasoning chain and explanation via *slots* in the template.

Towards this, our contributions in this paper are two-fold. First, we present a dataset of commonsense reasoning templates and their corresponding expansions that are valid completions of the template. The slots in the templates are open-ended and are not restricted to any particular vocabulary and enable controlling the reasoning chain. We also augment templates with an

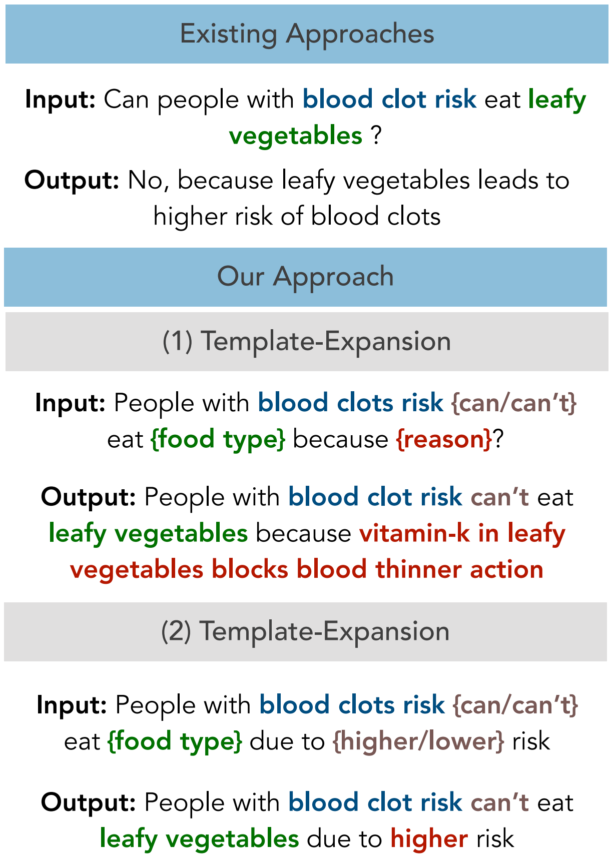

Existing Approaches
Input: Can people with blood clot risk eat leafy vegetables ?
Output: No, because leafy vegetables leads to higher risk of blood clots
Our Approach
(1) Template-Expansion
Input: People with blood clots risk {can/can't} eat {food type} because {reason}?
Output: People with blood clot risk can't eat leafy vegetables because vitamin-k in leafy vegetables blocks blood thinner action
(2) Template-Expansion
Input: People with blood clots risk {can/can't} eat {food type} due to {higher/lower} risk
Output: People with blood clot risk can't eat leafy vegetables due to higher risk

Figure 8.1: In this example, we show how existing a multihop QA sample with explanation reformulated as two different template-expansion pairs. This formulation allows to evaluate the ability of the model for the same reasoning in multiple ways where in each examples different aspects of the reasoning can be controlled.

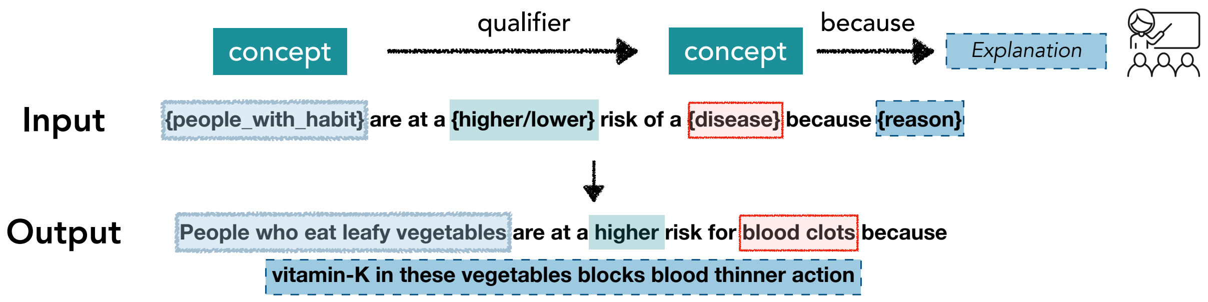

concept
qualifier
concept
because
Explanation
Input
{people_with_habit} are at a {higher/lower} risk of a {disease} because {reason}
Output
People who eat leafy vegetables are at a higher risk for blood clots because vitamin-K in these vegetables blocks blood thinner action

Figure 8.2: An overview of the overall template structure for our approach. Our goal is to reason across concepts in a multihop setting. Each template contains concept slots - *people_with_habit* and *disease*, and a multiple choice qualifier slot - *higher/lower* that describes their relationship and an explanation *reason* slot that aims to get a free-form text explanation for the relationship.

optional free-form explanation slot that explains the multihop reasoning connection between the concepts. Our dataset comprises of about 3600 unique template-expansion pairs collected from diverse sources, and we hope to enable PLM systems to effectively learn to fill multihop reasoning templates.

Next, we present *prompt-template-filling* approach, that models the multihop reasoning challenge as a PLM task where given a template, the goal of the model is to produce meaningful completed sentences for the template. The concept in each slot in the template is provided via a *prompt* [23], which indicates an abstraction of the nature of the slot. The multiple choice *qualifier* slot helps model the relationship between the concepts and the *explanation* slot generates a free-form text explanation for the reasoning chain. Specifying each slots in free-form text enables control allowing multihop reasoning questions to specify concepts, the qualitative relationship and the nature of explanation.

In our experiments from the commonsense reasoning template-expansion task, *prompt-template-filling* outperforms baseline approaches for template filling both in terms of generation metrics such as ROUGE and BERTSCORE, and factual correctness (factuality) metrics such as FACTCC. We also present a factuality evaluation using human judges and a detailed analysis of the model outputs. While we still observe factual errors, our approach provides a more nuanced understanding of the mistakes, potentially leading to several future research directions.

To summarize, we present a *prompt-template-filling* approach to enable PLM models to perform controllable and explainable multihop reasoning by training them for template-expansion task. Towards this, we present a dataset and we show that our approach outperforms baselines both in terms of generation metrics and factuality metrics.

8.2 Dataset

To investigate whether PLM models are effective at cross domain reasoning, we collect a dataset of templates that are composed of cross-domain reasoning chains and corresponding sentences that match the template. Figure **??** shows an example of a sample from our dataset. Each template in our dataset is composed of the following basic units:

1. *concept slot* : contains an abstract category form of a concept from one of the domains.
2. *qualifier slot* : a word or phrase that describes the nature of the effect of concept of one domain on the other (e.g. higher, lower,...)
3. *explanation slot* : this optional field consists of a free-form explanation that explains the reasoning across the concepts from the different domains.

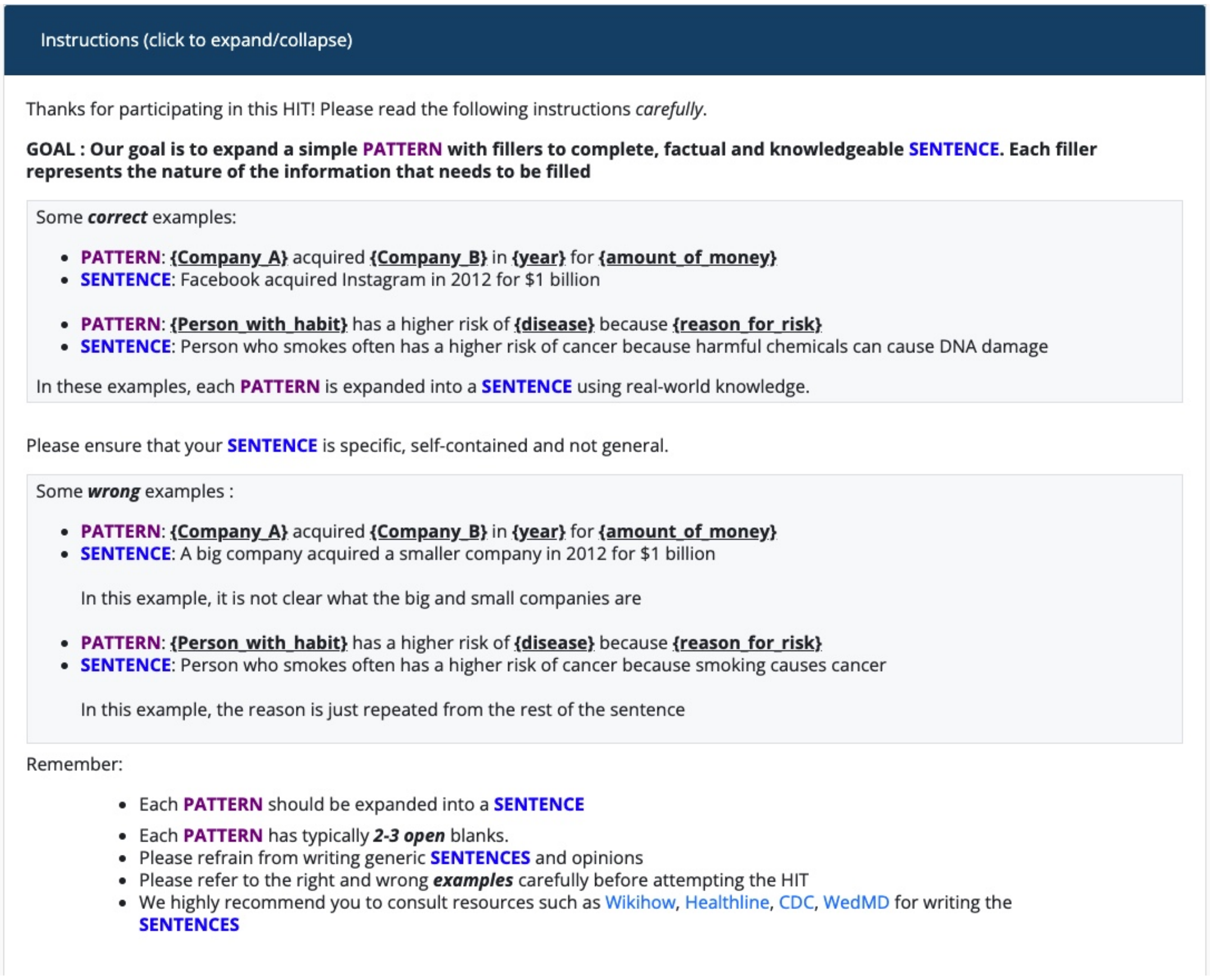
For our use-case, we use the *commonsense* domain and the *health and well-being* domain. In reasoning, it is a long-standing challenge to address *commonsense* reasoning with approaches ranging from building commonsense knowledge bases [111, 170] and neural-network based approaches [19, 152]. There has also been specialized knowledge resources for reasoning in the *health and well-being* domain [16, 153]. Due to their significant impact over the years, we chose these domains to collect corpus for our use-case.

