

INTERPRETABLE NATURAL LANGUAGE PROCESSING WITH
APPLICATIONS TO INFORMATION EXTRACTION

by

Zheng Tang

Copyright © Zheng Tang 2022

A Dissertation Submitted to the Faculty of the

DEPARTMENT OF COMPUTER SCIENCE

In Partial Fulfillment of the Requirements

For the Degree of

DOCTOR OF PHILOSOPHY

In the Graduate College

THE UNIVERSITY OF ARIZONA

2022

THE UNIVERSITY OF ARIZONA
GRADUATE COLLEGE

As members of the Dissertation Committee, we certify that we have read the dissertation prepared by: Zheng Tang, titled: *Interpretable Natural Language Processing with Applications to Information Extraction*.

and recommend that it be accepted as fulfilling the dissertation requirement for the Degree of Doctor of Philosophy.

	Date: <u>Dec 27, 2022</u>
_____ <i>Dr. Mihai Surdeanu</i>	
	Date: <u>Dec 23, 2022</u>
_____ <i>Dr. Chicheng Zhang</i>	
	Date: <u>Dec 20, 2022</u>
_____ <i>Dr. Clayton Morrison</i>	
	Date: <u>Dec 20, 2022</u>
_____ <i>Dr. Gus Hahn-Powell</i>	

Final approval and acceptance of this dissertation is contingent upon the candidate's submission of the final copies of the dissertation to the Graduate College.

I hereby certify that I have read this dissertation prepared under my direction and recommend that it be accepted as fulfilling the dissertation requirement.

	Date: <u>Dec 27, 2022</u>
_____ <i>Dr. Mihai Surdeanu</i> Dissertation Committee Chair Computer Science Department	

ACKNOWLEDGEMENTS

I would like to give heartfelt thanks to my advisor and chair of my committee Dr. Mihai Surdeanu who offered continuing support and constant encouragement. I also could not have undertaken this journey without my defense committee (ordered by the last names) Dr. Clayton T Morrison, Dr. Gus Hahn-Powell and Dr. Chicheng Zhang who generously provided knowledge and expertise.

I would like to acknowledge everyone in CLU lab. It is my honor to work with a group of such brilliant minds. Thank you for the stimulating discussions, for the sleepless nights we were working together before deadlines, and for all the fun we have had in the last five years.

I would also like to thank Dr. Jinan Xu at Beijing Jiaotong University who introduced me to the wonderful NLP research and Dr. Craig Knoblock at USC Information Science Institute who offered me the first job as a researcher.

And thanks to all the colleagues/friends who provided all the constructive, fun, and inspiring activities/discussions. Thanks to all the University of Arizona and the Department of Computer Science's faculties and staff and all the students I had in my TA experience for an eventful and enjoyable overseas graduate study experience.

Lastly, I would be remiss in not mentioning my family, especially my parents. Their belief in me has kept my spirits and motivation high during this process.

DEDICATION

Talk is cheap. Show me the code.

TABLE OF CONTENTS

LIST OF FIGURES	7
LIST OF TABLES	10
ABSTRACT	14
CHAPTER 1 INTRODUCTION	16
1.1 The Definition of Rules	18
1.2 Rules as Explanations	18
1.3 Contributions	19
1.3.1 Interpretability with Explanation Decoder	19
1.3.2 Interpretability with Sequence Modeling	20
1.3.3 Bootstrapped Self Training with Rules	21
1.4 Overview	22
CHAPTER 2 RELATED WORK	23
2.1 Relation Extraction	23
2.1.1 Relation extraction before deep learning	23
2.1.2 Deep learning methods for relation extraction	25
2.2 Explainability	26
2.2.1 A taxonomy of explanations	26
2.2.2 Finding rationales	28
2.3 Semi-supervised Learning	29
2.3.1 Bootstrapped Self Training	30
2.3.2 Co-training	30
2.3.3 Prompt-based Zero- or Few-shot Learning	30
CHAPTER 3 Jointly Learning of a Neural Classifier and an Explanation Decoder	32
3.1 Approach	34
3.1.1 Task 1	34
3.1.2 Task 2	37
3.2 Experimental Results	38
3.2.1 Datasets	38
3.2.2 Baseline	39
3.2.3 Implementation and Evaluation Details:	40
3.2.4 Results and Discussion	40
3.3 Conclusion	46

TABLE OF CONTENTS – *Continued*

CHAPTER 4	Multitask Learning of Neural Relation and Explanation Classifiers	49
4.1	Approach	52
4.1.1	Walkthrough Example	52
4.1.2	Joint Relation and Explainability Classifiers	54
4.1.3	Aggregating Local Explanations into a Global, Rule-based Model	62
4.2	Experimental Results	63
4.2.1	Data Preparation	63
4.2.2	Baselines	64
4.2.3	Implementation and Evaluation Details	68
4.2.4	Results and Discussion	70
4.3	Conclusion	88
CHAPTER 5	Rule-enhanced Bootstrapped Self Training for Weakly-supervised Relation Extraction	90
5.1	Approach	91
5.1.1	Training Procedure	93
5.2	Experimental Results	93
5.2.1	Data Preparation	93
5.2.2	Baselines	94
5.2.3	Implementation and Evaluation Details:	95
5.2.4	Results and Discussion	96
5.3	Conclusion	97
APPENDIX	103
REFERENCES	106

LIST OF FIGURES

1.1	Example of information extraction using CoreNLP (Manning et al., 2014a). The IE system can locate the named entities and identify the relation between entity pairs.	17
1.2	An example of a relation extraction rule in the Odin language that extracts the <code>per:employee_of</code> relation. The rule is driven by verbal triggers such as <i>work</i> , <i>play</i> or <i>serve</i> . The relation’s arguments (the subject and object) are identified through both semantic constraints (subject must be <code>Person</code>), and syntactic ones (subject must be attached to the trigger through a certain syntactic dependency pattern: an optional (?) adnominal clause (<code>ac1</code>), followed by a nominal subject (<code>nsubj</code>). This rule would extract a <code>per:employee_of</code> relation from the text “... <i>Joe is a research scientist working at IBM...</i> ”. . .	19
2.1	General workflow of Sainz et al. (2021)’s entailment-based RE approach.	31
3.1	Neural architecture of the proposed multitask learning approach. The input is a sequence of words together with NER labels and POS tags. The pair of entities to be classified (“subject” in blue and “object” in orange) are also provided. We use a concatenation of several representations, including embeddings of words, NER labels, and POS tags. The encoder uses a sentence-level bidirectional LSTM (biLSTM) and graph convolutional networks (GCN). There are pooling layers for the subject, object, and full sentence GCN outputs. The concatenated pooling outputs are fed to the classifier’s feedforward layer. The decoder is a LSTM with an attention mechanism.	33
4.1	Flow of our semi-supervised training procedure for an individual training example. All the “Train ...” blocks (green background) involve parameter updates of the corresponding classifiers. These updates are shown here for an individual training example, but are batched in the actual implementation.	55

LIST OF FIGURES – *Continued*

4.2	An example of a relation extraction rule in the Odin language that extracts the <code>per:employee_of</code> relation. The rule is driven by verbal triggers such as <i>work</i> , <i>play</i> or <i>serve</i> . The relation’s arguments (the subject and object) are identified through both semantic constraints (subject must be <code>Person</code>), and syntactic ones (subject must be attached to the trigger through a certain syntactic dependency pattern: an optional (?) adnominal clause (<code>acl</code>), followed by a nominal subject (<code>nsubj</code>). This rule would extract a <code>per:employee_of</code> relation from the text “... <i>Joe is a research scientist working at IBM</i> ...” . . .	58
4.3	Neural architecture of the proposed multitask learning approach. The entity tokens (subject in <code>blue</code> and object in <code>orange</code>) are masked with their named entity labels, e.g., <code>SUBJ-Person</code> , in the actual implementation.	60
4.4	Examples of explainability annotations on TACRED for a <i>correct</i> RC prediction. The subject and object entities, which are provided in the task input, are highlighted in <code>blue</code> and <code>orange</code> . The important tokens for explainability identified by the various methods are highlighted in <code>red</code> . The bottom of the figure shows the heatmap of <code>[CLS]</code> attention weights for BERT’s 16 heads (the lighter the color the higher the weight). We shrink the weight range margin to make the color scale distinguishable. For a fair comparison, we masked the subject and object in the attention weights.	77
4.5	Examples of explainability annotations on TACRED for an <i>incorrect</i> RC prediction. This figure follows the same convention as Figure 4.4.	78
4.6	Examples of explainability annotations on CoNLL04 for an <i>incorrect</i> RC prediction. This figure follows the same convention as Figure 4.4.	79
4.7	Examples of explainability annotations on CoNLL04 for a <i>correct</i> RC prediction. This figure follows the same convention as Figure 4.4. . . .	80
4.8	The distributions of POS tags in the TACRED test partition. The top figure shows how many times each POS tag appears in the test data. The bottom figure shows how many times each POS tag appears in the generated explanations from the same partition.	82
5.1	Overall procedure of our bootstrapped self training approach. At the beginning, we come up with a set of manual rules. In each iteration, (1) we apply the rules to the rule executor and move the matched data from raw data set D_R to labeled data set D_L (2) we train the neural model with D_L , (3) we generate rules using the outputs from the neural model, (4) we feed the new rules in rule executor and repeat this procedure from (1).	92

LIST OF FIGURES – *Continued*

5.2	Learning curve of our approach based on the iterations.	97
5.3	Example of nested event extraction from the BioNLP task.	101

LIST OF TABLES

3.1	Results for the three events in the BioNLP 2013 test partition. T1 and T2 indicate the two tasks in my MTL approach, i.e., the event classifier and the rule decoder, respectively. Silver indicates that that configuration used the silver data created by the rule-based system (see §3.2.1). BioNLP best and median indicate the best/median results during the 2013 shared task. I do not include T1 + T2 results because in this configuration I observed that there is not sufficient data to train the decoder.	41
3.2	Evaluation of decoded rules, on the BioNLP development partition. BLEU measures the overlap with hand-written rules. Exact Matches shows the percentage of decoded rules that exactly match hand-written ones. Explainable Matches shows the number of decoded rules that do not match exactly hand-written ones, but were considered good explanations by human experts.	43
3.3	Examples of mistakes in the decoded rules. The first column shows hand-written rules, while the second shows the rules decoded by my approach from sentences where the corresponding hand-written rules matched. I highlight in the hand-written rules the tokens that were missed during decoding (false negatives) in green, and in the decoded rules I highlight the spurious tokens (false positives) in red. The first row lists a partial mistake, which does not affect the interpretability of the decoded rule, since it only misses one token that can be inferred by the human experts from context. The second row lists a partial mistake, which impacts the semantics of the rule. For example, the decoder missed that the path between the trigger and the theme argument starts with an optional prop_of and appos . This rule was marked as partially correct because some simple syntactic patterns, e.g., nn , can still be correctly matched by the decoded rule. The last row lists a larger decoding error that was marked as completely incorrect by the annotator. For example, in the last decoded rule, the decoder generated an incorrect cause argument, which does not exist in the data, as well as an incorrect syntactic pattern for the theme argument, i.e., the protein being phosphorylated.	44

LIST OF TABLES – *Continued*

3.4	Results on the TACRED test partition, including ablation experiments (the “w/o” rows). I experimented with two configurations: <i>Rule-only data</i> uses only training examples generated by rules; <i>Rules + TACRED training data</i> applies the previous rules to the training dataset from TACRED.	45
3.5	Learning curve of my approach based on amount of rules used, in the <i>rule-only data</i> configuration. These results are on TACRED development.	45
3.6	Examples of mistakes in the decoded rules. I highlight in the hand-written rules the tokens that were missed during decoding (false negatives) in green, and in the decoded rules I highlight the spurious tokens (false positives) in red	47
4.1	Walkthrough example of our approach. The task input includes information about the entities participating in the relation (denoted as subject and object) and their types (<code>PERSON</code> here). Our neural architecture, which includes both a relation and explanation classifier, predicts the relation that holds between the two entities (<code>per:children</code> here, i.e., the object is the child of the subject), as well as which words best explain the decision (in red). In step (c), the rule generator collects the necessary information from the annotated sentence, i.e., the shortest syntactic dependency path that connects the two entities with the explanation words (in red in the figure). Step (d) shows the generated rule in the Odin language.	53
4.2	Relation extraction results on the TACRED test partition. We used the pre-trained SpanBERT-large. Our full model trains on the entire training partition using the SSL method discussed in Section 4.1.2. The “burn-in only” setting trains just on the training subset that has annotations from rules.	71
4.3	Relation extraction results on the CoNLL04 test partition. We used the pre-trained SpanBERT-large. Our full model trains on the entire training partition using the SSL method discussed in Section 4.1.2. The “burn-in only” setting trains just on the training subset that has annotations from rules.	71
4.4	Automated evaluation of explainability on TACRED, in which we compare explainability annotations produced by these methods against the lexical artifacts of rules.	73
4.5	Automated evaluation of explainability on CoNLL04, in which we compare explainability annotations produced by these methods against the lexical artifacts of rules.	74

LIST OF TABLES – *Continued*

4.6	Learning curve of our approach on TACRED based on amount of rules used. In each experiment, we use up to top k rules per relation type; the number in parentheses is the actual average number of rules per type.	74
4.7	TACRED evaluation of the plausability of explanations, which measures the overlap between machine explanations and human annotations. For each method, we pick the higher scores between the two human annotators.	75
4.8	CoNLL04 evaluation of the plausability of explanations, which measures the overlap between machine explanations and human annotations. For each method, we pick the higher scores between the two human annotators.	76
4.9	Ablation results on the TACRED test partition, i.e., “–” indicates that the corresponding component was removed from the full system, and “N/A” indicates that that metric is not applicable.	83
4.10	Ablation results on the CoNLL04 test partition, i.e., “–” indicates that the corresponding component was removed from the full system, and “N/A” indicates that that metric is not applicable.	83
4.11	Performance of the rule-based model on the TACRED test partition. [1] is the set of manually-written surface rules of Angeli et al. (2015) coupled with our syntactic rules (see Section 4.2.1). [2] is the set of rules generated from our explainability classifier’s outputs with gold labels on the training partition. [3] is the set of rules from the explainability classifier’s outputs with predicted labels on the test partition. We also evaluate the performance on combinations of these sets of rules: [2]+[3] contain all rules generated by our approach; [1]+[2]+[3] combine machine-generated rules with the manually-written rules.	84
4.12	Performance of the rule-based model on the CoNLL04 test partition. This table follows the same conventions as Table 4.11, except, in this case, [1] is the set of manually-written Odin rules we wrote for CoNLL04.	85
4.13	Typical errors that our explainability classifier commits. These include errors of under prediction (first two rows), misleading prediction (middle two rows), and errors of over prediction (last two rows). This figure follows the same convention as Figure 4.5.	87
5.1	Relation extraction results on the TACRED test partition.	95
5.2	NLI-prompt results on the TACRED test partition. We convert the rules in that iteration to verbalization template.	96
5.3	Details of our neural architecture.	103

LIST OF TABLES – *Continued*

5.4	Hyperparameter details for training the neural models for relation classification (for SpanBERT) and both components (Unsupervised Rationale and our approach). The numbers with * are the default values from the SpanBERT implementation available at: https://github.com/facebookresearch/SpanBERT	105
-----	--	-----

ABSTRACT

Interpretability is very important for many NLP applications. Many such applications (e.g., information extraction, sentiment analysis) are applied to important decision-making areas like government policy, financial, law, and others. In these scenarios, machines must explain the information produced, if they are to be deployed in the real world.

In this dissertation, we present some approaches for information extraction that mitigates the tension between generalization and explainability by jointly training for the two goals. First we investigate an approach uses an encoder-decoder architecture, which jointly trains a classifier for information extraction, and a rule decoder that generates syntactico-semantic rules that explain the decisions of the classifier. We evaluate the proposed approach on two different information extraction tasks and show that the decoder generates interpretable rules that serve as accurate explanations for the classifier’s decisions, and, importantly, that the joint training generally improves the performance of the classifier. We show that our approach can be used for semi-supervised learning, and that its performance improves when trained on automatically-labeled data generated by a rule-based system. Second, we investigate another approach uses a multi-task learning architecture, which jointly trains a classifier for relation extraction, and a sequence model that labels words in the context of the relation that explain the decisions of the relation classifier. We also convert the model outputs to rules to bring global explanations to this approach. This sequence model is trained using a hybrid strategy: supervised, when supervision from pre-existing patterns is available, and semi-supervised otherwise. In the latter situation, we treat the sequence model’s labels as latent variables, and learn the best assignment that maximizes the performance of the relation classifier. We evaluate

the proposed approach on the two relation extraction datasets and show that the sequence model provides labels that serve as accurate explanations for the relation classifier’s decisions, and, importantly, that the joint training generally improves the performance of the relation classifier. We also evaluate the performance of the generated rules and show that the new rules are great add-on to the manual rules and bring the rule-based system much closer to the neural models. Third, we also explore the usages of the model outputs in two ways: 1. Convert them to rules to bring global explanations to this approach; and 2. Use them for bootstrapping when we do not have enough data. Our globally-explainable models approach the performance of neural ones within a reasonable gap.

CHAPTER 1

INTRODUCTION

Information extraction (IE) is one of the key challenges in the natural language processing (NLP) field. With the explosion of unstructured information on the Internet, the demand for high-quality tools that convert free text to structured information continues to grow (Chang et al., 2010; Lee et al., 2013; Valenzuela-Escarcega et al., 2018). Formally, an IE task is defined by its input and its extraction target (Chang et al., 2006). The input is usually a raw text which is written in natural language and the target can be formed in many different ways. In named entity recognition (NER) task, the target is to extract the spans from raw text as entities and recognize the entities as one of several categories such as location (LOC), persons (PER) or organizations (ORG). In relation extraction (RE) task, the goal is to identify the pre-define relations between two given entities. For example, in Stanford TACRED (Zhang et al., 2017) task, the relation held between the entities are 41 TAC KBP (Getman et al., 2017) canonical relation types, or no_relation label if no relation was found. Or in event extraction (EE) task, it aims to classify the event within n-grams. This can be a phosphorylation over a gene, a positive regulation over two events, or a binding event over multiple events and genes in the medical field (Nédellec et al., 2013). Figure 1.1 demonstrates an IE application from Stanford CoreNLP, in which the system handles both NER and RE tasks.

The past decades have seen a steady transition from rule-based IE systems Appelt et al. (1993) to methods that rely on machine learning (ML) (see Related Work). Currently, the neural networks based solutions have dominate most of the IE tasks. From convolutional neural networks (CNN), recurrent neural networks (RNN) such as long-short term memory (LSTM) networks, to most recent self-attention transformer

— Text to annotate —

John Doe, a professor at Oxford in England likes running.

— Annotations —

named entities ✕

relations ✕

— Language —
 English ▾

Submit

Named Entity Recognition:

PERSON
TITLE
CITY
COUNTRY

1 John Doe , a professor at Oxford in England likes running .

KBP Relations:

1 John Doe , a professor at Oxford in England likes running .

Figure 1.1: Example of information extraction using CoreNLP (Manning et al., 2014a). The IE system can locate the named entities and identify the relation between entity pairs.

networks. These neural network methods brought us tremendous improvement on most of the NLP tasks. This, of course, includes the IE tasks.

However, there is a tension between generalization and interpretability, as interpretable models are often generated by “distilling” a model with better generalization, e.g., a deep learning one that relies on distributed representations, into models that are more interpretable but lose some generalization, e.g., linear models or decision trees (Craven and Shavlik, 1996; Ribeiro et al., 2016; Frosst and Hinton, 2017). Here, we argue that both generalization and interpretability are equally important. For example, in the medical space, a patient will likely reject a treatment recommended by an algorithm without an explanation. Closer to natural language processing (NLP), a statistical information extraction method that converts free text in a specific domain to structured knowledge should also provide human-understandable explanations of its extractions. This allows the subject matter expert to quality check such output without a deep knowledge of the underlying machinery, which is a necessity in successful inter-disciplinary NLP collaborations.

In this work we propose an approach for relation extraction (RE) that *jointly* learns how to explain and predict. To achieve that, we use rules as the explanation

to defend the model prediction. Next, we elaborate what is rule and why we think rule can be used as explanations.

1.1 The Definition of Rules

The term “rule” is ambiguous. Because of this, it is important to define what we mean by “rule” in this work. We show two walkthrough example rules to demonstrate it.

(1) Surface rules: The surface rules defines regular expressions over text and tokens, and maps matched text to semantic objects. For example, in the Tokensregex language (Chang and Manning, 2014), the rule `SUBJ-PER was born in * OBJ-CITY`¹ extracts a `per:city_of_birth` relation between a person named entity (the subject) and a city named entity (the object) if the sequence *was born in* occurs somewhere between the two entities.

(2) Syntactic rules: We also use syntax-based rules using the Odin language Valenzuela-Escárcega et al. (2016b).² Figure 1.2 shows an example of such a rule.

1.2 Rules as Explanations

In contrast to modern deep neural networks methods, the predictions of rule-based approaches are easily explainable, as a small number of rules tends to apply to each extraction. Further, in many situations, rule-based methods can be developed by domain experts with minimal training data. For these reasons, rule-based IE methods remain widely used in industry Chiticariu et al. (2013).

As described in Section 1.1, we defined the rules in two forms: the surface ones and the syntactic ones. In surface rules, the tokens contained in the rule (e.g., *was*, *born*, *in*) can explain the relation between two entity mentions (the subject and object). Similarly, in syntactic rules, all their lexical elements (typically the trigger

¹We simplified the Tokensregex syntax for readability.

²All these rules are included in this submission as supplemental material.

```

label: per:employee_of
pattern: |
  trigger =
    [lemma=/work|write|play|consult|serve/]
  subject: SUBJ_Person = <acl? nsubj
  object: OBJ_Organization = nmod

```

Label(s) to assign to a match.

Lexical constraints on the relation’s predicate.

argName:ArgType, where ArgType indicates the named-entity category expected for this argument.

Figure 1.2: An example of a relation extraction rule in the Odin language that extracts the `per:employee_of` relation. The rule is driven by verbal triggers such as *work*, *play* or *serve*. The relation’s arguments (the subject and object) are identified through both semantic constraints (subject must be `Person`), and syntactic ones (subject must be attached to the trigger through a certain syntactic dependency pattern: an optional (?) adnominal clause (`acl`), followed by a nominal subject (`nsubj`). This rule would extract a `per:employee_of` relation from the text “...*Joe is a research scientist working at IBM...*”.

predicates such as *work* or *write* in the figure) can explain the relation between two arguments.

1.3 Contributions

1.3.1 Interpretability with Explanation Decoder

First we proposed an interpretable approach for event extraction (EE) that mitigates the tension between generalization and interpretability through multitask learning (MTL). Our approach uses an attention-based encoder to encode the input text and given entities of interest (e.g., proteins in the biomedical domain), and a decoder that jointly trains two tasks. The first task is event classification, which identifies which event applies for a given entity (e.g., phosphorylation). The second task decodes a rule in the some pattern language like Odin (Valenzuela-Escárcega et al., 2018;

Valenzuela-Escárcega et al., 2016), which explains the prediction of the classifier in a format that can be read and understood by human end users. Importantly, both tasks share the same encoder, and are trained using a joint objective function. We later expand this approach to relation extraction task. In this scenario, the first task becomes relation classification, which identifies which relation applied for a given entities pairs. The second task remains the same as decoding a rule which defends the prediction of the classifier.

Supporting earlier findings, we observe that joint training leads to performance improvements both within and across tasks. However, the quality of the generated rules is not perfect. There are two major flaws: 1. With no grammar checking in decoding period, the rules are not guaranteed to be executable; and 2. The decoder prefer to repeat what it saw in the training examples. Due to these issues, the generated rules can be only seen as explanations to the classification results and prevent us from apply them to further usages such as model induction³ or semi-supervised learning (SSL).

1.3.2 Interpretability with Sequence Modeling

In the Section 1.3.1, we show that joint training a classifier with an explanation decoder leads to performance improvements on the classifier and the decoder can generate rules which can defend the classifier’s decision. However, we also address the drawbacks of the decoded rules, thus cannot be used for model induction or semi-supervised learning.

To solve these flaws, we propose an approach uses a multi-task learning architecture, which jointly trains a classifier for relation extraction, and a sequence model that labels words in the context of the relation that explain the decisions of the relation classifier. We evaluate the proposed approach on multiple datasets and show that the sequence model provides labels that serve as accurate explanations for the relation classifier’s decisions, and, importantly, that the joint training generally improves the performance of the relation classifier. We also convert the model

³Infer explainable s models (in our case, rules) as transparent alternative to the neural model.

outputs to rules to bring global explanations to this approach. We evaluate the performance of the generated rules and show that the new rules are great add-on to the manual rules and bring the rule-based system much closer to the neural models.

1.3.3 Bootstrapped Self Training with Rules

In the section 1.3.2, we show that the explanation classifier can provide us high-quality rationales to the relation classifier’s decisions. Further, converting the model outputs to rules gives us a rule-based system that is much closer to the neural models. This inspires us that this approach can be applied in semi-supervised learning (SSL) where the amount of data annotation is limited.

In the typical pseudo labeling method in SSL, people first train the model with limited labeled data and later apply the trained model to unlabeled data to gather more annotations, repeat this process until no more new data can be labeled. Empirically, this method works well when there is only small labeled data available. However, the model cannot guarantee the quality of all its predictions and it may cause the problem like semantic drift or large human effort on filtering the model annotations.

To improve the pseudo labeling approach, we use the model outputs to generate rules and apply the rules to unlabeled data to get more annotations. Unlike the model predictions, we argue that the rules are more interpretable and easier to filter and modify. In this work, we evaluate the proposed approach on TACRED dataset in a no supervision setting, where we hide all the gold labels from TACRED training set and only use the high-precision manual rules to get the seed annotation. We show that using rules in bootstrapping is better than using the model predictions directly and we can reach similar results compared to NLI prompt based models without the extra NLI component.

