PatternRank: Jointly Ranking Patterns and Extractions for Relation Extraction Using Graph-Based Algorithms

Robert Vacareanu[♠,♣], Dane Bell[*], and Mihai Surdeanu[♠]

[♠]University of Arizona, Tucson, AZ, USA
[*]LUM.ai, Tucson, AZ, USA
[♣]Technical University of Cluj-Napoca, Cluj-Napoca, Romania
{rvacareanu,msurdeanu}@arizona.edu, *dane@lum.ai*

Abstract

In this paper we revisit the direction of using lexico-syntactic patterns for relation extraction instead of today's ubiquitous neural classifiers. We propose a semi-supervised graph-based algorithm for pattern acquisition that scores patterns and the relations they extract jointly, using a variant of PageRank (Page et al., 1999). We insert light supervision in the form of seed patterns or relations, and model it with several custom teleportation probabilities that bias random-walk scores of patterns/relations based on their proximity to correct information. We evaluate our approach on Few-Shot TACRED (Zhang et al., 2017; Sabo et al., 2021), and show that our method outperforms (or performs competitively with) more expensive and opaque deep neural networks. Lastly, we thoroughly compare our proposed approach with the seminal RlogF pattern acquisition algorithm of Riloff (1996), showing that it outperforms it for all the hyper parameters tested, in all settings.[1]

1 Introduction

Rule-based methods hastily fell out of favor after the "deep learning tsunami" hit natural language processing (Manning, 2015). However, deep learning methods are not perfect: they continue to remain "blackboxes," despite recent effort towards untangling their representations (Jain and Wallace, 2019; Wiegreffe and Pinter, 2019; Kobayashi et al., 2020). Moreover, the presence of various linguistic phenomenons in the hidden representation of deep models does not imply that the model will use them (McCoy et al., 2019). On the other hand, rules are interpretable by design. Furthermore, changing something in a deep learning model often changes everything (Sculley et al., 2015; Arpteg et al., 2018), while rules are disentangled, i.e., modifying a rule impacts only its own matches.

Here, we introduce a novel pattern acquisition method for relation extraction, which uses graph-based techniques that operate over the entire topology of the bipartite graph that contains candidate patterns and their extracted relations. More specifically, we leverage a variant of PageRank (Page et al., 1999) to *jointly* score candidate patterns and their extractions. Such an approach has the advantage of softening the sparsity problem: an unknown, but potentially in-domain candidate relation is not automatically considered incorrect (as most "traditional" semi-supervised algorithms would consider it); instead it receives a non-zero score that depends on how reachable it is in this graph.

Our main contributions are:

- We propose a graph-based algorithm, PatternRank, for jointly scoring patterns and extractions by considering the whole topology of the graph that contains them. Our algorithm captures light supervision (in the form of seed relations or patterns) with several custom teleportation probabilities that bias random-walk scores of patterns/relations based on their proximity to correct information.

- We evaluate our proposed approach on the Few-Shot TACRED task (Zhang et al., 2017; Sabo et al., 2021). Our results show that the performance is better than (or at least equivalent with) several state-of-the-art neural approaches, while also being fully interpretable.

- We perform an extensive comparison between our proposed method and the seminal pattern acquisition algorithm RLogF of Riloff (1996), which is closest in spirit to our direction, explaining why it outperforms it.

[1]Code available at `https://github.com/clulab/releases/tree/master/pandl2022-patternrank`

2 Terminology

Before moving on, we introduce here key terminology used throughout the paper.

Pattern We define a *pattern* as a linear sequence of surface and syntactic constraints. For example, the pattern `[lemma=born] >nmod_in [ne=LOC]` matches if the text contains the lemma *born*, linked through a *nmod_in* dependency to a token labeled as a *location* named entity.

Relation We evaluate our proposed method on relation classification. As such, we use the term *relation* to refer to the semantic relation holding between two given entities. For example, for the sentence `John was born in Tampa`, where the two entities are *John* and *Tampa*, respectively, the relation will be *per:born_in*.

3 Related Work

Rule-based systems received tremendous attention the "pre deep-learning era." In a seminal work, Hearst (1992) proposes a method to learn hyponymy relations using hand-written patterns of the form NP_0 `such as {`NP_1 `.., (and|or)}` NP_n. Riloff (1993) introduced AutoSlog, a system for automatically learning domain-specific dictionaries using a few hand-written patterns. The system is improved in (Riloff, 1996) by combining the original AutoSlog with statistical techniques and introduced the RlogF pattern scoring function, defined as: $RlogF(pattern_i) = \frac{F_i}{N_i} log_2(F_i)$, where F_i is the number of extractions for the corresponding class, and N_i is the total number of extractions.

The duality between patterns and relations has also been explored in (Brin, 1998). More concretely, the authors first generate a set of patterns from a set of relations. Then, using the previously generated set of patterns they generate a new set of relations. On a high level, we employ a similar algorithm. Conceptually similar method has also been in explored in (Riloff and Jones, 1999; Riloff and Wiebe, 2003).