For the use-case to reason across *commonsense* and *health and well-being*, we collect a set of template (x) and its corresponding expansions (y) based on this overall schema of reasoning

Template	Sentences
{person_at_location} has a {higher/lower} risk of {disease} because {reason_for_risk}	Person who lives in a city has a higher risk of depression - because of stress due to noise Person who lives near a village has a lower risk of respiratory illness - because of lower pollution
{person_taking_prescription} has a higher risk of {disease} due to {reason}	Someone on steroids have a higher risk for heart disease because - steroids compromise heart pumping People on insulin have a lower risk of hyperglycemia - because of lower glucose levels.
{food_item_1} should not be consumed with {food_item_2} because {reason}	Steak should not be consumed with mashed potatoes because - pairing fried foods with starchy carbohydrates increases the risk of diabetes. Pizza should not be consumed with French fries because proteins require - a much different stomach environment than starches for proper digestion
A change in behavior such as {behavior_change} is often associated with {a_medical_condition} because {reason_for_condition}	A change in behavior such as becoming more sedentary is - often associated with obesity because less activity leads to less calorie burning. A change in behavior such as no longer drinking coffee is often - associated with diminished insomnia because less caffeine equals improved sleep.
When severe symptoms like {a_symptom} for a {a_medical_condition} shows up, immediately one should perform {an_action}	When severe symptoms like confusion or disorientation for heatstroke show up, immediately - one should perform cooling actions, such as applying cooling towels. When severe symptoms like unconsciousness for a heart attack show up, immediately - one should call 911 and perform CPR while awaiting help.
People often do {an_activity} before going to bed in night to prevent risk of {disease}. This is because {reason_for_activity}	People often do reading before going to bed in night to prevent risk of insomnia. - This is because doing some light reading helps lull you to sleep. People often do teeth brushing before going to bed in night to prevent risk - of tooth decay. This is because brushing removes cavity-causing plaque from teeth.

Table 8.1: Examples from our dataset. Each template has two corresponding sentences. [concept] is a commonsense knowledge concept, [concept] is a health and well-being concept, and [text] represents the explanation and [text] represents a qualifier. We show two sentences each for a template.

across *commonsense* and *health and well-being* domain. An example is shown in figure **??**. Each template has atleast one *concept* slot, one from each domain (*people eating leafy vegetables* from commonsense domain and *blood clot* from the medical domain in the example shown in the figure). A qualifier slot optionally specifies *how* the concept in a domain interacts with the concept from other domain. In the example in figure **??**, *higher risk* indicates the qualifier. The template also includes an optional *explanation* slot that specifies in free-form text how leafy vegetable intake is connected to blood clots.


Instructions (click to expand/collapse)

Thanks for participating in this HIT! Please read the following instructions *carefully*.

GOAL : Our goal is to expand a simple PATTERN with fillers to complete, factual and knowledgeable SENTENCE. Each filler represents the nature of the information that needs to be filled

Some ***correct*** examples:

- **PATTERN**: **{Company_A}** acquired **{Company_B}** in **{year}** for **{amount_of_money}**
- **SENTENCE**: Facebook acquired Instagram in 2012 for $1 billion

- **PATTERN**: **{Person_with_habit}** has a higher risk of **{disease}** because **{reason_for_risk}**
- **SENTENCE**: Person who smokes often has a higher risk of cancer because harmful chemicals can cause DNA damage

In these examples, each **PATTERN** is expanded into a **SENTENCE** using real-world knowledge.

Please ensure that your **SENTENCE** is specific, self-contained and not general.

Some ***wrong*** examples :

- **PATTERN**: **{Company_A}** acquired **{Company_B}** in **{year}** for **{amount_of_money}**
- **SENTENCE**: A big company acquired a smaller company in 2012 for $1 billion

  In this example, it is not clear what the big and small companies are

- **PATTERN**: **{Person_with_habit}** has a higher risk of **{disease}** because **{reason_for_risk}**
- **SENTENCE**: Person who smokes often has a higher risk of cancer because smoking causes cancer

  In this example, the reason is just repeated from the rest of the sentence

Remember:

- Each **PATTERN** should be expanded into a **SENTENCE**
- Each **PATTERN** has typically ***2-3 open*** blanks.
- Please refrain from writing generic **SENTENCES** and opinions
- Please refer to the right and wrong ***examples*** carefully before attempting the HIT
- We highly recommend you to consult resources such as Wikihow, Healthline, CDC, WedMD for writing the **SENTENCES**

Figure 8.3: The mechanical turk interface for data collection. The human annotators were given instructors and examples to introduce them to the task.

8.2.1 Task Setup

To collect our dataset, we use amazon mechanical turk platform [1]. The interface is shown in figure 8.3. Each datapoint took ∼120 seconds to annotate, and we paid an average of $15 per

[1]`https://www.mturk.com/`

Category	Statistic
#sent len	14.57
#datapoints	6909
# avg slots per template	2.4

Table 8.2: Dataset Statistics

Template	Output
The first blank is **person_at_location**. The second blank is **higher/lower**. The third blank is **disease**. The fourth blank is a **reason_for_risk**. **[MASK]** has a **[MASK]** risk of **[MASK]** because **[MASK]**	Person who lives in a city has a higher risk of depression because of stress due to noise

Table 8.3: Task Setup. Each concept category is given as a prompt to the input and the slots are represented via the `[MASK]` token. The task for PLM is to generate the *output*

hour. Additionally, we used a filtering step to select master annotators with an approval rate of more than 90%. All the turkers were given specific instructions to input only factual information and not opinionated statements. Specifically, the turkers were instructed to use the following sources: *CDC*[2], *WebMD*[3], *Healthline*[4] and *Mayo Clinic*[5]. The annotators were instructed to give a template, and atleast two corresponding sentences that matches the template. The statistics of the data are shown in table 8.2 and some qualitative examples from the dataset are given in the table 8.1. Overall, our dataset contains about 7000 template-sentence pairs with about 3600 unique templates.

Once the templates are collected, we post-process the data to validate that we do not have any identifying information like proper names. We then create a standard 70/10/20 train, validation test split with this dataset.

8.3 *Prompt Template-Filling Framework*

Early NLP systems have often relied with templated rule-based systems [3, 22, 34, 146] due to their simplistic nature. Compared to machine learning methods, they were often rigid [204]. Despite their rigidity, template based systems are often easy to comprehend, and lend themselves to easily incorporate domain knowledge [29]. Our goal is to combine the strengths of both template-based systems and recent pretrained PLM models for the task of cross-domain reasoning.

In our *prompt-template-filling* formulation, we setup the template filling task as a prompt-

[2]https://www.cdc.gov/
[3]https://www.webmd.com/
[4]https://www.healthline.com/
[5]https://www.mayoclinic.org/

tuning task inspired by the recent advances in prompt-tuning. Prompt-based approaches have achieved state-of-the-art performance in several few-shot learning experiments [23, 58, 97]. Table **??** shows an example of our task setup. The template filling task takes an input template x, containing one or more template slots represented as spans (`[MASK]`) as input, and produce an expanded sequences y as output. Given a template x, the task is to model $p(y|x)$. Since there could be multiple sentences in the output y, we concatenate these sentences as one for model training.

In comparison to approaches such as Donahue et al. [45], our approach does not strictly enforce that that sentences only fill missing spans of text. Rather, the expanded sentences can have additional modifications. For instance, for the following input template - **{`person_at_location`}** has a **{`higher/lower`}** risk of **{`disease`}** because **{`reason_for_risk`}**, a valid sentence is *person who lives in the city has a higher risk of depression due to noise*. In this example, the word *because* does not match the output sentence phrase "*due to*" but it is considered a valid output for the template.

8.3.1 Training

Given a template $x \in \mathcal{X}$ and its corresponding expansion $y \in \mathcal{Y}$, we can train any sequence-to-sequence model that models $p_\theta(y|x)$. Towards this, we use a pretrained sequence-to-sequence model $\mathcal{M}$ to estimate the filled template y for an input x. We model the conditional distribution $p_\theta(y \mid x)$ parameterized by θ: as

$$p_\theta(y \mid x) = \prod_{k=1}^{M} p_\theta(y^k \mid x, y^1, .., y^{k-1})$$

where M is the length of y.