1.4 Overview

The rest of this work is organized as follows. In Chapter 2 we outline the previous work relevant to our task. In Chapter 3 we present the work on jointly training a neural classifier and an explanation decoder and show that the decoder not only brings interpretability to the neural model and helps the performance of the neural model. Then in Chapter 4 we discuss the approach where we train a neural relation and expansion classifiers in a multitask learning framework and generate rules for model induction purpose using the model outputs. In Chapter 5, we discuss how applying the generated rules from the model we proposed in Chapter 4 to bootstrapped self training approaches improves performance in comparison to traditional pseudo-labeling methods that rely on self-trained model's predictions.

CHAPTER 2

RELATED WORK

Our interpretability works lie at the intersection of relation extraction and explainability. The third work belongs to the semi-supervised learning. We summarize these three research areas next.

2.1 Relation Extraction

Information extraction (IE), i.e., extracting structured information from text such as events and their participants, is one of the fundamental tasks in NLP that was shown to be useful for many end-user applications such as question answering Srihari and Li (1999, 2000), and summarization Rau et al. (1989); Zechner (1997). Our work focuses on a subtask of IE: relation extraction (RE), which addresses the extraction of (mostly) binary relations between entities such as `place_of_birth`, which connects a person named entity with a location.

RE has received tremendous attention in the past several decades. We group the works on RE into two categories: before the “deep learning tsunami” Manning (2015), and after.

2.1.1 Relation extraction before deep learning

The first approaches for RE were rule-based. For example, Hearst (1992) proposed a method to learn hyponymy relations using hand-written patterns. Riloff (1996) introduced a pattern acquisition method that alternates between learning patterns and extracting relation mentions. Brin (1998) proposed a dual iterative pattern/relation expansion, which exploited the duality between patterns and relations. Hassan et al. (2006) used Hyperlink-Induced Topic Search (HITS) Kleinberg (1999) to

jointly learn patterns and relations in an unsupervised manner. In general, these rule-based methods usually obtain high precision but suffer from low recall. While our explanations can be interpreted as rules, our work differs from these directions in two significant ways. First, most of these directions are iterative, alternating between learning patterns (or rules) and relations. In contrast, our approach trains relation and explanation classifiers jointly. Second, and probably more importantly, we show that our explanations often focus on parts of speech that are necessary for plausability (according to the human annotators) but are semantically-ambiguous such as prepositions and determiners. On the other hand, most pattern acquisition methods usually focus on clear syntactic structures such as subject-verb-object and words with more clear semantics such as nominals and verbs.

Statistical methods that followed the above rule-based approaches address the limited generality of rules. In terms of supervision, “traditional” machine learning approaches for RE include fully supervised methods Zelenko et al. (2003); Bunescu and Mooney (2005), or methods that rely on distant supervision, where training data is generated automatically by (noisily) aligning existing knowledge bases with texts Mintz et al. (2009); Riedel et al. (2010); Hoffmann et al. (2011); Surdeanu et al. (2012). Most of these approaches used explicit features such as lexical, syntactic, and semantic. For example, Kambhatla (2004) proposed a maximum entropy classifier using these features. Zhou et al. (2005) found that additional features such as syntactic chunks further help the classification performance. Jiang and Zhai (2007) evaluate the effectiveness of different feature spaces for RE. Similarly, Chan and Roth (2011) expanded feature representations to include syntactico-semantic structures that improve RE.

Our work is conceptually similar to the method of Chan and Roth (2011). Similar to them, we extract relations only from the smaller context identified by a distinct component (the explainability classifier in our case). However, there are several important differences between these two efforts. First, the method of Chan and Roth (2011) operates as a pipeline: they start by matching syntactico-semantic structures potentially indicative of relations, and then they apply a relation classifier only on

the texts that match them. In contrast, our method jointly trains the relation and explainability classifiers. Second, the syntactico-semantic structures in Chan and Roth (2011) were manually extracted and categorized, whereas our explanations are learned in a semi-supervised way from data and a small number of rules. Last but not least, the patterns of Chan and Roth (2011) are non-lexicalized. In contrast, the explanations produced by our explainability classifier are lexicalized, which is critical for human understanding.

Kernel methods were also a popular direction for relation extraction due to their advantage of avoiding feature engineering. To this end, Miller et al. (2000) introduced a sequence kernel for relation extraction. Several researchers proposed kernels designed around constituent parse trees to capture sentence grammatical structure Miller et al. (2000); Zelenko et al. (2003); Moschitti (2006). Bunescu and Mooney (2005); Nguyen et al. (2009) introduced kernels based on syntactic dependencies, a simpler representation that flattens constituent trees while preserving most syntactic information. To combine the information captured by individual kernels that model different representations, Zhao and Grishman (2005) presented a composite kernel which combines multiple such individual kernels.

2.1.2 Deep learning methods for relation extraction

Deep learning approaches for RE that rely on sequence models range from using CNNs or RNNs Zeng et al. (2014); Zhang and Wang (2015), to augmenting RNNs with different components Xu et al. (2015); Zhou et al. (2016), or to combining RNNs and CNNs Vu et al. (2016); Wang et al. (2016). Other approaches take advantage of graph neural networks Zhang et al. (2018) or attention mechanisms Zhang et al. (2017).

More recently, transformer-based Vaswani et al. (2017) approaches have shown considerable improvements on many natural language tasks including RE. For example, Wu and He (2019) applied BERT Devlin et al. (2018) to the TACRED RE task. Devlin et al. (2018); Yamada et al. (2020) showed that further improvements are possible with a better representation for the pre-trained language model.

Our approach also fits in this space. We deploy a transformer-based classifier to capture relation mentions, but we also include a novel component dedicated to explainability, which tags the words important for the relation at hand. Importantly, our direction has the relation classifier operate directly on top of the words deemed important for the relation by the explainability classifier, which guarantees that our explanations are *faithful*, i.e., our explanations correctly depict how the relation classifier makes a decision Vafa et al. (2021). Further, we propose an efficient semi-supervised strategy to jointly train the relation and explainability classifiers using a small amount of linguistic supervision for explainability.

2.2 Explainability

Explainable artificial intelligence (XAI) has recently experienced a resurgence in the context of deep learning Adadi and Berrada (2018); Gunning and Aha (2019); Arrieta et al. (2020); Danilevsky et al. (2020).

2.2.1 A taxonomy of explanations

Explanations can be categorized along two main aspects: whether they explain a complete model (*global*) or individual predictions (*local*); and whether they are built in the classification model itself (*self-explaining*) or are generated through a post-processing step (*post-hoc*).

Global vs. Local Rule-based approaches Hearst (1992); Brin (1998) or decision trees Béchet et al. (2000); Boros et al. (2017) provide global explainability by constructing transparent models that people can understand. However, these directions were slowly replaced by deep learning, which tends to yield better classifiers (at least with respect to accuracy). Several efforts aimed at bringing back global explainability into deep learning. For example, in the non-NLP context of high-stakes decision-making at population level, Rawal and Lakkaraju (2020) proposed a model-agnostic framework that constructs global counterfactual explanations

that provide an interpretable and accurate summary of recourses for an entire population affected by a certain problem such as bad financial credit. Closer to our work, Craven and Shavlik (1996); Frosst and Hinton (2017) both proposed distilling a neural network into a globally-interpretable model such as a decision tree.

However, most recent approaches focus on local model explainability, which preserves the underlying neural classifier and interprets its individual predictions. In this category, Hendricks et al. (2016) produced natural language explanations of individual model outputs. Han et al. (2020) used influence-based training-point ranking to study spurious training artifacts in NLP settings. Wachter et al. (2018); Karimi et al. (2020) used counterfactual explanations to understand model decisions.

Self-explaining vs. Post-hoc Self-explaining strategies make explanations an integral part of model predictions. For example, Tang et al. (2020) proposed an encoder-decoder method for relation extraction, which jointly classifies relations and decodes rules that explain the relation classifier’s decisions. Rajani et al. (2019) proposed a framework that provide both answer and explanation for a commonsense QA task. In contrast, post-hoc explanations include an additional component that generates explanations after the main model produces its decisions. In this space, Liu et al. (2018) learn a taxonomy post-hoc to better interpret network embeddings. As mentioned above, Craven and Shavlik (1996); Frosst and Hinton (2017) both proposed post-hoc strategies to distill neural network into decision trees. Li et al. (2016a); Fong et al. (2019); Hoover et al. (2020) provided post-hoc visualizations as model explanations. Belinkov et al. (2017); Peters et al. (2019); Zhao and Bethard (2020); Hewitt et al. (2021) introduced probes, i.e., models trained to predict certain linguistic properties in order to verify that the underlying neural models have learned the desired linguistic knowledge.

With respect to this taxonomy, our approach is self-explaining because our relation extractor has access solely to the context identified as important by the explainability classifier, and local because our core method explains individual predictions. However, in the latter part of this article we propose a simple strategy that converts local

explainability into global by converting the entire neural model into a set of rules using the words deemed as important in a dataset by the explainability classifier.

2.2.2 Finding rationales

From a different perspective, our approach can be seen as finding *rationales*, i.e., subsets of context that explain individual model decisions Vafa et al. (2021). Although these directions fit under local explainability (and mostly post-hoc) we discuss them separately due to their recent popularity and proximity to our work.

Some efforts in this space used gradient-based saliency mapping to determine the importance of tokens in context Baehrens et al. (2010); Simonyan et al. (2013); Devlin et al. (2018); Voita et al. (2021). However, gradients can be saturated, i.e., they may be close to zeros and, thus, lose explanatory signal. Ghorbani et al. (2019); Wang et al. (2020) also warn that gradients are fragile and they can be distorted while keeping the same prediction.

As an alternative, some researchers focused instead on attention weights in transformer networks Wiegrefe and Pinter (2019a); Mohankumar et al. (2020). However, there is also evidence that attention weights may not be good explanations Jain and Wallace (2019); Brunner et al. (2019); Kobayashi et al. (2020). Other efforts have used adversarial attacks on inputs to identify their importance. For example, HotFlip Ebrahimi et al. (2017) used word-level substitutions to impact predictions. CXPlain Schwab and Karlen (2019) calculates feature importance by masking them and comparing differences in output confidences. Feng et al. (2018); Li et al. (2016b) focused on input reduction to identify the importance of input features. Instead of reducing, Vafa et al. (2021) greedily added input information to locate meaningful rationales. However, other research has showed that input perturbation cannot always guarantee a good explanation Poerner et al. (2018).

In a different direction, surrogate approaches Ribeiro et al. (2016); Lundberg and Lee (2017) generated artificial data in the neighborhood of a prediction to be explained, by randomly hiding features from the instance and learning a surrogate model to explain the predictions. AllenNLP Wallace et al. (2019) combined adver-

serial attacks and gradient-based saliency mapping in their toolkit. Lastly, Lei et al. (2016); Situ et al. (2021) trained a generator model to produce feature importance.

Other than the problems we mentioned above, most of these approaches are either passively reflecting the model behavior or learning rationales in an unsupervised way. Because of this, these methods cannot guarantee faithfulness and plausibility. In contrast, our proposed approach provides local explanations (or rationales) that are designed to be faithful. Further, our empirical evaluation shows that our explanations are also more plausible than other rationale finding methods (see Section 4.2).

All of the approaches discussed above address the task of finding rationales. However, a relatively new direction focuses on the opposite effort: if rationales are provided by a human expert, how can they be integrated in a statistical model? For example, Bao et al. (2018) proposed a method to map discrete rationales to continuous attention, and showed that the performance on low-resource tasks can be improved by transferring these mappings from resource-rich tasks. Hancock et al. (2018) showed that human-provided natural language explanations for labeling decisions can be converted to noisy labels using a semantic parser. They empirically demonstrated that through this process they can train classifiers with comparable F1 scores considerably faster. Incorporating rationales in a classifier is a key part of the our approach. However, our method jointly trains the explanation classifier with the relation classifier, rather than depending on human rationales for the entire training data.

2.3 Semi-supervised Learning

Current neural network methods need labeled data to train. However, it is not easy to get enough labeled data in most of the real-world scenarios. It is hard, expensive and time-consuming to get enough annotations from qualified annotators. Thus, researchers propose semi-supervised learning approaches which aim to train the labeled data and unlabeled data together. Approaches relevant in NLP include, but are not limited to, bootstrapped self training, co-training, and recent prompt-based zero- or few-shot algorithms.

2.3.1 Bootstrapped Self Training

Typical bootstrapped self training approach includes three steps, first we annotate some seed data, then we train the model with the seed annotations, finally, we apply the model to the unlabeled data to get more annotations. We repeat these three steps until no more newly labeled data. This kind of approaches has been discussed over decades, Yarowsky (1995) observe that words tend to have one sense in a given discourse/document and they come up with a bootstrapped approach on word sense disambiguation. Gupta and Manning (2015) propose a bootstrapping algorithm for named entity extraction that expands the seed set using embeddings and kNN. Eyal et al. (2021) takes a syntactic search engine (Shlain et al., 2020) and applies it for bootstrapping relation extraction. They also utilize the NLG data argumentation to further improve the results.

2.3.2 Co-training

Blum and Mitchell (1998) show that there are multiple independent ways of looking at the data and the views will help each other by providing supervision to each other. Collins and Singer (1999) propose a semi-supervised algorithm that is a combination of Yarowsky (1995)’s bootstrapping with Blum and Mitchell (1998)’s co-training. Qu et al. (2018) propose a co-training framework which jointly trains a pattern module and a distributional module. The pattern module aims to select high quality patterns while the distributional module tries to learn relation representation between given entities.

However, the model in bootstrapping and co-training cannot guarantee the quality of all its predictions and it may cause the problem like semantic drift or large human effort on filtering the model annotations.

2.3.3 Prompt-based Zero- or Few-shot Learning

Recent large pre-trained language models (PLMs) with huge amount of parameters have showed the ability to handle NLP tasks with only a few examples. Researchers

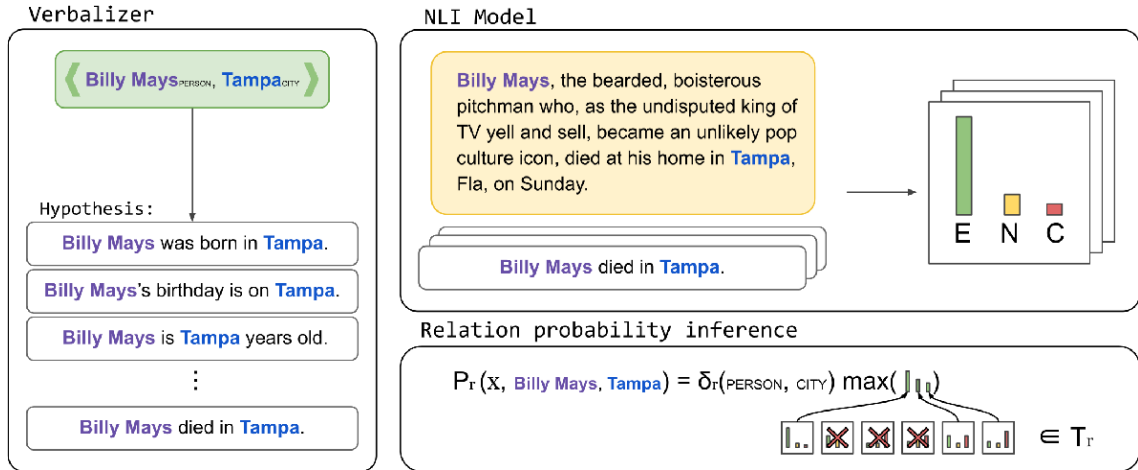


Figure 2.1: General workflow of Sainz et al. (2021)'s entailment-based RE approach.

show that we can reformatting the NLP tasks with prompt and get dramatic improvements compared to traditional fine-tuning models. Sainz et al. (2021) reformat the relation extraction task as a natural language inference (NLI) task and get improvement on TACRED dataset. Figure 2.1 shows the general workflow of this work. They show that with the help of NLI language model, they can reach decent performance on TACRED dataset in the zero- and few-shot scenarios. The limitation of this work is that it requires the PLM with NLI engine, which is not applicable to all PLMs. Wei et al. (2022) show that PLM can perform multi-hop reasoning over a chain-of-thought. Zhang et al. (2022) propose a prompt-based rule discovery and model boosting. However there is also evidences (Webson and Pavlick, 2022) show that the PLMs do not actually understand the prompt which makes their decisions unreliable. Besides, due to the large size of parameters, the large PLMs are expensive to get access and use.

CHAPTER 3

Jointly Learning of a Neural Classifier and an Explanation Decoder

In this work we demonstrate that this transition from rule- to ML-based IE can be performed such that the benefits of both worlds are preserved. In particular, we start with a rule-based relation extraction (RE) system and bootstrap a neural RE approach that is trained jointly with a decoder that learns to generate the rules that best explain each particular extraction. The contributions of our idea are the following:

(1) We introduce a strategy that jointly learns a classifier with a decoder that generates explanations for these extractions in the form of rules. We demonstrate that this approach is applicable to both event extraction and relation extraction.

(2) We extend a subset of the BioNLP 2013 GENIA event extraction (Kim et al., 2013) dataset with a set of rules designed to extract and explain three of the GENIA biomedical events: protein *phosphorylation*, *localization*, and *gene expression*. The result is a parallel dataset that aligns some of the GENIA event labels with rules that extract them. We will release this dataset for reproducibility.

(3) We evaluate our approach on the TACRED dataset (Zhang et al., 2017) and extended BioNLP 2013 GENIA event extraction (Kim et al., 2013) dataset. We demonstrate that: (a) our neural RE classifier outperforms considerably the rule-based one we started from; (b) our decoder generates explanations with high accuracy, i.e., a BLEU overlap score between the generated rules and the gold, hand-written rules of over 90%; and, (c) joint learning improves the performance of both the classifier and decoder.

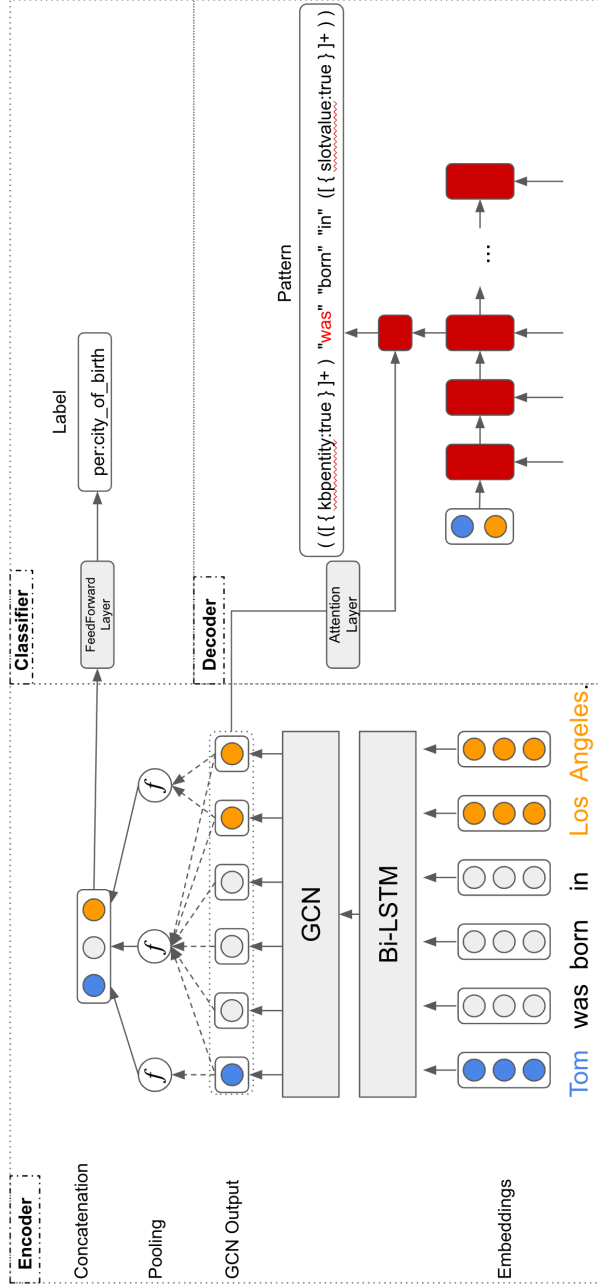


Figure 3.1: Neural architecture of the proposed multitask learning approach. The input is a sequence of words together with NER labels and POS tags. The pair of entities to be classified (“subject” in blue and “object” in orange) are also provided. We use a concatenation of several representations, including embeddings of words, NER labels, and POS tags. The encoder uses a sentence-level bidirectional LSTM (biLSTM) and graph convolutional networks (GCN). There are pooling layers for the subject, object, and full sentence GCN outputs. The concatenated pooling outputs are fed to the classifier’s feedforward layer. The decoder is a LSTM with an attention mechanism.

(4) Our approach can be easily extended to a semi-supervised setting, where we use the rules associated with the events of interest to extract additional training data with “silver” labels, i.e., where we use the rule predictions as training labels for the classifier. We show that despite the inherent noise in this process, the performance of our approach improves considerably in this semi-supervised setting.

3.1 Approach

Our approach jointly addresses classification and interpretability through an encoder-decoder architecture, where the decoder uses MTL for extraction (Task 1) and rule generation (Task 2). Figure 3.1 summarizes my approach on TACRED task¹.

3.1.1 Task 1

Due the differences between BioNLP task and TACRED task, the encoder and classifier are different from each other, We will discuss them separately.

BioNLP 2013 GENIA Given a sentence and an entity in focus, it must identify which event applies to the entity, and what is its trigger, i.e., the verbal or nominal predicates that drives the lexicalization of the event (e.g., “phosphorylation”).

We train a binary event classifier for each event type, which must identify if the corresponding event type applies to the entity under consideration, and, if so, which token in the input sentence is the event’s trigger.

The classifier uses an encoder with entity attention to encode its input. For each sentence with words w_1, \dots, w_n and a given entity z , we associate each word i with a representation x_i that concatenates three embeddings: $x_i = e(w_i) \circ e(p_i) \circ char(w_i)$, where $e(w_i)$ is the word embedding of token i , p_i is the word’s relative position to the *entity* under consideration, and $char(w_i)$ is the output of a bidirectional character-level LSTM (charLSTM) applied over w_i . $e(w_i)$ is initialized with the pretrained embeddings of Hahn-Powell et al. (2016) using the word2vec Skip-gram

¹The architecture in BioNLP is slightly different from this, we will discuss the details later.

model (Mikolov et al., 2013) trained on the full text of over 1 million biomedical papers taken from the PubMed Central Open Access Subset.² while $e(p_i)$ and $char(w_i)$ are initialized randomly. The sequence of x_i s serves as input to a word-level bidirectional LSTM (biLSTM), whose hidden states h_i s serve as input to the attention layer below.

The entity-attention layer computes a sequence of context vectors (the matrix \mathbf{C} in the equations below), which weighs the biLSTM’s hidden states by their importance to the entity z . my attention mechanism is inspired by the transformer network (Vaswani et al., 2017). Similarly, we compute the attention function on a set of keys and values that are packed together into matrices \mathbf{K} and \mathbf{V} . The difference is that my approach is entity-focused in its query, so we only compute the attention on a single query vector \mathbf{q} . Further, unlike the conventional encoder in a transformer network, we don’t produce a single vector, but a sequence of vectors (the matrix \mathbf{C}).

$$\mathbf{q} = \mathbf{W}_q \mathbf{h}_z \tag{3.1}$$

$$\mathbf{K} = \mathbf{W}_k \mathbf{H}^E \tag{3.2}$$

$$\mathbf{V} = \mathbf{W}_v \mathbf{H}^E \tag{3.3}$$

$$\mathbf{s} = \mathbf{qK} \tag{3.4}$$

$$\mathbf{a} = \text{softmax}(\mathbf{s}) \tag{3.5}$$

$$\mathbf{C} = \mathbf{V} \odot \mathbf{a} \tag{3.6}$$

where \mathbf{W}_q , \mathbf{W}_k , \mathbf{W}_v are learned matrices of dimension 200×200 , \mathbf{H}^E contains the biLSTM’s hidden states, and \mathbf{h}_z is the hidden state of the entity z from \mathbf{H}^E . We concatenate each context vector (\mathbf{C}_i) with the entity vector (\mathbf{H}_i^E) and feed the concatenated vector to two feedforward layers with a softmax function, and use its output to predict if there is a trigger in this position. We calculate the classifier’s loss using the binary log loss function.

²<https://www.ncbi.nlm.nih.gov/pmc/tools/openftlist/>

TACRED The inputs consist of a sentence $W = [w_1, \dots, w_n]$, and a pair of entities (called “subject” and “object”) corresponding to two spans in this sentence: $W_s = [w_{s_1}, \dots, w_{s_n}]$ and $W_o = [w_{o_1}, \dots, w_{o_n}]$. The goal is to predict a relation $r \in R$ (from a predefined set of relation types) that holds between the subject and object or “no relation” otherwise.

For each sentence, we associate each word w_i with a representation \mathbf{x}_i that concatenates three embeddings: $\mathbf{x}_i = \mathbf{e}(w_i) \circ \mathbf{e}(n_i) \circ \mathbf{e}(p_i)$, where $\mathbf{e}(w_i)$ is the word embedding of token i , $\mathbf{e}(n_i)$ is the NER embedding of token i , $\mathbf{e}(p_i)$ is the POS Tag embedding of token i . We feed these representations into a sentence-level bidirectional LSTM encoder (Hochreiter and Schmidhuber, 1997):

$$[\mathbf{h}_1, \dots, \mathbf{h}_n] = \text{LSTM}([\mathbf{x}_1, \dots, \mathbf{x}_n]) \quad (3.7)$$

Following Zhang et al. (2018), we extract the “K-1 pruned” dependency tree that covers the two entities, i.e., the shortest dependency path between two entities enhanced with all tokens that are directly attached to the path, and feed it into a GCN (Kipf and Welling, 2016) layer:

$$\mathbf{h}_i^{(l)} = \sigma\left(\sum_{j=1}^n \tilde{A}_{ij} \mathbf{W}^{(l)} \mathbf{h}_j^{(l-1)} / d_i + \mathbf{b}^{(l)}\right) \quad (3.8)$$

where \mathbf{A} is the corresponding adjacency matrix, $\tilde{\mathbf{A}} = \mathbf{A} + \mathbf{I}$ with \mathbf{I} being the $n \times n$ identity matrix, $d_i = \sum_{j=1}^n \tilde{A}_{ij}$ is the degree of token i in the resulting graph, and $\mathbf{W}^{(l)}$ is linear transformation.