3.1 Automatic pattern learning

The typical approach to semi-supervised pattern learning is to initialize the learning algorithm with a small set of known seed relations (Riloff and Jones, 1999; Riloff and Wiebe, 2003; Gupta and Manning, 2014). Generally, the approach is to consider the matches outside the seed relations as incorrect matches (Riloff and Jones, 1999; Riloff and Wiebe, 2003). Gupta and Manning (2014) improved the approach by allowing soft matches. Concretely, they predict the labels on unlabeled entities using a concatenation of different features, including word embeddings. We approach this issue by interpreting the matches of all the patterns as a graph and scoring everything jointly. In a sense, our approach can be thought of as a guided wisdom of the crowd approach.

3.2 Graph-based pattern learning

Treating pattern acquisition from a graph-based perspective is an under-explored approach. However, we are not the first to view it from this point of view. Kozareva et al. (2008) briefly explored using PageRank, among other scoring techniques, but in another setting and only for relation scoring. They focused on learning hyponym relations starting from a single doubly-anchored pattern template and a single seed instance. They build a directed graph G, where an edge (u, v) represents that using the relation u in the pattern template extracted the relation v. Then, they used graph-based scoring techniques to select the best relations. In contrast, our proposed method learns both patterns and relations jointly, starting from either a pattern or a seed relation.

Perhaps the work of (Hassan et al., 2006) is closest in nature with our approach. They proposed using Hyperlink-Induced Topic Search (HITS) (Kleinberg, 1999) to jointly learn patterns and relations without any supervision. One limitation of their method is that it is unable to accommodate initial information about which seeds relations or patterns are considered correct, information which can come from a previous component in the pipeline, or from human supervision. In contrast, our algorithm can incorporate human supervision through seed relations or patterns, and we use a new topic-sensitive variant of PageRank (Page et al., 1999) to model human supervision during the random walk.

Although rule-based approaches received a lot of attention prior to the deep-learning era, graph-based approaches for pattern acquisition remain under explored. In this paper we model the pattern scoring problem as a random walk over a graph consisting of patterns and relations, and mitigate sparsity through custom teleportation probability. We empirically show that this strategy leads to better results in realistic few-shot settings for relation extraction (Sabo et al., 2021).

4 Method

Our approach, called PatternRank, follows an iterative approach, which alternates between learning patterns from known relations (or seeds), and extracting new relations matched by these patterns. The key contribution of our work is the novel scoring strategy, which scores patterns and extracted relations jointly, using a graph-based algorithm. We describe the three components of our method in greater detail below.

4.1 Generating Candidate Patterns

We generate candidate patterns starting from a small set of seed relations and some (unannotated) text corpus. Our seed relations take the form (`agent`, `predicate`, `patient`), and the patterns are the syntactic paths connecting the three elements. For example, from the seed relation (*drought*, *causes*, *famine*), we consider all the syntactic paths connecting *drought* with *famine* that pass through the predicate *causes* in the corpus.[2] Then, to reduce search space, we filter these candidate patterns as follows:

1. We remove candidate patterns observed less than k times in the corpus.

2. We avoid mirror patterns. For example, if in the set of candidate patterns we have both `<nsubj causes >dobj` and `<dobj causes >nsubj` we keep the patterns that extracted more relations overall.

3. We remove patterns that contain the same type of dependencies on both sides of the predicate. For example, we filter out patterns like `<nsubj causes >nsubj >dobj`.

Our method accommodates the case where the starting point are seed patterns instead of seed relations. In this case we apply the seed patterns to generate seed relations. Then, we continue with pattern generation as described above.

An important difference from previous work is that our method does not use a pre-computed index with all the patterns on the corpus (Lin and Pantel, 2001). In realistic settings, such an index of rules is prohibitively large due to the sparsity of language. Instead, we generate them *on the fly* using Odinson, a rule-based information extraction framework that indexes *atomic* syntactic dependencies (Valenzuela-Escárcega et al., 2020). To bridge the fact that the Odinson index contains individual syntactic dependencies, whereas our rules are compositional (i.e. they aggregate multiple atomic (word- or dependency-level) constraints such as matching lemmas or part-of-speech tags), we create patterns by searching the Odinson for all the syntactic paths connecting the seed relations. For example, for the seed relation (*drought*, *causes*, *famine*) we search for the paths connecting *drought* to *famine* via *causes* with the Odinson query: `[lemma=drought] («|»)* [lemma=cause] («|»)* [lemma=famine]`, where `(«|»)*` stands for zero or more syntactic dependencies connecting the token to the left to the one on the right. This significantly speeds up pattern generation without relying on an explicit pattern index.

4.2 Extracting Candidate Relations

Using the previously generated patterns, we apply them over our corpora and record the relations matched by them. We filter the extractions based on the part-of-speech tags of the constituent words and based on their concreteness. For example, we remove relations where the agent or patient are pronouns or symbols. Further, we use the concreteness norm database of Brysbaert et al. (2014)[3] to filter out relations that are too generic (e.g., (`someone`, `causes`, `something`)). The threshold is an application specific hyper parameter.

4.3 Scoring Patterns and Relations

We score patterns and relations jointly using a graph-based algorithm. Specifically, we view the patterns and relations as a bipartite graph. Nodes represent patterns or relations. Edges between nodes of the same type are prohibited. Two nodes are connected with an edge if the corresponding pattern matched the corresponding relation. Edges are undirected, as the pattern/relation matching can be seen as bidirectional. Additionally, edges are weighted, where the weight represents the number of times the pattern extracted the relation.

