Weakly-supervised Relation Extraction by Pattern-enhanced Embedding Learning

Meng Qu[1], Xiang Ren[2], Yu Zhang[1], Jiawei Han[1]

[1]University of Illinois at Urbana-Champaign, IL, USA
[2]University of Southern California, CA, USA
[1]{mengqu2, yuz9, hanj}@illinois.edu [2]xiangren@usc.edu

ABSTRACT

Extracting relations from text corpora is an important task with wide applications. However, it becomes particularly challenging when focusing on weakly-supervised relation extraction, that is, utilizing a few relation instances (i.e., a pair of entities and their relation) as seeds to extract from corpora more instances of the same relation. Existing distributional approaches leverage the corpus-level co-occurrence statistics of entities to predict their relations, and require a large number of labeled instances to learn effective relation classifiers. Alternatively, pattern-based approaches perform boostrapping or apply neural networks to model the local contexts, but still rely on a large number of labeled instances to build reliable models. In this paper, we study the integration of distributional and pattern-based methods in a weakly-supervised setting such that the two kinds of methods can provide complementary supervision for each other to build an effective, unified model. We propose a novel co-training framework with a distributional module and a pattern module. During training, the distributional module helps the pattern module *discriminate* between the informative patterns and other patterns, and the pattern module *generates* some highly-confident instances to improve the distributional module. The whole framework can be effectively optimized by iterating between improving the pattern module and updating the distributional module. We conduct experiments on two tasks: knowledge base completion with text corpora and corpus-level relation extraction. Experimental results prove the effectiveness of our framework over many competitive baselines.

ACM Reference Format:
Meng Qu[1], Xiang Ren[2], Yu Zhang[1], Jiawei Han[1]. 2018. Weakly-supervised Relation Extraction by Pattern-enhanced Embedding Learning. In *WWW 2018: The 2018 Web Conference, April 23–27, 2018, Lyon, France.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3178876.3186024

1 INTRODUCTION

Relation extraction is an important task in data mining and natural language processing. Given a text corpus, relation extraction aims at extracting a set of relation instances (i.e., a pair of entities and their relation) based on some given examples. Many efforts [7, 22, 26] have been done on sentence-level relation extraction, where the goal is to predict the relation for a pair of entities mentioned in a sentence (e.g., predict the relation between "*Beijing*" and "*China*" in sentence 1 of Fig. 1). Despite its wide applications, these studies usually require a large number of human-annotated sentences as training data, which are expensive to obtain. In many cases (e.g., knowledge base completion [39]), it is also desirable to extract a set of relation instances by consolidating evidences from multiple sentences in corpora, which cannot be directly achieved by these studies. Instead of looking at individual sentences, corpus-level relation extraction [2, 12, 21, 27, 43] identifies relation instances from text corpora using evidences from multiple sentences. This also makes it possible to apply weakly-supervised methods based on corpus-level statistics [1, 8]. Such weakly-supervised approaches usually take a few relation instances as seeds, and extract more instances by consolidating redundant information collected from large corpora. The extracted instances can serve as extra knowledge in various downstream applications, including knowledge base completion [27, 34], corpus-level relation extraction [16, 43], hypernym discovery [30, 31] and synonym discovery [25, 36].

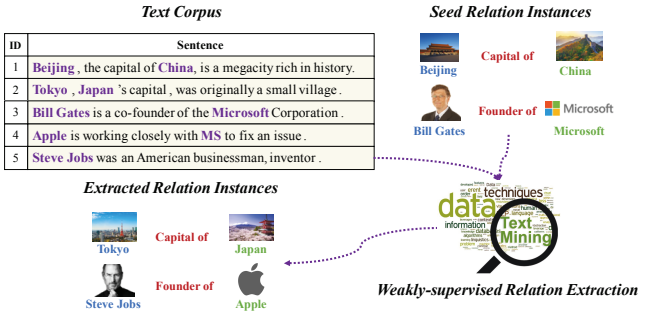


Figure 1: Illustration of weakly-supervised relation extraction. Given a text corpus and a few relation instances as seeds, the goal is to extract more instances from the corpus.

In this paper, we focus on corpus-level relation extraction in the *weakly-supervised setting*. There are broadly two types of weakly-supervised approaches for corpus-level relation extraction. Among them, pattern-based approaches predict the relation of an entity pair from multiple sentences mentioning both entities. To do that, traditional approaches [23, 28, 41] extract textual patterns (e.g., tokens between a pair of entities) and new relation instances in a bootstrapping manner. However, many relations could be expressed in a variety of ways. Due to such diversity, these approaches often have difficulty matching the learned patterns to unseen contexts, leading to the problem of semantic drift [8] and inferior performance. For example, with the given instance "*(Beijing, Capital of, China)*" in Fig. 1, "*[Head] , the capital of [Tail]*" will be extracted as a textual pattern from sentence 1. But we have difficulty in matching the

WWW 2018, April 23–27, 2018, Lyon, France

ACM ISBN 978-1-4503-5639-8/18/04.
https://doi.org/10.1145/3178876.3186024

pattern to sentence 2 even though both sentences refer to the same relation "*Capital of*". Recent approaches [17, 40] try to overcome the sparsity issue of textual patterns by encoding textual patterns with neural networks, so that pattern matching can be replaced by similarity measurement between vector representations. However, these approaches typically rely on large amount of labeled instances to train effective models [30], making it hard to deal with the weakly-supervised setting.

Alternatively, distributional approaches resort to the corpus-level co-occurrence statistics of entities. The basic idea is to learn low-dimensional representations of entities to preserve such statistics, so that entities with similar semantic meanings tend to have similar representations. With entity representations, a relation classifier can be learned using the labeled relation instances, which takes entity representations as features and predicts the relation of a pair of entities. To learn entity representations, some approaches [19, 24, 33] only consider the given text corpus. Despite the unsupervised property, their performance is usually limited due to the lack of supervision [39]. To learn more effective representations for relation extraction, some other approaches [37, 39] jointly learn entity representations and relation classifiers using the labeled instances. However, similar to pattern-based approaches, distributional approaches also require considerable amount of relation instances to achieve good performance [39], which are usually hard to obtain in the weakly-supervised setting.

The pattern-based and the distributional approaches extract relations from different perspectives, which are naturally complementary to each other. Ideally, we would wish to integrate both approaches, so that they can mutually enhance and reduce the reliance on the given relation instances. Towards integrating both approaches, several existing studies [25, 30, 34] try to jointly train a distributional model and a pattern model using the labeled instances. However, the supervision of their frameworks still totally comes from the given relation instances, which is insufficient in the weakly-supervised setting. Therefore, their performance is yet far from satisfaction, and we are seeking an approach that is more robust to the scarcity of seed instances.