Lastly, we concatenate the sentence representation, the subject entity representation, and the object entity representation as follows:

$$\mathbf{h}_{sent} = f(\mathbf{h}^{(L)}) = f(\text{GCN}(\mathbf{h}^{(0)})) \quad (3.9)$$

$$\mathbf{h}_s = f(\mathbf{h}_{s_1:s_n}^{(L)}) \quad (3.10)$$

$$\mathbf{h}_o = f(\mathbf{h}_{o_1:o_n}^{(L)}) \quad (3.11)$$

$$\mathbf{h}_{final} = \mathbf{h}_{sent} \circ \mathbf{h}_s \circ \mathbf{h}_o \quad (3.12)$$

where $\mathbf{h}^{(l)}$ denotes the collective hidden representations at layer l of the GCN, and $f : \mathbb{R}^{d \times n} \rightarrow \mathbb{R}^d$ is a max pooling function that maps from n output vectors to the representation vector. The concatenated representation \mathbf{h}_{final} is fed to a feedforward layer with a softmax function to produce a probability distribution over relation types.

3.1.2 Task 2

Decode a rule in a rule(such as Odin or Tokensregex) language that explains the prediction of the event classifier. That is, the rule decoder’s goal is to generate the pattern P that extracted the corresponding data point, where P is represented as a sequence of tokens in the corresponding pattern language: $P = [p_1, \dots, p_n]$. For example, the pattern $(([\{\text{kbpentity:true}\}]+)/\text{was}/\text{/born}/\text{/on}/([\{\text{slotvalue:true}\}]+))$ (where `kbpentity:true` marks subject tokens, and `slotvalue:true` marks object tokens) extracts mentions of the `per:date_of_birth` relation.

We implemented this decoder using a LSTM with an attention mechanism. In each timestep t , we generate the attention context vector \mathbf{c}_t^D by using the current hidden state of the decoder, \mathbf{h}_t^D :

$$\mathbf{s}_t = \mathbf{h}_{(L)}^E \mathbf{W}^A \mathbf{h}_t^D \quad (3.13)$$

$$\mathbf{a}_t = \text{softmax}(\mathbf{s}_t) \quad (3.14)$$

$$\mathbf{c}_t^D = \sum_j \mathbf{a}_t(j) \mathbf{h}_j^E \quad (3.15)$$

where \mathbf{W}^A is a learned matrix, and $\mathbf{h}_{(L)}^E$ are hidden representations from the encoder’s hidden states.

We feed this \mathbf{c}_t^D vector to a single feed forward layer that is coupled with a softmax function and use its output to obtain a probability distribution over the pattern vocabulary.

We use cross entropy to calculate the losses for both the classifier and decoder. To balance the loss between classifier and decoder, we normalize the decoder loss by the pattern length. Formally, the joint loss function is:

$$\text{loss} = \text{loss}_c + \text{loss}_d / \text{length}(P) \quad (3.16)$$

3.2 Experimental Results

3.2.1 Datasets

BioNLP 2013 GENIA Event Extraction I train and evaluate on three events from the BioNLP 2013 GENIA Events extraction shared task (Kim et al., 2013): Phosphorylation (P), Localization (L), and Gene Expression (GE). To facilitate comparison with previous work, I use the standard training, development, and test partitions from the original dataset. To generate data for the rule decoder, I extend this dataset with rules from the rule-based system of Valenzuela-Escárcega et al. (2018), which reported high-precision results for Phosphorylation (92%). I manually added new rules using existing syntactic templates that cover common syntactic forms of subject-verb-object patterns to cover more events. Further, because the system of Valenzuela-Escárcega et al. (2018) did not cover L and GE events, I extended it with rules for these two events. All in all, we used: 32, 20, and 21 rules for P, L, and GE, respectively. Most of these rules rely on syntactic structures denoted in terms of dependency paths to extract event arguments. From these rules, I obtained a token-level vocabulary for the rule decoder. This poses an additional challenge on my decoder, which must now decode from raw text both the semantics necessary for these events, and the syntactic patterns needed to match event arguments. Further, note that these rules do not have perfect recall, i.e., there are events in the data that are not covered by rules. In other words, the two tasks in my MTL framework are not perfectly aligned: there are data points which are part of the training examples of T1, but not of T2 (for those training examples, the loss of decoder is set to be 0).

In addition to using these rules for explainability, I used the rule-based system

to generate additional “silver” training data for these three events, by using its extractions from a collection of PubMed publications. From these papers, I extracted an additional 6592, 6321, and 2056 positive training examples for P, L, and GE, respectively. To avoid biasing the classifier to the positive classes, I also generated 3467, 3532, and 2876 negative training examples for P, L, and GE by extracting entities assign to extract evented to other event types in the BioNLP data.

TAC Relation Extraction I report results on the TACRED dataset (Zhang et al., 2017). I bootstrap my models from the patterns in the rule-based system of Angeli et al. (2015), which uses 4,528 surface patterns (in the Tokensregex language) and 169 patterns over syntactic dependencies (using Sengrex). I experimented with two configurations: *rule-only data* and *rules + TACRED training data*. In the former setting, I use solely positive training examples generated by the above rules. I combine these positive examples with negative ones generated automatically by assigning ‘no_relation’ to all other entity mention pairs in the same sentence where there is a positive example.³ I generated 3,850 positive and 12,311 negative examples for this configuration. In the latter configuration, I apply the same rules to the entire TACRED training dataset.⁴

3.2.2 Baseline

I compare my models with several competitive rule-based and neural sequence models.

Rule-based models. In my main experiments I compare with three types of rule-based models.(1) The rule-based system proposed by Valenzuela-Escárcega et al. (2018). They used their rule-based system to extract Phosphorylation events in BioNLP 2013 Genia Events (GE) task data using 42 manually written rules (which

³During the generation of these negative examples I filtered out pairs corresponding to inverse and symmetric relations. For example, if a sentence contains a relation (Subj, Rel, Obj), I do not generate the negative (Obj, no_relation, Subj) if Rel has an inverse relation, e.g., `per:children` is the inverse of `per:parents`.

⁴Thus, some training examples in this case will be associated with a rule and some will not. I adjusted the loss function to use only the classification loss when no rule applies.

we extended for my experiments). (2) The rule-based system of Zhang et al. (2017), which uses 4,528 surface patterns (in the Tokensregex language) and 169 patterns over syntactic dependencies (using Sengrex) on TACRED task.

Neural sequence models. We also compare our model on TACRED with the best non-combination method of Zhang et al. (2018). The latter method uses a LSTM and GCN combination similar to our encoder.

3.2.3 Implementation and Evaluation Details:

We use pre-trained GloVe vectors Pennington et al. (2014) to initialize our word embeddings. We use the *Adagrad* optimizer Duchi et al. (2011). For TACRED task, we apply entity masking to subject and object entities in the sentence, which is replacing the original token with a special <NER>-SUBJ or <NER>-OBJ token where <NER> is the corresponding name entity label provided by TACRED.

We used micro precision, recall, and F1 scores to evaluate the RE classifier. We used the BLEU score which captures up to 4-gram, to better capture the syntax in a rule, to measure the quality of generated rules, i.e., how close they are to the corresponding gold rules that extracted the same output. We used the BLEU implementation in NLTK Loper and Bird (2002), which allows us to calculate multi-reference BLEU scores over 1 to 4 grams. For TACRED task, we report BLEU scores only over the non 'no_relation' extractions with the corresponding testing data points that are matched by one of the rules in Zhang et al. (2017).

3.2.4 Results and Discussion

BioNLP 2013 Tables 3.1 analyzes the performance of my approach for the three events, compared against the rule-based system described in §3.2.1. These results highlight several important observations:

(1) T1 by itself performs generally worse than the rule baseline and the median BioNLP result. This is caused by: (a) the small size of this dataset, e.g., there are

	Phosphorylation (P)			Localization (L)			Gene Expression (GE)		
	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1
Rule baseline	92.68	48.12	63.35	66.13	44.44	53.16	51.08	69.79	58.98
T1	87.78	49.38	63.20	100.00	4.04	7.77	89.32	64.30	74.77
T1 + Silver	62.75	82.50	71.28	54.55	34.34	42.15	68.43	74.31	71.25
T1 + Silver + T2	84.38	68.75	75.77	76.60	39.39	52.03	69.92	71.24	70.58
BioNLP best	83.95	85.62	84.78	86.21	53.54	66.05	91.29	82.55	86.70
BioNLP median	79.83	81.57	80.64	88.55	40.91	55.89	82.58	78.11	80.09

Table 3.1: Results for the three events in the BioNLP 2013 test partition. T1 and T2 indicate the two tasks in my MTL approach, i.e., the event classifier and the rule decoder, respectively. Silver indicates that that configuration used the silver data created by the rule-based system (see §3.2.1). BioNLP best and median indicate the best/median results during the 2013 shared task. I do not include T1 + T2 results because in this configuration I observed that there is not sufficient data to train the decoder.

only 117 training examples for P; and (b) the fact that my approach uses no part-of-speech (POS) or syntactic information, which have been shown to be important for this BioNLP task (Kim et al., 2013). However, adding the silver data improves T1 performance considerably, e.g., 35 F1 points for Localization. This demonstrates that my approach provides a simple but effective platform for semi-supervised learning.

(2) Most importantly, jointly training for classification and explainability helps the classification task (T1) itself. As shown in the tables, combining T1 and T2 generally improves F1 scores considerably, e.g., 4 F1 points for Phosphorylation and 10 for Localization. To my knowledge, this is the first NLP work to demonstrate that aiming for interpretability also helps the main task addressed. All in all, I approach the median performance in the shared task, a respectable result considering that my approach uses only raw text as input, whereas all participants in this shared task used some form of syntactic representation. Importantly, my approach outperforms considerably the rule-based method of Valenzuela-Escárcega et al. (2018), which served as the starting point of this work (see Section 3.2.2).

(3) The only negative results in my experiments are the GE results in the test partition, where T1 outperforms both T1 + Silver and T1 + Silver + T2. I hypothesize that this is caused by the larger training data for this event, e.g., there are 6 times more training samples for GE than P, which allows the T1 classifier to learn by itself, without the scaffolding offered by MTL, and the additional (noisy) data in the silver dataset. This suggests that my approach is best suited for EE scenarios with minimal training data, an important subset of information extraction tasks.

	BLEU	Exact Matches	Non-exact, Explainable Matches
P	93.80	86.11	2/15
L	83.78	84.33	1/9
GE	78.99	76.45	10/43

Table 3.2: Evaluation of decoded rules, on the BioNLP development partition. BLEU measures the overlap with hand-written rules. Exact Matches shows the percentage of decoded rules that exactly match hand-written ones. Explainable Matches shows the number of decoded rules that do not match exactly hand-written ones, but were considered good explanations by human experts.

But are the decoded rules actually interpretable? To answer this, I compared in Table 3.2 the decoded rules against the hand-written rules that matched in the BioNLP development partition. That is, I performed this analysis on the subset of the development partition, where each data point is accompanied by a matching hand-written rule. This reduced this dataset to approximately 60% of the total BioNLP development set. In particular, I analyzed 108, 82, and 296 event instances with matching rules for P, L, and GE events, respectively. The table shows that my rules have high BLEU overlap with hand-written rules, e.g., 93% for P, and, by and large, they exactly match them. I believe this is an exciting result, as it shows that my approach is able to decode directly from the raw text the declarative semantics necessary for the task, as well as the syntactic patterns that match the event arguments.

Lastly, Table 3.3 shows examples of typical decoding errors, ranging from partial mistakes that do not affect the interpretability of rules to complete decoding mistakes. As I mentioned above, I cannot ensure the validity of the generated rules with my current approach. Table 3.3 shows that this indeed happens in my output. For example, the decoder generates a binary operator such “!=” without the left operand (first row in the table).

Hand-written Rule	Decoded Rule
<pre>trigger = [lemma = /phosphorylate/ & ! word = /(?)(de auto)/ & tag = /^(V JJ)/ & ! mention = ModificationTrigger]</pre>	<pre>trigger = [lemma = /phosphorylate/ & ! word = /(?)(de auto)/ & tag = /^(V JJ)/ & ! = ModificationTrigger]</pre>
<pre>theme : BioChemicalEntity = > nsubjpass prep_of ? /conj_(and or nor) nn cc/ { , 2 }</pre>	<pre>theme : BioChemicalEntity = > nsubjpass prep_of ? /conj_(and or nor) nn cc/ { , 2 }</pre>
<pre>trigger = [lemma = /detect localiz local releas secret translocat/ & ! word = /(?)(de prep_(by of)/]</pre>	<pre>trigger = [lemma = /detect localiz local releas secret translocat/ & ! word = /(?)(de prep_(by of)/]</pre>
<pre>theme : BioChemicalEntity = prep_of ? appos ? /conj_(and or nor) cc nn/ { , 2 }</pre>	<pre>theme : BioChemicalEntity = < /conj_(and or nor) / ? /conj_(and or nor) cc nn prep_of/ { , 2 }</pre>
<pre>trigger = [lemma = /phosphorylate/ & ! word = /(?)(de auto) / & ! outgoing = / prep_(by of) /]</pre>	<pre>trigger = [lemma = /phosphorylate/ & ! word = /(?)(de auto) / & ! outgoing = / mention = ModificationTrigger]</pre>
<pre>theme : BioChemicalEntity = < /conj_(and or nor) / ? /conj_(and or nor) cc nn prep_of / { 2 } site : Site ? = nn < dobj ? / prep_(at on) / num ?</pre>	<pre>cause : BioChemicalEntity ? = < xcomp ? (nsubj agent < vmod) / appos nn conj_ (and or nor) cc / { 2 }</pre>
<pre>theme : BioChemicalEntity = (dobj xcomp) / conj_ (and or nor) dep cc nn prep_of / { 2 } { > > [word = by] { 2 } site : Site ? = dobj ? / prep_(at on) nn conj_ (and or nor) cc / { 2 }</pre>	<pre>theme : BioChemicalEntity = (dobj xcomp) / conj_ (and or nor) dep cc nn prep_of / { 2 } { > > [word = by] { 2 } site : Site ? = dobj ? / prep_(at on) nn conj_ (and or nor) cc / { 2 }</pre>

Table 3-3: Examples of mistakes in the decoded rules. The first column shows hand-written rules, while the second shows the rules decoded by my approach from sentences where the corresponding hand-written rules matched. I highlight in the hand-written rules the tokens that were missed during decoding (false negatives) in green, and in the decoded rules I highlight the spurious tokens (false positives) in red. The first row lists a partial mistake, which does not affect the interpretability of the decoded rule, since it only misses one token that can be inferred by the human experts from context. The second row lists a partial mistake, which impacts the semantics of the rule. For example, the decoder missed that the path between the trigger and the theme argument starts with an optional **prop_of** and **appos**. This rule was marked as partially correct because some simple syntactic patterns, e.g., **nn**, can still be correctly matched by the decoder. The last row lists a larger decoding error that was marked as completely incorrect by the annotator. For example, in the last decoded rule, the decoder generated an incorrect cause argument, which does not exist in the data, as well as an incorrect syntactic pattern for the theme argument, i.e., the protein being phosphorylated.

Approach	Precision	Recall	F1	BLEU
Rule-only data				
Rule baseline	86.9	23.2	36.6	–
My approach	60.0	36.7	45.5	90.3
w/o decoder	58.7	36.4	44.9	–
w/o classifier	–	–	–	88.3
Rules + TACRED training data				
C-GCN	69.9	63.3	66.4	–
My approach	70.2	64.0	67.0	92.4
w/o decoder	71.2	62.3	66.5	–
w/o classifier	–	–	–	91.6

Table 3.4: Results on the TACRED test partition, including ablation experiments (the “w/o” rows). I experimented with two configurations: *Rule-only data* uses only training examples generated by rules; *Rules + TACRED training data* applies the previous rules to the training dataset from TACRED.

Model	Precision	Recall	F1	BLEU
20% of rules	74.9	20.1	31.7	96.9
40% of rules	69.0	26.9	38.8	90.8
60% of rules	62.7	29.7	40.3	88.8
80% of rules	57.3	36.5	44.6	89.4

Table 3.5: Learning curve of my approach based on amount of rules used, in the *rule-only data* configuration. These results are on TACRED development.

TACRED Table 3.4 reports the overall performance of my approach, the baselines, and ablation settings, for the two configurations investigated. I draw the following observations from these results:

(1) The rule-based method of Zhang et al. (2017) has high precision but suffers from low recall. In contrast, my approach that is bootstrapped from the same information has 13% higher recall and almost 9% higher F1 (absolute). Further, my approach decodes explanatory rules with a high BLEU score of 90%, which indicates that it maintains almost the entire explanatory power of the rule-based method.

(2) The ablation experiments indicate that *joint* training for classification and explainability helps both tasks, in both configurations. This indicates that performance and explainability are inter-connected.

(3) The two configurations analyzed in the table demonstrate that my approach performs well not only when trained solely on rules, but also when rules are combined with a training dataset annotated for RE. This suggests that my direction may be a general strategy to infuse some explainability in a statistical method, when rules are available during training.

(4) Table 3.5 lists the learning curve for my approach in the *rule-only data* configuration when the amount of rules available varies.⁵ This table shows that my approach obtains a higher F1 than the complete rule-based RE classifier even when using only 40% of the rules.⁶

(5) Note that the BLEU score provides an incomplete evaluation of rule quality, I also did a error analysis on those decoded rules which did not exactly match the gold rules. I found that some of decoded rules can still be used as explanation for the extraction even it did not perfectly match the gold one (as shown in Table 3.6).

3.3 Conclusion

In this work We introduce an interpretable approach for event extraction that jointly trains an event classifier with a component that translates the classifier’s decisions into interpretable extraction rules. We implement this approach using an encoder-decoder architecture, where the decoder jointly optimizes the decoding of extraction rules and event classification. We show that the performance of our approach on BioNLP further improves when trained on automatically-labeled data generated by a rule-based system. We introduce a strategy that jointly bootstraps a relation extraction classifier with a decoder that generates explanations for these extractions, using as sole supervision a set of example patterns that match such relations. Our experiments on the TACRED dataset demonstrated that our approach outperforms the strong rule-based method that provided the training patterns by

⁵For this experiment I sorted the rules in descending order of their match frequency in training, and kept the top $n\%$ in each setting.

⁶The high BLEU score in the 20% configuration is due to the small sample in development for which gold rules exist.

Hand-written Rule	Decoded Rule
(({{kbentity:true}}+)"based""in"({{slotvalue:true}}+))	(({{kbentity:true}}+)"in"({{slotvalue:true}}+))
(({{kbentity:true}}+)"CEO"({{slotvalue:true}}+))	(({{kbentity:true}}+)"president"({{slotvalue:true}}+))
{{ner:TITLE}=slot>/nsubj.*/{ner:PERSON}=entity>cop}	{ner:PERSON}=entity>/nn—compound—amod—appos/{ner:TITLE}=slot

Table 3.6: Examples of mistakes in the decoded rules. I highlight in the hand-written rules the tokens that were missed during decoding (false negatives) in green, and in the decoded rules I highlight the spurious tokens (false positives) in red

9 F1 points, while decoding explanations at over 90% BLEU score. Further, our evaluations on both BioNLP and TACRED datasets with this encoder-decoder architecture demonstrate that the decoder generates interpretable rules, and that the joint training improves the performance of the event classifier. All in all, our work suggests that it is possible to marry the interpretability of rule-based methods with the performance of neural approaches.

However, the generated rules are not of perfect quality. There are two significant flaws: 1. The rules are not guaranteed to be executable because there is no grammar checking during the decoding period; and 2. The decoder prefers to repeat what it saw in the training examples. Further, we plan to use other approaches to generate the rules. The key information in most of the rules is the lexical elements (e.g. the trigger predicates in the syntactic rules). We discuss such a solution in the next chapter, a sequence modeling task that locate these important words and generate rules from the sequence model output.

CHAPTER 4

Multitask Learning of Neural Relation and Explanation Classifiers

Many domains such as medical, legal, or finance, require that decision making be not only accurate but also trustworthy. Thus, understanding what the underlying model captures is a critical requirement in such applications. To this end, previous efforts addressed this limitation by adding explainability to neural models, which have come to dominate natural language processing (NLP) (Manning, 2015). These explanations can be categorized along two main aspects: whether they explain a complete model (*global*) or individual predictions (*local*); and whether they are an integral part of the classification model itself (*self-explaining*) or are generated through a post-processing step (*post-hoc*) (see Related Work for a longer discussion). Most of recent proposed efforts focus on the *local* and *post-hoc* explanations (Ribeiro et al., 2016; Shapley, 1952; Schwab and Karlen, 2019). These directions have a few advantages such as modularity and simplicity. However, they also have two important drawbacks: these types of explanations are not guaranteed to be *faithful* to the original model to be explained, and are not *actionable*, i.e., even if they correctly explain an imperfect classification, there is no clear path towards correcting the underlying model because “changing one thing changes everything” in a neural network (Sculley et al., 2015).

This chapter focuses on addressing the limitations of these local and post-hoc explainability approaches by providing a self-explanatory neural architecture (i.e., explanations are part of classification) that can provide both local and global explanations. In particular, we propose an approach for relation extraction (RE) that *jointly* learns how to explain and predict. Intuitively, our approach trains two classifiers: an explainability classifier (EC), which labels words in the textual context where the relation is expressed as important or not for the relation to be extracted, and a relation classifier (RC), which predicts the relation that holds between two

given entities using *only* the words deemed as important. As such, our approach is *self-explanatory* because of inter-dependency between RC and EC, and generates *faithful* explanations that correctly depict how the relation classifier makes a decision (Vafa et al., 2021).

Also, in the Chapter 3, we show that joint training a classifier with an explanation decoder leads to performance improvements on the classifier and the decoder can generate rules which can defend the classifier’s decision. However, there are two major flaws in the decoded rules: 1. With no grammar checking in decoding period, the rules are not guaranteed to be executable; and 2. The decoder prefer to repeat what it saw in the training examples. Due to these issues, the generated rules can be only seen as explanations to the classification results and prevent us from apply them to further usages such as model induction¹ or semi-supervised learning (SSL). In this work, we address these issues by converting the model outputs into rules.

The contributions of this chapter are the following:

(1) We introduce a hybrid strategy to jointly train the EC and RC. Our method trains the EC as a supervised classifier when information about which words are important for a relation exists. For example, in this article we use a small set of linguistic rules to identify the important words in the relation’s context. For example, in the sentence “*John was born in France,*” such a rule may identify the words *born* and *in* as important. Importantly, our approach requires minimal supervision for explanations, e.g., we report results when using an average of 7 rules per relation type on one dataset and fewer on another dataset. For the more common situation where training examples are not associated with such rules, we train using a semi-supervised strategy: we treat EC’s labels as latent variables, and learn the best assignment that maximizes the performance of the RC.

(2) We evaluate our approach on two datasets: TACRED (Zhang et al., 2017) and CoNLL04 (Roth and Yih, 2004). For (partial) explainability information, we select from the surface rules provided with the dataset (Zhang et al., 2017; Chang and

¹Infer explainable s models (in our case, rules) as transparent alternative to the neural model.

Manning, 2014) as well as from a small set of syntactic rules developed in-house using the Odin framework (Valenzuela-Escárcega et al., 2016b). Our evaluation demonstrates that jointly training for prediction and explainability improves the performance of the relation classifier considerably on CoNLL04, and maintains the same level of performance on TACRED when compared with a state-of-the-art neural relation classifier. Importantly, our method achieves its best performance when using an average of 7 rules per relation type on TACRED and 4 rules per relation type for CoNLL04, which indicates that only minimal guidance from such rules is needed.

(3) More relevant for the goals of this work, we also evaluate our method for explainability using two strategies. The first strategy is automated and focuses on the capacity of our method to identify the same words in the context as the ones identified by rules, to verify that our approach indeed encodes the proper linguistic knowledge. Thus, this evaluation looks at examples associated with rules. In this situation, we measure the overlap between the words identified by the EC as important and the words used by rules using standard precision, recall, and F1 scores. The second strategy relies on *plausability*, i.e., can the machine explanations be understood and interpreted by humans (Wiegrefe and Pinter, 2019b; Vafa et al., 2021)? To this end, we compare the tokens identified by the EC against human annotations of the context words marked as important for the relation. In both evaluations, our approach achieves considerably higher overlap with rules/human annotations than other strong baselines such as saliency mapping (Simonyan et al., 2013), LIME (Ribeiro et al., 2016), SHAP (Lundberg and Lee, 2017), CXPlain (Schwab and Karlen, 2019), and greedy rationales (Vafa et al., 2021).

(4) We also explore the feasibility of transforming the local explanations into global ones. That is, instead of using the EC to explain individual predictions, we introduce a simple algorithm that converts the tokens marked as important into a set of rules that becomes a new, fully-explainable model that approximates the behavior of the neural RC. We compare the performance of this rule-based model with the performance of the rules written by domain experts, as well as with the neural

RC model. The results show that our rule-based model has a considerably higher performance than the manually-written rules, approaching the performance of the neural classifier within a reasonable gap. In some real-world scenarios, this gap may be an acceptable cost, as the generated rule-based model provides *actionable* explainability. That is, when a rule is incorrect, a domain expert can improve it without impacting other parts of the models (Valenzuela-Escárcega et al., 2016a).

4.1 Approach

At a high level, our approach consists of two main components: a neural relation classifier with an integrated explainability classifier, and a rule generation component, which generates a rule-based model from the explainability information, i.e., context words that explain a relation, provided by the neural model.