8.4 Experiments

In this section, we describe the experimental setup, baselines for our approach. Since our approach is agnostic to the pretrained encoder-decoder architecture type, we perform experiments on several state-of-the-art PLM models.

8.4.1 Experimental Setup

Metrics : We use the following evaluation metrics for comparison against baselines: (i) `ROUGE` [102] and (ii) `BERTSCORE` [208]. N-gram metrics such as `ROUGE` are known to be limited, specifically for reasoning tasks. To mitigate this, we use `BERTSCORE`, which uses the similarity score between the reference and generated output using conceptual embeddings from `BERT` [42] model, which correlates better towards human judgements.

To perform the evaluation, we compare the generated sentence for the template against the gold annotations in our dataset. We remove the template words from the output and only compare the slot filler concepts to avoid score inflation due to copying. All the experiments were performed on a cluster of 8 NVIDIA V100 GPUs for a total of 32 GPU hours.

Model	Type	ROUGE-1	ROUGE-2	ROUGE-L	BERTSCORE
BERT-BASE	[MASK]	5.33	0.72	4.94	-0.39*
BERT-LARGE	[MASK]	8.05	0.63	7.85	-0.27*
T5-BASE	SPL TOKEN	14.00	2.71	12.58	2.2
T5-BASE	PROMPT	14.01	2.60	12.57	6.1
T5-LARGE	SPL TOKEN	13.74	3.11	13.74	4.8
T5-LARGE	PROMPT	16.74	4.33	15.37	6.7
BART-BASE	SPL TOKEN	17.17	5.60	16.32	3.9
BART-BASE	PROMPT	18.89	5.87	17.96	6.3
BART-LARGE	SPL TOKEN	19.54	7.57	18.49	7.0
BART-LARGE	PROMPT	20.58	7.32	19.58	7.6

Table 8.4: Overview of the results compared to baselines. The table shows that BART-BASE performs better than T5-BASE model and BART-LARGE outperforms both. Both in terms of ROUGE and BERTSCORE, we also observe that our PROMPT approach outperforms SPL TOKEN approach. * - a negative score in BERTSCORE implies that the reference was dissimilar to the generated output.

8.4.2 Models

We follow the same experimental settings across the baseline and our approach for all the models. We initialize all the models with their pretrained weights. We use commonly used encoder-decoder architectures for our experiments - BART-BASE, BART-LARGE, T5-BASE and T5-LARGE. The model settings are given below:

- BART-BASE: This pretrained encoder-decoder transformer architecture is based on Lewis et al. [100]. It consists of 12 transformer layers each with 768 hidden size, 16 attention heads and overall with 139M parameters.
- BART-LARGE: Larger version of BART-BASE, consisting of 24 transformer layers, 1024 hidden size, 16 heads and 406M parameters.
- T5-BASE: The T5 model is also a transformer encoder-decoder model based on Raffel et al. [139] with 220M parameters with 12-layers each with 768 hidden-state, 3072 feed-forward hidden-state and 12 attention heads.
- T5-LARGE: T5-Large model version comprises of 770M parameters with 24-layers with 1024 hidden-state, 4096 feed-forward hidden-state and 16 attention heads [6].

8.4.3 Baseline Methods

- BERT [MASK]: To understand whether pretrained models contain the knowledge already, we try a masked language modeling baseline [42] where we query the template using

[6]We use the implementation of all the models from the huggingface [197] repository

[MASK] tokens[7].

- SPL TOKEN: In this approach, we use the special token approach (SPL TOKEN) [45], where we indicate the start and end of each template slot in the input and generate the output sentence

Table 8.3 shows the baseline setup of the models for our task with a corresponding example.

8.4.4 Results

The results across various pretrained encoder-decoder approaches are shown in table 8.4. In this table, we see that on average, BART models perform better than T5 models on average. We hypothesize this might be an effect of their pretraining task choices and corresponding datasets. We also observe that PROMPT based models outperform the SPL TOKEN based approach. For all of the models and baselines, we used the greedy decoding strategy.

Firstly, we find that [MASK] approach does not perform competitively compared to fine-tuning, showing that pretrained models are not easily amenable towards template filling without finetuning. Across all the experiments, we found that the PROMPT approach outperforms SPL TOKEN approach across both ROUGE and BERTSCORE scores for all models.

8.5 Factual Correctness Evaluation

To further assess the quality of generated output, we perform additional factuality evaluation towards our best performing models - SPL TOKEN and PROMPT approach using BART-LARGE. Towards this we use the FACTCC factuality metric [**?**], which uses entailment classification to predict a binary factuality label between the source document and generated output.

Computing factuality using FACTCC metric requires an input source document; (i.e.) the generated output is compared against the source document for factual correctness. For this evaluation setup, we augmented each generated output y with a source document. Towards this, we use a large scale retrieval corpus based on **?**], and retrieve the top similar document D [**?**] to a generated template expansion. Using the (D, y) pairs, we compute the factual correctness of our best performing models. From the table 8.5, we observe that our PROMPT approach outperforms the SPL TOKEN approach for factual correctness by $\sim$14 points in accuracy.

Additionally, we also perform human evaluation of factual correctness. For this experiment, three human judges annotated 100 unique samples for *correctness* - that indicates how many samples were correct from a human perspective. We used our best performing BART-BASE-PROMPT model for this evaluation. In this experiment, a sentence generated by the model for a given template was given to each human judge and they were asked to evaluate whether the sentence was correct, given the template. The inter-annotator agreement on graph correctness was substantial with a Fleiss' Kappa score [56] of 0.73. From our evaluation, we found that human judges rated about 69% of the sentences to be correct given a template, comparable to our

[7]Since mask tokens in BERT needs to be predetermined for this experiment, we try different variations with number of [MASK] tokens and report the best results.

Model	Type	FACTCC
BART-LARGE	SPL TOKEN	65.27
BART-LARGE	PROMPT	79.88

Table 8.5: Factual consistency results. In this experiment, we show that our PROMPT approach the SPL TOKEN approach in terms of factuality metric FACTCC, showing its relative effectiveness

FACTCC evaluation metric numbers. Both the automated and human evaluation suggests that our PROMPT based approach has better factual consistency.

8.6 Error Analysis

In this section, we analyze in detail how well language models perform cross-domain reasoning. Automated metrics such as ROUGE are restrictive in terms of understanding the reasoning abilities and we complement our automated evaluation with manual error analysis. For this analysis, we randomly select 100 samples from the validation set predictions where the ROUGE scores were low. We observe the following categories of errors that language models exhibit. Table 8.6 shows the common type of errors and a corresponding example for each type.

Error Type	Template	Gold Answer	Generated Answer
Correct but not in gold	Children who are exposed to {environmental_factor} are often at a higher risk for {disease} because {reason}	Children exposed to second hand smoke are at a higher risk for lung disease because of breathing in the cigarette smoke	Children who are exposed to lead paint are often at a higher risk for kidney failure because lead causes kidney damage
Wrong commonsense concept	People with {certain_socioeconomic_condiiton} are at higher risk of {disease} as they are more exposed to {reason}	Person who often inhales a lot of dirt is at a higher risk of hay fever because of allergen content.	Person who often does less medications is at a higher risk of hay fever because of the drug can help clear it up
Generic Explanation	When people with {certain_co-morbidities} shows {symptoms}, this is because of {reason_for_patient_state}	When people with diabetes shows lethargy, this is because of high glucose levels.	When people with heart disease shows chest pain, this is because of the strain on the heart
Factually Incorrect	People with a {health_condition} should do {an_activity} because {reason}	People with a cardiovascular disease should do exercise since exercise burns excess fat	People with a flu diagnosis should do exercise

Table 8.6: Error Analysis based on the BART-BASE-PROMPT model. We select 100 samples from the validation set and each row shows an example of each class of error.

Error Type - Correct but not in gold (17%) : In several cases, we observe that the output produced by the language models are correct despite not matching the gold answer. This phenomenon is evident when the input template contains multiple possible answers. While the gold answer in the example shown in Table 8.6 (first row) fills the template using **smoking**, the language models generates an answer that relates to **kidney damage**. While correct, the automated metrics score this answer lower.

Error Type - Wrong commonsense concept (8%) : In this category of error, the model generates the wrong specification for the given slot. For instance (second row in table 8.6), the model mistakenly assumes `person taking less medication` as a `socioeconomic condition`.

Error Type - Generic Explanation (53%): In several cases, the model resorts to generic explanation that are *obvious*. A generic explanation repeats the same information as the rest of sentence as an explanation, thereby not providing any new information compared to the rest of the sentence. In the example shown in Table 8.6 (row 3), the explanation `because of the strain of the heart` is already clear from the concept `chest pain`.

Error Type - Factually Incorrect (22%) : Factual correctness is one of the biggest challenges in NLP applications [125, 130]. The incorrect factual information is also acute for cross-domain reasoning applications as well. As shown in the example (row 4 in table 8.6), the model incorrectly generates that `people with flu diagnosis` should do `exercise`.

Our errors highlight the difficulty of the task for language models. This leaves room for several research questions that requires future work. Overall, cross-domain reasoning is still an uphill task for language models with promising directions.

8.7 Related Work

Knowledge Bases : Knowledge Bases (KBs) have been the predominant approach to perform cross-domain reasoning in the past. Some of the prominent cross domain knowledge bases include DBPedia [117], YAGO [172] and NELL [120]. Most of these knowledge bases despite being cross-domain, the focus is primarily on the encyclopedic knowledge. In our work, we focus on ability of PLM for cross-domain reasoning, which can be viewed as a complementary approach to KBs.