In this paper, we propose such an approach called REPEL (Relation Extraction with Pattern-enhanced Embedding Learning) to weakly-supervised relation extraction. Our approach consists of a pattern module and a distributional module (see Fig. 2). The pattern module aims at learning a set of reliable textual patterns for relation extraction; while the distributional module tries to learn a relation classifier on entity representations for prediction. Different from existing studies, we follow the co-training [3] strategy and encourage both modules to provide extra supervision for each other, which is expected to complement the limited supervision from the given seed instances (see Fig. 3). Specifically, the pattern module acts as a *generator*, as it can extract some candidate instances based on the discovered reliable patterns; whereas the distributional module is treated as a *discriminator* to evaluate the quality of each generated instance, that is, whether an instance is reasonable. To encourage the collaboration of both modules, we formulate a joint optimization process, in which we iterate between two sub-processes. In the first sub-process, the discriminator (distributional module) will evaluate the instances generated by the generator (pattern module), and the results serve as extra signals to adjust the generator. In the second sub-process, the generator (pattern module) will in turn

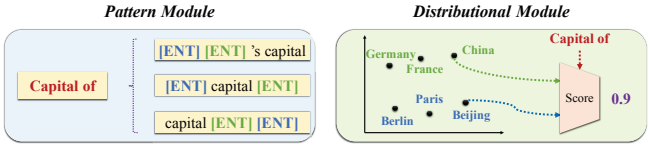


Figure 2: Illustration of the modules. The pattern module aims to learn reliable textual patterns for each relation. The distributional module tries to learn entity representations and a score function to estimate the quality of each instance.

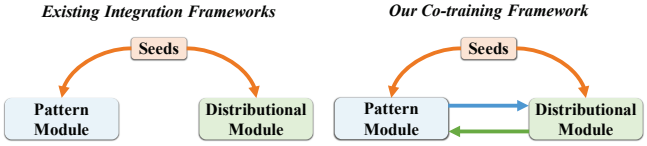


Figure 3: Comparison with existing integration frameworks. Existing frameworks totally rely on the seed instances to provide supervision. Our framework encourages both modules to provide extra supervision for each other.

generate a set of highly confident instances, which serve as extra training seeds to improve the discriminator (distributional module). During training, we keep iterating between the two sub-processes, so that both modules can be consistently improved. Once the training converges, both modules can be applied to relation extraction, which extract new relation instances from different perspectives.

In summary, in this paper we make the following contributions:

- We propose a principled framework to integrate the distributional and pattern-based methods for weakly-supervised relation extraction, which is effective in overcoming the scarcity of seeds.
- We develop a joint optimization algorithm for solving the unified objective, alternating between adjusting the pattern module and improving the distributional module.
- We conduct experiments on two downstream applications over two real-world datasets. Experimental results prove the effectiveness of our framework in the weakly-supervised setting.

2 PROBLEM DEFINITION

In this section, we formally define our problem.

Entity Name. An *entity name* is a string referring to a real-world entity, which usually appears in multiple sentences of a corpus. For example in Fig. 1, all strings with purple colors (e.g., Beijing, Bill Gates) are valid entity names.

To extract relations between different entities, a prerequisite is to detect those entity names in text corpora. In this paper, for simplicity, we will not focus on entity name detection. Instead, we will use existing tools to do that. Specifically, we first apply some named entity recognition tools [18] to the corpus, which are able to detect entity names in text. In practice, many detected entity names can refer to the same entity. For example in Fig. 1, "*Microsoft*" in sentence 3 and "*MS*" in sentence 4 both refer to *Microsoft Corporation*. For entity names representing the same entity, since they have exactly the same meaning, we may expect to treat them equally instead of treating them independently. Therefore, we further leverage some entity linking tools [9], which can link synonymous entity names to the same entity in an external knowledge (e.g., Freebase). After entity linking, for each entity, we use a unified id to replace all

entity names referring to that entity. For example, we can use the Freebase id of the entity *Microsoft Corporation* to replace "*Microsoft*" and "*MS*" in Fig. 1.

Relation Instance. A *relation instance* describes the relation between a pair of entities. Formally, a relation instance is composed of an entity pair (e_h, e_t) and a relation r, meaning that entity e_h and entity e_t have the relation r.

Relation instances are ubiquitous. For example in Fig. 1 *(Beijing, China)* with *capital of*, *(Bill Gates, Microsoft)* with *founder of* are both valid relation instances. Extracting such instances from text corpora is an essential task, which has wide applications.

Problem Definition. In this paper, we study weakly-supervised relation extraction. Specifically, given a text corpus D and some target relations R, with each target relation r specified by a set of relation instances $\{(e_{h_k}, e_{t_k}, r)\}_{k=1}^{N_r}$, our goal is to leverage the given instances as seeds and extract more instances from the corpus (Fig. 1). Formally, we define our problem as follows:

Definition 2.1. **(Problem Definition)** ***Given*** *a text corpus D and some target relations R, where each target relation r is characterized by a few seed instances $\{(e_{h_k}, e_{t_k}, r)\}_{k=1}^{N_r}$ or in other words a few seed entity pairs $\{(e_{h_k}, e_{t_k}\}_{k=1}^{N_r}$, the weakly-supervised relation extraction task* ***aims to*** *extract more instances $\{(e_{h_i}, e_{t_i}, r_i)\}_{i=1}^{M}$ from the corpus. In other words, we aim at discovering more entity pairs $\{(e_{h_i}, e_{t_i})\}_{i=1}^{M_r}$ under each target relation $r \in R$.*

3 THE REPEL FRAMEWORK

3.1 Framework Overview

In this section, we introduce our approach to weakly-supervised relation extraction. The major challenge comes from the deficiency of supervision, since we only have a few relation instances as seeds. Therefore, the performance of existing approaches, including the pattern-based [1, 16, 44] and the distributional approaches [4, 20, 39], is not satisfactory. Although some studies [25, 30, 34] trying to reduce the reliance on seeds by integrating both approaches, they simply employ a joint training framework, which still requires considerable relation instances to train effective models.

To better overcome the challenge of seed scarcity, in this paper we propose a framework called REPEL based on the co-training strategy [3]. Our framework consists of two modules, a pattern module and a distributional module (see Fig. 2), which extract relations from different perspectives. The pattern module aims at finding a set of reliable textual patterns for relation extraction. Meanwhile, the distributional module tries to learn entity representations and train a score function, which measures the quality of a relation instance. Different from existing studies, both modules are encouraged to provide extra supervision to each other, which is expected to complement the limited supervision from seed instances (see Fig. 3). Specifically, the pattern module is treated as a generator since it can extract some candidate relation instances, and meanwhile the distributional module acts as a discriminator to evaluate each instance. During training, the discriminator evaluates the instances generated by the generator, and the results serve as extra signals to adjust the generator. On the other hand, the generator will in turn generate some highly confident instances, which act as extra seeds to improve the discriminator. We keep iterating between adjusting

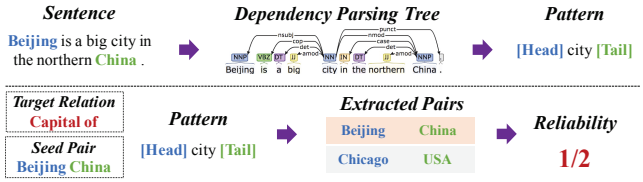


Figure 4: Illustration of the pattern module. We extract patterns by considering the dependency parsing trees. The pattern reliability is calculated based on the seed entity pairs.

the pattern module and improving the distributional module. Once the training process converges, both modules can be utilized to discover more instances.

The overall objective is summarized below:

$$\max_{P,D} O = \max_{P,D}\{O_p + O_d + \lambda O_i\}. \tag{1}$$

In the objective, P represents the parameters of the pattern module, that is, a given number of reliable patterns for each target relation. D denotes the parameters of the distributional module, that is, entity representations and a score function. The objective function consists of three terms. Among them, O_p is the objective of the pattern module, in which we leverage the given seed instances for pattern selection. O_d is the objective of the distributional module, which learns relevant parameters under the guidance of seed instances. Finally, O_i models the interactions of both modules.