Formally, we have the bipartite graph $G = (P, R, E)$, with two partitions: P, which contains

[2] We use lemmas instead of the verbatim words for lexical items in the paths, and universal dependencies for the syntactic annotations. Our method accommodates seed relations provided without a predicate, in which case we generate syntactic paths that simply connect the agent and patient.

[3] This database contains psycholinguistic concreteness norms for 40,000 generally known English lemmas on a numerical scale from 1 (highly abstract) to 5 (highly concrete).

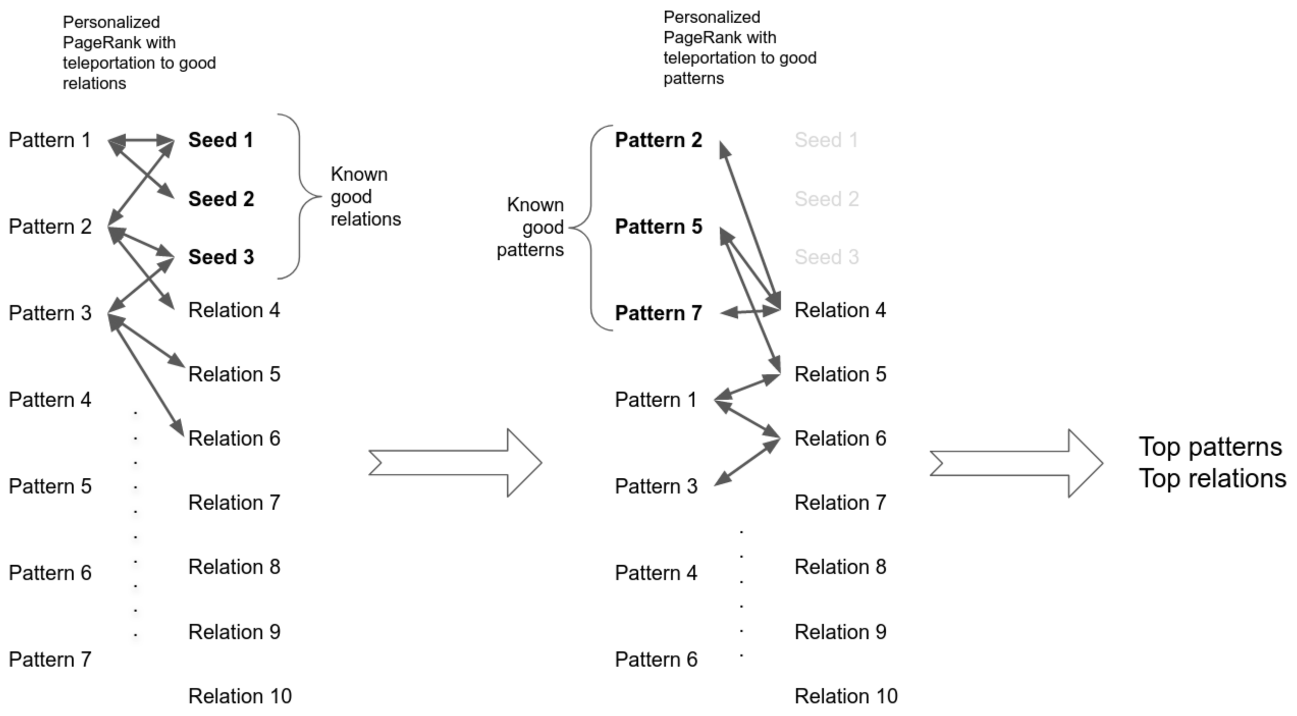

Figure 1: Our proposed method for jointly scoring patterns and relations. We apply the personalized PageRank algorithm twice. First, we use a teleportation vector where we teleport only to the seed relations, i.e., example relations known to be correct. After this step, we take the top scoring patterns and consider them to be correct. We apply the personalized PageRank algorithm a second time, but this time we teleport only to the previously selected patterns. The algorithm outputs a list of patterns and relations, each with an associated score.

the nodes corresponding to the patterns, and R, which has the nodes corresponding to the relations that were extracted using the patterns from P. An undirected edge $(p, r) \in E$ between a pattern $p \in P$ and an extracted relation $r \in R$ is added if the pattern p matches the relation r in the corpora. We attach a weight $w \in \mathbb{N}$ to the (p, r) edge, representing the number of time p extracted r.

Equipped with this representation, we apply graph-based algorithms for jointly scoring the patterns and the relations. We adapt Topic-Sensitive PageRank (Haveliwala, 2002), a variant of PageRank (Page et al., 1999) with custom teleportation probabilities. We apply this algorithm twice, as follows:

1. We first apply it to obtain the scores for each of the generated patterns. In this setting, we set the teleportation probability to uniformly teleport *only* to seed relations, i.e., extractions specified ahead of time by the user to be correct.

2. Using the PageRank scores generated in the previous step, we select the desired number of patterns by keeping the top patterns with the highest scores.

3. We apply the topic-sensitive PageRank a second time. For this run, we teleport *only* to the patterns selected in the previous step. Further, this time we do not teleport uniformly, but proportional with the score of a given pattern. This is done to prevent a pattern which extracted very few relations to dominate the random walk.