4.1.1 Walkthrough Example

Before getting into the details of our approach, we highlight its key functionality with the walkthrough example shown in Table 4.1. Consider the sentence “*John’s daughter, Emma, likes swimming.*”. As shown in Table 4.1 (a), the task input includes: the raw text in the sentence, the entities participating in the relation (denoted as subject and object) and their types (**PERSON** here), and the syntactic dependency parse tree. Table 4.1 (b) shows the output of our relation classifier (RC) and explanation classifier (EC): the RC returns the predicted relation `per:children`, while the EC labels the word *daughter* as the trigger of the predicted relation. Step (c) shows the information that is collected for rule generation. This information includes: the two entities, the relation predicted, the tokens identified by the EC as the rationale for the relation, and the shortest syntactic path connecting the two entities with the rationale words. The output rule generated by our approach is shown in step (d). This rule is written in the Odin language (Valenzuela-Escarcega et al., 2015; Valenzuela-Escárcega et al., 2016b). The rule captures the relation to be predicted (`per:children`), its trigger (*daughter*), the two arguments and their type,

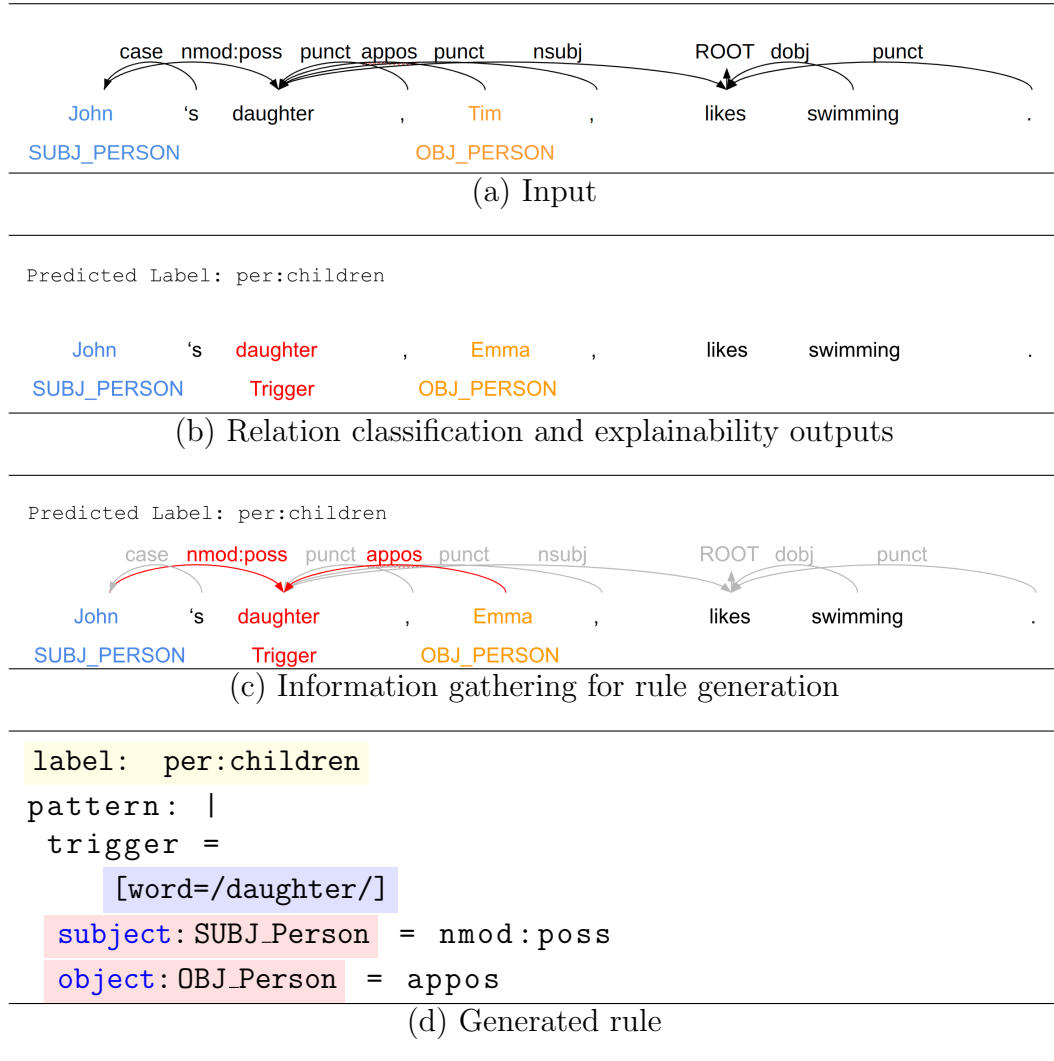


Table 4.1: Walkthrough example of our approach. The task input includes information about the entities participating in the relation (denoted as subject and object) and their types (PERSON here). Our neural architecture, which includes both a relation and explanation classifier, predicts the relation that holds between the two entities (per:children here, i.e., the object is the child of the subject), as well as which words best explain the decision (in red). In step (c), the rule generator collects the necessary information from the annotated sentence, i.e., the shortest syntactic dependency path that connects the two entities with the explanation words (in red in the figure). Step (d) shows the generated rule in the Odin language.

e.g., `subject` with the type `SUBJ_Person`, and the syntactic paths between each argument and the trigger phrase, e.g., `nmod:poss` for the subject argument. Note that in this simple example, the trigger consists of a single word, but, in general, an Odin rule can take any arbitrary sequence of words as its trigger.

This example shows that our method can be deployed in two ways. First, one can use the joint RC and EC neural classifiers, which predict relations that hold between pairs of entities, as well as local explanations (or rationales) that explain the prediction. Alternatively, a different class of users may use the output of step (d), which, once applied on large text collections, contains a set of rules that describes multiple relation classes. This usage may be preferred in real-world situations that have to mitigate the “technical debt” of neural methods, i.e., reduce the cost of maintaining these models over time (Sculley et al., 2015). Although not within the scope of this work, other works have shown that rule-based methods for IE can be improved and maintained at a low cost (Valenzuela-Escárcega et al., 2016a).

4.1.2 Joint Relation and Explainability Classifiers

As mentioned, our approach jointly trains an explainability classifier (EC) and a relation classifier (RC). The RC is a multiclass classifier that distinguishes between actual relation labels seen in training. We couple the RC with a binary classifier that first predicts if the current example contains an actual relation or no relation (marked as `no_relation`). For conciseness, we call this classifier the no relation classifier (NRC). The EC is a binary word-level classifier, which labels words in the sentence that contains the relation with 1, if they are important for the underlying relation, or 0, otherwise.

We start this section with the description of the overall training procedure, and follow with details about the individual classifiers.

Training Procedure

The overall flow of the training procedure is shown in Figure 4.1. This flow is temporally split in two periods: a burn-in period, which is fully supervised, followed

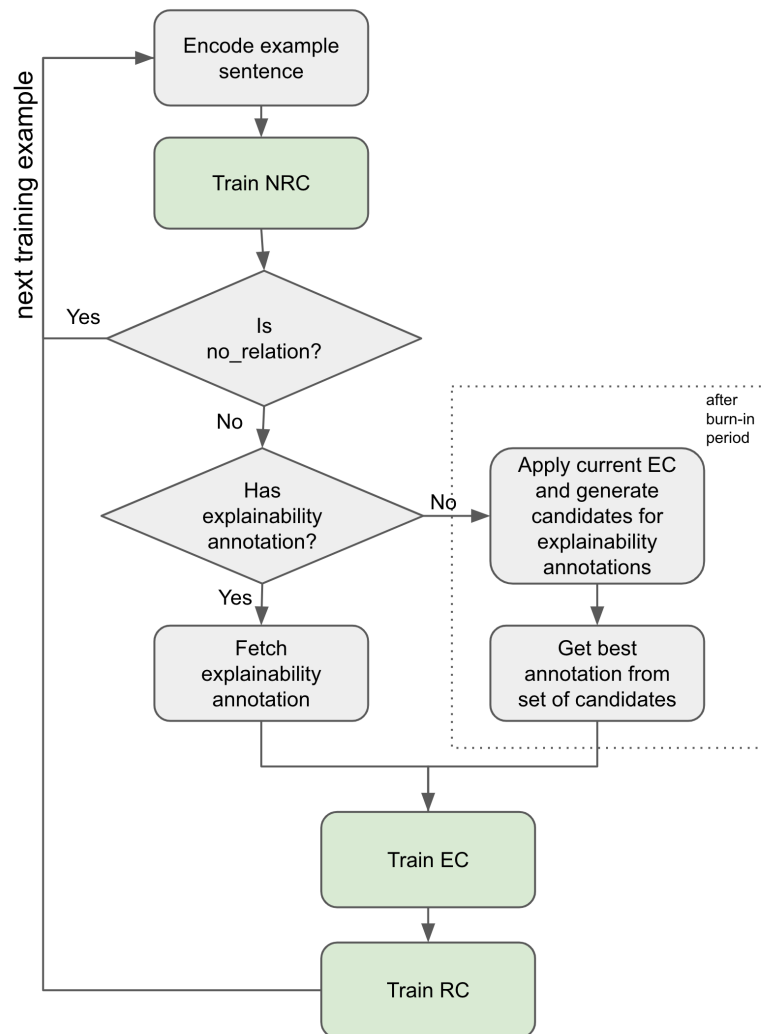


Figure 4.1: Flow of our semi-supervised training procedure for an individual training example. All the “Train . . .” blocks (green background) involve parameter updates of the corresponding classifiers. These updates are shown here for an individual training example, but are batched in the actual implementation.

by a period that includes semi-supervised learning (SSL). This distinction is necessary because while all training examples in this task are guaranteed to have RC labels, most examples will *not* have gold explainability annotations. For example, for the sentence “[CLS] John was born in London.”, the training data contains information that there is a `per:city_of_birth` relation between *John* and *London*, but may not contain information about which words are critical for this relation (*born* and *in*).

Burn-in period In this stage, shown in the left-hand side of Figure 4.1, we only use the training examples that are associated with explainability annotations (see Section 4.1.2 for details on how these annotations are generated). Here we train initial versions of the three classifiers: NRC, EC, and RC (see Section 4.1.2 for details on the three classifiers).

The purpose of this stage is to initialize the three classifiers such that they can be successfully used to reduce the search space for explainability annotations in the next SSL stage.

After burn-in In this stage, the training procedure is exposed to all training examples, including those without annotations for explainability. That is, for such training examples, we simply have annotations for the relation labels (or `no_relation`), without knowing which context words explain the underlying relation. In such situations, the right-hand side of the flow in Figure 4.1 is used, which triggers two additional components: one to generate candidates for explainability annotations, and one to choose the best sequence of word labels (i.e., which words are important and which are not).

For the former component, exhaustively generating all possible label assignments is prohibitively expensive (i.e., $O(2^N)$ for a sequence of length N). To mitigate this cost, we rely on the prediction scores of the EC classifier to reduce the number of candidates. That is, if the score of the binary EC for a given token is higher than a threshold (t_{up}), we directly annotate the corresponding token as important (i.e., assign label 1); if this score is lower than a second threshold (t_{low}), we annotate

the token as not important (label 0); and, lastly, if the the score is between the two thresholds, we generate two candidate labels for this token (both 0 and 1). For example, given an input sentence “[CLS] [SUBJ-PER] was born in [OBJ-CITY] .”,² and these prediction scores from the EC: [0.12, 0.14, 0.19, 0.86, 0.25, 0.15, 0.01], using $t_{up} = 0.8$ and $t_{low} = 0.2$, we produce the following candidate label sequences: [0, 0, 0, 1, 0, 0, 0] and [0, 0, 0, 1, 1, 0, 0], because the assignment for the token *in* is ambiguous according to the two thresholds.

Once these candidates are generated, we loop through all the generated sequences of word labels, and pick the sequence \hat{c} that yields the highest score for the correct relation label according to the current RC:

$$\hat{c} = \underset{c}{\operatorname{argmax}} p(R|c) \quad (4.1)$$

where R is the gold label of the instance, $p(R|c)$ is the score at the gold label R predicted by the RC for a given annotation candidate c . In the previous example, if the RC scores of the two candidates for the correct relation label `per:city_of_birth` are 0.8 and 0.5, we select the first candidate over the second one.

Then this sequence of labels is used as (pseudo) gold data to train the EC on this training example. This guarantees that each training example has annotations (gold, or generated through the above procedure) for both EC and RC.

Because these two components rely on having reasonable predictions from the EC and RC classifiers, we found it beneficial to include the previous burn-in period, where these classifiers are trained using the (small) amount of supervision available.

Explainability Annotation

As mentioned, a key part of our approach requires that EC annotations be available for a few of the training examples. To this end, rather than relying on manual annotations, which are expensive, we repurpose rules that extract the same relation. The intuition behind our approach is that if a rule exists that extracts the same

²The entities participating in a relation are masked with their named entity labels (see Section 4.1.2).

relation label as the gold label in a training example, then this rule (and, specifically, its lexical elements) can be seen as an explanation of the extraction. In particular, in this article we focus on the TACRED dataset (Zhang et al., 2017), and select explanations from two sets of rules:

(1) **Surface rules:** The TACRED project generated a set of high-precision rules for the task, implemented in the Tokensregex language (Chang and Manning, 2014). For example, the rule `SUBJ-PER was born in * OBJ-CITY`³ extracts a `per:city_of_birth` relation between a person named entity (the subject) and a city named entity (the object) if the sequence *was born in* occurs somewhere between the two entities. For such rules, we label all tokens contained in the rule (e.g., *was, born, in*) with the label 1 (i.e., they are important for explainability), and all other tokens in the sentence with 0.

```
label: per:employee_of
pattern: |
  trigger =
    [lemma=/work|write|play|consult|serve/]
  subject: SUBJ_Person = <acl? nsubj
  object: OBJ_Organization = nmod
```

Label(s) to assign to a match.

Lexical constraints on the relation’s predicate.

`argName:ArgType`, where `ArgType` indicates the named-entity category expected for this argument.

Figure 4.2: An example of a relation extraction rule in the Odin language that extracts the `per:employee_of` relation. The rule is driven by verbal triggers such as *work*, *play* or *serve*. The relation’s arguments (the subject and object) are identified through both semantic constraints (subject must be `Person`), and syntactic ones (subject must be attached to the trigger through a certain syntactic dependency pattern: an optional (?) adnominal clause (`acl`), followed by a nominal subject (`nsubj`)). This rule would extract a `per:employee_of` relation from the text “... *Joe is a research scientist working at IBM...*”.

³We simplified the Tokensregex syntax for readability.

(2) Syntactic rules: In initial experiments, we observed that the TACRED surface rules have high precision but low recall. To improve generalization, we also wrote 38 syntax-based rules using the Odin language (Valenzuela-Escárcega et al., 2016b).⁴ Figure 4.2 shows an example of such a rule. For these syntactic rules, we marked all their lexical elements (typically the trigger predicates such as *work* or *write* in the figure) as important (label 1), and all other words as not important (label 0).

Classifiers

As mentioned, the building blocks of our approach consist of three classifiers: the no-relation classifier (NRC), the relation classifier (RC), and the explainability classifier (EC). These are jointly trained using the schema previously described in this section. Below we describe their individual details, which are also visualized in Figure 4.3.

SpanBERT Encoder and NRC: We follow the entity masking schema from (Zhang et al., 2017) and replace the subject and object entities with their provided named entity (NE) labels, e.g., “[CLS] [SUBJ-PER] was born in [OBJ-CITY] ...”. We feed this input to a SpanBERT-based (Joshi et al., 2020) encoder:

$$[\mathbf{h}_0, \dots, \mathbf{h}_n] = \text{Encoder}([w_0, \dots, w_n]) \quad (4.2)$$

where w_n is the id of the word at position n , and h_n is the hidden representation generated by the encoder.

We add the special masking tokens for SUBJ--* and OBJ--* to the vocabulary so that the encoder can handle them properly. We implement the NRC using a feedforward layer with a sigmoid function on top of the encoder’s [CLS] token.

Explainability Classifier (EC): We implement the EC as a binary token-level classifier, where the positive label indicates that the corresponding token is important for the underlying relation. Section 4.1.2 discusses how these annotations are

⁴All these rules are included in this submission as supplemental material.

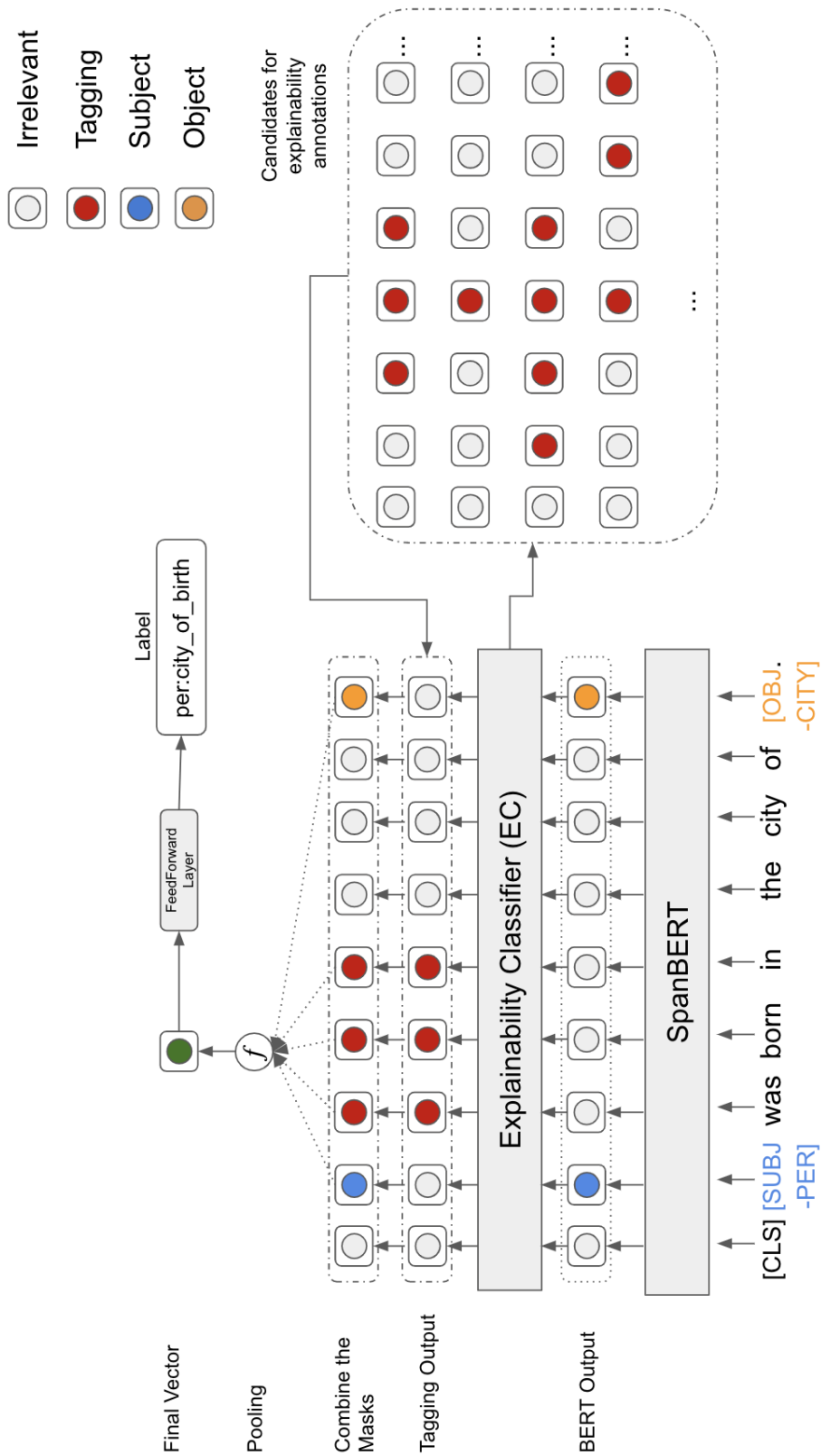


Figure 4.3: Neural architecture of the proposed multitask learning approach. The entity tokens (subject in blue and object in orange) are masked with their named entity labels, e.g., SUBJ–Person, in the actual implementation.

generated from rules; Section 4.1.2 explains the SSL training procedure when these annotations are not available.

Relation Classifier (RC): Crucially, the RC relies *only* on words that are marked as important by the EC, or are part of the subject/object entity. This is an important distinction between our approach and other relation extraction methods, which typically rely on the [CLS] representation for classification.

In the next section, we empirically show that this latter strategy is considerably less explainable than ours. This is because the [CLS] representation aggregates information from all tokens in the sentence, whereas our method focuses only on the important ones.

We build the aggregated representation of the important context words, subject and object as follows:

$$\mathbf{h}_{final} = f(\mathbf{h}_{ctx_1:ctx_n}) \circ f(\mathbf{h}_{subj_1:subj_n}) \circ f(\mathbf{h}_{obj_1:obj_n}) \quad (4.3)$$

where \mathbf{h} denotes the hidden representations produced by the encoder, $f : \mathbb{R}^{d \times n} \rightarrow \mathbb{R}^d$ is the average pooling function that maps from n output vectors into one; and \circ is the concatenation operator. Importantly, \mathbf{h}_{ctx} iterates only over words marked as important by the EC.

The concatenated representation \mathbf{h}_{final} is fed to a feedforward layer with a softmax function to produce a probability distribution \mathbf{p} over relation types.

The three classifiers are trained using the following joint loss function:

$$loss = loss_{nrc} + loss_{ec} + loss_{rc} \quad (4.4)$$

$$loss_{nrc} = -(t^n * \log(y^n) + (1 - t^n) * \log(1 - y^n)) \quad (4.5)$$

$$loss_{ec} = -(t^e * \log(y^e) + (1 - t^e) * \log(1 - y^e)) \quad (4.6)$$

$$loss_{rc} = -\log(p(R)) \quad (4.7)$$

where the losses of the NRC and EC ($loss_{nrc}$ and $loss_{ec}$, respectively) are implemented

using binary cross entropy. For both, t indicates the corresponding gold label, and y is the respective sigmoid’s activation. The loss of the RC ($loss_{rc}$) is implemented using categorical cross entropy, where $p(R)$ is the likelihood predicted by the model for the correct relation R .

4.1.3 Aggregating Local Explanations into a Global, Rule-based Model

As mentioned, the last component of our approach aggregates all local RC and EC predictions into a single rule-based model that explains the overall behavior of the RC and EC models. As such, the produced rule-based model brings global explainability to the task. We will show in Section 4.2 that this transformation comes with a cost in performance, but this cost might be acceptable in scenarios where such RE extraction must be deployed, maintained, and improved over a long period of time.

Rule Generation

As shown in Table 4.1 (c), our relation and explanation classifiers produce all the information necessary to generate an Odin rule. At a high-level, the Odin rules we employ here follow a predicate (or `trigger` in the Odin language) and argument template, where all arguments are connected to the trigger using a syntactic dependency path. This information is either provided by our classifiers, e.g., we use the rationale tokens identified by the EC as triggers, or can be automatically extracted from the sentence, e.g., we represent the syntactic connections between predicate and arguments using the shortest path that connects them in the syntactic dependency tree. Algorithm 1 describes this entire rule generation process:

Algorithm 1 Rule Generator

Require: set of annotated sentences, \mathbb{S}
Require: model output \mathbb{L} (from RC) and \mathbb{T} (from EC)

```

0:  $\mathbb{R} \leftarrow \emptyset$ 
0: for every sentence  $s$  in  $\mathbb{S}$  do
0:   Get the subject and object entity  $e_s$  and  $e_o$  from  $s$ 
0:   Get the predicted relation label  $l$  from  $\mathbb{L}$  and the rationale words  $t$  from  $\mathbb{T}$ 
0:   if  $s$  hasn't been extracted by any manual rule then
0:     Find the shortest path  $p_s$  between  $t$  and  $e_s$  in the dependency tree
0:     Find the shortest path  $p_o$  between  $t$  and  $e_o$  in the dependency tree
0:      $r \leftarrow$  empty Odin rule template
0:     Assign  $l$  to  $r$  as the label to match
0:     Assign  $t$  to  $r$  as the relation predicate (or trigger)
0:     Assign  $p_s$  and  $p_o$  to  $r$  as the argument patterns.
0:      $\mathbb{R} \leftarrow \mathbb{R} \cup \{r\}$ 
0:   end if
0: end for
Ensure: set of generated rules  $\mathbb{R} = 0$ 

```

4.2 Experimental Results

4.2.1 Data Preparation

We report results on the TACRED dataset (Zhang et al., 2017) and CoNLL04 dataset (Roth and Yih, 2004). As discussed in Section 4.1.2, we provided rules for explanation supervision. For the TACRED data, we selected rules from the surface patterns of Angeli et al. (2015), and we combined them with an additional set of 38 syntactic rules in the Odin language (Valenzuela-Escárcega et al., 2016b) that were manually created by one of the authors from the training data. For CoNLL04 data, we selected from a set of 19 syntactic rules in Odin language, 10 of which are borrowed from the TACRED syntactic rules, since the two datasets shared some overlapping relations.

These rules match 20.7% of positive examples in the TACRED training set and 24.2% of positive examples in the CoNLL04 training set. On average, 7.27 rules are assigned to each TACRED relation, and 3.8 rules are assigned to each CoNLL04

relation.

Importantly, our approach does *not* use rules at evaluation time. However, we take advantage of all existing rules to automatically evaluate the quality of the explanations generated by our method. In the TACRED dataset, the combined set of rules from (Angeli et al., 2015) and our syntactic rules match 23.9% data points in the development set, and 23.9% examples in the test set; in the CoNLL04 dataset, the syntactic rules match 20.1% of examples and 20.9% of examples respectively. We use only these matches for an automated evaluation of explainability (discussed below).

4.2.2 Baselines

Relation Extraction Baselines

For the relation extraction task, we compare our approach with three baselines: an extended version of the rule-based approach of Angeli et al. (2015), a neural state-of-the-art RE approach based on SpanBERT (Joshi et al., 2020), and a neural approach with built-in explainability (Lei et al., 2016):

- **Rule-based Extraction.** As mentioned in Section 4.2.1, we employ two sets of rules. First, we use the tokensregex surface rules from (Angeli et al., 2015), which are executed in the Stanford CoreNLP pipeline (Manning et al., 2014a). Second, we include the Odin syntactic rules we developed in-house, which are executed in the Odin framework (Valenzuela-Escárcega et al., 2016b).⁵
- **SpanBERT.** SpanBERT (Joshi et al., 2020) is an extension of the original BERT (Devlin et al., 2018) that: (1) masks continuous random spans instead of random tokens, and (2) trains the span boundary representations to predict the full content of the masked span without depending on individual token representations within it. SpanBERT outperforms BERT in many tasks including relation extraction. Further, SpanBERT is currently the best TACRED

⁵The rule set from (Angeli et al., 2015) also included some syntactic rules, but we found out that they only matched the simpler `per:title` relation, so we did not use them.