Next, we introduce the model details. **Note that** for simplicity, we only consider one relation when introducing the model. To deal with multiple relations, we can simply combine their objectives.

3.2 Pattern Module

In the pattern module, our goal is to select a given number of the most reliable patterns P for the target relation, and further leverage them to discover more relation instances from corpora.

Following previous studies on pattern based approaches [5, 23, 40], we define a pattern as the tokens in the shortest dependency path between a pair of entities in a sentence. Fig. 4 presents an example, in which the pattern of the sentence is *[Head] city [Tail]*. Given this definition, we can go back to the corpus and extract a pattern for every pair of entities in a sentence, forming a set of candidate patterns and many entity pairs linked to each pattern.

Among all the candidate patterns, we hope to extract the most reliable ones for the target relation. Towards this goal, we leverage the seed relation instances as guidance, and estimate the reliability of a pattern π with the following measurement $R(\pi)$:

$$R(\pi) = \frac{|G(\pi) \cap S_{pair}|}{|G(\pi)|}, \tag{2}$$

where $G(\pi)$ represents all the entity pairs extracted by the pattern π, and S_{pair} is the set of seed entity pairs under the target relation. The numerator of $R(\pi)$ is the number of seed entity pairs which can be discovered by the pattern π, and the denominator counts all extracted entity pairs. For example in the right part of Fig. 4, we focus on the relation *capital of*, and the pattern *[Head] city [Tail]* extracts two entity pairs. Among them, the red pair is in the seed set, and therefore the reliability is 1/2. Such definition of $R(\pi)$ is quite intuitive. Basically, if a pattern can extract many seed entity pairs under the target relation, then it will be considered reliable.

Based on the measurement, we try to select the top-K most reliable patterns, in which K is a given number. Such goal can be achieved by optimizing the following objective function with respect to P:

$$O_p = \sum_{\pi \in P} R(\pi), \tag{3}$$

where P is the pattern set with size K.

Once the most reliable patterns P are learned for the target relation, we can leverage them to extract new entity pairs under the target relation. Formally, we denote the set of entity pairs extracted by the pattern set P as $G(P)$, which is calculated as follows:

$$G(P) = \cup_{\pi \in P} G(\pi), \tag{4}$$

where $G(\pi)$ is the set of entity pairs extracted by pattern π.

3.3 Distributional Module

The distributional module of our approach focuses on the global distributional information of entities. Specifically, it aims at learning distributed entity representations from corpora, so that similar entities are likely to have similar representations. Meanwhile, we utilize the given relation instances as seeds to train a score function, which takes entity representations as features to estimate whether a relation instance is reasonable.

To learn entity representations from text corpora, we follow [32] and build a bipartite network between all the entities and words. The weight between an entity and a word is defined as the number of sentences in which they co-occur. Then for an entity e and a word w, we infer the conditional probability $P(w|e)$ as follows:

$$P(w|e) = \frac{\exp(\mathbf{x}_e \cdot \mathbf{c}_w)}{Z}, \tag{5}$$

where $\mathbf{x}_e$ is the vector representation of entity e, $\mathbf{c}_w$ is the embedding vector of word w and Z is a normalization term.

Given the estimated conditional probability $p(\cdot|e)$, we try to minimize its KL divergence from the empirical distribution $p'(\cdot|e)$ for every entity e, so that the distributional information can be preserved into the learned entity representations. Specifically, the empirical distribution is defined as $p'(w|e) \propto n_{w,e}$, where $n_{w,e}$ is the weight of the edge between word w and entity e. After some simplification, we obtain the following objective function:

$$O_{text} = \sum_{w,e} n_{w,e} \log P(w|e), \tag{6}$$

The above objective function can be efficiently optimized with the negative sampling [20] and edge sampling [33] techniques. In each epoch, a positive edge and several negative edges are sampled for optimization. For details, readers may refer to [32, 33].

Meanwhile, we also leverage the given seed instances to learn a score function, which estimates the quality of a instance, that is, how likely an entity pair has the target relation. Following the previous work [4], for an entity pair $f = (e_h, e_t)$, its score under the target relation is defined as follows:

$$L_D(f|r) = -||x_{e_h} + y_r - x_{e_t}||_2^2, \tag{7}$$

where $||\cdot||_2$ is the Euclidean norm of a vector, x_e is the representation of entity e, r is the target relation and y_r is a parameter vector for the target relation.

Intuitively, we expect a seed entity pair could have larger scores than some randomly sampled pairs under the target relation. Therefore, we adopt the following ranking based objective for training:

$$O_{seed} = \sum_{f \in S_{pair}} \sum_{f' = (e'_h, e'_t)} \min\{1, L_D(f|r) - L_D(f'|r)\}. \tag{8}$$

S_{pair} is all seed pairs, e'_h and e'_t are randomly sampled entities.

Finally, we integrate Eqn. 6 and Eqn. 8 as the objective of the distributional module, and we try to optimize it with respect to D.

$$O_d = O_{text} + \eta O_{seed}, \tag{9}$$

where η is used to control the weights of the two parts, D represents all parameters of the distributional module, including entity representations $\{x_e\}$ and the parameter vector y_r of the relation.

Once the representations are learned, we can use the score function L_D to measure the score of each entity pair under the target relation, and thus discover some highly confident relation instances.

3.4 Modeling the Module Interaction

So far, the supervision of both modules totally comes from the given relation instances, which is insufficient in the weakly-supervised setting. To solve this problem, we follow the co-training strategy [3], and encourage both modules to provide extra supervision for each other.

Specifically, we introduce the following objective function, and try to maximize it with respect to both of P and D:

$$O_i = E_{f \in G(P)}[L_D(f|r)], \tag{10}$$

where $f \in G(P)$ is an entity pair extracted by the reliable pattern set P with $G(P)$ defined in Eqn. 4, $L_D(f|r)$ is the score of pair f under the target relation. From the objective function, we see that the selected patterns P acts as a generator, since it generates some candidate entity pairs under the target relation; whereas the distributional module serves as a discriminator, trying to score the generated entity pairs under the target relation. The goal of the objective function is to encourage the agreement of the pattern module and the distributional module. More specifically, we hope that the entity pairs generated by the pattern module can be considered reasonable by the distributional module. The intuition underlying the objective comes from the co-training algorithms [3], where it has been proved that the error rate of two predictive models can be decreased by minimizing their disagreement [6].

To intuitively understand how this objective function will improve both modules, let us consider how to optimize with respect to both modules. For the pattern module, to maximize the above objective, the pattern set P should include patterns which are considered reliable by the distributional module. That is, the entity pairs generated by those patterns should obtain large scores from the distributional score function L_D. In this way, the distributional module provides extra supervision to estimate the pattern reliability. Meanwhile, for the distributional module, to maximize the objective function, it should assign larger scores to the entity pairs generated by the pattern module. Therefore, the highly confident entity pairs generated by the pattern module serve as extra seeds to help improve the distributional module.

With the above objective function, both modules can tightly interact with each other, and provide extra supervision to overcome the challenge of seed scarcity.