Language Models for Knowledge Generation: Using pretrained language models to generate knowledge has been studied for commonsense reasoning tasks. [20, 21, 152, 162]. Our work closely aligns with Bosselut et al. [20, 21]. Compared to Bosselut et al. [20], our focus in this work to extend this line of work from only commonsense reasoning to perform reasoning cross domain.

Language Model Infilling : Our work also closely relates to the language model infilling work in the literature such as Fedus et al. [53] and [45]. Compared to these works which only look at cloze-test infilling, our work aims to expand templates that cannot be directly modeled as cloze-style. Our work is also related to the story generation efforts such as Fan et al. [51], Ippolito et al. [78], Rashkin et al. [144], Yao et al. [203] but our application differs from them in that we focus on cross-domain reasoning instead of content planning for stories.

There has also been efforts to transfer knowledge cross-domain via transfer learning [41, 118, 195] but our work focuses on cross-domain reasoning in the same input sample unlike transfer learning based approaches.

8.8 Conclusion and Future Work

In this chapter, we present a novel *prompt-template-filling* approach that adapts language models to perform cross-domain reasoning via prompting. To study this, we present a dataset via a use-case of reasoning across *commonsense* and *health and well-being* domain. Through both automated and human metrics, we find that there is immense room for progress towards improving language models' capability for cross-domain reasoning. For future work, we want to extend this work for multiple other cross-domain scenarios and understand the nature of cross-domain reasoning in depth.

Chapter 9

Conclusion

9.1 Contributions and Insights

9.1.1 Goals of Explanations

In our first chapter, we established some fundamental goals towards building explainable NLP systems. Our approach takes a holistic view of their stakeholders' utility and simplicity requirements. It helps establish how we should view explanations Depending on the end task, developing explainable systems would require the developers to understand stakeholder requirements, and formulate them as machine learning tasks using utility and simplicity terms.

9.1.2 Explanations can help Downstream Tasks

In the further chapters, we saw two examples of how explanations can help downstream tasks. In the first task, we have presented a static explanation system that can explain the effects of perturbation and how that can help predict the task of predicting perturbation. QUARTET not only predicts meaningful explanations, but also achieves a new state-of-the-art on the end-task itself, leading to an interesting finding that models can make better predictions when forced to explain. In the second application, CURIE presents an LM approach that : (i) is effective at generating *st*-reasoning graphs, validated by automated metrics and human evaluations, (ii) improves performance on two downstream tasks by simply augmenting their input with the generated *st* graphs. Our dynamic explanation framework supports recursively querying for any node in the *st*-graph. In essence, we show that explanations **can** help models reason better for end tasks.

9.1.3 Explanations can help Humans

We also show in our next chapter how humans can benefit from explainable NLP systems. Towards this, we take the idea of using inference graphs for defeasible inference and scales up its usability by automatically generating and augmenting them to a downstream defeasible task that both humans and machines are known to find difficult. Humans perform significantly better (20% absolute improvement) across diverse defeasible datasets and overwhelmingly attribute their success to the explanations in form of graphs. While we establish that humans are helped by these graphs, a further investigation on how (and if) the graphs reinforced their beliefs, and what additional information in the graphs was beneficial to their understanding is essential.

9.1.4 Explanations can be incorporated into Models

We also proposed a self-explaining model that provides interpretability at both local and global levels by using architectural changes alone. This approach does not involve any additional data collection. Our method uses human understandable phrasal concepts for interpretability and it is trainable end-to-end. This opens an exciting research direction for building inherently interpretable models text classification. This approach provides alternative tools to explore models' inner working without relying on specialized tools post-hoc.

9.1.5 Recursive Descent for Explanation

In this chapter, we propose a framework to bridge the gap between theory and practice towards building explainable NLP systems. This framework embraces the relatively dynamic nature of explanations and uses depth to characterize how explanations can be expanded based on any user-given criteria.

9.1.6 Prompt Template Filling for Explainable Reasoning

In the next chapter, we presented a novel *prompt-template-filling* approach that implements *recursive descent* using the idea of templates. This approach adapts language models to perform cross-domain reasoning via prompting. To study this, we present a dataset via a use-case of reasoning across *commonsense* and *health and well-being* domain. We show in this work how templates can control the nature of explanations for a reasoning task and use text-to-text generation models to achieve the same.

9.2 Limitations and Future Work

9.2.1 Reviewable Explanations

One of the important aspects of explanations is that they need to be accompanied by evidence that can be attributed to specific sources - be it knowledge bases or retrieved documents. Towards this, we want to emphasize the importance of evidence as a fundamental aspect of an explanation and how in the future, we should aspire to build explainable systems that can integrate evidence as a fundamental aspect. In short term, we want to explore retrieval + reasoning architectures [67, 88] towards this goal.

9.2.2 Explanations beyond Input and Training Data

In this thesis, we explored two different styles of explanations - input level and training data level. There is also ample evidence in literature that pretraining using large-scale data [42, 139] contributes a lot to model performance. Explanation methods that trace the model reasoning to pretraining is not adequately addressed in the literature.

9.2.3 Faithfulness

Another important open question for explanations is how well they reflect the model's decision making process. Model functions such as attention [81, 192] and gradients [2] have shown to be unfaithful. Instance based approaches such as influence functions [68, 92] are shown to be faithful yet computationally limiting. While this thesis does not explore the faithfulness aspect in depth, we hope to develop explanation and evaluation methods for faithfulness.

Bibliography

[1] Peter Achinstein. *The nature of explanation*. Oxford University Press on Demand, 1983. 7.4

[2] J. Adebayo, M. Muelly, I. Liccardi, and Been Kim. Debugging tests for model explanations. *ArXiv*, abs/2011.05429, 2020. 9.2.3

[3] Eugene Agichtein and L. Gravano. Extracting relations from large plain-text collections. 1999. 8.3

[4] Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and D. Klein. Neural module networks. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 39–48, 2016. 2.2.1

[5] Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C. Lawrence Zitnick, and Devi Parikh. Vqa: Visual question answering. *ICCV*, 2015. 3.2

[6] Akari Asai and Hannaneh Hajishirzi. Logic-guided data augmentation and regularization for consistent question answering. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5642–5650, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.499. URL `https://www.aclweb.org/anthology/2020.acl-main.499`. 3.5

[7] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In *ICLR*, 2014. 2.2.1, 6.2

[8] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015*, 2015. 4.3.1

[9] Alan Baker. Simplicity. In Edward N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, winter 2016 edition, 2016. 2.2.3

[10] Barr and Feigenbaum Edward A. Avron. The handbook of artificial intelligence. 1981. 2.2.2

[11] Jonathan Baxter. A bayesian/information theoretic model of learning to learn via multiple task sampling. *Machine Learning*, 28(1):7–39, Jul 1997. ISSN 1573-0565. doi: 10.1023/A:1007327622663. URL `https://doi.org/10.1023/A:1007327622663`. 3.4

[12] J. Bentahar, B. Moulin, and M. Bélanger. A taxonomy of argumentation models used for knowledge representation. *Artificial Intelligence Review*, 33:211–259, 2010. 5.1

[13] D. Bernoulli. Exposition of a new theory on the measurement of risk. 1954. 7.1

[14] Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. Piqa: Reasoning about physical commonsense in natural language. In *Thirty-Fourth AAAI Conference on Artificial Intelligence*, 2020. 7.3

[15] Yonatan Bisk, Rowan Zellers, Ronan LeBras, Jianfeng Gao, and Yejin Choi. Piqa: Reasoning about physical commonsense in natural language. In *AAAI*, pages 7432–7439, 2020. 4.1

[16] Olivier Bodenreider. {The Unified Medical Language System (UMLS): integrating biomedical terminology}. *Nucleic Acids Research*, 32(suppl_1):D267–D270, 01 2004. ISSN 0305-1048. doi: 10.1093/nar/gkh061. URL `https://doi.org/10.1093/nar/gkh061`. 8.2

[17] Mario Bollini, Stefanie Tellex, Tyler Thompson, Nicholas Roy, and Daniela Rus. Interpreting and executing recipes with a cooking robot. In *ISER*, 2012. 3.1

[18] Antoine Bosselut, Omer Levy, Ari Holtzman, Corin Ennis, Dieter Fox, and Yejin Choi. Simulating action dynamics with neural process networks. *ICLR*, 2018. 3.1, 3.2

[19] Antoine Bosselut, Hannah Rashkin, Maarten Sap, Chaitanya Malaviya, Asli Celikyilmaz, and Yejin Choi. COMET: Commonsense transformers for automatic knowledge graph construction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4762–4779, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1470. URL `https://aclanthology.org/P19-1470`. 8.2

[20] Antoine Bosselut, Hannah Rashkin, Maarten Sap, Chaitanya Malaviya, Asli Çelikyilmaz, and Yejin Choi. Comet: Commonsense transformers for automatic knowledge graph construction. In *ACL*, 2019. 8.7

[21] Antoine Bosselut, Ronan Le Bras, and Yejin Choi. Dynamic neuro-symbolic knowledge graph construction for zero-shot commonsense question answering. In *Proceedings of the 35th AAAI Conference on Artificial Intelligence (AAAI)*, 2021. 8.7

[22] S. Brin. Extracting patterns and relations from the world wide web. In *WebDB*, 1998. 8.3

[23] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc., 2020. URL `https://proceedings.neurips.cc/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf`. 8.1, 8.3