BERT-based model available in the HuggingFace transformer library (Wolf et al., 2020) that does not use any external resources, or does not rely on complex hybrid architectures.

- **Unsupervised Rationale.** Lei et al. (2016) proposed an approach that combines an *unsupervised* rationale generator with a task-specific classifier, both of which are trained to operate together (similar to our approach). However, there are several key differences between their method and ours. First, their explanation generator cannot incorporate human input (as we do through rules); instead, it is indirectly guided by the loss of the downstream task. Second, their architecture is more complex, i.e., they use two distinct encoders: one for explanation generation and another for the downstream task (both of which are implemented with recurrent networks). We adapt this method to our RE framework, by replacing our EC with their rationale generation algorithm (which is a token-level binary classifier that produces an output compatible with our EC). For a fair comparison with our method, we kept the other components unchanged. That is: we encode the input text using the same SpanBERT, then we use their generated rationales and the given entities as pooling mask to construct the final vector to feed into the relation classifier⁶. Originally, Lei et al. (2016) proposed their approach to sentiment analysis and text retrieval. Bastings et al. (2019) extended this method and adapted it to a natural language inference task. To our knowledge, this is the first attempt to apply this explainability strategy to relation extraction.

Note that all baselines as well as our method receive inputs in the standard TACRED format,⁷ which contains tokenized sentences, spans of the subject and object mentions, and the types of the two entity mentions. The only difference between the RC baselines and our method is that, as discussed in Section 4.2.1, our approach receives information on which sentence tokens were matched by rules during the burn-in

⁶We also observed that our architecture that uses a single, shared transformer encoder performs better than their original architecture with two distinct encoders.

⁷We converted the CoNLL04 data into the same format as TACRED.

training period.

Explainability Baselines

For explainability, we compare our approach against eight baselines, detailed below. These are all popular explanation approaches published in recent years. Most of them provide a feature importance score for each feature⁸ and most of them are post-hoc⁹. Here, we labeled the top N positive features identified by the baselines as important.¹⁰ In the first quantitative evaluation of explainability (Section 4.2.4), for all baselines we set N to be equal to the number of words in the gold explanation. Importantly, this means that all baselines have an unfair advantage over our approach, which is non-parametric with respect to N , i.e., it identifies N on the fly for each sentence. In the second, qualitative evaluation of explainability (Section 4.2.4), N is a hyper parameter that we tuned to maximize the baselines’ performance.¹¹

We detail the eight explainability baselines below:

- **Attention.** Attention weights have been proposed as an explanation mechanism by Bahdanau et al. (2014). Followup work debated the validity of this strategy (Jain and Wallace, 2019; Wiegrefe and Pinter, 2019a; Kobayashi et al., 2020). However, because this remains a popular approach, we include attention weights as a baseline in this work. In particular, we use the attention weights from the last layer of a “vanilla” SpanBERT model, i.e., one that is trained on top of the [CLS] representation, without an EC. For this baseline, we label as important the top N tokens with the highest [CLS] attention weights.
- **Saliency Mapping.** The feature importance score of the token x_i is determined by the highest prediction’s accumulated gradients in each dimension of the

⁸Except for greedy adding and unsupervised rationale approaches which rely on labeling the features to be included in the rationale, similar to what we do.

⁹Except for unsupervised rationale approach which trains a generator together with the rest of the model, similar to what we do.

¹⁰We ignored tokens part of the subject and object entities for a fair comparison.

¹¹We used $N = 3$ for TACRED, and $N = 1$ for CoNLL04.

token in the embedding layer. These scores are obtained through a back-propagation of the highest prediction’s probability. Although there are different implementations of the gradient saliency mapping approach (Devlin et al., 2018; Voita et al., 2021), we use the simple back-propagation approach from (Simonyan et al., 2013).

- **LIME.** Ribeiro et al. (2016) proposed the LIME framework, which provides explanations to any black-box classifier. LIME samples the neighbors of the local instance \mathbf{x} to be explained, by generating perturbations of the tokens in \mathbf{x} . Then, it trains a linear separator from these samples to approximate the local behavior of the model. The coefficients of the separator are later used as the feature importance score.
- **Unsupervised Rationale.** As mentioned in the previous sub-section, this baseline replaces our EC with the unsupervised method of Lei et al. (2016). Here we use this method as an explainability baseline.
- **SHAP.** The Shapley value (Shapley, 1952) is a cooperative game theory concept that calculates the score of feature x_i by taking into account its interactions with all other subsets of features. Similar to what LIME does, Lundberg and Lee (2017) also train a linear model to approximate the local behavior around the sampled neighbors. However, unlike LIME, which uses cosine similarity or L2 distance as its kernel, they propose a SHAP kernel which is determined by the number of permutations of features.
- **CXPlain.** Schwab and Karlen (2019) proposed an approach called CXPlain that explains the decisions of any machine-learning model by measuring the importance of the model’s features. To this end, CXPlain masks each token x_i in \mathbf{x} , and calculates the score of x_i by comparing the output with the masked input $\bar{\mathbf{x}}$ against the output that relies on the original input \mathbf{x} . The difference between the two is calculated using a causal objective.
- **Greedy Adding.** Instead of randomly sampling from perturbations or masking

the features, Vafa et al. (2021) proposed a method that greedily adds the features to the input data point. That is, it starts with an empty rationale, and each time it selects and adds the feature that increases the probability of the correct label y_t the most. The process repeats as long as the confidence in predicting y_t keeps increasing.

- **All Words between Subject and Object.** We have observed that most of the important words that determine the relation between the entities occur in the span between the two entities. To capture this intuition, we implemented this simple baseline, which simply includes all the words between subject and object in its rationale.

Similarly to the RC settings discussed in the previous sub-section, these baselines and our method rely on the standard TACRED input format. However, our EC is semi-supervised, i.e., during burn-in it receives explainability annotations generated by rules. In contrast, the EC baselines do not rely on rule information.

4.2.3 Implementation and Evaluation Details

Before introducing our results, we discuss key details about our implementation and evaluation.

To avoid the RC classifier overfitting on the names in the sentence (Suntwal et al., 2019), we mask the subject and object entities by replacing the original tokens in these entities with a special token, i.e., `SUBJ--<NE>` or `OBJ--<NE>`, where `<NE>` is the corresponding name entity type provided in the dataset. We use the pre-trained SpanBERT to encode the input sentence. For the TACRED dataset, which is organized to contain a single relation per sentence, we feed the `[CLS]` token to the final linear layer for relation classification. However, for the CoNLL04 data, which typically contains more than one relation per sentence, we used the concatenation of the `[CLS]` hidden state and the average pooling of `[SUBJ]` and `[OBJ]` hidden state embeddings. This was necessary to distinguish between the different relations that co-occur in the same sentence. We used the AdamW optimizer (Loshchilov and

Hutter, 2019) for all training processes. We evaluated all RC classifiers using the standard micro precision, recall, and F1 scores. All neural models were trained using 5 different random seeds; we report the average scores and standard deviation over these seeds for RC.

For explainability, we report two evaluations.¹² For the first, automated evaluation, we use only the data points that are associated with a rule that produces the same relation label as the gold data. For these examples, we consider the lexical artifacts of the rule as gold information for explainability (as explained in §4.1.2). We measure the overlap between the important words produced by the analyzed methods and this data using precision, recall, and F1 scores. We also include a second, qualitative evaluation on the *plausability* of the generated explanations (Vafa et al., 2021), where a more plausible explanation will overlap more with a relation explanation manually generated by domain experts. For this evaluation, we sampled 100 and 60 data points from the test sets of TACRED and CoNLL04, respectively. These are sentences where our model predicted a relation, and where there is *no* gold annotation from rule-based method (i.e., no rule matched). We split these data points into two sets: a subset where our method predicted the correct relation, and one where it did not. In other words, in the former set, we investigate the capacity of the explainability methods to explain correct predictions, while in the latter we analyze their capacity to explain why the machine was incorrect. Two domain experts¹³ manually annotated rationales for these sentences and the provided relation labels. The annotators were asked to identify the minimal set of tokens that explain the provided relation. Or, in other words, identify the tokens that when replaced with other words change the relation to be predicted. For example, in the sentence *SUBJ-PER was born in OBJ-CITY.*, if we replace the words *born in* with other words (e.g., *moved to*), the relation between the subject and object changes. Importantly, to avoid any potential bias, the two annotators worked completely

¹²We did not include an evaluation of faithfulness, which is typically done by post-hoc explainability approaches (Ribeiro et al., 2016; Schwab and Karlen, 2019) because our approach is faithful by design, i.e., our RC only relies on the tokens identified by the EC.

¹³These were two of the authors.

independently of each other, and had no access to explanations provided by any algorithm.¹⁴ We evaluate the overlap between the machine and human rationales using the same standard precision, recall, and F1 measures.

Appendix A lists the hyperparameters used to train all RC and EC models.

Lastly, we evaluate the quality of the generated rule-based model. To this end, we evaluated two sets of rules: rules generated from the training sentences,¹⁵ and rules generated over the test set. In the latter scenario, we do not use any gold data. That is, we rely on the predicted relation labels (from the RC) and rationales (from the EC) to generate rules. Thus, the latter setting is akin to transductive learning, i.e., where the model has access to the unlabeled data from the testing partition, but no access to any human annotations. We evaluate the performance of these rule-based models using the same micro precision, recall, and F1 scores as the first RC evaluation.

4.2.4 Results and Discussion

In this section, we introduce and discuss the results for both relation and explainability classification. We conclude this section with an error analysis that highlights some typical errors in our models.

Relation Extraction

Tables 4.2 and 4.3 report the RE performance of all methods discussed on the TACRED and CoNLL04 datasets. The results of all statistical approaches are averaged over three random seeds. For all these models we report average performance and standard deviation in the tables. We draw the following observations from these tables:

- First, the SSL variant of our approach improves considerably over the equivalent burn-in only setting (i.e., training just on the data points that have matching

¹⁴To encourage reproducibility, we release the annotations at <https://github.com/clulab/releases/tree/master/cl2022-twoflints/dataset>

¹⁵We filter our training relations which matched a gold rule, since there is already a rule assigned to them

Approach	Precision	Recall	F1
Baselines			
Rules	85.82	24.21	37.77
SpanBERT (Joshi et al., 2020)	69.97±0.58	70.20±1.73	70.07±0.73
Unsupervised Rationale	69.24±0.40	69.05±1.86	69.14±0.83
Our Approach			
Burn-in Only	51.06±3.57	48.32±2.33	49.61±2.42
Full Model	72.02±0.90	69.11±1.82	70.52±0.54

Table 4.2: Relation extraction results on the TACRED test partition. We used the pre-trained SpanBERT-large. Our full model trains on the entire training partition using the SSL method discussed in Section 4.1.2. The “burn-in only” setting trains just on the training subset that has annotations from rules.

Approach	Precision	Recall	F1
Baselines			
Rules	81.6	16.82	27.90
SpanBERT (Joshi et al., 2020)	81.30±4.89	71.01±5.11	75.78±4.79
Unsupervised Rationale	83.91±2.88	74.88±1.44	79.11±1.01
Our Approach			
Burn-in Only	62.71±2.27	53.32±0.95	57.63 ±1.39
Full Model	83.01±2.16	76.30±3.08	79.46±0.92

Table 4.3: Relation extraction results on the CoNLL04 test partition. We used the pre-trained SpanBERT-large. Our full model trains on the entire training partition using the SSL method discussed in Section 4.1.2. The “burn-in only” setting trains just on the training subset that has annotations from rules.

rules). The improvement is 20.91% F1 (absolute) on TACRED, and 21.83% (absolute) on CoNLL04. These results highlight the importance of SSL for this task.

- Second, our approach is slightly better than SpanBERT on TACRED, and yields a statistically-significant improvement of nearly 4% F1 (absolute) on CoNLL04.¹⁶ This indicates that jointly training for classification and explainability helps the classification task itself (or, in the worst case, does not hurt relation classification). Table 4.3 also shows that our approach has the highest RE recall on CoNLL04, higher than the vanilla SpanBERT by 5%. All in all, this suggests that explainability also serves as a disambiguator in situations where multiple relations co-occur in the same sentence (the common setting in CoNLL04) by narrowing the text to just the context necessary for the relation at hand. As further evidence that performing RC on top of explanations helps disambiguate the underlying text, the standard deviation of our approach on CoNLL04 is five times smaller than that of SpanBERT.
- Interestingly, the unsupervised rationale method approaches the performance of our full model on both datasets. However, as we will show in the next sub-section, this comes with considerably worse explanations.
- Lastly, our approach nearly doubles the F1 score of the rule-based approach on TACRED, and more than doubles it on CoNLL04. This is caused by large improvements in recall, which highlights the importance of hybrid strategies that combine rules and neural components.

To understand the runtime overhead introduced by the EC, we compared our method’s runtimes during training and inference against the runtime of the vanilla SpanBERT. The average training time of our method is 0.37 sec/batch in the burn-in period and 0.38 after burn-in. In contrast, the average training time of SpanBERT

¹⁶We performed statistical significance analysis using non-parametric bootstrap resampling with 1000 iterations.

Approach	Precision	Recall	F1
Attention	30.28	30.28	30.28
Saliency Mapping	30.22	30.22	30.22
LIME	30.45	36.84	32.49
Unsupervised Rationale	4.65	79.53	8.51
SHAP	31.27	31.27	31.27
CXPlain	53.60	53.60	53.60
Greedy Adding	40.47	50.53	40.81
All words in between SUBJ & OBJ	71.48	86.33	78.21
Our Approach	95.63	97.92	95.76

Table 4.4: Automated evaluation of explainability on TACRED, in which we compare explainability annotations produced by these methods against the lexical artifacts of rules.

is 0.06 sec/batch.¹⁷ The inference time for both our model and SpanBERT is 0.10 sec/batch on the same device. The larger overhead in training is caused by: (a) back-propagating through a larger computational graph due to the joint EC and RC loss, and (b) iterating through multiple candidate explanations. We measured the average number of explanation candidates to be 85 in the first training epoch after burn-in period, and 22 after 10 epochs. However, considering that inference time are similar, we believe that the training overhead is justified by the additional explainability functionality included in the framework.

Quantitative Evaluation of Explainability

The results of the automated evaluation of explainability in Tables 4.4 and 4.5 show that our approach generally improves explainability quality considerably. Post-hoc explanation methods do not provide the same explanation quality compared to our method, which actively models explainability. Note that the high performance of annotating all the words between subject and object is caused by the fact that most data points in this evaluation are associated with surface rules, which prefer shorter contexts that are more likely to contain only significant information. Nevertheless, the 20% F1 gap between this strong baseline and our method indicates that our

¹⁷All times measured on an NVIDIA RTX 3090 GPU.

Approach	Precision	Recall	F1
Attention	69.44	69.44	69.44
Saliency Mapping	42.42	42.42	42.42
LIME	62.45	89.39	68.45
Unsupervised Rationale	5.47	86.94	9.84
SHAP	34.85	34.85	34.85
CXPlain	50.00	50.00	50.00
Greedy Adding	23.24	54.55	29.58
All words in between SUBJ & OBJ	72.99	96.59	77.29
Our Approach	99.29	100	99.52

Table 4.5: Automated evaluation of explainability on CoNLL04, in which we compare explainability annotations produced by these methods against the lexical artifacts of rules.

Num of Rules	Precision	Recall	F1
Relation Classification			
Up to top 1 (0.98 rules/relation)	72.48	66.23	69.21
Up to top 5 (3.56 rules/relation)	72.97	69.02	70.94
Up to top 10 (5.02 rules/relation)	69.30	71.64	70.45
All rules (7.27 rules/relation)	71.15	71.13	71.14
Explainability Classification			
Up to top 1 (0.98 rules/relation)	74.62	85.35	75.02
Up to top 5 (3.56 rules/relation)	92.19	94.06	91.28
Up to top 10 (5.02 rules/relation)	91.06	95.62	91.22
All rules (7.27 rules/relation)	95.63	97.92	95.76

Table 4.6: Learning curve of our approach on TACRED based on amount of rules used. In each experiment, we use up to top k rules per relation type; the number in parentheses is the actual average number of rules per type.

method successfully learns how to generalize beyond these simple scenarios.

However, we note that these results are not terribly surprising: our method is trained to generate explanations that mimic lexical artifacts of rules, while the other explainability baselines have not been exposed to rules during their training. Thus, this evaluation is necessary (to validate that our approach is learning to do what we intended, which is to mimic the lexical artifacts of rules) but not sufficient. In the next sub-section, we will show that our approach overlaps with human explanations much more than all other explainability baselines.

Approach	Precision	Recall	F1
Attention	41.39	20.60	26.50
Saliency Mapping	18.73	35.58	23.41
LIME	14.31	26.03	18.09
Unsupervised Rationale	4.73	69.66	8.30
SHAP	13.86	22.85	16.79
CXPlain	28.84	55.06	36.48
Greedy Adding	31.59	33.52	30.16
Our Approach	74.72	61.20	62.05

Table 4.7: TACRED evaluation of the plausability of explanations, which measures the overlap between machine explanations and human annotations. For each method, we pick the higher scores between the two human annotators.

Table 4.6 lists a learning curve for our approach on TACRED, as we vary the amount of rules available per relation. That is, for each relation, we use up to top k rules, where k varies from 1 to 10. In the table we include results for both relation and explainability classification using the same measures as the previous tables. The table shows that even in the “up to top 5 rules” configuration (which means an average of 3.6 rules per relation type in practice), our model obtains a close F1 to the our best model with good explainability. This result indicates that our approach performs well with minimal human supervision for explanation guidance. Note that we do not include the learning curve for CoNLL04 since there are only 19 rules applied to this dataset, which translates into only 3.8 per relation type.

Qualitative Evaluation of Explainability

Tables 4.7 and 4.8 lists the results of our evaluation of the plausability of explanations by comparing them against human annotations of explainability. Similar to evaluations of machine translation, we choose the higher scores between the machine methods and any of the two human annotators. Note that the human annotators had a Kappa agreement (McHugh, 2012) of 69.8% on labeling the same tokens as part of an explanation. This is considered moderate (Landis and Koch, 1977), which we found encouraging considering the complexity of the task and the fine granularity of the annotations. We investigated the differences between the human

Approach	Precision	Recall	F1
Attention	61.06	30.30	38.94
Saliency Mapping	18.79	39.39	24.43
LIME	22.14	53.33	30.09
Unsupervised Rationale	5.35	74.55	9.31
SHAP	18.18	36.36	23.27
CXPlain	21.21	44.55	27.82
Greedy Adding	33.33	38.03	32.21
Our Approach	65.15	59.24	58.97

Table 4.8: CoNLL04 evaluation of the plausability of explanations, which measures the overlap between machine explanations and human annotations. For each method, we pick the higher scores between the two human annotators.

annotators and observed that they are caused either by legitimate annotation errors or by the fact that there are multiple valid rationales for a given relation. For example, in the sentence *OBJ-PER is the CEO and president of SUBJ-ORG*, the relation `org:top_membersemployees` can be explained either by the tokens *CEO* or *president*.

The two tables indicate that our approach generates explanations that have considerably higher overlap with human-generated explanations, even though all data points part of this evaluation were chosen to *not* have a matching rule. This suggests that our approach generates high-quality explanations of its predictions regardless of whether it has seen the underlying pattern or not. Moreover, the recall of our approach is much higher than that of the other post-hoc explanations, which have not been exposed to rules during training. This shows that with a small amount of supervision, the generated explanations can be better aligned with human intuitions. The fact that our method outperforms considerably the unsupervised rationale approach of Lei et al. (2016), which is driven solely by relation classification performance, further emphasizes that a “human-in-the-loop” method such as ours is necessary to yield meaningful explanations.

We include several examples of the generated rationales in Figures 4.5, 4.4, 4.7, and 4.6. These examples indicate that most of the baselines are noisier, i.e., they contain a considerable amount of false positives (words that should not be part of

Our Approach	Pauliina Miettinen , a defender and goalkeeper for Finland , was named Tuesday as the new coach of Women 's Professional Soccer inaugural champion Sky Blue . Gold label: per:origin; predicted label: per:countries_of_residence
Saliency	Gonzalez is the Pauliina Miettinen , a defender and goalkeeper for Finland , was named Tuesday as the new coach of Women 's Professional Soccer inaugural champion Sky Blue . Predicted label: per:countries_of_residence
LIME	Pauliina Miettinen , a defender and goalkeeper for Finland , was named Tuesday as the new coach of Women 's Professional Soccer inaugural champion Sky Blue . Predicted label: per:countries_of_residence
SHAP	Pauliina Miettinen , a defender and goalkeeper for Finland , was named Tuesday as the new coach of Women 's Professional Soccer inaugural champion Sky Blue . Predicted label: per:countries_of_residence
CXPlain	Pauliina Miettinen , a defender and goalkeeper for Finland , was named Tuesday as the new coach of Women 's Professional Soccer inaugural champion Sky Blue . Predicted label: per:countries_of_residence
Greedy Adding	Pauliina Miettinen , a defender and goalkeeper for Finland , was named Tuesday as the new coach of Women 's Professional Soccer inaugural champion Sky Blue . Predicted label: per:countries_of_residence

Attention Weights

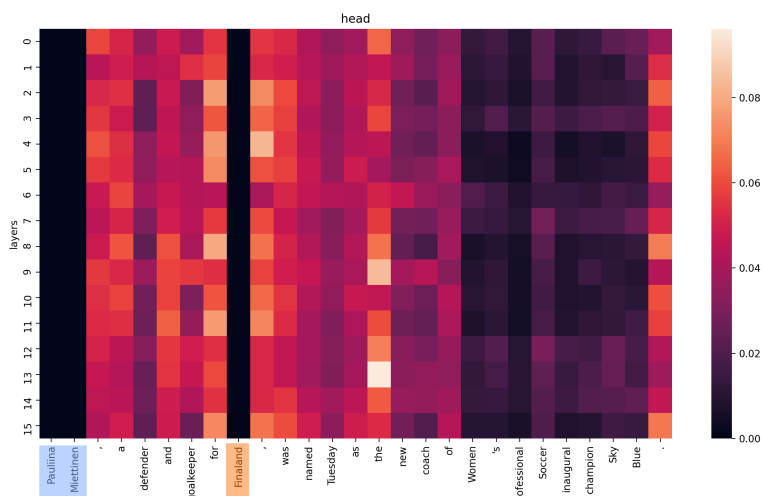


Figure 4.4: Examples of explainability annotations on TACRED for a *correct* RC prediction. The subject and object entities, which are provided in the task input, are highlighted in blue and orange. The important tokens for explainability identified by the various methods are highlighted in red. The bottom of the figure shows the heatmap of $[CLS]$ attention weights for BERT’s 16 heads (the lighter the color the higher the weight). We shrink the weight range margin to make the color scale distinguishable. For a fair comparison, we masked the subject and object in the attention weights.

Our Approach	The proportion stood at 38.7 percent , down 0.5 percentage points from the first half , said Xia Nong , deputy head of the Industrial Policy Department of the National Development and Reform Commission (NDRC) , at a press conference in Beijing . Gold label: org:subsidiary; predicted label: org:subsidiary
Saliency	The proportion stood at 38.7 percent , down 0.5 percentage points from the first half , said Xia Nong , deputy head of the Industrial Policy Department of the National Development and Reform Commission (NDRC) , at a press conference in Beijing . Predicted label: org:subsidiary
LIME	The proportion stood at 38.7 percent , down 0.5 percentage points from the first half , said Xia Nong , deputy head of the Industrial Policy Department of the National Development and Reform Commission (NDRC) , at a press conference in Beijing . Predicted label: org:subsidiary
SHAP	The proportion stood at 38.7 percent , down 0.5 percentage points from the first half , said Xia Nong , deputy head of the Industrial Policy Department of the National Development and Reform Commission (NDRC) , at a press conference in Beijing . Predicted label: org:subsidiary
CXPlain	The proportion stood at 38.7 percent , down 0.5 percentage points from the first half , said Xia Nong , deputy head of the Industrial Policy Department of the National Development and Reform Commission (NDRC) , at a press conference in Beijing . Predicted label: org:subsidiary
Greedy Adding	The proportion stood at 38.7 percent , down 0.5 percentage points from the first half , said Xia Nong , deputy head of the Industrial Policy Department of the National Development and Reform Commission (NDRC) , at a press conference in Beijing . Predicted label: org:subsidiary

Attention Weights

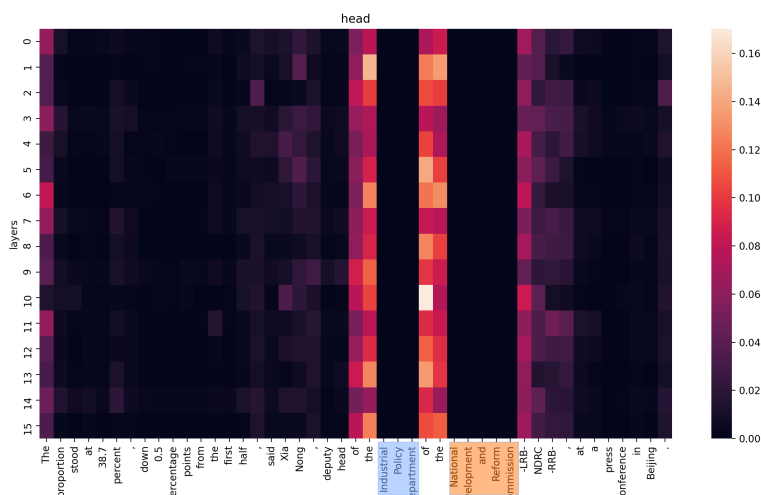


Figure 4.5: Examples of explainability annotations on TACRED for an *incorrect* RC prediction. This figure follows the same convention as Figure 4.4.