At the end of this process, the algorithm generates scores for both patterns and relations. Figure 1 summarizes this whole process.

An important thing to note is that our proposed method considers the entire topology for computing a score, without ignoring or considering an unknown extraction to be bad. As a consequence, a pattern which rarely matches the seed relations can still obtain a high score if its matches tend to overlap with patterns that predominantly match the seed relations. Traditional bootstrapping approaches for relation extraction typically mark unknown extractions as bad, ignore them, or (better) try to score them relative to the seed relations (Gupta and Manning, 2014). In a sense, we handle this problem via a guided wisdom of the crowd approach, where patterns which overlap in extractions with patterns that match predominantly the seed relations can

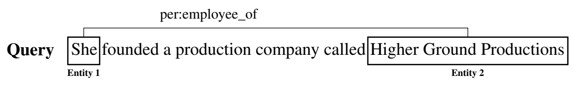
per:employee_of
Query
She
founded a production company called
Higher Ground Productions
Entity 1
Entity 2

Figure 2: Example of a sentence in the TACRED dataset (Zhang et al., 2017). The task is to predict the relation that holds between the two entities, which in this example is `per:employee_of`.

still obtain a high score.

5 Experiments

5.1 Experimental Setting

For all experiments we used Odinson for indexing (Valenzuela-Escárcega et al., 2020). Further, we used the UMBC corpus (Han et al., 2013) to obtain more robust statistics on a rule's extractions.

5.2 Relation Extraction

We evaluate PatternRank, our proposed method, on the Few-Shot TACRED dataset (Sabo et al., 2021), a few-shot variant of the original TACRED dataset (Zhang et al., 2017). TACRED is a relation classification task, where you are asked to predict the relation between the two (given) entities in a sentence.[4] An example of the input is presented in Figure 2. In the few-shot variant, the test relation classes are *not seen* during training. Instead, a method has to generalize to new relation types using only a few examples of the new relations.[5] More formally, at testing time, the task requires the classification of a relation that holds between two given entities in a *query sentence*, using only a *support set*, which consists of examples for each of the 5 possible relations for the query sentence. If none of the relations hold for the query sentence, the prediction should be `no_relation`.

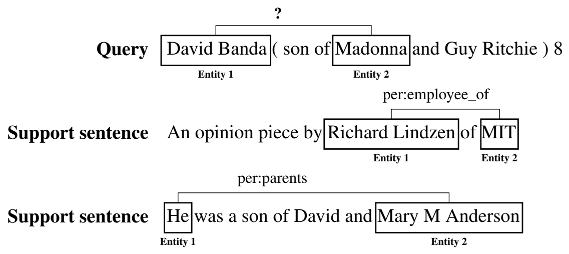

Figure 3: A shortened test example of Few-Shot TACRED. Here, we consider only two support sentences in total, which makes it a 2-way 1-shot setting. If the true relation of the query sentence is not found among the support sentences, then the label is `no_relation`. In this example the gold relation is `per:parents`.

[4]Entity types are also provided.

[5]Typically 1 or 5 sentences per relation.

Because our method starts from seed patterns or relations, we convert each support sentence into seed patterns[6] by artificially constructing patterns that extract the specified entities. We construct two such patterns, one which contains only surface constraints, and one that relies on syntax. For example, for the relation type *per:city_of_birth* and the support sentence ***Rothman*** *was born in* ***San Francisco*** *in 1932*, we automatically generate the surface pattern: `[ne=PER] [lemma=be] [lemma=born] [lemma=in] [ne=LOC]`, and the syntactic pattern: `[ne=PER] <nsubjpass [lemma=born] >nmod_in [ne=LOC]`. We then use UMBC (Han et al., 2013) to obtain robust statistics on their extractions, by making use of the seeds and the relations available so far. More precisely, we apply the patterns on UMBC and extract candidate relations. Then, we use this set of candidate relations to build more patterns. Lastly, we apply this set of patterns over UMBC to obtain the extractions which will then ranked.

We compare our proposed approach with one strong baseline, and several state-of-the-art neural methods.

Our baseline is driven by the type of the two participating entities. In particular, given a query sentence q, which has the entity types $(E1, E2)$ (e.g. $(\texttt{PER}, \texttt{PER})$ for the example in Figure 3), the algorithm:

1. Discards the sentences from the support set which do not have the entity types $(E1, E2)$. For example, for the example in Figure 3, we discard the first support sentence, as the entity types in it are $(\texttt{PER}, \texttt{ORG})$.

2. Adds one artificial support sentence with the relation type set to `no_relation`, if in the background training set there are sentences with the same entity types as the query sentence. We do this to ensure that the `no_relation` label remains a classification option, as there may be multiple relations that can hold between the two entities.[7] For example, we add *His son* ***Richmond Jr*** *and grandson* ***Richmond III*** *both became football stars.* with the relation label `no_relation`.

[6]We empirically observed that seed patterns are less noisy than seed relations.

[7]Examples of relations that can hold between `PER` and `PER`: `per:parents`, `per:spouse`, `per:children`, `per:siblings`, `per:other_family`.