[24] Oana-Maria Camburu, Tim Rocktäschel, Thomas Lukasiewicz, and Phil Blunsom. e-snli: Natural language inference with natural language explanations. In *NeurIPS*, 2018. 2.2.1,

3.2, 6.2

[25] Dallas Card, Michael Zhang, and Noah A Smith. Deep weighted averaging classifiers. In *Proceedings of the Conference on Fairness, Accountability, and Transparency*, pages 369–378, 2019. 6.2

[26] Rich Caruana, Yin Lou, Johannes Gehrke, Paul Koch, Marc Sturm, and Noemie Elhadad. Intelligible models for healthcare: Predicting pneumonia risk and hospital 30-day readmission. In *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1721–1730, 2015. 6.1

[27] Eugene Charniak and D. McDermott. Introduction to artificial intelligence. In *Addison-Wesley series in computer science*, 1986. 2.2.2

[28] Chaofan Chen, Oscar Li, Daniel Tao, Alina Barnett, Cynthia Rudin, and Jonathan K Su. This looks like that: deep learning for interpretable image recognition. In *Advances in neural information processing systems*, pages 8930–8941, 2019. 6.1

[29] Laura Chiticariu, Yunyao Li, and Frederick R. Reiss. Rule-based information extraction is dead! long live rule-based information extraction systems! In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 827–832, Seattle, Washington, USA, October 2013. Association for Computational Linguistics. URL `https://aclanthology.org/D13-1079`. 8.3

[30] Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D Manning. What does bert look at? an analysis of bert's attention. *arXiv preprint arXiv:1906.04341*, 2019. 3.4

[31] P. Clark, Oyvind Tafjord, and Kyle Richardson. Transformers as soft reasoners over language. In *IJCAI*, 2020. 2.2.2, 7.1

[32] Peter Clark, Oyvind Tafjord, and Kyle Richardson. Transformers as soft reasoners over language. In Christian Bessiere, editor, *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, pages 3882–3890. International Joint Conferences on Artificial Intelligence Organization, 2020. 2.2.1

[33] William W. Cohen, Haitian Sun, R. Hofer, and M. Siegler. Scalable neural methods for reasoning with a symbolic knowledge base. *ArXiv*, abs/2002.06115, 2020. 2.2.2

[34] M. Craven, Dan DiPasquo, Dayne Freitag, A. McCallum, Tom Michael Mitchell, K. Nigam, and Seán Slattery. Learning to construct knowledge bases from the world wide web. *Artif. Intell.*, 118:69–113, 2000. 8.3

[35] Danilo Croce, Daniele Rossini, and Roberto Basili. Auditing deep learning processes through kernel-based explanatory models. 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)*, pages 4028–4037, 2019. 6.2, 6.4.4

[36] Yin Cui, Guandao Yang, Andreas Veit, Xun Huang, and Serge J. Belongie. Learning to evaluate image captioning. *CVPR*, 2018. 3.2

[37] Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc V Le, and Ruslan Salakhutdinov. Transformer-xl: Attentive language models beyond a fixed-length context. *arXiv*

preprint arXiv:1901.02860, 2019. 6.4.2

[38] Bhavana Dalvi, Lifu Huang, Niket Tandon, Wen-tau Yih, and Peter Clark. Tracking state changes in procedural text: A challenge dataset and models for process comprehension. *NAACL*, 2018. 3.2, 3.3

[39] Fred D. Davis. Perceived usefulness, perceived ease of use, and user acceptance of information technology. *MIS Q.*, 13:319–340, 1989. 6.4.4

[40] Maria De-Arteaga, Alexey Romanov, Hanna Wallach, Jennifer Chayes, Christian Borgs, Alexandra Chouldechova, Sahin Geyik, Krishnaram Kenthapadi, and Adam Tauman Kalai. Bias in bios: A case study of semantic representation bias in a high-stakes setting. In *Proceedings of the Conference on Fairness, Accountability, and Transparency*, FAT* '19, page 120–128, New York, NY, USA, 2019. Association for Computing Machinery. ISBN 9781450361255. doi: 10.1145/3287560.3287572. URL `https://doi.org/10.1145/3287560.3287572`. 6.1

[41] Yang Deng, Ying Shen, Min Yang, Yaliang Li, Nan Du, Wei Fan, and Kai Lei. Knowledge as a bridge: Improving cross-domain answer selection with external knowledge. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3295–3305, Santa Fe, New Mexico, USA, August 2018. Association for Computational Linguistics. URL `https://aclanthology.org/C18-1279`. 8.7

[42] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *NAACL 2019*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423. URL `https://www.aclweb.org/anthology/N19-1423`. 3.2, 3.4, 3.5, 4.4.1, 6.4.2, 8.4.1, 8.4.3, 9.2.2

[43] Jay DeYoung, Sarthak Jain, Nazneen Fatema Rajani, Eric Lehman, Caiming Xiong, Richard Socher, and Byron C. Wallace. ERASER: A benchmark to evaluate rationalized NLP models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4443–4458, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.408. URL `https://www.aclweb.org/anthology/2020.acl-main.408`. 2.2.1

[44] Bhuwan Dhingra, Manzil Zaheer, Vidhisha Balachandran, Graham Neubig, Ruslan Salakhutdinov, and William W. Cohen. Differentiable reasoning over a virtual knowledge base. In *International Conference on Learning Representations*, 2020. URL `https://openreview.net/forum?id=SJxstlHFPH`. 2.2.2, 6.3.4

[45] Chris Donahue, Mina Lee, and Percy Liang. Enabling language models to fill in the blanks. In *ACL*, 2020. 8.3, 8.4.3, 8.7

[46] Finale Doshi-Velez and Been Kim. Towards a rigorous science of interpretable machine learning. *arXiv preprint arXiv:1702.08608*, 2017. 2.2.2, 6.1, 6.4.4, 6.5

[47] Finale Doshi-Velez, Mason Kortz, Ryan Budish, Chris Bavitz, Sam Gershman, D. O'Brien, Stuart Schieber, J. Waldo, D. Weinberger, and Alexandra Wood. Accountability of ai under the law: The role of explanation. *ArXiv*, abs/1711.01134, 2017. 6.4.4

[48] Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. Drop: A reading comprehension benchmark requiring discrete reasoning over paragraphs. *NAACL*, 2019. 3.2

[49] John Dunlosky, Katherine A. Rawson, Elizabeth Marsh, Mitchell J. Nathan, and Daniel T. Willingham. Improving students' learning with effective learning techniques: Promising directions from cognitive and educational psychology. *Psychological science*, 2013. 3.6.2

[50] J. Ebrahimi, Anyi Rao, Daniel Lowd, and D. Dou. Hotflip: White-box adversarial examples for text classification. In *ACL*, 2018. 2.2.2

[51] Angela Fan, Mike Lewis, and Yann Dauphin. Hierarchical neural story generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 889–898, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-1082. URL `https://aclanthology.org/P18-1082`. 8.7

[52] P. Farquhar. A fractional hypercube decomposition theorem for multiattribute utility functions. *Oper. Res.*, 23:941–967, 1975. 2.2.2

[53] William Fedus, Ian Goodfellow, and Andrew M. Dai. MaskGAN: Better text generation via filling in the ______. In *International Conference on Learning Representations*, 2018. URL `https://openreview.net/forum?id=ByOExmWAb`. 8.7

[54] Shi Feng, Eric Wallace, Alvin Grissom II, Mohit Iyyer, Pedro Rodriguez, and Jordan Boyd-Graber. Pathologies of neural models make interpretations difficult. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3719–3728, Brussels, Belgium, October-November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1407. 2.2.2, 6.2

[55] Yanlin Feng, Xinyue Chen, Bill Yuchen Lin, Peifeng Wang, Jun Yan, and Xiang Ren. Scalable multi-hop relational reasoning for knowledge-aware question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1295–1309, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.99. URL `https://aclanthology.org/2020.emnlp-main.99`. 8.1

[56] Joseph L Fleiss and Jacob Cohen. The equivalence of weighted kappa and the intraclass correlation coefficient as measures of reliability. *Educational and psychological measurement*, 33(3):613–619, 1973. 8.5

[57] Kenneth D. Forbus. Qualitative process theory. *Artificial Intelligence*, 24:85–168, 1984. 3.2

[58] Tianyu Gao, Adam Fisch, and Danqi Chen. Making pre-trained language models better few-shot learners. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3816–3830, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-long.295. URL `https://aclanthology.org/2021.acl-long.295`. 8.3

[59] Shalini Ghosh, Giedrius Burachas, Arijit Ray, and Avi Ziskind. Generating natural language explanations for vqa using scene graphs and visual attention. *arXiv preprint arXiv:1902.05715*, 2019. 3.2

[60] G. Gigerenzer and Henry Brighton. Homo heuristicus: Why biased minds make better inferences. *Topics in cognitive science*, 1 1:107–43, 2009. 7.1

[61] Yoav Goldberg. Neural network methods for natural language processing. *Synthesis Lectures on HLT*, 2017. 3.4

[62] H. Grice. Logic and conversation syntax and semantics. 1975. 4.1

[63] Nitish Gupta, Kevin Lin, Dan Roth, Sameer Singh, and Matt Gardner. Neural module networks for reasoning over text. In *International Conference on Learning Representations*, 2020. URL `https://openreview.net/forum?id=SygWvAVFPr`. 2.2.1

[64] Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel Bowman, and Noah A. Smith. Annotation artifacts in natural language inference data. In *NAACL 201*, pages 107–112, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-2017. URL `https://www.aclweb.org/anthology/N18-2017`. 6.1