4 THE JOINT OPTIMIZATION PROBLEM

To optimize the overall objective function (Eqn. 1), we leverage the coordinate gradient descent algorithm [38], by iterating between two sub-processes. In the first sub-process, we fix the pattern module, and update the distributional module under the guidance of the given seeds and the highly confident instances generated by the pattern module. In the second sub-process, the distributional module is fixed, and we update the selected patterns with the given seed instances and the supervision provided by the distributional module. During training, we keep iterating between the two sub-processes, so that both modules can be consistently improved.

1. Optimizing the Distributional Module. In this step, we fix the selected pattern set P to update the parameters D of the distributional module. Formally, maximizing the objective function with respect to D can be transformed as the following problem:

$$\max_D\{O_d + \lambda O_i\} = \max_D\{O_d + \lambda E_{f\in G(P)}[L_D(f|r)]\}, \tag{11}$$

which is a continuous optimization problem. We use the stochastic gradient descent algorithm for optimization. On the one hand, we adjust all parameters D to maximize the O_d part. On the other hand, some entity pairs f will be sampled based on the selected patterns P, which are treated as extra instances to update D.

2. Optimizing the Pattern Module. In this this, we fix the parameters D of the distributional module and adjust the reliable pattern set P. Formally, maximizing the objective function with respect to P is equivalent to the following optimization problem:

$$\max_P\{O_p + \lambda O_i\} = \max_P\{\sum_{\pi\in P}\Big(R(\pi) + \lambda E_{f\in G(\pi)}[L_D(f|r)]\Big)\}, \tag{12}$$

which is a discrete optimization problem, with the goal as selecting a given number of patterns P with the largest reliability. The reliability of a pattern π is calculated from two sources: O_p and O_i. In the O_p part, the reliability is measured with $R(\pi)$ defined in Eqn. 2, which leverages the given seeds for reliability estimation. In the O_i part, we utilize the score function L_D to score each entity pair f extracted by pattern π, and further average them to obtain another reliability estimation $E_{f\in G(\pi)}[L_D(f|r)]$. Finally, the two estimations are weighted as the overall reliability. In practice, we can first calculate the overall reliability of each pattern, and then select the top-K patterns to form the reliable pattern set P.

Finally, we summarize the optimization algorithm into Alg. 1. Once the training converges, our approach will return a set of discovered reliable patterns from the pattern module and a distributional score function from the distributional module. Both the learned patterns and score function can be leveraged for relation extraction, which extract new instances from different perspectives. Specifically, the learned reliable patterns extract relations from local contexts by matching the contexts with the patterns, which usually have high precision but low recall. This is because for a pair of entities, the local contexts mentioning both entities are usually more reliable for predicting their relations, leading to high precision. However, for many pairs of entities, they may never co-occur in any local contexts, and thus using local contexts can result in low recall. In practice, the learned reliable patterns can be applied to applications such as corpus-level relation extraction (see the details in Sec. 5.1.3 (2)). On the other hand, the learned distributional score function predicts entity relation from corpus-level statistics, leading to relatively low precision but high recall, and is more suitable for tasks like knowledge base completion with text corpora (see the details in Sec. 5.1.3 (1)).

Algorithm 1 Optimization algorithm of REPEL.

Input: A text corpus, a few seed relation instances, the number of reliable patterns K, the parameter λ, the parameter η.
Output: A set of reliable patterns P from pattern module, a score function L_D from distributional module, extracted relation instances.

```
Generate patterns and entity pairs extracted by each pattern.
Build the bipartite network between entities and words.
while not converge do
    ⊡ Update the distributional module:
    Extract some instances by using the set of reliable patterns P.
    Optimize D with both the seeds and extracted instances (Eqn. 11).
    ⊡ Update the pattern module:
    Calculate pattern reliability with the seeds and L_D (Eqn. 12).
    Select the top-K most reliable patterns to form the pattern set P.
end while
⊡ Extract relation instances:
Utilize the reliable patterns P to extract instances from local contexts.
Utilize the distributional score function L_D to extract instances.
```

5 EXPERIMENT

In this section, we evaluate our approach on two downstream applications: knowledge base completion with text corpora (KBC) and corpus-level relation extraction (RE).

In knowledge base completion with text corpora, the key task is to predict the missing relationships between each pair of entities in knowledge bases. Since some pairs of entities may not co-occur in any sentences in the given corpus, the learned pattern module can not provide information for predicting their relations. Therefore, for KBC we only use the entity representations and score function learned by the distributional module for extraction, and we expect to show that the pattern module can provide extra seeds during training, yielding a more effective distributional module. For corpus-level RE, it aims at predicting the relation of a pair of entities from several sentences mentioning both of them. In this case, the reliable patterns learned by the pattern module can capture the local context information from the sentences. Therefore, we focus on utilizing the learned pattern module for prediction in RE, and we expect to show that the distributional module can enhance the pattern module by providing extra supervision to select reliable patterns.

5.1 Experiment Setup

1. Datasets. In experiment, we leverage existing NER tool [18] for entity detection. Since the NER tool can only detect entities of several major types such as location, person and organization, we thus sample 10 common relations [1] related to person, location and organization from Freebase [2] as our target relations. Then two datasets are constructed based on the selected relations.

(1) **Wiki**: The first 150K articles in Wikipedia [3] are used as the corpus. For each target relation, we randomly sample 50 relation

[1] *location.country.capital, people.person.parents, people.person.children, location.administrative division.country, people.person.place of birth, location.neighborhood.neighborhood of, people.person.nationality, people.deceased person.place of death, location.location.contains, organization.organization.founders.*
[2] https://developers.google.com/freebase/
[3] https://www.wikipedia.org/

Table 1: Statistics of the Datasets.

Dataset	Wiki + Freebase	NYT + Freebase
# Documents	150,000	118,664
# Entities	92,443	23,120
# Candidate Patterns	621,782	232,892
# Seed Instances per Relation	50	50
# Relations in KBC	10	10
# Test Instances in KBC	10,734	6,094
# Relations in RE	5	6
# Test Entity Pairs in RE	131	222

instances from Freebase as seeds. In the knowledge base completion task, we select all the above 10 relations as the target relations, and we sample 10,734 extra instances from Freebase for prediction. In the corpus-level relation extraction task, the manually annotated sentences from [10] are used for evaluation. Among all relations in the annotated sentences, 5 relations [4] can be mapped to our selected 10 Freebase relations, and thus we only focus on these 5 relations. There are totally 194 manually annotated sentences and 131 entity pairs related to the relations.

(2) **NYT**: The 118,664 documents from 2013 New York Times news articles. Similar to the Wiki dataset, for each target relation we randomly sample 50 relation instances from Freebase as seeds. In the knowledge base completion task, we select all the above 10 relations as the target relations, and totally 6,094 extra instances are sampled for evaluation. In the corpus-level relation extraction task, we leverage the manually annotated sentences from [11] for evaluation. Among all relations in the sentences, 6 relations [5] can be mapped to the selected 10 Freebase relations, so we focus on these 6 relations. There are totally 322 manually annotated sentences and 222 entity pairs related to the relations.

For each text corpus, we adopt Stanford CoreNLP package [18][6] to do preprocessing. Then we leverage DBpedia Spotlight [9][7] to link the detected entity names to the Freebase.