3. Randomly picks one of the remaining sentences with matching entity types from the support set, and labels the query sentence with the corresponding relation type. For example, the baseline may randomly select the sentence ***He** was a son of David and **Mary M Anderson***, and predict `per:parents`.

Additionally, we compare our method with the following state-of-the-art supervised methods:

Sentence-Pair (Gao et al., 2019): Concatenates the query sentence with each support sentence and runs a sequence classification variant of BERT (Devlin et al., 2018) over it, predicting a two-element vector. The first element represents the probability of the pair sharing the same relation type and the second element represents the probability of the pair not sharing the same relation type. It takes the average score in case there are multiple sentences with the same relation type. Additionally, the score for `no_relation` is considered to be the smallest value assigned by the method for the probability that the pair does not share the same relation type.

Threshold (Sabo et al., 2021): Assigns the label `no_relation` if the score of the concatenation is smaller than a learned threshold. Otherwise, it assigns the relation type associated with the highest similarity score, as in Sentence-Pair.

NAV (Sabo et al., 2021): A transformer-based classifier which uses the background training set to learn a vector for the `no_relation` label. At test time, it computes the similarity score between the query sentence and each support sentence. Additionally, it computes the similarity score between the query sentence and the vector associated with `no_relation`. Finally, it outputs the relation associated with the highest score.

MNAV (Sabo et al., 2021): Conceptually similar with *NAV*, but instead of using a single vector for the `no_relation` label, it uses multiple vectors. The rationale behind adding multiple vectors is that it is expected to ease the embedding space constraints. The number of vectors is treated as a hyperparameter.

We present our results in Table 1. Note that our method obtains 12.72 F1 in the more challenging 5-way 1-shot setting (where only 1 support sentence is provided per relation label), and 22.13 F1 in the 5-way 5-shot setting (where 5 sentences are provided per type) using 100 patterns. The fact that our method outperforms all others in the 1-shot setting, indicates that our graph-based algorithm generalizes better in the minimal supervision setting. We discuss the results in detail below.

Method	5-way 1-shot	5-way 5-shot
Baseline	10.82±0.01%	10.90±0.01%
Riloff (5)	4.55±0.61%	15.28± 1.71
Riloff (100)	11.68±0.80%	22.01± 1.76
Sentence-Pair	10.19±0.81%	–
Threshold	6.87±0.48%	13.57±0.46%
NAV	8.38±0.80%	18.38±2.01%
MNAV	12.39±1.01%	**30.04±1.92%**
Ours (5)	6.53±0.49%	17.05± 1.87 %
Ours (100)	12.72±1.12%	22.13± 1.94 %
Ours (all rules)	**14.06±0.96%**	22.16± 1.67 %

Table 1: Performance comparison between our proposed method, a baseline driven by entity types, Riloff's RlogF approach (Riloff, 1996), and other state-of-the-art neural methods. For both Riloff and our method, we report the results using 5 and 100 patterns learned by the two approaches. Additionally, we add the performance when using all patterns.

6 Analysis and Discussion

6.1 PatternRank vs. Neural Methods

The results in Table 1 show that our method outperforms (or, at least, performs equivalently with) all neural methods in the 1-shot setting, and performs competitively in the 5-shot one. Using all rules we obtain a performance of 14.06 in the harder 5-way 1-shot case, outperforming the previous state-of-the-art of 12.39 by over 1.5 F1 points. This strong performance is preserved even if we use only 100 rules.[8] This confirms our intuition that there is value in graph-based approaches in the neural era, especially in settings with minimal supervision, which are closer to real-world applications of relation extraction. Our conjecture is that teleportation reduces sparsity by allowing patterns that overlap with patterns that predominantly match the seed relations to obtain a high score. At a high level, this approach acts akin to a guided wisdom of the crowd.

Importantly, note that our method produces a relatively small number of interpretable rules, which is a radical departure from these neural methods. We argue that this is a step forward towards reducing the high maintainability cost of neural systems.

One potential argument against our method is that it needs an external text corpus at training time. For example, in this work we used UMBC (Han

[8]Using the top 100 rules per support sentence translates to using approximately 13% of all the patterns found.

et al., 2013). However, so do neural methods: all neural methods reported here rely on transformer networks, which have been pretrained on much larger resources than the corpus we used.

6.2 PatternRank vs. Riloff's RlogF

We compare the scores proposed by our method with the scores proposed by the RlogF scoring function Riloff (1996), which is closest in spirit to our direction. We do not include in this analysis other bootstrapping approaches, such as (Gupta and Manning, 2014; Collins and Singer, 1999), which focus on other NLP tasks, such as named entity recognition. RlogF takes into consideration the number of times a rule matched a seed compared to the total number of matches, as follows:

$$RlogF(pattern_i) = \frac{F_i}{N_i} \cdot log_2(F_i)$$

where F_i is the number of times $pattern_i$ matched a known good relation and N_i is the total number of matches of $pattern_i$. This scoring function treats each pattern in isolation and considers every match outside the seed set as negative.

As Table 1 shows, our approach outperforms RlogF when using 5 patterns and 100 patterns respectively.[9] When using only 5 rules, our proposed approach gives a 40% relative improvement. When using 100 rules, our proposed approach improves RlogF by over 1 F1 points. Overall, we perform much better with few rules and when data is sparse, but our proposed approach outperforms Rlogf in all our experiments, regardless of the setting.