Our Approach	“ ‘ We ’re in for a long haul , ’ ’ said Dave Olson of the Payette National Forest in Idaho , where more than 200 fires continued to burn. Gold label: <code>no_relation</code> ; predicted label: <code>Work_For</code>
Saliency	“ ‘ We ’re in for a long haul , ’ ’ said Dave Olson of the Payette National Forest in Idaho , where more than 200 fires continued to burn. Predicted label: <code>Work_For</code>
LIME	“ ‘ We ’re in for a long haul , ’ ’ said Dave Olson of the Payette National Forest in Idaho , where more than 200 fires continued to burn. Predicted label: <code>Work_For</code>
SHAP	“ ‘ We re in for a long haul , ’ ’ said Dave Olson of the Payette National Forest in Idaho , where more than 200 fires continued to burn. Predicted label: <code>Work_For</code>
CXPlain	“ ‘ We ’re in for a long haul , ’ ’ said Dave Olson of the Payette National Forest in Idaho , where more than 200 fires continued to burn. Predicted label: <code>Work_For</code>
Greedy Adding	“ ‘ We ’re in for a long haul , ’ ’ said Dave Olson of the Payette National Forest in Idaho , where more than 200 fires continued to burn. Predicted label: <code>Work_For</code>

Attention Weights

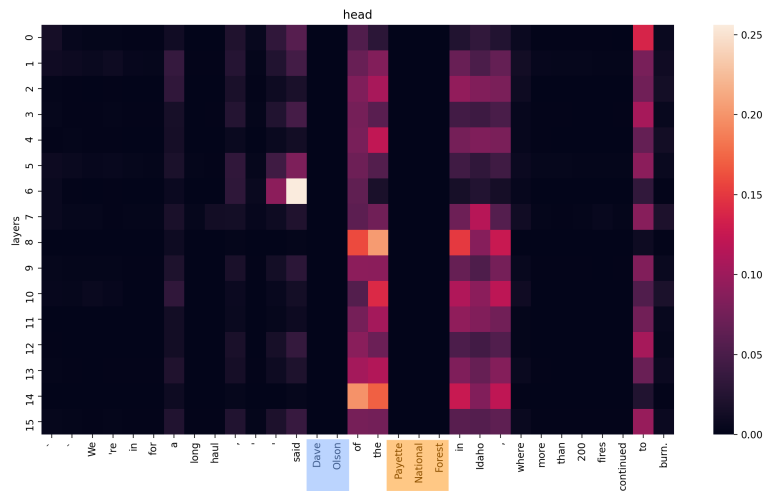


Figure 4.6: Examples of explainability annotations on CoNLL04 for an *incorrect* RC prediction. This figure follows the same convention as Figure 4.4.

Our Approach	‘ ‘ The U.S. has the biggest stock of chemical arms in the world , and it is trying to obstruct other countries from having their own , ’ ’ said Arab League Secretary-General Chedli Klibi . Gold label: Work_For; predicted label: Work_For
Saliency	█ ‘ ‘ The U.S. has the biggest stock of chemical arms in the world , and it is trying to obstruct other countries from having their own , ’ ’ said Arab League Secretary-General Chedli Klibi . Predicted label: Work_For
LIME	‘ ‘ The U.S. has the biggest stock of chemical arms in the world , and it is trying to obstruct other countries from having their own █ , ’ ’ said Arab League Secretary-General Chedli Klibi . Predicted label: Work_For
SHAP	‘ ‘ The U.S. has the biggest stock of chemical arms in the world , and it is trying to obstruct other countries from having their own , ’ ’ said Arab League Secretary-General Chedli Klibi . Predicted label: Work_For
CXPlain	‘ ‘ The U.S. has the biggest stock of chemical arms in the world , and it is trying to obstruct other countries from having their own , ’ █ said Arab League Secretary-General Chedli Klibi . Predicted label: Work_For
Greedy Adding	‘ ‘ The U.S. has the biggest stock of chemical arms in the world , and it is trying to obstruct other countries from having their own , ’ ’ said Arab League Secretary-General Chedli Klibi . Predicted label: Work_For

Attention Weights

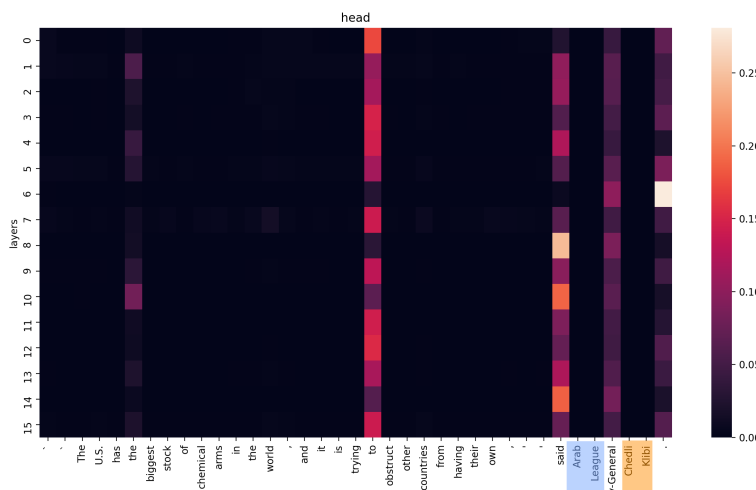


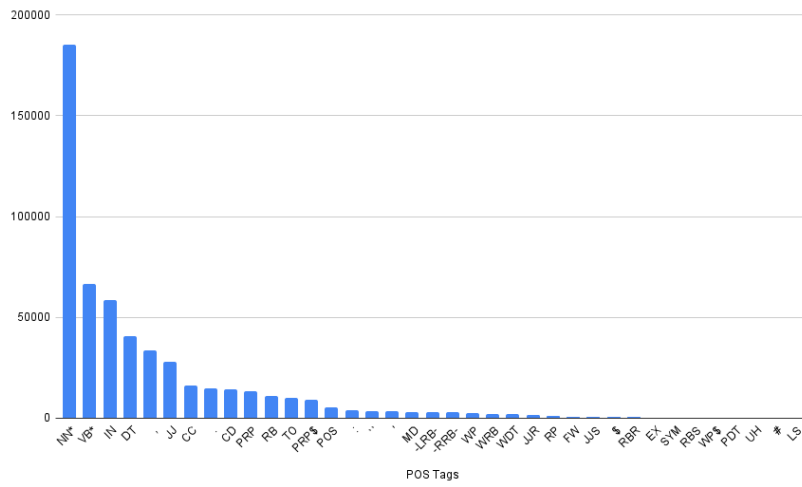
Figure 4.7: Examples of explainability annotations on CoNLL04 for a *correct* RC prediction. This figure follows the same convention as Figure 4.4.

the rationale) and false negatives (words that should be included but are not). In contrast, our method does a better job focusing on the right explanation tokens.

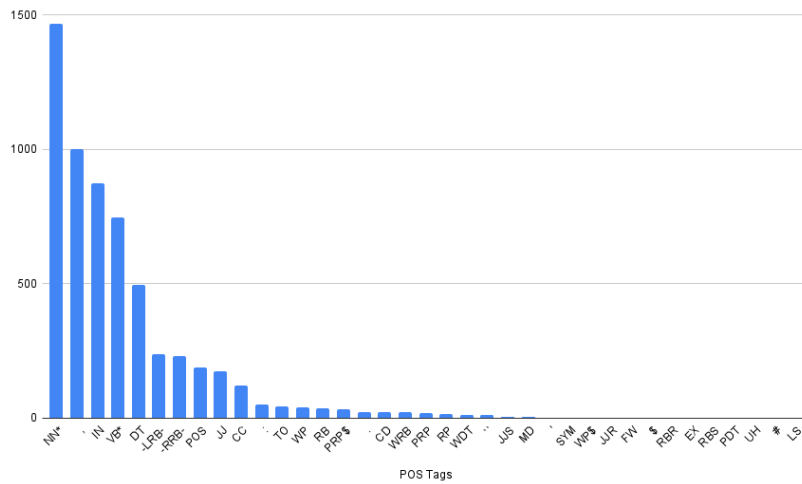
In the example in Figure 4.5, both our RC model and vanilla BERT predicted the correct relation. However, our method labels only the preposition *of* and the determiner *the* as its explanation, while other baselines such as LIME and SHAP completely missed them. Greedy adding and CXPlain label more irrelevant words in the context such as *(* and *press conference*. The attention weights do capture the key words, but we can clearly see additional noise surrounding the entities. In the example in Figure 4.4, both our model and vanilla model predicted the incorrect relation. Our model labels the preposition *for*, which provides a strong hint for its (possibly) incorrect prediction (`per:countries_of_residence`). In contrast, the baselines focus more on the nouns such as *defender* and *champion*. Applying the substitution heuristic indicates that the preposition *for* is necessary for the explanation (e.g., changing it to *against* changes the relation), while the nouns are not relevant. In this example, the attention weights are almost completely noisy.

In Figure 4.7, both our model and the vanilla SpanBERT model produce the correct prediction. The words *Secretary-General* clearly explain the *Work_For* relation in the explanations generated by our model and greedy adding. The other baselines do not provide meaningful explanations here. In Figure 4.6, which shows an incorrect prediction, only our model can defend its prediction by its explanation. The baseline approaches cannot provide valid explanations to defend the prediction at all. We also find that with the explanation provided from our model, one can argue that the predicted relation is actually correct, and we should change the gold label instead.

Lei et al. (2016) state that rationales should be short, coherent, and be sufficient for the correct prediction. However, short does not necessarily mean simple. To highlight this point, Figure 4.8 compares the distribution of POS tags in the TACRED test partition with the distribution of POS tags that participate in explanations in the same partition. We draw two observations from this data. First, to extract plausible rationales, our EC has to diverge from the distribution of POS tags in the data in a non-trivial way. For example, the frequency of verbs (VB*), prepositions



Counts of POS tags in the test partition.



Counts of POS tags in the explanations in the test partition.

Figure 4.8: The distributions of POS tags in the TACRED test partition. The top figure shows how many times each POS tag appears in the test data. The bottom figure shows how many times each POS tag appears in the generated explanations from the same partition.

(IN), and commas is considerably higher in the explanations than the raw data. Second, the figure indicates that our explanations often focus on parts of speech that are necessary for plausability (according to the human annotators) but are

	RC F1	Quantitative EC F1	Qualitative EC F1
Full Model	70.52±0.54	95.76	62.05
– NRC	67.47 ±0.54	92.95	54.70
– EC	70.62±0.46	N/A	N/A
Vanilla SpanBERT	70.07±0.73	N/A	N/A

Table 4.9: Ablation results on the TACRED test partition, i.e., “–” indicates that the corresponding component was removed from the full system, and “N/A” indicates that that metric is not applicable.

	RC F1	Quantitative EC F1	Qualitative EC F1
Full Model	79.46±0.92	99.52	58.97
– NRC	77.34±2.33	99.00	50.12
– EC	76.58±1.52	N/A	N/A
Vanilla SpanBERT	75.78±4.79	N/A	N/A

Table 4.10: Ablation results on the CoNLL04 test partition, i.e., “–” indicates that the corresponding component was removed from the full system, and “N/A” indicates that that metric is not applicable.

semantically-ambiguous such as prepositions (IN), commas,¹⁸ and determiners (DT). This is different from traditional pattern acquisition methods (Riloff, 1996), which usually focus on words with more clear semantics such as nominals and verbs.¹⁹

Ablation Study

To understand the impact of the classifiers employed by our approach (i.e., NRC, RC, and EC), we implemented ablation experiments on both datasets, which are summarized in Tables 4.9 and 4.10. Note that the method without both NRC and EC becomes equivalent to the vanilla SpanBERT (as we discussed in Section 4.2.2). Overall, this experiment re-emphasizes that not only does our approach outperform the vanilla SpanBERT, but it does so while generating an explanation for its decisions.

¹⁸Commas are necessary to capture appositive constructs, which are often indicative of relations, e.g., “Barack Obama, the former president.” In cases such as these, the subject and object of the relation (e.g., “Barack Obama” and “former president”, respectively) cover most lexical information relevant to the relation. In these cases, the remaining signal that indicates the apposition is the comma.

¹⁹Note that traditional patterns may include prepositions and particles, e.g., in verb constructs such as SUBJECT was born in OBJECT. However, these patterns are usually semantically headed by verb phrases or nominalized predicates, e.g., born, and seldom by prepositions.

Approach	Precision	Recall	F1
Baseline			
Manual Rules ^[1]	85.93	24.24	37.81
Our Approach			
Rules from Training ^[2]	49.39	30.26	37.52
Rules from Test ^[3]	59.69	55.04	57.27
Combination of [1] and [2]	54.12	62.95	58.20
Combination of [1] and [3]	65.28	71.64	68.31
Combination of [2] and [3]	56.34	40.90	47.40
Combination of [1], [2] and [3]	57.36	72.00	63.85

Table 4.11: Performance of the rule-based model on the TACRED test partition. [1] is the set of manually-written surface rules of Angeli et al. (2015) coupled with our syntactic rules (see Section 4.2.1). [2] is the set of rules generated from our explainability classifier’s outputs with gold labels on the training partition. [3] is the set of rules from the explainability classifier’s outputs with predicted labels on the test partition. We also evaluate the performance on combinations of these sets of rules: [2]+[3] contain all rules generated by our approach; [1]+[2]+[3] combine machine-generated rules with the manually-written rules.

Removing the NRC drops the relation classification F1 score by approximately 3 points on TACRED, and 2 points on CoNLL04. This impact is explained by that fact the using the NRC avoids the meaningless scenario where the EC (which was trained only on positive examples) is applied to negative examples. Interestingly, removing the EC has no statistical impact on relation classification performance on TACRED, but it reduces the relation classification F1 by approximately 3 points on CoNLL04. As discussed in Section 4.2.4, this is caused by the fact that the EC serves as a useful disambiguator in CoNLL04, where multiple relations co-occur in the same sentence. The EC is not that impactful in TACRED, which has a more artificial setting with much fewer relations per sentence.²⁰

Interpretability: from Local to Global

Lastly, we evaluate the performance of our rule-based model that relies solely on rules, some of which were manually written (see Section 4.2.1), while some were

²⁰The average number of relations per sentence in TACRED is approximately 2 in training, and 1 in development and test.

Approach	Precision	Recall	F1
Baseline			
Manual Rules ^[1]	81.82	17.06	28.24
Our Approach			
Rules from Training ^[2]	66.10	27.73	39.07
Rules from Test ^[3]	67.95	50.24	57.77
Combination of [1] and [2]	71.06	39.57	50.84
Combination of [1] and [3]	68.48	59.72	63.80
Combination of [2] and [3]	64.01	55.21	59.29
Combination of [1], [2] and [3]	66.67	63.03	64.80

Table 4.12: Performance of the rule-based model on the CoNLL04 test partition. This table follows the same conventions as Table 4.11, except, in this case, [1] is the set of manually-written Odin rules we wrote for CoNLL04.

automatically generated by our approach, as described in Section 4.1.3. The results are summarized in Tables 4.11 and 4.12. We draw two observations from these results:

- Automatically-generated rules can outperform manually-written ones. However, in order to approach the performance of the neural RC, our method benefits from being aware of the distribution of words in each testing sentence to be processed (setting [3] in the tables). Importantly, we reiterate that when using the test sentences, our approach does *not* have access to any gold human annotations for RC and EC. That is, the rules generated from test sentences rely only on predicted relation labels and predicted explanations for each given sentence. The fact that rules need to be exposed to more data before they generalize is not extremely surprising: the rule matching engine we currently use relies on exact lexical matching, which means that the actual tokens to be matched must be present in the rule. However, the fact that the knowledge necessary to encode a relation extraction *can* be encoded into rules is exciting. The combination of these observations suggests that a future avenue for research that focuses on “soft rule matching” (Zhou et al., 2020), might be the direction that captures the advantages of both rules and neural methods.

- Interestingly, automatically-generated rules tend to be complementary to the manual ones. The combination of all three rule sets ([1], [2], and [3] in the tables) outperforms considerably both the setting that relies solely on manual rules and the configuration that relies only on automatically-generated ones. The combination of all rule sets outperforms the manually-generated rules by 31% F1 and 38% F1 (absolute) in TACRED and CoNLL04, respectively. Furthermore, the TACRED result of the combined rule set approaches the performance of the neural RC within less than 3% F1. The performance gap between the combined rule set and neural RC in CoNLL04 is larger (over 14% F1).²¹ Nevertheless, all in all, this result suggests that humans and machines can collaborate towards building a fully-explainable model that comes reasonably close to the performance of neural classifiers.

Error Analysis

We conclude this section with a brief error analysis of our explainability classifier in the TACRED and CoNLL04 datasets. Table 4.13 summarizes a few typical errors observed in the two datasets.

The first two rows in the table show examples where the EC generates explanations that rely solely on the subject and object entities, without including any word in the relations’ contexts. Note that the example shown in the first row is potentially correct: it is likely that a location name that immediately precedes an organization name indicates the location of that organization. However, the second example is clearly incorrect: the correct explanation to justify the `no_relation` label should minimally include *not* and *relative*. Further, please note that a hypothetical RC that had access to the *unmasked* entities could potentially perform even better. For example, in the first case, one could infer that *O Globo* is based in *Rio de Janeiro* because the former organization name is Portuguese. However, our RC only sees

²¹We conjecture that the cause for this larger gap is the lower quality of the rules used for the CoNLL04 dataset. That is, the TACRED rules were developed by a larger team over a longer period of time, whereas the CoNLL04 rules were developed by one of the authors in only a few hours.

<p>Rio de Janeiro O GLOBO</p> <p>Gold label: <code>OrgBased_In</code>; predicted label: <code>OrgBased_In</code></p>
<p>I had an e-mail exchange with Benjamin Chertoff of Popular Mechanics in the original Loose Change thread that showed that he was not a close relative of Michael Chertoff .</p> <p>Gold label: <code>no_relation</code>; predicted label: <code>per:other_family</code></p>
<p>In Beijing Thursday, spokesman Li repeated China's position that the key to the solution of the Cambodian conflict “ lies in the genuine and complete Vietnamese troop withdrawal at the earliest possible date and effective international supervision. ”</p> <p>Gold label: <code>Live_in</code>; predicted label: <code>Work_for</code></p>
<p>There's been a sea change in my lifetime, “ said Jefferson Keel , lieutenant governor of the Chickasaw Nation in Oklahoma and a first vice president of the National Congress of American Indians. ”</p> <p>Gold label: <code>no_relation</code>; predicted label: <code>org:top_members/employees</code></p>
<p>Overview of Arrow Missile Program , Tasks 94AA0008Z Jerusalem THE JERUSALEM POST in English 15 Oct 93 pp 6-8 , 10-FOR OFFICIAL USE ONLY</p> <p>Gold label: <code>OrgBased_In</code>; predicted label: <code>OrgBased_In</code></p>
<p>Like al-Shabab , the ADF is primarily a Muslim radical group .</p> <p>Gold label: <code>org:politicalreligious.affiliation</code>; predicted label: <code>org:politicalreligious.affiliation</code></p>

Table 4.13: Typical errors that our explainability classifier commits. These include errors of under prediction (first two rows), misleading prediction (middle two rows), and errors of over prediction (last two rows). This figure follows the same convention as Figure 4.5.

masked subjects and objects. Nevertheless, we believe that our strategy of masking entities participating in relations is a valuable exercise, as it investigates the capacity of neural methods to identify explicit context necessary for relation extraction.

Rows 3 and 4 in the table show examples when our RC makes incorrect predictions due to incorrect tokens labeled by the EC. For example, the token *president* in row 4 guides the RC towards the incorrect prediction `org:top_members/employees`. The situation in the third row is more subtle: one might argue that *China* here can also be referring to the government, which makes the prediction `Work_for` correct. In any case, these errors indicate that our explanations can be used for debugging purposes when the RC makes incorrect predictions.

The last two rows in the table show examples where our EC over included words in its explanations. For example, in the last row, a likely interpretation is that the verb *is* should be part of the correct explanation, but all the other words are unnecessary. This happens because the rule lexical triggers in TACRED tend to contain multiple words, which encouraged the EC to learn to include additional words in its explanation. In contrast, in CoNLL04 (second to last row), most triggers are single-word phrases. This prompted the EC to include one token in its explanation, even though it is unnecessary for the prediction of the relation label in this case.

For a more complete bigger picture, we analyzed the overall frequency of these error types on the same sampled instances we used for the qualitative explanation evaluation (Section 4.2.4). Errors where the EC provided no explanations²² occurred in 4.12% of examples in TACRED, and 19.41% in CoNLL04. Errors where the explanations caused false positive relations to be predicted appeared 25.95% times in TACRED, and 16.49% in CoNLL04. Nevertheless, as Tables 4.4, 4.5, 4.7, and 4.8 show, our EC makes considerably fewer errors than all other explainability methods. There is no reason to believe that its current errors cannot be fixed with human feedback that would provide a (hopefully small) number of rules to adjust imperfect explanations.

4.3 Conclusion

We introduced an explainable approach for relation extraction that jointly trains for prediction and explainability. Our approach uses a multi-task learning framework with a shared encoder, and jointly trains a classifier for relation extraction with a second explainability classifier that labels which words in the context of the relation explain the underlying relation. Further, our method is semi-supervised, as annotations for the latter classifier are usually not available.

We evaluated the proposed approach on a relation extraction task in two datasets: TACRED and CoNLL04. Our evaluation showed that, even with minimal supervi-

²²We included in this category the situations where the explanation was completely empty or it included only the subject and/or object entity mentions.

sion for explanation guidance, our method generates explanations for the relation classifier’s decisions that are considerably more accurate and plausible than other strong baselines such as LIME, or relying on attention weights (Simonyan et al., 2013; Bahdanau et al., 2014; Ribeiro et al., 2016; Lundberg and Lee, 2017; Schwab and Karlen, 2019; Vafa et al., 2021). Further, our results indicated that jointly training for explainability and prediction improves the prediction task itself, i.e., the relation classifier performs better when it is exposed only to the textual context deemed important by the explainability classifier.

We also showed that it is possible to convert these local explanations into global ones. We converted the outputs of our explainability classifier into a set of rules that globally explains the behavior of the neural relation classifier. Our results showed that our strategy for generating a rule-based model pushes the performance of rule-based approaches closer to that of neural methods.

For the next step, we propose our approach being used in an iterative semi-supervised learning scenario akin to co-training (Blum and Mitchell, 1998). That is, the newly generated rules can be converted to executable rules that can be applied over large, unannotated texts to generate new training examples for the relation classifier, and vice versa. Further, our method could potentially benefit from traditional pattern bootstrapping approaches (Riloff, 1996; Lin and Pantel, 2001), which could reduce the amount of human supervision necessary by automatically expanding the set of initial patterns available.

CHAPTER 5

Rule-enhanced Bootstrapped Self Training for Weakly-supervised Relation
Extraction

Neural networks have brought us tremendous improvement on most of the NLP tasks in recent years. However, training a neural network requires a large amount of labeled data. Collecting the labeled data is hard and expensive.

In the typical pseudo labeling method in SSL, people first train the model with limited labeled data and later apply the trained model to unlabeled data to gather more annotations, repeat this process until no more new data can be labeled. Empirically, this method works well when there is only small labeled data available. However, the model cannot guarantee the quality of all its predictions and it may cause the problem like semantic drift or large human effort on filtering the model annotations. To improve the pseudo labeling approach, we take the EC-RC model we mentioned in Chapter 4, use its outputs to generate rules and apply the rules to unlabeled data to get more annotations. The contributions of our idea are the following:

- (1) We introduced a strategy that uses the EC-RC model (Tang and Surdeanu, 2022) outputs to generate rules and uses the rules to bootstrap the relation classifier performance. We evaluate this approach in a low-resource scenario where there is no labeled data.
- (2) We evaluate this approach on TACRED dataset Zhang et al. (2017) and demonstrate that: (a) our neural RE classifier outperforms considerably the rule-based one we started from; (b) using rules in bootstrapping is better than solely relying on relation classifier’s predictions, (c) we can reach similar results compared to prompt based models without the extra NLI component or human effort.

5.1 Approach

Similar to typical bootstrapped self training approaches, Our approach iteratively train the model with annotated data and apply the automate machine annotator to the raw data to get new annotations. Unlike those approaches which use the classifier as annotator, our approach uses the generated rules as the annotator. Figure 5.1 shows the overall training procedure. There are three major components in this procedure:

(1) Rule Executor: We use Odin (Valenzuela-Escárcega et al., 2016b) system as our rule executor. Figure 4.2 in Chapter 4 shows an example of Odin rule. The rule executor looks for the verbal trigger and semantically constrained arguments in the given sentences and checks if the arguments are attached to the trigger with certain syntactic dependency patten. If all the constraints are satisfied, the relation label from the rule will be assigned to the sentence.

(2) Neural Model: Our approach utilize the same model we described in Chapter 4. It contains two main classifiers: the relation classifier (RC), and the explanation classifier (EC). These are jointly trained using the schema previously described in Section 4.1. The RC predicts the relation holds between two entities and the EC label the words which explain the prediction (We call the explanation words “trigger” in this work).

(3) Rule Generator: The rule generator has two major components: the generator and the filter. The generator takes the model output from the neural model we discussed above. This entire rule generation process was described in Algorithm 1 in Section 4.1. The filter takes generated rules from the generator, applies them to a validation set and evaluate the rule performance by precision. If a rule’s performance is low than a certain threshold, the filter will discard that rule and not send it to the rule executor.