2. Compared Algorithms. In the knowledge base completion task, we select the following baseline algorithms to compare:
(1) **word2vec [20]**: A distributional approach for word embedding learning, which can learn entity representations from text corpora. Once the representations are learned, we utilize the seed instances to train a relation classifier (Eqn. 7) for extraction. (2) **TransE [4]**: A distributional approach for knowledge base completion, which only uses the given seed instances for training. (3) **RK [36]**: A distributional approach for knowledge base completion, which leverages both the text corpus and the given relation instances to learn entity representations. (4) **DPE [25]**: An approach that integrates the distributional and pattern-based methods. It jointly models the distributional information in text corpora, the given relation instances and the textual patterns. (5) **CONV [34]**: A knowledge base completion approach, which integrates the distributional and pattern-based methods by jointly optimizing the given seed instances and the instances extracted by textual patterns.

In the corpus-level relation extraction task, the following approaches are selected to compare:

(1) **SnowBall [1]**: A pattern approach for relation extraction, which discovers reliable patterns with the seed instances in a bootstrapping way. (2) **CNN-ATT [16]**: A pattern approach for corpus-level relation extraction. It leverages convolutional neural networks to encode and classify each sentence, and then consolidates the results of different sentences using an attention mechanism. (3) **PCNN-ATT [16]**: A pattern approach for corpus-level relation extraction. Compared with CNN-ATT, it also involves the position embedding for each word and entity. (4) **PathCNN [45]**: A pattern approach for corpus-level relation extraction. For each entity pair, besides sentences mentioning both entities, it also considers some other sentences mentioning only one of them. (5) **LexNET [29, 30]**: An approach combining the distributional and pattern-based methods for relation extraction. Formally, it uses a recurrent layer to encode local textual patterns, and then uses the encoding vector together with entity representations for prediction.

For our proposed approach, we consider the following variants: (1) **REPEL-P**: A variant of our approach with only the pattern module (O_p). (2) **REPEL-D**: A variant of our approach with only the distributional module (O_d). (3) **REPEL**: Our proposed approach, which encourages the collaboration of both modules during training. Once the training converges, we leverage the entity representations and score function learned by the distributional module for the KBC task; whereas the reliable patterns discovered by the pattern module are used for the RE task.

3. Evaluation Setup. (1) **Knowledge Base Completion:** For each compared algorithm, we first learn entity representations and relation classifiers (or score function for our approach) by using the given text corpus and relation instances. Then the learned representations and classifiers are leveraged for evaluation. Specifically, for each test instance (e_h, e_t, r), we remove its head entity or tail entity, obtaining two incomplete instances, including $(e_h, ?, r)$ and $(?, e_t, r)$, and our goal is to select the correct entity from the entity set to fill the incomplete instances. To do that, for each candidate entity in the entity set, we calculate its score by measuring the quality of the formed instance using the relation classifier. Then we sort different entities in the descending order based on their scores, and calculate the rank of the correct entity. Finally, we report the mean value of those ranks (i.e., MR) and also the proportion of the correct entities ranked within top 10 (i.e., Hits@10).
(2) **Corpus-level Relation Extraction:** For each compared algorithm, we first use it to predict the relation expressed in each test sentence. Specifically, for neural network based approaches (PathCNN, CNN-ATT, PCNN-ATT, LexNET), the test sentences can be directly classified based on the learned neural classifiers. For approaches based on textual patterns (Snowball, REPEL, REPEL-P), we first match the local context of the test sentence to a discovered reliable pattern π^*, then we classify the sentence based on the relation expressed by pattern π^*. To do such matching, we represent each learned reliable pattern and the local patterns of the test sentences with a low-dimensional vector. The pattern vector is calculated by averaging the embeddings of tokens in each pattern, with the token embeddings learned by our approach in Eqn. 5. Once the pattern vectors are obtained, each local pattern in test sentences is matched to its most similar reliable pattern, in which the similarity is measured as the cosine similarity between the pattern vectors. After

[4] *people.person.children,people.person.place of birth,people.person.nationality,people.deceased person.place of death,organization.organization.founders.*
[5] *people.person.children,people.person.nationality,location.location.contains, people.deceased person.place of death,organization.organization.founders, location.administrative division.country.*
[6] http://stanfordnlp.github.io/CoreNLP/
[7] https://github.com/dbpedia-spotlight/dbpedia-spotlight

all test sentences are classified, for each test entity pair, we consolidate the prediction results from the test sentences mentioning both entities, and return the predicted relation together with the confidence score. During consolidate, we either average the prediction results of all test sentences (LexNET, PathCNN, Snowball, REPEL, REPEL-P), or leverage the learned attention mechanism (CNN-ATT, PCNN-ATT). Finally, we sort all test entity pairs in the descending order based on the calculated confidence scores, and compare the ranked list with the ground-truth. Based on the results, we report both the precision at position K (i.e., P@K), recall at position K (i.e., R@K) and the precision-recall curve.

4. Parameter Settings. For all knowledge base completion methods and the distributional module of our approach, we set the dimension of all representations as 100. The number of iterations for TransE, word2vec, RK, DPE are set as 1000, 20, 20, 3B respectively to ensure the convergence. Other parameters are set as the default values suggested in the original papers. For the neural based approaches to corpus-level relation extraction, the dimension of the embedding layer and the hidden layer is set as 100. Other parameters are set as the default values suggested in the original papers. For our proposed approach, the parameter λ for controlling the weight of the interaction term is set as 1 by default. For the distributional module, the learning rate is set as 0.01, the parameter η is set as 0.005, the number of training edges in each iteration is set as 3B. For the pattern module, we set the number of reliable patterns K for each relation as 100.

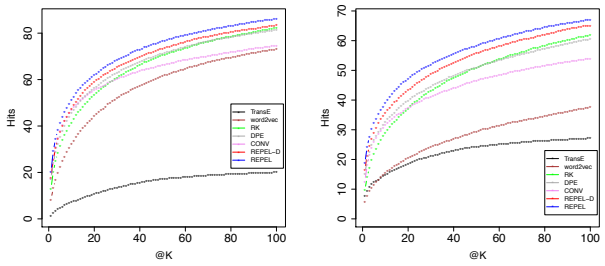
5.2 Performance Comparison

1. Knowledge Base Completion with Text Corpora (KBC). We present the quantitative results in Table 2, and the hits curve in Fig. 5. For the approach only considering the given seed instances (TransE), we see the performance is very limited due to the scarcity of seeds. Along the other line, the approach considering text corpora (word2vec) achieves relatively better results, but are still far from satisfactory, since it ignores the supervision from the seed instances. If we consider both the text corpus and seed instances for entity representation learning (RK), we obtain much better results. Moreover, by further jointly training a pattern model (DPE, CONV), the hits ratio can be further significantly improved.

Table 2: Quantitative results on the KBC task.

Algorithm	Wiki + Freebase		NYT + Freebase	
	Hits@10	MR	Hits@10	MR
TransE [4]	7.13	4328.40	15.94	3833.47
word2vec [20]	32.12	203.53	15.56	913.04
RK [36]	41.49	72.87	29.01	307.89
DPE [25]	45.45	78.87	32.47	279.99
CONV [34]	46.84	139.81	31.51	903.38
REPEL-D	47.49	67.28	35.79	234.23
REPEL	**51.18**	**62.18**	**38.98**	**199.44**

For our proposed approach, with only the distributional module (REPEL-D), it already outperforms all the baseline approaches. Compared with DPE, the performance gain of REPEL-D mainly comes from the usage of the score function in Eqn. 7, which can better model different relations. Compared with CONV, REPEL-D achieves better results, as the distributional information in text corpora can be better captured with Eqn. 6. Moreover, by encouraging



(a) Wiki (b) NYT

Figure 5: Hits curves on the KBC task.

the collaboration of both modules (REPEL), the results are further significantly improved. This observation demonstrates that the pattern module can indeed help improve the distributional module by providing some highly confident instances.