To better understand the differences between the two approaches, we perform an analysis on the patterns resulted from the support sentences of Few-Shot TACRED dev partition. That is, for each support sentence we learn patterns that will be representative for that particular relation. Each pattern has an associated score, given by our proposed method. In the process, we keep track of the number of good and total matches respectively, which are used in RlogF. When comparing the performance, we repeat our experiments 5 times and report the mean and standard deviation. We used the 5-way 1-shot split, unless otherwise stated.

[9]We omit running RlogF with all the rules because there is limited benefit of ranking if everything is used, regardless of score. As such, when using all the rules both methods obtain the same score, as they should.

6.2.1 Score agreement

First, we compare the agreement between the scores assigned by both methods. Since we are interested in ranking and not in the values of the scores, we use the Kendall rank correlation coefficient. Naively calculating Kendall's tau between the scores given by RlogF and the scores given by PatternRank yields $\tau = 0.98 \pm 0.02$. However, this is misleading because both scoring methods tend to agree for the low-scoring patterns, which represents the majority of the cases. If, instead, we restrict our analysis to patterns that match seed relations, we obtain $\tau = -0.19 \pm 0.07$.[10] This indicates that the rankings are more dissimilar than the naive ranking would suggest. Manually inspecting the scoring, we observed that RlogF favors high precision patterns, regardless of the number of matches. For example, for the relation *org:country_of_headquarters*, our method scored `[ne=ORG] [lemma=in] [ne=LOC]` the highest, while RlogF scored `[ne=ORG] <conj_and [lemma=president] >nmod_of [ne=LOC]` the highest.

In order to better understand why RlogF over emphasizes precision, we mathematically compared what it means for $pattern_j$ to be scored higher than $pattern_i$ according to this algorithm. Assuming that the $pattern_i$ is known, starting from $RlogF(pattern_i) - RLogF(pattern_j) < 0$, assuming $F_i, F_j, N_i, N_j > 0$ and defining: $\alpha = \frac{F_i}{N_i}$, where $0 < \alpha \leq 1$, and $\beta = \frac{F_j}{N_j}$, where $0 < \beta \leq 1$, we have: $\alpha log_2(F_i) < \beta log_2(F_j)$, and, thus, $F_i < F_j^{\frac{\beta}{\alpha}}$. As such, if α is close to 1 (i.e., the expected value for correct patterns), $pattern_j$ needs exponentially more correct matches the smaller β gets. For example, if $F_i = 100, \alpha = 1$, for a $\beta = 0.75$ we would need $F_j = 465$.

We show the histogram of the resulting scores from both methods in Figure 4. Because RLogF is an unbounded score, we normalize the rule scores between $[0, 1]$. Additionally, we consider only the sentences which resulted in more than 20 patterns, in order to have robust statistics across sentences.[11] We observe that our proposed method produces scores that are more spread in the domain.

Additionally, we note that RlogF scored a higher number of patterns in the higher end of the spectrum ($[0.8, 1]$) than our proposed method. Manu-

[10]Additionally, RlogF gives a score of 0 for patterns that did not match at least 2 times any seed relations.

[11]That includes over 80% of the sentences.

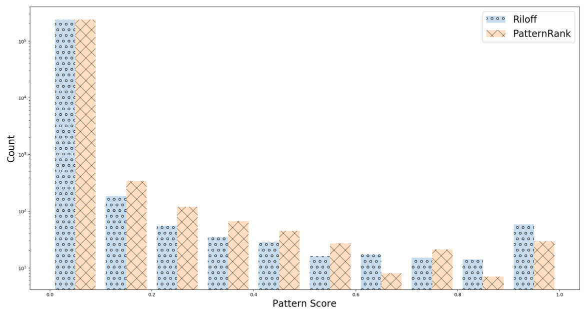

Figure 4: Histogram of scores for the obtained patterns, scored with RlogF and PatternRank. For robust statistics, we consider only sentences which result in more than 20 patterns. Additionally, we normalize the pattern scores per sentence to be in $[0, 1]$. We show the count in log scale for increased visibility.

ally analyzing such cases, we observe that RlogF tends to assign a high scores to patterns with low number of total matches N_i, if its precision F_i is high. When the good matches are sparse in the corpora, this can result in rules that matched only a few times the relations of interest to obtain a high score, when compared with the other rules.[12] This is in line with our previous observation that RLogF over-emphasizes precision.

6.2.2 Pattern importance

Following the observation in Section 6.2.1 that RlogF tends to score more patterns higher than our proposed method, we investigated the performance of the system when taking the n highest-scoring patterns, with both our method and RlogF. We show the results in Figure 5. We note that our method outperforms RlogF consistently until the number of patterns for each sentence gets saturated. We interpret this as further evidence that RlogF assigns overly optimistic scores to patterns with high precision, which can be detrimental when the statistics are not robust.

6.2.3 Random Jump Probability

Our proposed method makes use of the random jump probability, which ensures that the resulting matrix is ergodic. We assess the influence of this hyper-parameter by analyzing the performance of the method when varying it. We set the number of patterns used for each method to 10. As highlighted in Figure 6, we found our system to be robust to hyper-parameter changes, outperforming RlogF for all the values we tested.[13]

[12]Changing $\frac{F_i}{N_i} \cdot log_2(F_i)$ to $\frac{F_i}{N_i} \cdot log_2(N_i)$ does not change the histogram, nor the performance.