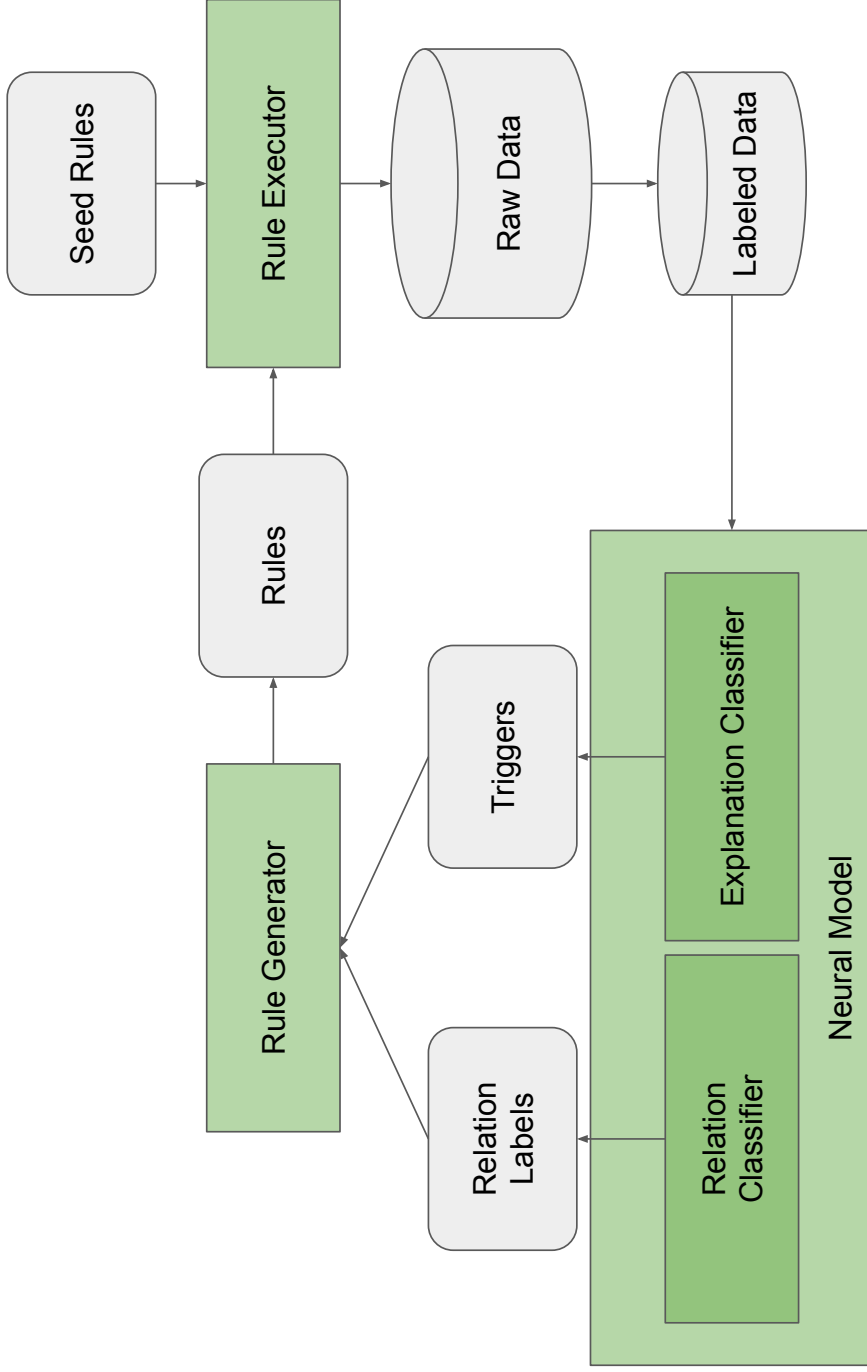


Figure 5.1: Overall procedure of our bootstrapped self training approach. At the beginning, we come up with a set of manual rules. In each iteration, (1) we apply the rules to the rule executor and move the matched data from raw data set D_R to labeled data set D_L (2) we train the neural model with D_L , (3) we generate rules using the outputs from the neural model, (4) we feed the new rules in rule executor and repeat this procedure from (1).

5.1.1 Training Procedure

As we mentioned at the beginning of this section, overall training procedure is shown in the Figure 5.1.

In iteration 0, we first feed the seed rules r_0 in the rule executor and apply them to the raw data set D_R . These rules are small set of manual rules written by human annotators. We add the rule-matched data D_0 as seed annotations to labeled data set D_L and remove them from D_R .

In iteration t , we add the new rule-matched data D_t to D_L and remove them from D_R , train the neural model m_t with all labeled data in D_L and get the outputs from both relation and explanation classifier. Then, we generate and filter the rules in the rule generator. Eventually we feed the newly generated rules r_{t+1} in the rule executor and apply them to D_R and get new rule-matched data D_{t+1} . We repeat this procedure until we saw no more performance improvement on the validation set.

5.2 Experimental Results

5.2.1 Data Preparation

We report results on the TACRED dataset Zhang et al. (2017). To mimic the low-resource scenarios, we hide all gold labels from the training set. We keep only 1% of development set for validation purpose. Like other bootstrapping approaches, we need some seed annotations to start the training procedure. In Chapter 4, for the TACRED data, we selected rules from the surface patterns of Angeli et al. (2015), and we combined them with an additional set of 38 syntactic rules in the Odin language Valenzuela-Escárcega et al. (2016b) that were manually created by one of the authors from the training data. This is what we use as the initial rules set. We apply these rules to the unlabeled training set, and use the rule extraction results as the seed annotation.

5.2.2 Baselines

We compare our results with four baselines: an extended version of the rule-based approach of Angeli et al. (2015), a typical pseudo labeling bootstrapping approach, a prompt-based RE approach based on natural language inference (NLI) (Sainz et al., 2021), and a prompt-based rule discovery and bootsting approach Zhang et al. (2022):

- **Rule-based Extraction.** As mentioned in Section 5.2.1, we employ two sets of rules. First, we use the tokensregex surface rules from Angeli et al. (2015), which are executed in the Stanford CoreNLP pipeline Manning et al. (2014a). Second, we include the Odin syntactic rules we developed in-house, which are executed in the Odin framework Valenzuela-Escárcega et al. (2016b).¹
- **Pseudo Labeling.** In the typical pseudo labeling method in SSL, people first train the model with limited labeled data and later apply the trained model to unlabeled data to gather more annotations, repeat this process until no more new data can be labeled.
- **NLI-Prompt.** Sainz et al. (2021) reformulate the relation extraction task as an entailment task. They come up with a bunch of verbalization templates for each relation in TACRED. For example, they verbalize `per:city_of_birth` relation as `{subj} was born in {obj}`, where `{subj}` and `{obj}` are the given entities. Given a sentence with two entities, they treat the sentence as premise, they replace the subject and object in each verbalization template with the given entities and use them as hypothesis. The RE task becomes to the task to find the best entailment template for the given example.
- **PRBOOST.** Zhang et al. (2022) propose method called PRBOOST which iteratively generate rules from prompting, ask human to filter the rules, use the rules to generate annotation, and use the annotations to train new model.

¹The rule set from Angeli et al. (2015) also included some syntactic rules, but we found out that they only matched the simpler `per:title` relation, so we did not use them.

Approach	Precision	Recall	F1
Baselines			
Rules	85.82	24.21	37.77
Pseudo Labeling	68.20	38.38	49.11
NLI-Prompt (Sainz et al., 2021)	55.46	52.09	53.72
PRBOOST (Zhang et al., 2022)	–	–	48.1
Our Approach			
Iteration 5	60.79	46.35	52.59

Table 5.1: Relation extraction results on the TACRED test partition.

5.2.3 Implementation and Evaluation Details:

We use Odin (Valenzuela-Escárcega et al., 2016b) system as our rule executor. We use the same neural model as we described in Chapter 4. We follow the same implementation details as we discussed in Section 4.2.2. Instead of using the full development set, we randomly select 1% data from TACRED development set for validation purpose. We evaluate the performance of the model using the micro precision, recall, and F1 scores. In the rule generator, we generate rules following the Algorithm 1 in Section 4.1. We measure the quality of each generated rule by its precision on the 1% development data. We use 0.5 as the threshold, if the precision of a rule is lower than the threshold, we will discard that rule.

In NLI-prompt approach (Sainz et al., 2021), they do the experiments with different scenarios with different language models. To achieve a fair comparison, we choose the zero-shot scenario and RoBERTa (Liu et al., 2019)² in our experiments. We also convert our generated rules to the verbalization templates as NLI-prompt approach (Sainz et al., 2021) and feed the template in their NLI-prompt framework. The NLI-prompt approach requires a detection function for `no_relation` examples, Sainz et al. (2021) show that the threshold-based detection works better for this task. The threshold is estimated using the 1% development data.

Approach	Precision	Recall	F1
NLI-Prompt (Sainz et al., 2021)	55.46	52.09	53.72
Iteration 0	45.91	65.17	53.87
Iteration 5	42.60	51.49	46.62

Table 5.2: NLI-prompt results on the TACRED test partition. We convert the rules in that iteration to verbalization template.

5.2.4 Results and Discussion

Table 5.1 reports the overall performance of our approach and the baselines. We use the numbers reported in the papers for the prompt baselines. We draw the following observations from these results:

- (1) The rule-based method of Zhang et al. (2017) has high precision but suffers from low recall. In contrast, our best model that is bootstrapped from the same information has 22% higher recall and almost 15% higher F1 (absolute).
- (2) Our approach has better performance compared to the pseudo labeling approach. This shows that using rules in the bootstrapped self training is better than solely replying on the classifier’s prediction.
- (3) Our method approaches the performance of the NLI-prompt method within 2% gap. Moreover, our method works better than the PRBOOST. This indicates that we can still use bootstrapped self training to get the competitive performance with the prompt-based methods. More importantly, our approach does not need the extra NLI component.

Table 5.2 shows the performance of using different sets of templates. The manual templates designed for entailment task has a similar performance compared to the templates generated by the manual rules. This indicates that we do not need too much effort on prompt template engineering if there are some existing resources to this task, such as rules.

²The NLI-prompt requires a fine-tuned NLI layer in the language model, unfortunately, the model (SpanBERT) we use does not have it. We believe that RoBERTa is the closet alternative.

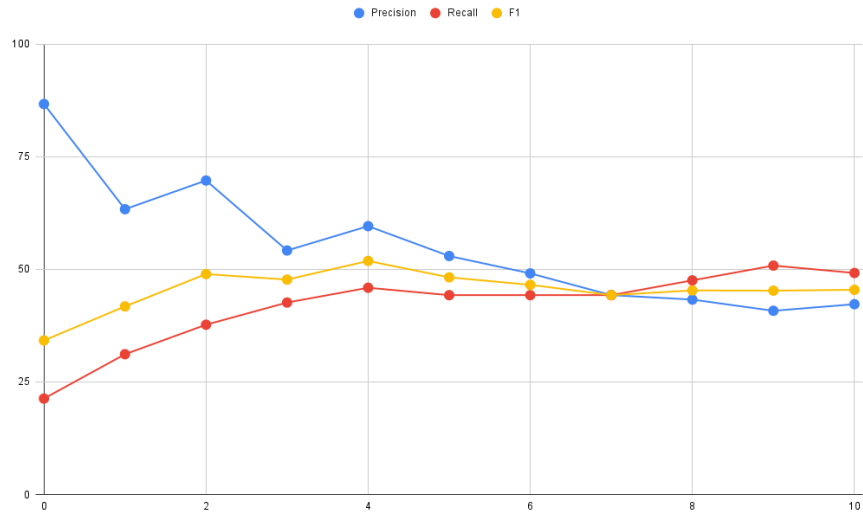


Figure 5.2: Learning curve of our approach based on the iterations.

Figure 5.2 shows the changes of precision, recall and F1 scores over multiple iterations. As shown, the recall and F1 are steadily increase during this procedure. This is inspiring since it shows that our approach can help improve the generalizability of the neural model in the low-resource scenario. Besides, we notice that the drop of the precision is the reason the F1 score stop improve after iteration 5. However, this is solvable since our annotations are from the rules and there are ways to control the quality of the rules other than just filtering out the low precision ones. We will leave it as future work.

5.3 Conclusion

We introduced a strategy that uses the EC-RC model (Tang and Surdeanu, 2022) outputs to generate rules and uses the rules to bootstrap the relation classifier performance. We evaluate this approach in a low-resource scenario where there is no labeled data.

Our experiments on the TACRED dataset demonstrated that our approach outperforms the strong rule-based method that provided the training patterns by 15 F1 points. It also demonstrated that getting rules involved in the training procedure

is better than the typical bootstrapping approach which solely relies on model prediction. Further, we show that we can reach similar results compared to prompt based models without the extra NLI component or keeping human in the loop. ‘

Conclusion and Future Work

In this dissertation, we have introduced several efforts on migrating interpretability and generalizability in neural networks by combining rules and neural models.

In Chapter 3, we presented a strategy for jointly training a classifier and a decoder that generates explanations for classifier’s decision. We evaluated this approach with two different tasks, BioNLP event extraction and TACRED relation extraction. The experiments show that this approach considerably outperforms the strong rule-based methods. Furthermore, our evaluations with this encoder-decoder architecture on the BioNLP and TACRED datasets showed that the decoder generates interpretable rules and that joint training improves the classifier’s performance.

However, the decoded rules are not guaranteed to be executable. Further, they lack of generalizability to unseen examples. To solve these issues, in Chapter 5 we introduced an explainable method for relation extraction that trains for both prediction and explainability. Our method employs a multi-task learning framework with a shared encoder to jointly train a relation extraction classifier with a second explainability classifier that labels which words in the context of the relation explain the underlying relation. Furthermore, because annotations for the latter classifier are rarely available, our method is semi-supervised. The proposed method was tested on a relation extraction task in two datasets: TACRED and CoNLL04. Our evaluation revealed that, even with minimal explanation guidance, our method generates explanations for the relation classifier’s decisions that are significantly more accurate and plausible than other strong baselines such as LIME or relying on attention weights. Furthermore, our findings show that training for explainability and prediction concurrently improves the prediction task itself, i.e., the relation classifier performs better when exposed only to the textual context deemed important by the explainability classifier.

We also demonstrated that these local explanations can be converted into global explanations by converting the local rationales to rules. Furthermore, we deployed these rules to a bootstrapping framework and show that the bootstrapping with rules is better than the typical pseudo labeling methods in bootstrapping, and competitive with the prompt-based approaches which rely on extra components such as specific task layers or human effort.

All in all, our work suggests that it is possible to marry the interpretability of rule-based methods with the performance of neural approaches. We hope this work contributes to the improvement of NLP applications due to neural networks, while maintaining the interpretability offered by rules.

Future Works

We have identified several potential future directions that continue our work on combining rules with neural networks.

EC-RC Method in New Scenarios and Domains

At a higher level, we hope that this work will support meaningful collaborations between NLP researchers and subject matter experts in other domains (e.g., medical, legal), who benefit from the output of NLP systems (e.g., large-scale extraction of biomedical events) but may not understand the intricacies of the neural methods that underlie these NLP approaches.

For now, we only applied our approach to unary event extraction and binary relation extraction. However, we expect this approach to be applicable to other information extraction tasks. For example, Stanford CoreNLP (Manning et al., 2014b) integrates both statistical and rule-based approaches, and obtains a decent performance for named entity recognition (NER). This suggests that our approach should also be applicable to the NER task. We expect that our approach help build a more transparent and explainable NER tool. In Chapter 3, we only explored the unary events in the BioNLP 2013 task. In the future, we hope that our approach

can be also applied to other event types, e.g., binary or n -ary events.

We also hope that our approach can be adapted to other tasks other than information extraction, such as sentiment analysis, question answering, and natural language inference.

Explanations for Nested Event Extraction

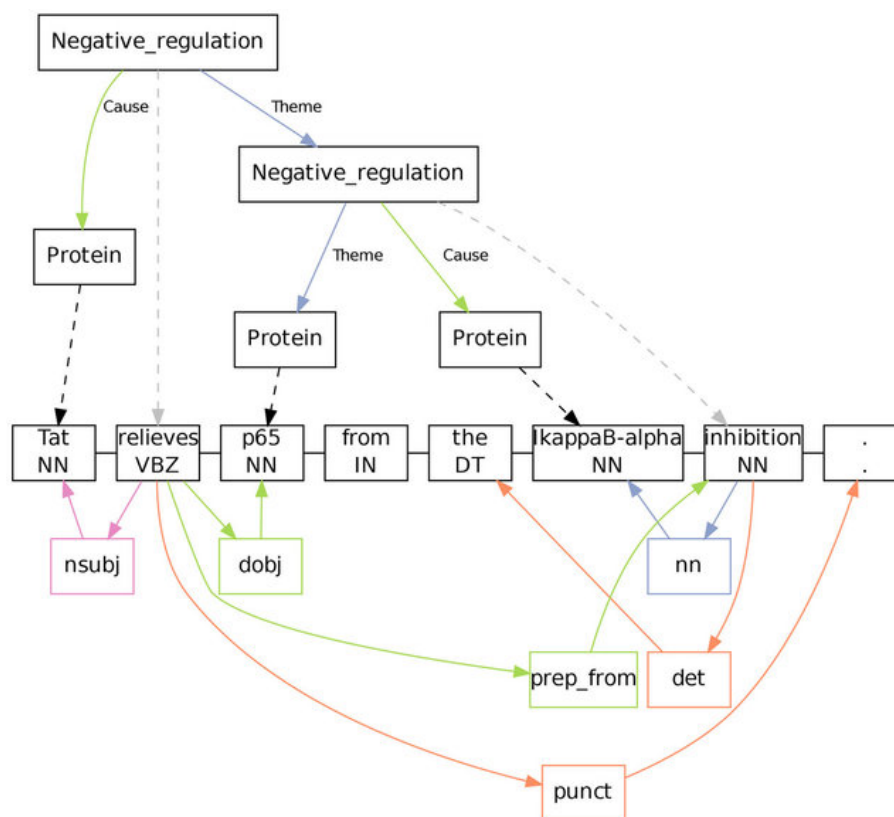


Figure 5.3: Example of nested event extraction from the BioNLP task.

Event extraction for individual events has been well studied. However, in real-world NLP applications, many events build on top of other events. In other words, event structures tend to be nested. Figure 5.3 shows nested events from the BioNLP task, e.g., a “regulation” event has another biochemical reaction e.g., another “regulation” event, as an argument (Kim et al., 2013). To capture this complexity, such events are often annotated as trees or directed graphs (Baker et al., 2007; Surdeanu et al.,

2008; Kim et al., 2013; Banarescu et al., 2013). To handle such event extraction tasks, researchers need to come up with more complex patterns in the rule-based approaches or custom statistical features and parsing pipelines in the neural network systems, which makes this task challenging for our current approach. We hope we can explore our approach to nested event extraction tasks and show that our approach can also handle complex extraction tasks more suitable for real-world scenarios.

Rules Enhanced with Neural Networks

In general, the disadvantage of rules is their lack of coverage. Researchers switch from rule-based systems to statistical methods such as neural networks due to a lack of generalization. We argue that we should not completely disregard rules.

Recent large pre-trained language models (PLMs) have demonstrated the ability to handle NLP tasks with just a few examples. Researchers show that we can reformat the NLP tasks with prompts and obtain considerable improvements compared to traditional fine-tuned models. The key issue in prompting is generating the prompt templates. However, prompt templates are not easy to obtain; they require either human effort on template design or extensive research on automated template generation. In Chapter 5, we demonstrated that we can adapt prompts to rules for NLI-prompting. We believe these rules can also be adapted to other prompt problems, e.g., for question answering.

There exist approaches for fuzzy matching for patterns (Li et al., 2011; Ferré et al., 2018). For example, embeddings have been used for measuring the similarity between entities for lexicon acquisition (Mikolov et al., 2013; Pennington et al., 2014; Trisedya et al., 2019). With the help of recent large pre-trained language models (PLMs), we can also compare the similarity between sentences and documents. These neural representations help machines understand the nuances in a language and to do a better job in matching sentences. We think we can also encode the rules in the neural representations and discover the matching sentences using similarity scores. By doing so, we hope we can achieve better coverage from rules.

APPENDIX

Chapter 3 Appendix

Experimental Details

We use the dependency parse trees, POS and NER sequences as included in the original release of the TACRED dataset, which was generated with Stanford CoreNLP Manning et al. (2014b). We use the pretrained 300-dimensional GloVe vectors Pennington et al. (2014) to initialize word embeddings. We use a 2 layers of bi-LSTM, 2 layers of GCN, and 2 layers of feedforward in our encoder. And 2 layers of LSTM and 1 layer of feedforward in our decoder. Table 5.4 shows the details of the proposed neural network. We apply the ReLU function for all nonlinearities in the GCN layers and the standard max pooling operations in all pooling layers. For regularization we use dropout with $p = 0.5$ to all encoder LSTM layers and all but the last GCN layers.

Encoder and classifier components	Size
Vocabulary	53953
POS embedding dimension	30
NER embedding dimension	30
LSTM hidden layers	200
Feedforward layers	200
GCN layers	200
Relation	41
Decoder component	Size
LSTM hidden layers	200
Pattern embedding dimension	100
Feedforward layer	200
Maximum decoding length	100
Pattern	1141

Table 5.3: Details of our neural architecture.

For training, we use Adagrad Duchi et al. (2011) an initial learning rate, and from epoch 1 we start to anneal the learning rate by a factor of 0.9 every time the F1 score on the development set does not increase after one epoch. We tuned the initial learning rate between 0.01 and 1; we chose 0.3 as this obtained the best performance on development. We trained 100 epochs for all the experiments with a batch size of 50. There were 3,850 positive data points and 12,311 negative data in the rule-only data. For this dataset, it took 1 minute to finish one epoch in average. And for Rules + TACRED training data, it took 4 minutes to finish one epoch in average³.

All the hyperparameters above were tuned manually. We trained our model on PyTorch 3.8.5 with CUDA version 10.0, using one NVIDIA Titan RTX.

Chapter 4 Appendix

We use the dependency parse trees, POS tags and NER labels as included in the original release of the TACRED dataset. All these were generated with Stanford CoreNLP Manning et al. (2014b).

We use the pretrained SpanBERT model Joshi et al. (2020) available in the HuggingFace transformer library Wolf et al. (2020) as our encoder.⁴ Table 5.4 shows the hyperparameter details for training the neural models for relation classification (SpanBERT) and both relation and explainability classification (Unsupervised Rationale and our approach). Note that we relied mostly on the default hyperparameter values from SpanBERT, but used a larger number of epochs with a smaller learning rate to fine-tune the additional explainability component. The Unsupervised Rationale method was tuned for relation classification, which boosted its RC performance (Tables 2 and 3), but negatively impacted its explainability power (Tables 4 and 5).

Some of the explainability baselines do not have hyper parameters, including: attention, saliency mapping, greedy adding, and all words in between. For SHAP, we use all default settings from the API provided by the authors at: <https://shap.readthedocs.io/en/latest/index.html>. For LIME, the number of samples we

³The software is available at this URL: `hidden_for_review`.

⁴<https://huggingface.co/SpanBERT/spanbert-large-cased>

Approach	SpanBERT	Unsupervised Rationale	Our Approach
Number of epochs	10*	20	20
Learning rate	2e-5*	1e-5	1e-5
Dropout rate	0.1*	0.1	0.1
Batch size	32*	32	32
Max sequence length	128*	128	128
Scheduler		Linear scheduler with warm up*	

Table 5.4: Hyperparameter details for training the neural models for relation classification (for SpanBERT) and both components (Unsupervised Rationale and our approach). The numbers with * are the default values from the SpanBERT implementation available at: <https://github.com/facebookresearch/SpanBERT>.

used is 2000. And for CXPlain, the explanation model we use is a 2-layers RNN model, with learning rate of 0.001, dropout rate of 0.2, and trained for 2 epochs.

REFERENCES

- Adadi, A. and M. Berrada (2018). Peeking inside the black-box: a survey on explainable artificial intelligence (XAI). *IEEE access*, **6**, pp. 52138–52160.
- Angeli, G., V. Zhong, D. Chen, A. Chaganty, J. Bolton, M. J. J. Premkumar, P. Pasupat, S. Gupta, and C. D. Manning (2015). Bootstrapped Self Training for Knowledge Base Population. *Theory and Applications of Categories*.
- Appelt, D. E., J. R. Hobbs, J. Bear, D. Israel, and M. Tyson (1993). FASTUS: A finite-state processor for information extraction from real-world text. In *IJCAI*, volume 93, pp. 1172–1178.
- Arrieta, A. B., N. Díaz-Rodríguez, J. Del Ser, A. Bennetot, S. Tabik, A. Barbado, S. García, S. Gil-López, D. Molina, R. Benjamins, et al. (2020). Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. *Information Fusion*, **58**, pp. 82–115.
- Baehrens, D., T. Schroeter, S. Harmeling, M. Kawanabe, K. Hansen, and K.-R. Müller (2010). How to explain individual classification decisions. *The Journal of Machine Learning Research*, **11**, pp. 1803–1831.
- Bahdanau, D., K. Cho, and Y. Bengio (2014). Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.
- Baker, C., M. Ellsworth, and K. Erk (2007). SemEval’07 task 19: frame semantic structure extraction. In *Proceedings of the 4th International Workshop on Semantic Evaluations*, pp. 99–104. Association for Computational Linguistics.
- Banarescu, L., C. Bonial, S. Cai, M. Georgescu, K. Griffitt, U. Hermjakob, K. Knight, P. Koehn, M. Palmer, and N. Schneider (2013). Abstract meaning representation

- for sembanking. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, pp. 178–186.
- Bao, Y., S. Chang, M. Yu, and R. Barzilay (2018). Deriving Machine Attention from Human Rationales. *arXiv preprint arXiv:1808.09367*.
- Bastings, J., W. Aziz, and I. Titov (2019). Interpretable Neural Predictions with Differentiable Binary Variables. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 2963–2977. Association for Computational Linguistics, Florence, Italy. doi:10.18653/v1/P19-1284.
- Béchet, F., A. Nasr, and F. Genet (2000). Tagging unknown proper names using decision trees. In *proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*, pp. 77–84.
- Belinkov, Y., N. Durrani, F. Dalvi, H. Sajjad, and J. Glass (2017). What do Neural Machine Translation Models Learn about Morphology? In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 861–872. Association for Computational Linguistics, Vancouver, Canada. doi:10.18653/v1/P17-1080.
- Blum, A. and T. Mitchell (1998). Combining labeled and unlabeled data with co-training. In *Proceedings of the eleventh annual conference on Computational learning theory*, pp. 92–100. ACM.
- Boros, T., S. D. Dumitrescu, and S. Pipa (2017). Fast and Accurate Decision Trees for Natural Language Processing Tasks. In *Proceedings of the International Conference Recent Advances in Natural Language Processing, RANLP 2017*, pp. 103–110. INCOMA Ltd., Varna, Bulgaria. doi:10.26615/978-954-452-049-6_016.
- Brin, S. (1998). Extracting Patterns and Relations from the World Wide Web. In *WebDB*, pp. 172–183.
- Brunner, G., Y. Liu, D. Pascual, O. Richter, M. Ciaramita, and R. Wattenhofer (2019). On identifiability in transformers. *arXiv preprint arXiv:1908.04211*.