[65] Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel Bowman, and Noah A Smith. Annotation artifacts in natural language inference data. In *NAACL*, pages 107–112, 2018. 3.2, 3.5

[66] Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A Smith. Don't stop pretraining: Adapt language models to domains and tasks. *arXiv preprint arXiv:2004.10964*, 2020. 4.4

[67] Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. Realm: Retrieval-augmented language model pre-training. *arXiv preprint arXiv:2002.08909*, 2020. 6.3.4, 9.2.1

[68] Xiaochuang Han, Byron C. Wallace, and Yulia Tsvetkov. Explaining black box predictions and unveiling data artifacts through influence functions. In *ACL*, 2020. 2.2.2, 6.2, 6.3.4, 6.4.4, 9.2.3

[69] Sally Haslanger. What is a (social) structural explanation? *Philosophical Studies*, 173(1): 113–130, 2016. 2.2.1

[70] Carl G Hempel and Paul Oppenheim. Studies in the logic of explanation. *Philosophy of science*, 15(2):135–175, 1948. 7.4

[71] Mikael Henaff, Jason Weston, Arthur Szlam, Antoine Bordes, and Yann LeCun. Tracking the world state with recurrent entity networks. In *ICLR*, 2017. 3.1, 3.2

[72] Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. Teaching machines to read and comprehend. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 1693–1701. Curran Associates, Inc., 2015. 6.2

[73] John Hewitt and Percy Liang. Designing and interpreting probes with control tasks.

In *Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2019. 2.2.2

[74] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997. 4.3.1

[75] John H Holland, Keith J Holyoak, Richard E Nisbett, and Paul R Thagard. *Induction: Processes of inference, learning, and discovery*. MIT press, 1989. 7.4

[76] Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The curious case of neural text degeneration. *arXiv preprint arXiv:1904.09751*, 2019. 4.2.4

[77] Naoya Inoue, Pontus Stenetorp, and Kentaro Inui. R4c: A benchmark for evaluating rc systems to get the right answer for the right reason. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6740–6750, 2020. 2.2.1

[78] Daphne Ippolito, David Grangier, Chris Callison-Burch, and Douglas Eck. Unsupervised hierarchical story infilling. In *Proceedings of the First Workshop on Narrative Understanding*, pages 37–43, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/W19-2405. URL `https://aclanthology.org/W19-2405`. 8.7

[79] D. Israel. What's wrong with non-monotonic logic? In *AAAI*, 1980. 5.1

[80] Alon Jacovi and Yoav Goldberg. Towards faithfully interpretable NLP systems: How should we define and evaluate faithfulness? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4198–4205, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.386. URL `https://www.aclweb.org/anthology/2020.acl-main.386`. 3

[81] Sarthak Jain and Byron C. Wallace. Attention is not Explanation. 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)*, pages 3543–3556, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1357. 6.1, 6.2, 9.2.3

[82] Peter Jansen, Elizabeth Wainwright, Steven Marmorstein, and Clayton Morrison. Worldtree: A corpus of explanation graphs for elementary science questions supporting multi-hop inference. In *LREC 2018*, 2018. 3.2

[83] H Jeffreys. Theory of probability, (clarendon press, oxford). 1961. 2.2.3

[84] Harsh Jhamtani and Peter Clark. Learning to explain: Datasets and models for identifying valid reasoning chains in multihop question-answering. *EMNLP*, 2020. 2.2.1

[85] Haozhe Ji, Pei Ke, Shaohan Huang, Furu Wei, Xiaoyan Zhu, and Minlie Huang. Language generation with multi-hop reasoning on commonsense knowledge graph. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 725–736, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.54. URL `https://aclanthology.org/2020.emnlp-main.54`. 8.1

[86] Yichen Jiang and Mohit Bansal. Self-assembling modular networks for interpretable multi-

hop reasoning. 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)*, pages 4474–4484, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1455. 2.2.1, 6.2

[87] Xisen Jin, Zhongyu Wei, Junyi Du, Xiangyang Xue, and Xiang Ren. Towards hierarchical importance attribution: Explaining compositional semantics for neural sequence models. In *International Conference on Learning Representations*, 2020. URL `https://openreview.net/forum?id=BkxRRkSKwr`. 2.2.2, 6.2, 6.4.4

[88] Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.550. URL `https://aclanthology.org/2020.emnlp-main.550`. 9.2.1

[89] R. Keeney, H. Raiffa, and D. Rajala. Decisions with multiple objectives: Preferences and value trade-offs. *IEEE Transactions on Systems, Man, and Cybernetics*, 9:403–403, 1979. 2.2.2

[90] Been Kim, Cynthia Rudin, and Julie A Shah. The bayesian case model: A generative approach for case-based reasoning and prototype classification. In *Advances in neural information processing systems*, pages 1952–1960, 2014. 6.1

[91] Nikita Kitaev and D. Klein. Constituency parsing with a self-attentive encoder. In *ACL*, 2018. 6.4.2

[92] Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1885–1894. JMLR. org, 2017. (document), 2.2.2, 6.2, 6.3.4, 7.1, 9.2.3

[93] Pang Wei Koh, Thao Nguyen, Yew Siang Tang, Stephen Mussmann, Emma Pierson, Been Kim, and Percy Liang. Concept bottleneck models. *NeurIPS*, 2020. 6.1, 7.1

[94] Eric Kolve, Roozbeh Mottaghi, Winson Han, Eli VanderBilt, Luca Weihs, Alvaro Herrasti, Daniel Gordon, Yuke Zhu, Abhinav Gupta, and Ali Farhadi. AI2-THOR: An Interactive 3D Environment for Visual AI. *arXiv*, 2017. 2.2.2

[95] Chih kuan Yeh, Been Kim, Sercan Arik, Chun-Liang Li, Pradeep Ravikumar, and Tomas Pfister. On completeness-aware concept-based explanations in deep neural networks. 2020. 6.1

[96] Sachin Kumar, Shuly Wintner, Noah A. Smith, and Yulia Tsvetkov. Topics to avoid: Demoting latent confounds in text classification. In *Proc. EMNLP*, pages 4151–4161, 2019. 6.1

[97] Teven Le Scao and Alexander Rush. How many data points is a prompt worth? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2627–2636, Online, June 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.naacl-main.

208. URL https://aclanthology.org/2021.naacl-main.208. 8.3

[98] Piyawat Lertvittayakumjorn and Francesca Toni. Human-grounded evaluations of explanation methods for text classification. In *EMNLP/IJCNLP*, 2019. 6.5

[99] David Lewis. *Counterfactuals*. John Wiley & Sons, 2013. 3.2

[100] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.703. URL https://aclanthology.org/2020.acl-main.703. 8.4.2

[101] Xin Li and Dan Roth. Learning question classifiers. In *Proceedings of the 19th international conference on Computational linguistics-Volume 1*, pages 1–7. Association for Computational Linguistics, 2002. 6.4.1

[102] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81, 2004. 4.3.2, 5.2.1, 8.4.1

[103] Kevin Lin, Oyvind Tafjord, Peter Clark, and Matt Gardner. Reasoning over paragraph effects in situations. In *MRQA@EMNLP*, 2019. 4.5

[104] Zachary C Lipton. The mythos of model interpretability. *Queue*, 16(3):31–57, 2018. 1, 2.2.3, 6.1

[105] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019. 6.3.2, 6.4.2

[106] Minh-Thang Luong, Hieu Pham, and Christopher D Manning. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, 2015. 4.3.1

[107] Thang Luong, Hieu Pham, and Christopher D. Manning. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, Lisbon, Portugal, September 2015. Association for Computational Linguistics. doi: 10.18653/v1/D15-1166. URL https://www.aclweb.org/anthology/D15-1166. 6.2

[108] Aman Madaan and Yiming Yang. Neural language modeling for contextualized temporal graph generation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 864–881, Online, June 2021. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/2021.naacl-main.67. 5.2.1

[109] Aman Madaan, Dheeraj Rajagopal, Yiming Yang, Abhilasha Ravichander, E. Hovy, and Shrimai Prabhumoye. Eigen: Event influence generation using pre-trained language models. *ArXiv*, abs/2010.11764, 2020. 3

[110] Christopher D. Manning and Bill MacCartney. Natural language inference. 2009. 2.2.2

[111] Cynthia Matuszek, John Cabral, M. Witbrock, and John DeOliveira. An introduction to the syntax and content of cyc. In *AAAI Spring Symposium: Formalizing and Compiling Background Knowledge and Its Applications to Knowledge Representation and Question Answering*, 2006. 8.2

[112] J. McAllister. Model selection and the multiplicity of patterns in empirical data. *Philosophy of Science*, 74:884 – 894, 2007. 2.2.3

[113] J. McCarthy. Some philosophical problems from the standpoint of artificial intelligence. *Machine intelligence*, 1981. 5.1

[114] R. Thomas McCoy, Ellie Pavlick, and Tal Linzen. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In *Proc. ACL*, 2019. 6.1

[115] Quinn McNemar. Note on the sampling error of the difference between correlated proportions or percentages. *Psychometrika*, 12(2):153–157, 1947. (document), 5.4

[116] David Alvarez Melis and Tommi Jaakkola. Towards robust interpretability with self-explaining neural networks. In *Advances in Neural Information Processing Systems*, pages 7775–7784, 2018. 6.1, 6.2, 6.3.5, 6.5