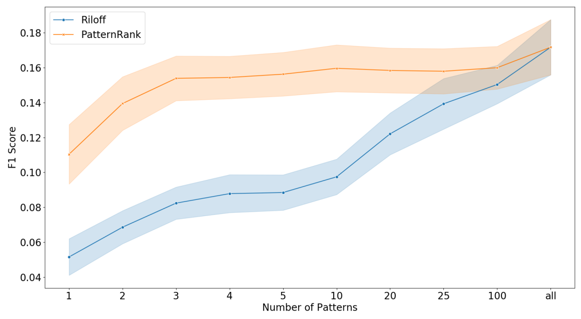

Figure 5: Performance of our method compared with RlogF, varying the number of patterns accepted. We note that our proposed method consistently outperforms RLogF up until the number of patterns are saturated.

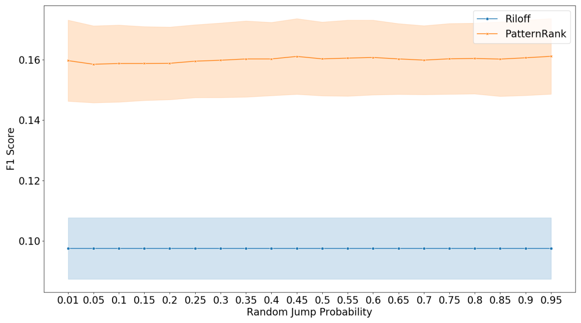

Figure 6: Performance of our method compared with RlogF, when varying the random jump probability. We used the top 10 best patterns according to each method. We average over all the dev episodes. We note that our proposed method consistently outperforms RLogF, and that the performance is robust to changes in the random jump probability.

6.3 Resulting Rules

Quantitatively, our proposed method outperforms Riloff on the relation classification task regardless of the number of patterns considered. Qualitatively, analyzing the top scored patterns, we observe that our proposed method tends to prefer patterns which offer a better balance between precision and recall due to using the whole topology of the pattern/relation graph. We show a few examples of the top scored patterns with both methods in Table 2 for two relations from development: `org:country_of_headquarters` and `per:age`.

6.4 Limitations

Our proposed method provides some advantages, such as simplicity and greater interpretability. Nevertheless, it has some limitations. First, our pro-

[13]A random jump probability of 0 is not possible because then the graph might not be connected. A random jump probability of 1 is not useful because it translates to jumping from node to node, randomly.

Relations	PatternRank (ours)	Riloff

`org:country_of_headquarters`	`[ne=ORG]+ [lemma=in] [ne=LOC]+`
	`[ne=ORG]+ [lemma=","] [ne=LOC]+`
		`[ne=ORG]+ <nmod:poss [lemma=campaign] >nmod_against [ne=LOC]+`
		`[ne=ORG]+ <appos [lemma=institute] >nmod_in [ne=LOC]+`

`per:age`	`[ne=PER]+ [lemma=","] [ne=NUMBER]+`
	`[ne=PER]+ [lemma=be] [ne=NUMBER]+`
		`[ne=PER]+ >appos [lemma=kan] >nummod [ne=NUMBER]+`
		`[ne=PER]+ >appos [lemma=student] >det:qmod [ne=NUMBER]+`

Table 2: Comparison between the top scoring patterns according to our method and to RlogF. We observe that our proposed method tends to score higher patterns that generalize better, i.e., patterns with fewer lexical or syntactic constraints.

posed method depends on an external explicit corpus. Typical pre-trained LMs compress the underlying text used for training in its parameters, while our proposed method needs the explicit text.

Second, our assumption that the same relation holds between entities of type (ENTITY1,ENTITY2) might not always be true. Nevertheless, this assumption was empirically proven using distant supervision (Mintz et al., 2009). Our empirical results add evidence to its efficacy. However, this assumption must be evaluated before employing this method in downstream applications.

Third, though rules are generally less prone to overfitting and offer good out-of-domain generalization, they may have limited expressivity when compared to neural methods. Nevertheless, the capacity needs of a model should be evaluated on a per-application basis. For example, (Tang and Surdeanu, 2023) found that using only rules can achieve a performance of over 65% F1 on the supervised TACRED task. This is comparable to state-of-the-art neural methods, which by now obtain 70+% F1,[14] suggesting that rule-based methods can be competitive on information extraction tasks.

[14] https://paperswithcode.com/sota/relation-extraction-on-tacred

7 Conclusion

We propose a new pattern acquisition method for relation extraction, which uses graph-based techniques that operate over the entire topology of the bipartite graph that contains candidate patterns and their extracted relations.

We evaluate our proposed approach on the Few-Shot TACRED task (Sabo et al., 2021), a more realistic and harder variant of TACRED (Zhang et al., 2017). Our proposed approach outperforms or performs comparably with more opaque neural methods. Further, we empirically show that our proposed method performs better than the seminal pattern scoring method proposed in (Riloff, 1996), RLogF. Lastly, we highlight some of the limitations of our proposed approach.