- Bunescu, R. and R. Mooney (2005). A shortest path dependency kernel for relation extraction. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pp. 724–731.
- Chan, Y. S. and D. Roth (2011). Exploiting Syntactico-Semantic Structures for Relation Extraction. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pp. 551–560. Association for Computational Linguistics, Portland, Oregon, USA.
- Chang, A. X. and C. D. Manning (2014). TokensRegex: Defining cascaded regular expressions over tokens. *Stanford University Computer Science Technical Reports. CSTR*, **2**, p. 2014.
- Chang, A. X., V. I. Spitkovsky, E. Yeh, E. Agirre, and C. D. Manning (2010). Stanford-UBC entity linking at TAC-KBP.
- Chang, C.-H., M. Kayed, M. R. Girgis, and K. F. Shaalan (2006). A survey of web information extraction systems. *IEEE transactions on knowledge and data engineering*, **18**(10), pp. 1411–1428.
- Chiticariu, L., Y. Li, and F. Reiss (2013). 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*, pp. 827–832.
- Collins, M. and Y. Singer (1999). Unsupervised models for named entity classification. In *1999 Joint SIGDAT conference on empirical methods in natural language processing and very large corpora*.
- Craven, M. and J. W. Shavlik (1996). Extracting tree-structured representations of trained networks. In *Advances in neural information processing systems*, pp. 24–30.
- Danilevsky, M., K. Qian, R. Aharonov, Y. Katsis, B. Kawas, and P. Sen (2020). A Survey of the State of Explainable AI for Natural Language Processing. In

- Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing*, pp. 447–459. Association for Computational Linguistics, Suzhou, China.
- Devlin, J., M.-W. Chang, K. Lee, and K. Toutanova (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.
- Duchi, J., E. Hazan, and Y. Singer (2011). Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research*, **12**(7).
- Ebrahimi, J., A. Rao, D. Lowd, and D. Dou (2017). Hotflip: White-box adversarial examples for text classification. *arXiv preprint arXiv:1712.06751*.
- Eyal, M., A. Amrami, H. Taub-Tabib, and Y. Goldberg (2021). Bootstrapping Relation Extractors using Syntactic Search by Examples. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pp. 1491–1503. Association for Computational Linguistics, Online. doi:10.18653/v1/2021.eacl-main.128.
- Feng, S., E. Wallace, A. Grissom II, M. Iyyer, P. Rodriguez, and J. Boyd-Graber (2018). Pathologies of Neural Models Make Interpretations Difficult. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 3719–3728. Association for Computational Linguistics, Brussels, Belgium. doi:10.18653/v1/D18-1407.
- Ferré, A., L. Deléger, P. Zweigenbaum, and C. Nédellec (2018). Combining rule-based and embedding-based approaches to normalize textual entities with an ontology. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*. European Language Resources Association (ELRA), Miyazaki, Japan.

- Fong, R., M. Patrick, and A. Vedaldi (2019). Understanding Deep Networks via Extremal Perturbations and Smooth Masks. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 2950–2958. doi:10.1109/ICCV.2019.00304.
- Frosst, N. and G. Hinton (2017). Distilling a neural network into a soft decision tree. *arXiv preprint arXiv:1711.09784*.
- Getman, J., J. Ellis, Z. Song, J. Tracey, and S. M. Strassel (2017). Overview of Linguistic Resources for the TAC KBP 2017 Evaluations: Methodologies and Results. In *TAC*.
- Ghorbani, A., A. Abid, and J. Zou (2019). Interpretation of neural networks is fragile. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 3681–3688.
- Gunning, D. and D. Aha (2019). DARPA’s explainable artificial intelligence (XAI) program. *AI Magazine*, **40**(2), pp. 44–58.
- Gupta, S. and C. D. Manning (2015). Distributed Representations of Words to Guide Bootstrapped Entity Classifiers. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 1215–1220. Association for Computational Linguistics, Denver, Colorado. doi:10.3115/v1/N15-1128.
- Hahn-Powell, G., D. Bell, M. A. Valenzuela-Escárcega, and M. Surdeanu (2016). This before That: Causal Precedence in the Biomedical Domain. In *Proceedings of the 2016 Workshop on Biomedical Natural Language Processing*, pp. 146–155. Association for Computational Linguistics. doi:10.18653/v1/W16-2920.
- Han, X., B. C. Wallace, and Y. Tsvetkov (2020). Explaining Black Box Predictions and Unveiling Data Artifacts through Influence Functions. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp.

- 5553–5563. Association for Computational Linguistics, Online. doi:10.18653/v1/2020.acl-main.492.
- Hancock, B., M. Bringmann, P. Varma, P. Liang, S. Wang, and C. Ré (2018). Training classifiers with natural language explanations. In *Proceedings of the conference. Association for Computational Linguistics. Meeting*, volume 2018, p. 1884. NIH Public Access.
- Hassan, H., A. H. Awadallah, and O. Emam (2006). Unsupervised Information Extraction Approach Using Graph Mutual Reinforcement. In *EMNLP*, pp. 501–508.
- Hearst, M. A. (1992). Automatic Acquisition of Hyponyms from Large Text Corpora. In *COLING 1992 Volume 2: The 14th International Conference on Computational Linguistics*, p. 539–545.
- Hendricks, L. A., Z. Akata, M. Rohrbach, J. Donahue, B. Schiele, and T. Darrell (2016). Generating visual explanations. In *European Conference on Computer Vision*, pp. 3–19. Springer.
- Hewitt, J., K. Ethayarajh, P. Liang, and C. Manning (2021). Conditional probing: measuring usable information beyond a baseline. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 1626–1639. Association for Computational Linguistics, Online and Punta Cana, Dominican Republic.
- Hochreiter, S. and J. Schmidhuber (1997). Long short-term memory. *Neural computation*, **9**(8), pp. 1735–1780.
- Hoffmann, R., C. Zhang, X. Ling, L. Zettlemoyer, and D. S. Weld (2011). Knowledge-based weak supervision for information extraction of overlapping relations. In *Proceedings of the 49th annual meeting of the association for computational linguistics: human language technologies*, pp. 541–550.

- Hoover, B., H. Strobelt, and S. Gehrmann (2020). exBERT: A Visual Analysis Tool to Explore Learned Representations in Transformer Models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pp. 187–196. Association for Computational Linguistics, Online.
- Jain, S. and B. C. Wallace (2019). Attention is not explanation. *arXiv preprint arXiv:1902.10186*.
- Jiang, J. and C. Zhai (2007). A Systematic Exploration of the Feature Space for Relation Extraction. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, pp. 113–120. Association for Computational Linguistics, Rochester, New York.
- Joshi, M., D. Chen, Y. Liu, D. S. Weld, L. Zettlemoyer, and O. Levy (2020). SpanBERT: Improving Pre-training by Representing and Predicting Spans. *Transactions of the Association for Computational Linguistics*, **8**, pp. 64–77. doi: 10.1162/tacl_a-00300.
- Kambhatla, N. (2004). Combining lexical, syntactic, and semantic features with maximum entropy models for information extraction. In *Proceedings of the ACL Interactive Poster and Demonstration Sessions*, pp. 178–181.
- Karimi, A.-H., G. Barthe, B. Balle, and I. Valera (2020). Model-agnostic counterfactual explanations for consequential decisions. In *International Conference on Artificial Intelligence and Statistics*, pp. 895–905. PMLR.
- Kim, J.-D., Y. Wang, and Y. Yasunori (2013). The genia event extraction shared task, 2013 edition-overview. In *Proceedings of the BioNLP Shared Task 2013 Workshop*, pp. 8–15.
- Kipf, T. N. and M. Welling (2016). Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

- Kleinberg, J. M. (1999). Hubs, authorities, and communities. *ACM Comput. Surv.*, **31**, p. 5.
- Kobayashi, G., T. Kuribayashi, S. Yokoi, and K. Inui (2020). Attention is Not Only a Weight: Analyzing Transformers with Vector Norms. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 7057–7075. Association for Computational Linguistics, Online. doi:10.18653/v1/2020.emnlp-main.574.
- Landis, J. R. and G. G. Koch (1977). The measurement of observer agreement for categorical data. *Biometrics*, pp. 159–174.
- Lee, H., A. Chang, Y. Peirsman, N. Chambers, M. Surdeanu, and D. Jurafsky (2013). Deterministic Coreference Resolution Based on Entity-Centric, Precision-Ranked Rules. *Computational Linguistics*, **39**(4), pp. 885–916. ISSN 0891-2017. doi:10.1162/COLI_a_00152. Copyright: Copyright 2020 Elsevier B.V., All rights reserved.
- Lei, T., R. Barzilay, and T. Jaakkola (2016). Rationalizing Neural Predictions. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 107–117. Association for Computational Linguistics, Austin, Texas. doi:10.18653/v1/D16-1011.
- Li, G., D. Deng, and J. Feng (2011). Faerie: efficient filtering algorithms for approximate dictionary-based entity extraction. In *Proceedings of the 2011 ACM SIGMOD International Conference on Management of data*, pp. 529–540.
- Li, J., X. Chen, E. Hovy, and D. Jurafsky (2016a). Visualizing and Understanding Neural Models in NLP. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 681–691. Association for Computational Linguistics, San Diego, California. doi:10.18653/v1/N16-1082.

- Li, J., W. Monroe, and D. Jurafsky (2016b). Understanding Neural Networks through Representation Erasure. *ArXiv*, **abs/1612.08220**.
- Lin, D. and P. Pantel (2001). DIRT – Discovery of Inference Rules from Text. KDD '01, p. 323–328. Association for Computing Machinery, New York, NY, USA. ISBN 158113391X. doi:10.1145/502512.502559.
- Liu, N., X. Huang, J. Li, and X. Hu (2018). On Interpretation of Network Embedding via Taxonomy Induction. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '18, p. 1812–1820. Association for Computing Machinery, New York, NY, USA. ISBN 9781450355520. doi:10.1145/3219819.3220001.
- Liu, Y., M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov (2019). Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.
- Loper, E. and S. Bird (2002). NLTK: The Natural Language Toolkit. In *In Proceedings of the ACL Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics*. Philadelphia: Association for Computational Linguistics.
- Loshchilov, I. and F. Hutter (2019). Decoupled Weight Decay Regularization. In *International Conference on Learning Representations*.
- Lundberg, S. M. and S.-I. Lee (2017). A Unified Approach to Interpreting Model Predictions. In Guyon, I., U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.) *Advances in Neural Information Processing Systems 30*, pp. 4765–4774. Curran Associates, Inc.
- Manning, C., M. Surdeanu, J. Bauer, J. Finkel, S. Bethard, and D. McClosky (2014a). The Stanford CoreNLP Natural Language Processing Toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System*

- Demonstrations*, pp. 55–60. Association for Computational Linguistics, Baltimore, Maryland. doi:10.3115/v1/P14-5010.
- Manning, C. D. (2015). Last Words: Computational Linguistics and Deep Learning. *Computational Linguistics*, **41**(4), pp. 701–707. doi:doi:10.1162/COLI_a_00239.
- Manning, C. D., M. Surdeanu, J. Bauer, J. Finkel, S. J. Bethard, and D. McClosky (2014b). The Stanford CoreNLP Natural Language Processing Toolkit. In *Association for Computational Linguistics (ACL) System Demonstrations*, pp. 55–60.
- McHugh, M. L. (2012). Interrater reliability: the kappa statistic. *Biochemia medica*, **22**(3), pp. 276–282.
- Mikolov, T., K. Chen, G. Corrado, and J. Dean (2013). Efficient Estimation of Word Representations in Vector Space. In *Proceedings of the International Conference on Learning Representations (ICLR)*.
- Miller, S., H. Fox, L. Ramshaw, and R. Weischedel (2000). A Novel Use of Statistical Parsing to Extract Information from Text. In *1st Meeting of the North American Chapter of the Association for Computational Linguistics*, p. 226–233.
- Mintz, M., S. Bills, R. Snow, and D. Jurafsky (2009). Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pp. 1003–1011.
- Mohankumar, A. K., P. Nema, S. Narasimhan, M. M. Khapra, B. V. Srinivasan, and B. Ravindran (2020). Towards Transparent and Explainable Attention Models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 4206–4216. Association for Computational Linguistics, Online. doi:10.18653/v1/2020.acl-main.387.

- Moschitti, A. (2006). Making tree kernels practical for natural language learning. In *11th conference of the European Chapter of the Association for Computational Linguistics*.
- Nédellec, C., R. Bossy, J.-D. Kim, J.-J. Kim, T. Ohta, S. Pyysalo, and P. Zweigenbaum (2013). Overview of BioNLP shared task 2013. In *Proceedings of the BioNLP shared task 2013 workshop*, pp. 1–7.
- Nguyen, T.-V. T., A. Moschitti, and G. Riccardi (2009). Convolution Kernels on Constituent, Dependency and Sequential Structures for Relation Extraction. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pp. 1378–1387. Association for Computational Linguistics, Singapore.
- Pennington, J., R. Socher, and C. D. Manning (2014). Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pp. 1532–1543.
- Peters, M. E., S. Ruder, and N. A. Smith (2019). To Tune or Not to Tune? Adapting Pretrained Representations to Diverse Tasks. In *Proceedings of the 4th Workshop on Representation Learning for NLP (RepL4NLP-2019)*, pp. 7–14. Association for Computational Linguistics, Florence, Italy. doi:10.18653/v1/W19-4302.
- Poerner, N., B. Roth, and H. Schütze (2018). Evaluating neural network explanation methods using hybrid documents and morphological agreement. *arXiv preprint arXiv:1801.06422*.
- Qu, M., X. Ren, Y. Zhang, and J. Han (2018). Weakly-supervised relation extraction by pattern-enhanced embedding learning. In *Proceedings of the 2018 World Wide Web Conference*, pp. 1257–1266.
- Rajani, N. F., B. McCann, C. Xiong, and R. Socher (2019). Explain Yourself! Leveraging Language Models for Commonsense Reasoning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp.

- 4932–4942. Association for Computational Linguistics, Florence, Italy. doi:10.18653/v1/P19-1487.
- Rau, L. F., P. S. Jacobs, and U. Zernik (1989). Information extraction and text summarization using linguistic knowledge acquisition. *Information Processing & Management*, **25**(4), pp. 419–428.
- Rawal, K. and H. Lakkaraju (2020). Beyond Individualized Recourse: Interpretable and Interactive Summaries of Actionable Recourses. *Advances in Neural Information Processing Systems*, **33**.
- Ribeiro, M. T., S. Singh, and C. Guestrin (2016). Why should i trust you?: Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 1135–1144. ACM.
- Riedel, S., L. Yao, and A. McCallum (2010). Modeling relations and their mentions without labeled text. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 148–163. Springer.
- Riloff, E. (1996). Automatically generating extraction patterns from untagged text. In *Proceedings of the national conference on artificial intelligence*, pp. 1044–1049.
- Roth, D. and W.-t. Yih (2004). A Linear Programming Formulation for Global Inference in Natural Language Tasks. In *Proceedings of the Eighth Conference on Computational Natural Language Learning (CoNLL-2004) at HLT-NAACL 2004*, pp. 1–8. Association for Computational Linguistics, Boston, Massachusetts, USA.
- Sainz, O., O. Lopez de Lacalle, G. Labaka, A. Barrena, and E. Agirre (2021). Label Verbalization and Entailment for Effective Zero and Few-Shot Relation Extraction. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 1199–1212. Association for Computational Linguistics, Online and Punta Cana, Dominican Republic. doi:10.18653/v1/2021.emnlp-main.92.

- Schwab, P. and W. Karlen (2019). CXPlain: Causal Explanations for Model Interpretation under Uncertainty. In *Advances in Neural Information Processing Systems (NeurIPS)*.
- Sculley, D., G. Holt, D. Golovin, E. Davydov, T. Phillips, D. Ebner, V. Chaudhary, M. Young, J.-F. Crespo, and D. Dennison (2015). Hidden technical debt in machine learning systems. *Advances in neural information processing systems*, **28**.
- Shapley, L. S. (1952). *A Value for N-Person Games*. RAND Corporation, Santa Monica, CA. doi:10.7249/P0295.
- Shlain, M., H. Taub-Tabib, S. Sadde, and Y. Goldberg (2020). Syntactic Search by Example. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pp. 17–23. Association for Computational Linguistics, Online. doi:10.18653/v1/2020.acl-demos.3.
- Simonyan, K., A. Vedaldi, and A. Zisserman (2013). Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*.
- Situ, X., I. Zukerman, C. Paris, S. Maruf, and G. Haffari (2021). Learning to Explain: Generating Stable Explanations Fast. 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)*, pp. 5340–5355. Association for Computational Linguistics, Online. doi:10.18653/v1/2021.acl-long.415.
- Srihari, R. and W. Li (1999). Information extraction supported question answering. Technical report, CYMFONY NET INC WILLIAMSVILLE NY.
- Srihari, R. K. and W. Li (2000). A question answering system supported by information extraction. In *Sixth Applied Natural Language Processing Conference*, pp. 166–172.

- Suntwal, S., M. Paul, R. Sharp, and M. Surdeanu (2019). On the Importance of Delexicalization for Fact Verification. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 3413–3418. Association for Computational Linguistics, Hong Kong, China. doi:10.18653/v1/D19-1340.
- Surdeanu, M., R. Johansson, A. Meyers, L. Màrquez, and J. Nivre (2008). The CoNLL-2008 shared task on joint parsing of syntactic and semantic dependencies. In *Proceedings of the Twelfth Conference on Computational Natural Language Learning*, pp. 159–177. Association for Computational Linguistics.
- Surdeanu, M., J. Tibshirani, R. Nallapati, and C. D. Manning (2012). Multi-instance Multi-label Learning for Relation Extraction. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pp. 455–465. Association for Computational Linguistics, Jeju Island, Korea.
- Tang, Z., G. Hahn-Powell, and M. Surdeanu (2020). Exploring Interpretability in Event Extraction: Multitask Learning of a Neural Event Classifier and an Explanation Decoder. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*, pp. 169–175. Association for Computational Linguistics, Online. doi:10.18653/v1/2020.acl-srw.23.
- Tang, Z. and M. Surdeanu (2022). It Takes Two Flints to Make a Fire: Multitask Learning of Neural Relation and Explanation Classifiers. *Computational Linguistics*, pp. 1–40.
- Trisedya, B. D., J. Qi, and R. Zhang (2019). Entity alignment between knowledge graphs using attribute embeddings. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 297–304.
- Vafa, K., Y. Deng, D. M. Blei, and A. M. Rush (2021). Rationales for Sequential Predictions. In *Empirical Methods in Natural Language Processing*, p. 10314–10332.

- Valenzuela-Escarcega, M. A., O. Babur, G. Hahn-Powell, D. Bell, T. Hicks, E. Noriega-Atala, X. Wang, M. Surdeanu, E. Demir, and C. T. Morrison (2018). Large-scale Automated Machine Reading Discovers New Cancer Driving Mechanisms. *Database: The Journal of Biological Databases and Curation*. doi:10.1093/database/bay098.
- Valenzuela-Escárcega, M. A., Ö. Babur, G. Hahn-Powell, D. Bell, T. Hicks, E. Noriega-Atala, X. Wang, M. Surdeanu, E. Demir, and C. T. Morrison (2018). Large-scale Automated Machine Reading Discovers New Cancer Driving Mechanisms. *Database: The Journal of Biological Databases and Curation*. doi:10.1093/database/bay098.
- Valenzuela-Escárcega, M. A., G. Hahn-Powell, D. Bell, and M. Surdeanu (2016a). SnapToGrid: From statistical to interpretable models for biomedical information extraction. *arXiv preprint arXiv:1606.09604*.
- Valenzuela-Escarcega, M. A., G. Hahn-Powell, T. Hicks, and M. Surdeanu (2015). A Domain-independent Rule-based Framework for Event Extraction. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Assian Federation of Natural Language Processing: Software Demonstrations (ACL-IJCNLP)*, pp. 127–132.
- Valenzuela-Escárcega, M. A., G. Hahn-Powell, and M. Surdeanu (2016b). Odin’s Runes: A Rule Language for Information Extraction. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC’16)*, pp. 322–329. European Language Resources Association (ELRA), Portorož, Slovenia.
- Valenzuela-Escárcega, M. A., G. Hahn-Powell, and M. Surdeanu (2016). Odin’s Runes: A Rule Language for Information Extraction. In *Proceedings of the 10th International Conference on Language Resources and Evaluation*. LREC.
- Vaswani, A., N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin (2017). Attention is all you need. In *Advances in Neural Information Processing Systems*, pp. 5998–6008.

- Voita, E., R. Sennrich, and I. Titov (2021). Analyzing the Source and Target Contributions to Predictions in Neural Machine Translation. 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)*, pp. 1126–1140. Association for Computational Linguistics, Online. doi:10.18653/v1/2021.acl-long.91.
- Vu, N. T., H. Adel, P. Gupta, and H. Schütze (2016). Combining Recurrent and Convolutional Neural Networks for Relation Classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 534–539. Association for Computational Linguistics, San Diego, California. doi:10.18653/v1/N16-1065.
- Wachter, S., B. Mittelstadt, and C. Russell (2018). Counterfactual Explanations Without Opening the Black Box: Automated Decisions and the GDPR. *Harvard journal of law & technology*, **31**, pp. 841–887. doi:10.2139/ssrn.3063289.
- Wallace, E., J. Tuyls, J. Wang, S. Subramanian, M. Gardner, and S. Singh (2019). AllenNLP Interpret: A Framework for Explaining Predictions of NLP Models. In *Empirical Methods in Natural Language Processing*, pp. 7–12.
- Wang, J., J. Tuyls, E. Wallace, and S. Singh (2020). Gradient-based Analysis of NLP Models is Manipulable. *arXiv preprint arXiv:2010.05419*.
- Wang, L., Z. Cao, G. de Melo, and Z. Liu (2016). Relation Classification via Multi-Level Attention CNNs. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1298–1307. Association for Computational Linguistics, Berlin, Germany. doi:10.18653/v1/P16-1123.
- Webson, A. and E. Pavlick (2022). Do Prompt-Based Models Really Understand the Meaning of Their Prompts? In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human*

- Language Technologies*, pp. 2300–2344. Association for Computational Linguistics, Seattle, United States. doi:10.18653/v1/2022.naacl-main.167.
- Wei, J., X. Wang, D. Schuurmans, M. Bosma, E. Chi, Q. Le, and D. Zhou (2022). Chain of thought prompting elicits reasoning in large language models. *arXiv preprint arXiv:2201.11903*.
- Wiegrefe, S. and Y. Pinter (2019a). Attention is not not explanation. *arXiv preprint arXiv:1908.04626*.
- Wiegrefe, S. and Y. Pinter (2019b). 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)*, pp. 11–20. Association for Computational Linguistics, Hong Kong, China. doi:10.18653/v1/D19-1002.
- Wolf, T., L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, J. Davison, S. Shleifer, P. von Platen, C. Ma, Y. Jernite, J. Plu, C. Xu, T. L. Scao, S. Gugger, M. Drame, Q. Lhoest, and A. M. Rush (2020). Transformers: State-of-the-Art Natural Language Processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 38–45. Association for Computational Linguistics, Online.
- Wu, S. and Y. He (2019). Enriching pre-trained language model with entity information for relation classification. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, pp. 2361–2364.
- Xu, Y., L. Mou, G. Li, Y. Chen, H. Peng, and Z. Jin (2015). Classifying Relations via Long Short Term Memory Networks along Shortest Dependency Paths. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 1785–1794. Association for Computational Linguistics, Lisbon, Portugal. doi:10.18653/v1/D15-1206.

- Yamada, I., A. Asai, H. Shindo, H. Takeda, and Y. Matsumoto (2020). LUKE: deep contextualized entity representations with entity-aware self-attention. *arXiv preprint arXiv:2010.01057*.
- Yarowsky, D. (1995). Unsupervised word sense disambiguation rivaling supervised methods. In *33rd annual meeting of the association for computational linguistics*, pp. 189–196.
- Zechner, K. (1997). A literature survey on information extraction and text summarization. *Computational Linguistics Program*, **22**.
- Zelenko, D., C. Aone, and A. Richardella (2003). Kernel methods for relation extraction. *Journal of machine learning research*, **3**(Feb), pp. 1083–1106.
- Zeng, D., K. Liu, S. Lai, G. Zhou, and J. Zhao (2014). Relation classification via convolutional deep neural network. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pp. 2335–2344.
- Zhang, D. and D. Wang (2015). Relation classification via recurrent neural network. *arXiv preprint arXiv:1508.01006*.
- Zhang, R., Y. Yu, P. Shetty, L. Song, and C. Zhang (2022). Prompt-Based Rule Discovery and Boosting for Interactive Weakly-Supervised Learning. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 745–758. Association for Computational Linguistics, Dublin, Ireland. doi:10.18653/v1/2022.acl-long.55.
- Zhang, Y., P. Qi, and C. D. Manning (2018). Graph Convolution over Pruned Dependency Trees Improves Relation Extraction. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 2205–2215. Association for Computational Linguistics, Brussels, Belgium. doi:10.18653/v1/D18-1244.

- Zhang, Y., V. Zhong, D. Chen, G. Angeli, and C. D. Manning (2017). Position-aware Attention and Supervised Data Improve Slot Filling. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP 2017)*, pp. 35–45.
- Zhao, S. and R. Grishman (2005). Extracting Relations with Integrated Information Using Kernel Methods. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pp. 419–426. Association for Computational Linguistics, Ann Arbor, Michigan. doi:10.3115/1219840.1219892.
- Zhao, Y. and S. Bethard (2020). How does BERT’s attention change when you fine-tune? An analysis methodology and a case study in negation scope. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 4729–4747. Association for Computational Linguistics, Online. doi:10.18653/v1/2020.acl-main.429.
- Zhou, G., J. Su, J. Zhang, and M. Zhang (2005). Exploring Various Knowledge in Relation Extraction. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pp. 427–434. Association for Computational Linguistics, Ann Arbor, Michigan. doi:10.3115/1219840.1219893.
- Zhou, P., W. Shi, J. Tian, Z. Qi, B. Li, H. Hao, and B. Xu (2016). Attention-Based Bidirectional Long Short-Term Memory Networks for Relation Classification. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 207–212. Association for Computational Linguistics, Berlin, Germany. doi:10.18653/v1/P16-2034.
- Zhou, W., H. Lin, B. Y. Lin, Z. Wang, J. Du, L. Neves, and X. Ren (2020). Nero: A neural rule grounding framework for label-efficient relation extraction. In *Proceedings of The Web Conference 2020*, pp. 2166–2176.