[117] Pablo Mendes, Max Jakob, and Christian Bizer. DBpedia: A multilingual cross-domain knowledge base. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 1813–1817, Istanbul, Turkey, May 2012. European Language Resources Association (ELRA). URL `http://www.lrec-conf.org/proceedings/lrec2012/pdf/570_Paper.pdf`. 8.7

[118] Sewon Min, Minjoon Seo, and Hannaneh Hajishirzi. Question answering through transfer learning from large fine-grained supervision data. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 510–517, Vancouver, Canada, July 2017. Association for Computational Linguistics. doi: 10.18653/v1/P17-2081. URL `https://aclanthology.org/P17-2081`. 8.7

[119] Sewon Min, Eric Wallace, Sameer Singh, Matt Gardner, Hannaneh Hajishirzi, and Luke Zettlemoyer. Compositional questions do not necessitate multi-hop reasoning. *arXiv*, 2019. 3.2

[120] T. Mitchell, W. Cohen, E. Hruschka, P. Talukdar, B. Yang, J. Betteridge, A. Carlson, B. Dalvi, M. Gardner, B. Kisiel, J. Krishnamurthy, N. Lao, K. Mazaitis, T. Mohamed, N. Nakashole, E. Platanios, A. Ritter, M. Samadi, B. Settles, R. Wang, D. Wijaya, A. Gupta, X. Chen, A. Saparov, M. Greaves, and J. Welling. Never-ending learning. *Commun. ACM*, 61(5):103–115, April 2018. ISSN 0001-0782. doi: 10.1145/3191513. URL `https://doi.org/10.1145/3191513`. 8.7

[121] Richard Montague. English as a formal language. In Bruno Visentini, editor, *Linguaggi nella societa e nella tecnica*, pages 188–221. Edizioni di Communita, 1970. 6.3.1

[122] Grégoire Montavon, Sebastian Lapuschkin, Alexander Binder, W. Samek, and K. Müller. Explaining nonlinear classification decisions with deep taylor decomposition. *Pattern Recognit.*, 65:211–222, 2017. 6.3.3

[123] C. Morgan. The nature of nonmonotonic reasoning. *Minds and Machines*, 10:321–360,

2004. 5.1

[124] Sheshera Mysore, Zach Jensen, Edward Kim, Kevin Huang, Haw-Shiuan Chang, Emma Strubell, Jeffrey Flanigan, Andrew McCallum, and Elsa Olivetti. The materials science procedural text corpus: Annotating materials synthesis procedures with shallow semantic structures. *arXiv preprint arXiv:1905.06939*, 2019. 3.1

[125] Artidoro Pagnoni, Vidhisha Balachandran, and Yulia Tsvetkov. Understanding factuality in abstractive summarization with FRANK: A benchmark for factuality metrics. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4812–4829, Online, June 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.naacl-main.383. URL `https://aclanthology.org/2021.naacl-main.383`. 8.6

[126] Bo Pang and Lillian Lee. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. *arXiv preprint cs/0506075*, 2005. 6.4.1

[127] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics, 2002. 4.3.2, 5.2.1

[128] Dong Huk Park, Lisa Anne Hendricks, Zeynep Akata, Anna Rohrbach, Bernt Schiele, Trevor Darrell, and Marcus Rohrbach. Multimodal explanations: Justifying decisions and pointing to the evidence. *CVPR*, 2018. 3.2

[129] Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014. 4.3.1

[130] Fabio Petroni, Patrick Lewis, Aleksandra Piktus, Tim Rocktäschel, Yuxiang Wu, Alexander H. Miller, and Sebastian Riedel. How context affects language models' factual predictions. In *Automated Knowledge Base Construction*, 2020. URL `https://openreview.net/forum?id=025X0zPfn`. 8.6

[131] Vitali Petsiuk, Abir Das, and Kate Saenko. Rise: Randomized input sampling for explanation of black-box models. *arXiv preprint arXiv:1806.07421*, 2018. 3.2

[132] J. Pollock. Defeasible reasoning. *Cogn. Sci.*, 11:481–518, 1987. 5.1, 5.2

[133] J. Pollock. A recursive semantics for defeasible reasoning. In *Argumentation in Artificial Intelligence*, 2009. (document), 5.1, 5.1, 5.2

[134] Danish Pruthi, Mansi Gupta, Bhuwan Dhingra, Graham Neubig, and Zachary C Lipton. Learning to deceive with attention-based explanations. In *ACL*, 2020. 6.1, 6.2

[135] Xavier Puig, Kevin Ra, Marko Boben, Jiaman Li, Tingwu Wang, Sanja Fidler, and Antonio Torralba. Virtualhome: Simulating household activities via programs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8494–8502, 2018. 2.2.2

[136] Lianhui Qin, Antoine Bosselut, Ari Holtzman, Chandra Bhagavatula, Elizabeth Clark, and Yejin Choi. Counterfactual story reasoning and generation. *EMNLP*, 2019. 4.1, 4.5

[137] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. *URL https://s3-us-west-2. amazonaws. com/openai-assets/researchcovers/languageunsupervised/language understanding paper. pdf*, 2018. 4.3.1

[138] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8):9, 2019. 4.3.1, 5.2.1

[139] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140): 1–67, 2020. URL `http://jmlr.org/papers/v21/20-074.html`. 8.4.2, 9.2.2

[140] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21:1–67, 2020. 5.2.1

[141] Dheeraj Rajagopal, Niket Tandon, P. Clarke, Bhavana Dalvi, and E. Hovy. What-if i ask you to explain: Explaining the effects of perturbations in procedural text. *Findings of EMNLP*, 2020. 2, 2.2.1, 4.5

[142] Nazneen Fatema Rajani, Bryan McCann, Caiming Xiong, and Richard Socher. Explain yourself! leveraging language models for commonsense reasoning. *ACL*, 2019. 2.2.1, 3.2, 6.2

[143] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad: 100,000+ questions for machine comprehension of text. *EMNLP*, 2016. 3.2

[144] Hannah Rashkin, Asli Celikyilmaz, Yejin Choi, and Jianfeng Gao. PlotMachines: Outline-conditioned generation with dynamic plot state tracking. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4274–4295, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.349. URL `https://aclanthology.org/2020.emnlp-main.349`. 8.7

[145] Marco Ribeiro, Sameer Singh, and Carlos Guestrin. “why should I trust you?”: Explaining the predictions of any classifier. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations*, pages 97–101, San Diego, California, June 2016. Association for Computational Linguistics. doi: 10.18653/v1/N16-3020. 2.2.2

[146] E. Riloff. Automatically generating extraction patterns from untagged text. In *AAAI/IAAI, Vol. 2*, 1996. 8.3

[147] Cynthia Rudin. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, 1(5):206–215, 2019. 6.1

[148] Rachel Rudinger, Vered Shwartz, Jena D. Hwang, Chandra Bhagavatula, Maxwell Forbes,

Ronan Le Bras, Noah A. Smith, and Yejin Choi. Thinking like a skeptic: Defeasible inference in natural language. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4661–4675, Online, November 2020. Association for Computational Linguistics. URL `https://www.aclweb.org/anthology/2020.findings-emnlp.418`. 4.1, 4.2.2, 4.5, 5.1, 5.2.1, 5.3, 5.3

[149] Alexander M. Rush, Sumit Chopra, and Jason Weston. A neural attention model for abstractive sentence summarization. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 379–389, Lisbon, Portugal, September 2015. Association for Computational Linguistics. doi: 10.18653/v1/D15-1044. 6.2

[150] Swarnadeep Saha, S. Ghosh, Shashank Srivastava, and Mohit Bansal. Prover: Proof generation for interpretable reasoning over rules. *ArXiv*, abs/2010.02830, 2020. 2.2.1, 2.2.2

[151] Wesley C Salmon. *Scientific explanation and the causal structure of the world*. Princeton University Press, 1984. 7.4

[152] Maarten Sap, Ronan Le Bras, Emily Allaway, Chandra Bhagavatula, Nicholas Lourie, Hannah Rashkin, Brendan Roof, Noah A Smith, and Yejin Choi. Atomic: An atlas of machine commonsense for if-then reasoning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 3027–3035, 2019. 4.1, 8.2, 8.7

[153] Rainer Schmidt and Lothar Gierl. Case-based reasoning for medical knowledge-based systems. *Studies in health technology and informatics*, 77:720–5, 2000. 8.2

[154] Hanie Sedghi and Ashish Sabharwal. Knowledge completion for generics using guided tensor factorization. *ACL*, 2018. 3.2

[155] Ramprasaath Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. *ICCV*, 2017. 3.2

[156] Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. *arXiv preprint arXiv:1508.07909*, 2015. 3.4

[157] Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. Bidirectional attention flow for machine comprehension. *ICLR*, 2017. 3.4

[158] Sofia Serrano and Noah A. Smith. Is attention interpretable? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2931–2951, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1282. 6.1

[159] Shikhar Sharma, Layla El Asri, Hannes Schulz, and Jeremie Zumer. Relevance of unsupervised metrics in task-oriented dialogue for evaluating natural language generation. *CoRR*, abs/1706.09799, 2017. URL `http://arxiv.org/abs/1706.09799`. 2

[160] Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. Learning important features through propagating activation differences. volume 70 of *Proceedings of Machine Learning Research*, pages 3145–3153, International Convention Centre, Sydney, Australia, 06–11 Aug 2017. PMLR. 6.3.3

[161] Anshumali Shrivastava and Ping Li. Asymmetric lsh (alsh) for sublinear time maximum

inner product search (mips). In *Advances in Neural Information Processing Systems*, pages 2321–2329, 2014. 2