All in all, we provide compelling evidence that, for specific applications, rule-based methods continue to offer comparable or better performance than their neural counterparts, and, thus, they should not be overlooked by current and future research on information extraction.

Acknowledgments

This work was partially supported by the National Science Foundation (NSF) under grant #2006583. Mihai Surdeanu declares a financial interest in lum.ai. This interest has been properly disclosed to the University of Arizona Institutional Review Committee and is managed in accordance with its conflict of interest policies.

References

Anders Arpteg, Björn Brinne, Luka Crnkovic-Friis, and J. Bosch. 2018. Software engineering challenges of deep learning. *2018 44th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*, pages 50–59.

Sergey Brin. 1998. Extracting patterns and relations from the world wide web. In *WebDB*.

Marc Brysbaert, Amy Beth Warriner, and Victor Kuperman. 2014. Concreteness ratings for 40 thousand generally known english word lemmas. *Behavior research methods*, 46(3):904–911.

Michael Collins and Yoram Singer. 1999. Unsupervised models for named entity classification. In *EMNLP*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Tianyu Gao, Xu Han, Hao Zhu, Zhiyuan Liu, Peng Li, Maosong Sun, and Jie Zhou. 2019. Fewrel 2.0: Towards more challenging few-shot relation classification. In *EMNLP/IJCNLP*.

S. Gupta and Christopher D. Manning. 2014. Improved pattern learning for bootstrapped entity extraction. In *CoNLL*.

Lushan Han, Abhay Lokesh Kashyap, Timothy W. Finin, James Mayfield, and Jonathan Weese. 2013. Umbc_ebiquity-core: Semantic textual similarity systems. In **SEMEVAL*.

Hany Hassan, Ahmed Hassan Awadallah, and Ossama Emam. 2006. Unsupervised information extraction approach using graph mutual reinforcement. In *EMNLP*.

Taher H. Haveliwala. 2002. Topic-sensitive pagerank. In *Proceedings of the 11th International Conference on World Wide Web*, WWW '02, page 517–526, New York, NY, USA. Association for Computing Machinery.

Marti A. Hearst. 1992. Automatic acquisition of hyponyms from large text corpora. In *COLING 1992 Volume 2: The 14th International Conference on Computational Linguistics*.

Sarthak Jain and Byron C. Wallace. 2019. Attention is not explanation. *CoRR*, abs/1902.10186.

Jon M. Kleinberg. 1999. Hubs, authorities, and communities. *ACM Comput. Surv.*, 31:5.

Goro Kobayashi, Tatsuki Kuribayashi, Sho Yokoi, and Kentaro Inui. 2020. Attention module is not only a weight: Analyzing transformers with vector norms. *CoRR*, abs/2004.10102.

Zornitsa Kozareva, Ellen Riloff, and Eduard H. Hovy. 2008. Semantic class learning from the web with hyponym pattern linkage graphs. In *ACL*.

Dekang Lin and Patrick Pantel. 2001. Dirt @sbt@discovery of inference rules from text. In *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '01, page 323–328, New York, NY, USA. Association for Computing Machinery.

Christopher D. Manning. 2015. Computational linguistics and deep learning. *Computational Linguistics*, 41:701–707.

R. Thomas McCoy, Ellie Pavlick, and Tal Linzen. 2019. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. *CoRR*, abs/1902.01007.

Mike D. Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *ACL*.

Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. 1999. The pagerank citation ranking: Bringing order to the web. Technical Report 1999-66, Stanford InfoLab. Previous number = SIDL-WP-1999-0120.

Ellen Riloff. 1993. Automatically constructing a dictionary for information extraction tasks. In *AAAI*.

Ellen Riloff. 1996. Automatically generating extraction patterns from untagged text. In *AAAI/IAAI, Vol. 2*.

Ellen Riloff and R. Jones. 1999. Learning dictionaries for information extraction by multi-level bootstrapping. In *AAAI/IAAI*.

Ellen Riloff and Janyce Wiebe. 2003. Learning extraction patterns for subjective expressions. In *EMNLP*.

O. Mahamane Sani Sabo, Yanai Elazar, Yoav Goldberg, and Ido Dagan. 2021. Revisiting few-shot relation classification: Evaluation data and classification schemes. *Transactions of the Association for Computational Linguistics*, 9:691–706.

D. Sculley, Gary Holt, Daniel Golovin, Eugene Davydov, Todd Phillips, Dietmar Ebner, Vinay Chaudhary, Michael Young, Jean-François Crespo, and Dan Dennison. 2015. Hidden technical debt in machine learning systems. In *NIPS*.

Zheng Tang and Mihai Surdeanu. 2023. It takes two flints to make a fire: Multitask learning of neural relation and explanation classifiers. *Computational Linguistics*. Accepted on 2022.

Marco A. Valenzuela-Escárcega, Gus Hahn-Powell, and Dane Bell. 2020. Odinson: A fast rule-based information extraction framework. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 2183–2191, Marseille, France. European Language Resources Association.

Sarah Wiegreffe and Yuval Pinter. 2019. Attention is not not explanation. *CoRR*, abs/1908.04626.

Yuhao Zhang, Victor Zhong, Danqi Chen, Gabor Angeli, and Christopher 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)*, pages 35–45.