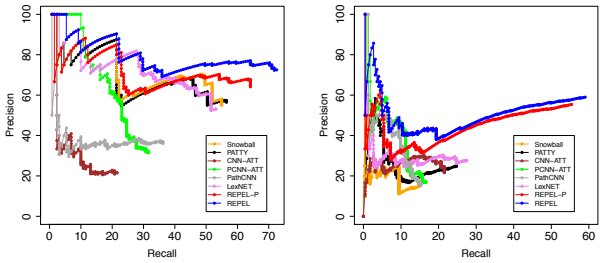
Overall, our approach achieves quite impressive results on the knowledge base completion task compared with several strong baseline approaches. Also, the pattern module can indeed enhance the distributional module with our co-training framework.

2. Corpus-level Relation Extraction (RE). Next, we show the results on the corpus-level relation extraction task. We present the quantitative results in Table 3 and the precision-recall in Fig. 6. For the approaches using textual patterns (Snowball), we see the results are quite limited especially on the NYT dataset. This is because it discovers informative patterns in a bootstrapping way, which can lead to the semantic drift problem [8] and thus harm the performance. For other neural network based pattern approaches (PathCNN, CNN-ATT, PCNN-ATT), although they are proved to be very effective when the given instances are abundant, their performance in the weakly-supervised setting is not satisfactory. The reason is that they typically deploy complicated convolutional layers or recurrent layers in their model, which rely on massive relation instances to tune. However, in our setting, the instances are very limited, leading to their poor performance. For the integration approach (LexNET), although it incorporates the distributional information, the performance is still quite limited especially on the NYT dataset. This is because the joint training framework of LexNET also requires considerable training instances.

Table 3: Quantitative results on the RE task.

Algorithm	Wiki + Freebase				NYT + Freebase			
	P@50	R@50	P@100	R@100	P@50	R@50	P@100	R@100
Snowball [1]	58.00	22.14	65.00	49.62	20.00	4.50	21.00	9.46
CNN-ATT [16]	26.00	9.92	22.00	16.79	24.00	5.41	29.00	13.06
PCNN-ATT [16]	58.00	22.14	36.00	27.48	46.00	10.36	26.00	11.71
PathCNN [45]	36.00	13.74	38.00	29.01	42.00	9.46	26.00	11.71
LexNET [29, 30]	74.00	28.24	61.00	46.56	32.00	7.21	26.00	11.71
REPEL-D	14.00	5.34	17.00	12.98	6.00	1.35	7.00	3.15
REPEL-P	64.00	24.43	70.00	53.44	32.00	7.21	33.00	14.86
REPEL	**78.00**	**29.77**	**76.00**	**58.02**	**48.00**	**10.81**	**43.00**	**19.37**

For our proposed approach, the performance of the distributional module (REPEL-D) is very bad. This is because each test entity is mentioned in only few test sentences, and thus the learned entity representations are not so effective due to the sparsity of the distributional information. On the other hand, the pattern module (REPEL-P) of our approach achieves surprisingly good results, which are comparable to the neural models. This is because we



(a) Wiki (b) NYT

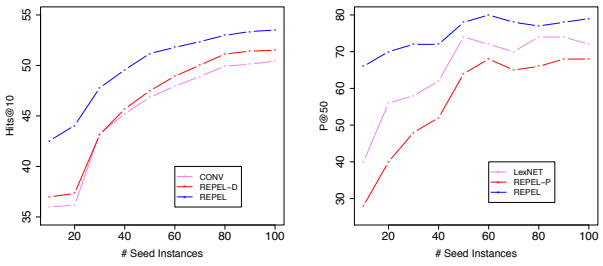
Figure 6: Precision-Recall curves on the RE task.

represent each pattern using the average embedding of tokens in the pattern for pattern matching, where the token embedding is learned from the given text corpus. Although such strategy is very naive compared with the neural encoding methods, it does not involve any extra parameters to learn. In the weakly-supervised setting, the neural methods are usually hard to train due to the large number of parameters, leading to inferior results. Whereas our approach achieves impressive results because of its simplicity. Furthermore, comparing the pattern module (REPEL-P) with the complete framework (REPEL), we see that the complete framework further outperforms the pattern module, which demonstrates that the distributional module can also enhance the pattern module by helping estimate pattern reliability.

Overall, in the weakly-supervised setting, our approach is able to achieve comparable results compared with the neural methods. Besides, the distributional module can indeed improve the pattern module with our co-training framework.

5.3 Performance Analysis

1. Performance w.r.t. the Number of Seed Instances. To overcome the challenge of seed scarcity, our approach encourages both modules to provide extra supervision for each other. In this section, we thoroughly study whether our framework is indeed robust to the scarcity of seed instances. We take the Wiki dataset as an example, and report the performance of different methods under differ number of seed instances.

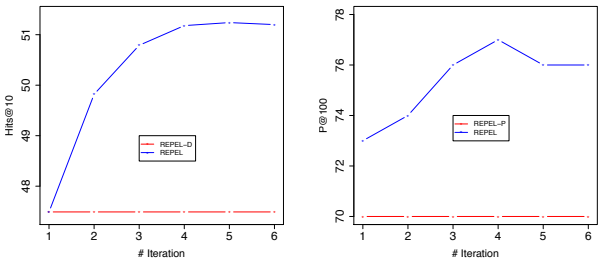


(a) KBC (b) RE

Figure 7: Performance w.r.t. # relation instances. Our approach consistently outperforms the compared algorithms especially when the given seeds are very limited.

Fig. 7 presents the results on the KBC and RE tasks. We see that our approach (REPEL) consistently outperforms other approaches (CONV, LexNET) integrating both the distributional and pattern-based methods. Besides, our approach (REPEL) also achieves better results than its variants (REPEL-P, REPEL-D), which deploy only one module. Moreover, we observe that when the given seed instances are quite sufficient, the results of different approaches are pretty close. Whereas under very limited seed instances, our approach (REPEL) significantly outperforms its variants (REPEL-P, REPEL-D) and the baseline approaches (CONV, LexNET). Based on the observation, we see that with the co-training framework, our approach is more robust to seed scarcity compared with existing integration approaches (CONV, LexNET).

2. Convergences Analysis. In our approach, we leverage the coordinate gradient descent algorithm for optimization, alternating between updating the distributional module and improving the pattern module. Next, we examine the optimization algorithm and study whether it converges during training. We take the Wiki dataset as an example, and present the performance of our approach at each iteration.

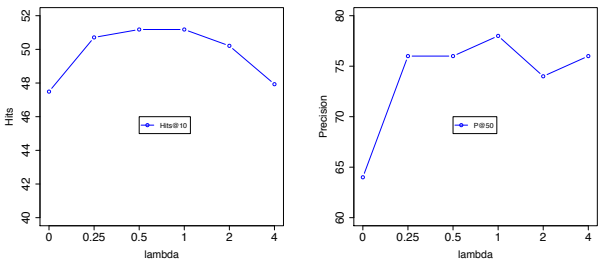


(a) KBC (b) RE

Figure 8: Convergence curves of our approach. Our approach quickly converges after several iterations.