[162] Vered Shwartz, Peter West, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Unsupervised commonsense question answering with self-talk. *arXiv preprint arXiv:2004.05483*, 2020. 8.7

[163] H. Simon. Rational choice and the structure of the environment. *Psychological review*, 63 2:129–38, 1956. 7.1

[164] H. Simon. The architecture of complexity. 1962. 2.2.3

[165] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *CoRR*, abs/1312.6034, 2014. 2.2.2, 6.4.4, 7.2

[166] Chandan Singh, W. James Murdoch, and Bin Yu. Hierarchical interpretations for neural network predictions. In *International Conference on Learning Representations*, 2019. URL `https://openreview.net/forum?id=SkEqro0ctQ`. 2.2.2, 6.2, 6.4.4

[167] Daniel Smilkov, Nikhil Thorat, Been Kim, Fernanda Viégas, and Martin Wattenberg. Smoothgrad: removing noise by adding noise. *arXiv preprint arXiv:1706.03825*, 2017. 2.2.2, 6.2, 7.2

[168] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642, 2013. 6.4.1, 6.4.4

[169] R. Solow and H. Simon. Models of man : Social and rational. 1957. 2.2.2

[170] Robyn Speer and Catherine Havasi. Conceptnet 5: A large semantic network for relational knowledge. In *The People's Web Meets NLP*, 2013. 8.2

[171] G. Stigler. The development of utility theory. i. *Journal of Political Economy*, 58:307 – 327, 1950. 2.2.2

[172] Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. Yago: A core of semantic knowledge. In *Proceedings of the 16th International Conference on World Wide Web*, WWW '07, page 697–706, New York, NY, USA, 2007. Association for Computing Machinery. ISBN 9781595936547. doi: 10.1145/1242572.1242667. URL `https://doi.org/10.1145/1242572.1242667`. 8.7

[173] Haitian Sun, Bhuwan Dhingra, M. Zaheer, Kathryn Mazaitis, R. Salakhutdinov, and William W. Cohen. Open domain question answering using early fusion of knowledge bases and text. In *EMNLP*, 2018. 2.2.2

[174] Tony Sun, Andrew Gaut, Shirlyn Tang, Yuxin Huang, Mai ElSherief, Jieyu Zhao, Diba Mirza, Elizabeth Belding, Kai-Wei Chang, and William Yang Wang. Mitigating gender bias in natural language processing: Literature review. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1630–1640, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1159. URL `https://www.aclweb.org/anthology/P19-1159`. 6.1

[175] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 3319–3328. JMLR. org, 2017. 2.2.2, 6.2, 7.2

[176] Oyvind Tafjord, Peter Clark, Matt Gardner, Wen-tau Yih, and Ashish Sabharwal. Quarel: A dataset and models for answering questions about qualitative relationships. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 7063–7071, 2019. 4.1, 4.2.2, 4.5

[177] Oyvind Tafjord, Matt Gardner, Kevin Lin, and Peter Clark. Quartz: An open-domain dataset of qualitative relationship questions. In *EMNLP/IJCNLP*, 2019. 4.4.2

[178] Oyvind Tafjord, Bhavana Dalvi Mishra, and Peter Clark. Proofwriter: Generating implications, proofs, and abductive statements over natural language, 2020. 2.2.2

[179] Alon Talmor and Jonathan Berant. The web as a knowledge-base for answering complex questions. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 641–651, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-1059. URL `https://aclanthology.org/N18-1059`. 8.1

[180] Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. Commonsenseqa: A question answering challenge targeting commonsense knowledge. In *NAACL*, 2019. 3.2

[181] Niket Tandon, Bhavana Dalvi Mishra, Joel Grus, Wen-tau Yih, Antoine Bosselut, and Peter Clark. Reasoning about actions and state changes by injecting commonsense knowledge. *EMNLP*, 2018. 3.1

[182] Niket Tandon, Bhavana Dalvi, Keisuke Sakaguchi, Peter Clark, and Antoine Bosselut. Wiqa: A dataset for "what if..." reasoning over procedural 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)*, pages 6078–6087, 2019. (document), 4.1, 4.2.2, 4.4.1, 4.7, 4.5, 5.1, 5.1, 5.2, 5.2.1, 5.2.1

[183] Niket Tandon, Bhavana Dalvi Mishra, Keisuke Sakaguchi, Antoine Bosselut, and Peter Clark. Wiqa: A dataset for "what if..." reasoning over procedural text. *EMNLP*, 2019. 3.1, 3.2, 3.3, 3.5, 3.7

[184] James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. FEVER: a large-scale dataset for fact extraction and VERification. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 809–819, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-1074. URL `https://www.aclweb.org/anthology/N18-1074`. 2.2.1

[185] D. Traum. Towards a computational theory of grounding in natural language conversation. 1991. 7.3

[186] Bas C Van Fraassen et al. *The scientific image*. Oxford University Press, 1980. 7.4

[187] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N

Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017. 6.3.2

[188] Elena Voita and Ivan Titov. Information-theoretic probing with minimum description length. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 183–196, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.14. URL `https://www.aclweb.org/anthology/2020.emnlp-main.14`. 2.2.2

[189] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Ed Chi, Quoc Le, and Denny Zhou. URL `https://arxiv.org/abs/2201.11903`. 8.1

[190] Gerhard Weikum and Martin Theobald. From information to knowledge: harvesting entities and relationships from web sources. In *PODS*, 2010. 3.2

[191] Daniel S Weld and Johan De Kleer. *Readings in qualitative reasoning about physical systems*. Morgan Kaufmann, 2013. 3.2

[192] Sarah Wiegreffe and Yuval Pinter. Attention is not not explanation. 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)*, pages 11–20, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1002. 6.1, 6.2, 9.2.3

[193] Sarah Wiegreffe and Yuval Pinter. Attention is not not explanation. *arXiv preprint arXiv:1908.04626*, 2019. 3.2

[194] Sarah Wiegreffe, Ana Marasović, and Noah A. Smith. Measuring association between labels and free-text rationales. *ArXiv*, abs/2010.12762, 2020. 3

[195] Georg Wiese, Dirk Weissenborn, and Mariana Neves. Neural domain adaptation for biomedical question answering. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 281–289, Vancouver, Canada, August 2017. Association for Computational Linguistics. doi: 10.18653/v1/K17-1029. URL `https://aclanthology.org/K17-1029`. 8.7

[196] Thomas Wolf, L Debut, V Sanh, J Chaumond, C Delangue, A Moi, P Cistac, T Rault, R Louf, M Funtowicz, et al. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv, abs/1910.03771*, 2019. 4.3.1

[197] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online, October 2020. Association for Computational Linguistics. URL `https://www.aclweb.org/anthology/2020.emnlp-demos.6`. 6

[198] Zhengnan Xie, S. Thiem, J. Martin, E. Wainwright, Steven Marmorstein, and Peter A. Jansen. Worldtree v2: A corpus of science-domain structured explanations and inference

patterns supporting multi-hop inference. In *LREC*, 2020. 2.2.1

[199] Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *EMNLP*, 2018. 2.2.1, 3.2, 8.1

[200] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in neural information processing systems*, pages 5754–5764, 2019. 6.3.2, 6.4.2

[201] Zi Yang and Eric Nyberg. Leveraging procedural knowledge for task-oriented search. In *SIGIR*, 2015. 3.1

[202] Thomas Wolf Yangfeng Ji, Antoine Bosselut and Asli Celikyilmaz. The amazing world of generation. *EMNLP tutorials*, 2020. URL `https://nlg-world.github.io/`. 4.5

[203] Lili Yao, Nanyun Peng, Ralph Weischedel, Kevin Knight, Dongyan Zhao, and Rui Yan. Plan-and-write: Towards better automatic storytelling. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):7378–7385, Jul. 2019. doi: 10.1609/aaai.v33i01.33017378. URL `https://ojs.aaai.org/index.php/AAAI/article/view/4726`. 8.7

[204] S. Yih. Template-based information extraction from tree-structured html documents. 1997. 8.3

[205] Omar Zaidan and Jason Eisner. Modeling annotators: A generative approach to learning from annotator rationales. In *Proceedings of the 2008 conference on Empirical methods in natural language processing*, pages 31–40, 2008. 6.2

[206] Fabio Massimo Zanzotto, Andrea Santilli, Leonardo Ranaldi, Dario Onorati, Pierfrancesco Tommasino, and Francesca Fallucchi. KERMIT: Complementing transformer architectures with encoders of explicit syntactic interpretations. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 256–267, Online, November 2020. Association for Computational Linguistics. URL `https://www.aclweb.org/anthology/2020.emnlp-main.18`. (document), 6.2

[207] Rowan Zellers, Yonatan Bisk, Ali Farhadi, and Yejin Choi. From recognition to cognition: Visual commonsense reasoning. *ArXiv*, abs/1811.10830, 2018. 3.2

[208] Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. Bertscore: Evaluating text generation with bert. *ArXiv*, abs/1904.09675, 2019. 3.2, 3.4, 8.4.1

[209] Jieyu Zhao, Tianlu Wang, Mark Yatskar, Vicente Ordonez, and Kai-Wei Chang. Men also like shopping: Reducing gender bias amplification using corpus-level constraints. In *Proc. of EMNLP*, pages 2979–2989, 2017. 6.1

[210] Ruiqi Zhong, Steven Shao, and Kathleen McKeown. Fine-grained sentiment analysis with faithful attention. *arXiv preprint arXiv:1908.06870*, 2019. 6.2