Fig. 8 presents the results. In both tasks, the performance of our approach is consistently improved at the first several iterations, which shows that both modules can keep improving each other in our framework. Besides, we see that our approach quickly converges after several (3~4) iterations, which demonstrates the efficiency of the optimization algorithm.

3. Performance w.r.t. λ. In our framework, the parameter λ controls the weight of the interaction term O_i (Eqn. 10). A large λ encourages strong interactions of both modules, whereas a small λ corresponds to weak interactions. In this part, we study the performance of our approach under different λ. We take the Wiki dataset as an example, and report the results on both tasks.



(a) KBC (b) RE

Figure 9: Performance w.r.t. λ. Encouraging the interaction of the both modules ($\lambda > 0$) improves the performance.

Fig. 9 presents the results. When λ is set as 0, meaning that there is no interaction between the modules, we see that the results are quite limited. Then the results are quickly improved as we gradually increase λ, which further remain stable in the range (0.5, 1). If we further increase λ, the results begin to drop in the knowledge base completion task, as a large λ emphasizes too much on the supervision provided by the modules, and thus ignores the supervision from the given seed instances.

4. Case Study. In our co-training framework, both modules will collaborate with each other to overcome the seed scarcity problem. Specifically, the distributional module provides extra signals to select reliable patterns, whereas the pattern module discovers some highly confident instances to improve the distributional module. Next, we show some case study results to intuitively illustrate that both modules can indeed mutually enhance each other.

Table 4: The most reliable patterns discovered by our approach. Blue patterns are incorrect ones by human.

Relation: people.person.place-of-birth	
REPEL-P	**REPEL**
[Tail] [Head] birthplace	[Head] born place [Tail]
[Head] born place [Tail]	[Head] born city [Tail]
[Head] father move [Tail]	[Tail] [Head] birthplace
[Head] born city [Tail]	[Head] born June [Tail]
[Head] born January [Tail]	[Head] live return [Tail]
Relation: people.person.parents	
REPEL-P	**REPEL**
[Tail] die succeed son [Head]	[Tail] die succeed son [Head]
Babur [Tail] [Head]	[Head] daughter [Tail]
[Tail] give boy [Head]	[Head] son [Tail]
[Head] son [Tail]	descendant [Tail] [Head]
have relationship [Head] [Tail]	[Tail] marry have son [Head]

We first present the most reliable path-based patterns (i.e., tokens along the shortest dependency path between two entities) discovered by our approach and its variant on the Wiki dataset in Table 4. Blue patterns are unreliable ones based on the human. Comparing our approach with its variant (REPEL-P), we see that by considering the supervision signals from the distributional module (REPEL), some unreliable patterns can be filtered out from the pattern list, and the patterns discovered by our approach (REPEL) are more reliable. Therefore, the distributional module can indeed help the pattern module for reliable pattern selection.

Table 5: Top ranked instances extracted by the reliable patterns. Blue instances are incorrect ones by human.

Relation: people.person.nationality
(Charles IV of France, France) (Benjamin Franklin, USA) (Adolf Hitler, German) (Thomas Jefferson, USA) (Pol Pot, Thailand)
Relation: location.country.capital
(Denmark, Copenhagen) (Vietnam, Ho Chi Minh City) (Norway, Oslo) (Yugoslavia, Belgrade) (Guinea, Conakry)

Meanwhile, we also randomly sample some instances extracted by the discovered reliable patterns, and we show them in Table 5, where the blue instances are the incorrect ones by human. From the results, we see that most instances extracted by the reliable patterns are correct and reasonable. Therefore, the pattern module can in turn benefit the distributional module by providing some reasonable relation instances.

6 RELATED WORK

Our work is related to pattern-based approaches for relation extraction. Given two entities, the pattern-based approaches predict their relation from sentences mentioning both entities. Traditional approaches [1, 14, 23, 28, 41] try to find some informative textual patterns using the given instances, and utilize the patterns for extraction. However, these approaches ignore the semantic correlations of patterns, and thus suffer from semantic drift [8]. Recent approaches [16, 17, 30, 34, 40, 45] address the problem by encoding textual patterns with neural networks. Despite their success, these approaches rely on considerable labeled instances to train effective models, which suffer from the seed scarcity problem in the weakly-supervised setting. Our approach solves the problem by letting the distributional module provide extra supervision.

Our work is also related to the distributional approaches. Typically, these approaches learn entity representations from corpus-level statistics, and meanwhile a relation classifier is trained with the relation instances, which takes entity representations as features for relation prediction. Some approaches learn entity representations from only text corpora [20, 24, 33]. However, their performance is usually limited due to the lack of supervision. Some other approaches [4, 13, 15, 36, 37, 39, 42] learn more predictive entity representations by using the given relation instances as supervision, achieving superior results. However, they also require abundant relation instances to learn effective relation classifiers, which are hard to obtain in the weakly-supervised setting. Our approach alleviates the problem by letting the pattern module generate some highly confident instances as extra seeds.

There are also handful studies [25, 27, 30, 34, 35] trying to integrate the distributional and pattern-based approaches. Typically, they jointly train a distributional model and a pattern model. However, the supervision of each model totally comes from the given relation instances, which is insufficient in the weakly-supervised setting. Our approach solves the seed scarcity problem with a co-training framework, which encourages both models to provide extra supervision for each other.

7 CONCLUSIONS

In this paper, we studied corpus-level relation extraction in the weakly-supervised setting. We proposed a novel co-training framework called REPEL to integrate a pattern module and a distributional module. Our framework encouraged both modules to provide extra supervision for each other, so that they can collaborate to overcome the scarcity of seeds. Experimental results proved the effectiveness of our framework. In the future, we plan to enhance the pattern module by using neural models for pattern encoding.

ACKNOWLEDGMENTS

Research was sponsored in part by U.S. Army Research Lab. under Cooperative Agreement No. W911NF-09-2-0053 (NSCTA), DARPA under Agreement No. W911NF-17-C-0099, National Science Foundation IIS 16-18481, IIS 17-04532, and IIS 17-41317, and also grant 1U54GM114838 awarded by NIGMS through funds provided by the trans-NIH Big Data to Knowledge (BD2K) initiative.

REFERENCES

[1] E. Agichtein and L. Gravano. Snowball: Extracting relations from large plain-text collections. In *ACM DL'00*, pages 85–94, 2000.

[2] L. Bing, S. Chaudhari, R. Wang, and W. Cohen. Improving distant supervision for information extraction using label propagation through lists. In *EMNLP'15*, pages 524–529, 2015.

[3] A. Blum and T. Mitchell. Combining labeled and unlabeled data with co-training. In *COLT'98*, pages 92–100, 1998.

[4] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, and O. Yakhnenko. Translating embeddings for modeling multi-relational data. In *NIPS'13*, pages 2787–2795, 2013.

[5] R. C. Bunescu and R. J. Mooney. A shortest path dependency kernel for relation extraction. In *HLT-EMNLP'05*, pages 724–731, 2005.

[6] O. Chapelle, B. Scholkopf, and A. Zien. Semi-supervised learning (chapelle, o. et al., eds.; 2006)[book reviews]. *IEEE Transactions on Neural Networks*, 20(3):542–542, 2009.

[7] A. Culotta and J. Sorensen. Dependency tree kernels for relation extraction. In *ACL'04*, pages 423–429, 2004.

[8] J. R. Curran, T. Murphy, and B. Scholz. Minimising semantic drift with mutual exclusion bootstrapping. In *PACLING'07*, pages 172–180, 2007.

[9] J. Daiber, M. Jakob, C. Hokamp, and P. N. Mendes. Improving efficiency and accuracy in multilingual entity extraction. In *I-SEMANTICS'13*, pages 121–124, 2013.

[10] J. Ellis, J. Getman, J. Mott, X. Li, K. Griffitt, S. Strassel, and J. Wright. Linguistic resources for 2013 knowledge base population evaluations. In *TAC'13*, 2013.

[11] O. Etzioni, A. Fader, J. Christensen, S. Soderland, and M. Mausam. Open information extraction: The second generation. In *IJCAI'11*, pages 3–10, 2011.

[12] R. Hoffmann, C. Zhang, X. Ling, L. Zettlemoyer, and D. S. Weld. Knowledge-based weak supervision for information extraction of overlapping relations. In *ACL'11*, pages 541–550, 2011.

[13] G. Ji, S. He, L. Xu, K. Liu, and J. Zhao. Knowledge graph embedding via dynamic mapping matrix. In *ACL'15*, pages 687–696, 2015.

[14] M. Jiang, J. Shang, T. Cassidy, X. Ren, L. M. Kaplan, T. P. Hanratty, and J. Han. Metapad: Meta pattern discovery from massive text corpora. In *KDD'17*, pages 877–886, 2017.

[15] Y. Lin, Z. Liu, M. Sun, Y. Liu, and X. Zhu. Learning entity and relation embeddings for knowledge graph completion. In *AAAI'15*, pages 2181–2187. AAAI, 2015.

[16] Y. Lin, S. Shen, Z. Liu, H. Luan, and M. Sun. Neural relation extraction with selective attention over instances. In *ACL'16*, pages 2124–2133, 2016.

[17] Y. Liu, F. Wei, S. Li, H. Ji, M. Zhou, and H. Wang. A dependency-based neural network for relation classification. In *ACL'15*, pages 285–290, 2015.

[18] C. D. Manning, M. Surdeanu, J. Bauer, J. Finkel, S. J. Bethard, and D. McClosky. The Stanford CoreNLP natural language processing toolkit. In *ACL'14 (System Demonstrations)*, pages 55–60, 2014.

[19] T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[20] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *NIPS'13*, pages 3111–3119. MIT Press, 2013.

[21] M. Mintz, S. Bills, R. Snow, and D. Jurafsky. Distant supervision for relation extraction without labeled data. In *ACL-IJCNLP'09*, pages 1003–1011, 2009.

[22] R. J. Mooney and R. C. Bunescu. Subsequence kernels for relation extraction. In *NIPS'06*, pages 171–178. MIT Press, 2006.

[23] N. Nakashole, G. Weikum, and F. Suchanek. Patty: A taxonomy of relational patterns with semantic types. In *EMNLP'12*, pages 1135–1145, 2012.

[24] J. Pennington, R. Socher, and C. D. Manning. Glove: Global vectors for word representation. In *EMNLP'14*, pages 1532–1543, 2014.

[25] M. Qu, X. Ren, and J. Han. Automatic synonym discovery with knowledge bases. In *KDD'17*, pages 997–1005, 2017.

[26] X. Ren, Z. Wu, W. He, M. Qu, C. R. Voss, H. Ji, T. F. Abdelzaher, and J. Han. Cotype: Joint extraction of typed entities and relations with knowledge bases. In *WWW'17*, pages 1015–1024, 2017.

[27] S. Riedel, L. Yao, A. McCallum, and B. M. Marlin. Relation extraction with matrix factorization and universal schemas. In *HLT-NAACL'13*, pages 74–84, 2013.

[28] M. Schmitz, R. Bart, S. Soderland, O. Etzioni, et al. Open language learning for information extraction. In *EMNLP-CoNLL'12*, pages 523–534, 2012.

[29] V. Shwartz and I. Dagan. Path-based vs. distributional information in recognizing lexical semantic relations. *arXiv preprint arXiv:1608.05014*, 2016.

[30] V. Shwartz, Y. Goldberg, and I. Dagan. Improving hypernymy detection with an integrated path-based and distributional method. In *ACL'16*, pages 2389–2398, 2016.

[31] R. Snow, D. Jurafsky, and A. Y. Ng. Learning syntactic patterns for automatic hypernym discovery. In *NIPS'05*, pages 1297–1304, 2005.

[32] J. Tang, M. Qu, and Q. Mei. Pte: Predictive text embedding through large-scale heterogeneous text networks. In *KDD'15*, pages 1165–1174, 2015.

[33] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei. Line: Large-scale information network embedding. In *WWW'15*, pages 1067–1077, 2015.

[34] K. Toutanova, D. Chen, P. Pantel, H. Poon, P. Choudhury, and M. Gamon. Representing text for joint embedding of text and knowledge bases. In *EMNLP'15*, pages 1499–1509, 2015.

[35] P. Verga, A. Neelakantan, and A. McCallum. Generalizing to unseen entities and entity pairs with row-less universal schema. In *EACL'17*, pages 613–622, 2017.

[36] H. Wang, F. Tian, B. Gao, C. Zhu, J. Bian, and T.-Y. Liu. Solving verbal questions in iq test by knowledge-powered word embedding. In *EMNLP'16*, pages 541–550, 2016.

[37] Z. Wang, J. Zhang, J. Feng, and Z. Chen. Knowledge graph and text jointly embedding. In *EMNLP'14*, pages 1591–1601, 2014.

[38] S. J. Wright. Coordinate descent algorithms. *Mathematical Programming*, 151(1):3–34, 2015.

[39] C. Xu, Y. Bai, J. Bian, B. Gao, G. Wang, X. Liu, and T.-Y. Liu. Rc-net: A general framework for incorporating knowledge into word representations. In *CIKM'14*, pages 1219–1228, 2014.

[40] Y. Xu, L. Mou, G. Li, Y. Chen, H. Peng, and Z. Jin. Classifying relations via long short term memory networks along shortest dependency paths. In *EMNLP'15*, pages 1785–1794, 2015.

[41] M. Yahya, S. Whang, R. Gupta, and A. Y. Halevy. Renoun: Fact extraction for nominal attributes. In *EMNLP'14*, pages 325–335. ACL, 2014.

[42] B. Yang, W.-t. Yih, X. He, J. Gao, and L. Deng. Embedding entities and relations for learning and inference in knowledge bases. In *ICLR'15*, 2015.

[43] D. Zeng, K. Liu, Y. Chen, and J. Zhao. Distant supervision for relation extraction via piecewise convolutional neural networks. In *EMNLP'15*, pages 1753–1762, 2015.

[44] D. Zeng, K. Liu, S. Lai, G. Zhou, and J. Zhao. Relation classification via convolutional deep neural network. In *COLING'14*, pages 2335–2344, 2014.

[45] W. Zeng, Y. Lin, Z. Liu, and M. Sun. Incorporating relation paths in neural relation extraction. In *EMNLP'17*, pages 1769–1778, 2017.