Discovering Correlations between Sparse Features in Distant Supervision for Relation Extraction

Jianfeng Qu
Dantong Ouyang*
College of Computer Science and Technology, Jilin University, Changchun, China
Key laboratory of Symbolic Computation and Knowledge Engineering (Jilin University), Minstry of Education, Changchun, China
qujfjlu@163.com,ouyd@jlu.edu.cn

Wen Hua
School of Information Technology and Electrical Engineering, The University of Queensland, Australia
w.hua@uq.edu.au

Yuxin Ye
College of Computer Science and Technology, Jilin University, Changchun, China
yeyx@jlu.edu.cn

Xiaofang Zhou
School of Information Technology and Electrical Engineering, The University of Queensland, Australia
zxf@itee.uq.edu.au

ABSTRACT

The recent art in relation extraction is distant supervision which generates training data by heuristically aligning a knowledge base with free texts and thus avoids human labelling. However, the concerned relation mentions often use the bag-of-words representation, which ignores inner correlations between features located in different dimensions and makes relation extraction less effective. To capture the complex characteristics of relation expression and tighten the correlated features, we attempt to discover and utilise informative correlations between features by the following four phases: 1) formulating semantic similarities between lexical features using the embedding method; 2) constructing generative relation for lexical features with different sizes of side windows; 3) computing correlation scores between syntactic features through a kernel-based method; and 4) conducting a distillation process for the obtained correlated feature pairs and integrating informative pairs with existing relation extraction models. The extensive experiments demonstrate that our method can effectively discover correlation information and improve the performance of state-of-the-art relation extraction methods.

CCS CONCEPTS

• **Information systems → Information extraction**.

KEYWORDS

Distant supervision; bag-of-words representation; lexical features; syntactic features; feature correlation

ACM Reference Format:
Jianfeng Qu, Dantong Ouyang, Wen Hua, Yuxin Ye, and Xiaofang Zhou. 2019. Discovering Correlations between Sparse Features in Distant Supervision for Relation Extraction. In *The Twelfth ACM International Conference on Web Search and Data Mining (WSDM '19), February 11–15, 2019, Melbourne, VIC, Australia.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3289600.3291004

1 INTRODUCTION

Relation extraction intends to extract structured information (in the format of triplet $r(e_1, e_2)$) from plain texts mentioning these relational facts. Since it is essential for many important applications such as question answering (QA) and knowledge graph construction (KGC), relation extraction becomes a hot research topic [16, 17]. Traditional supervised methods are able to achieve outstanding performance by hand-labelled training data [2, 10]. However, it is commonly known that human effort is laborious and time-consuming, resulting in limited scale and domain of the training data [1].

To break this limitation, [8, 14, 18] employed distant supervision strategy which generates training data by heuristic alignment between a knowledge base (KB) (e.g., Freebase, DBpedia, YAGO) and free texts (e.g., Wikipedia, New York Times). Due to the large scale of relation mentions[1] and the diversity of natural language expressions, lexical and syntactic features extracted by natural language processing (NLP) tools are very abundant. For instance, [18] matched Freebase with New York Times, producing 1,071,684 lexical and syntactic features. In order to equip the relation extractor with the capability to distinguish these features, existing approaches often put all the extracted features into a common feature space in which each dimension denotes an individual feature (lexical and

*Corresponding author

[1] A relation mention indicates a sentence containing the corresponding entity pair, which is believed to describe some ground facts about this pair.

WSDM '19, February 11–15, 2019, Melbourne, VIC, Australia
© 2019 Association for Computing Machinery.
ACM ISBN 978-1-4503-5940-5/19/02…$15.00
https://doi.org/10.1145/3289600.3291004

Table 1: Lexical features for "Basketball player Kobe Bryant was born in Philadelphia, USA."

Feature type	flag	Left window	NE1	Middle	NE2	Right window
Lexical	Inverse_false	[]	PERSON	[was/VED born/VBN in/IN]	LOCATION	[]
Lexical	Inverse_false	[player]	PERSON	[was/VED born/VBN in/IN]	LOCATION	[,]
Lexical	Inverse_false	[basketball player]	PERSON	[was/VED born/VBN in/IN]	LOCATION	[, USA]

Table 2: Syntactic features for "Basketball player Kobe Bryant was born in Philadelphia, USA."

Feature type	Left window	NE1	Dependency path	NE2	Right window
Syntactic	[NMOD]→	PERSON	[SBJ]→[ROOT]←[VC]←[ADV]	LOCATION	
Syntactic		PERSON	[SBJ]→[ROOT]←[VC]←[ADV]	LOCATION	[P]→
Syntactic		PERSON	[SBJ]→[ROOT]←[VC]←[ADV]	LOCATION	[NMOD]→

Table 3: Examples of features extracted for relation "contain(location,location)"

<table>
<tr><th>Feature type</th><th>Feature example</th></tr>
<tr><td>Lexical</td><td>1.inverse_true|LOCATION|<b>, a town in</b>|LOCATION
2.inverse_true|LOCATION|<b>, a small town in</b>|LOCATION
3.inverse_true|LOCATION|<b>, a mountain town in</b>|LOCATION
4.inverse_true|LOCATION|<b>, a town in southern</b>|LOCATION
5.inverse_true|LOCATION|<b>, a small town in the central Mexican state of</b>|LOCATION
6.inverse_true|LOCATION|<b>, a remote village in northeastern</b>|LOCATION
7.inverse_true|LOCATION|<b>, a holy city in southern</b>|LOCATION
8.inverse_true|LOCATION|<b>, which is in the southeast corner of</b>|LOCATION</td></tr>
<tr><td>Syntactic</td><td>1.dep:LOCATION|[PMOD]→[ADV]→[NMOD]←[NMOD] ←[PMOD]←|LOCATION|[NMOD]→
2.dep:LOCATION|[PMOD]→[ADV]→[NMOD]→<b>[NMOD]</b>←[NMOD] ←[PMOD]←|LOCATION|[NMOD]→</td></tr>
</table>

syntactic features), and then translate each relation mention into a feature vector using the bag-of-words[2] representation.

The bag-of-words representation, however, ignores underlying correlations between features and cannot sufficiently capture the complex linguistic characteristics of features [21]. Specifically, some features located in different dimensions play fundamentally the same role in predicting specific relations, but vary in their expressions. Unfortunately, due to the shallow expressive ability of the bag-of-words representation, these correlated features are treated as completely different by existing extractors. The extreme sparsity and high dimensionality of the feature space makes relation extractors incapable of capturing the key clues to make accurate predictions. In summary, the bag-of-words representation wastes the valuable information between features, hence degrading the effectiveness of relation extraction.

1.1 Preliminaries

To help readers better understand this work, we first give a brief introduction about the features used in relation extraction. According to [14, 18], the features extracted from relation mentions using NLP tools can be grouped into two classes: lexical features and syntactic features.

Lexical features describe surface context appearing between or surrounding the entity pair (e_1, e_2), including:

- The sequence of words between the entity pair and their part-of-speech (POS) tags;
- A flag reflecting which entity appears first in the sentence;
- A window of K words to the left of e_1;
- A window of K words to the right of e_2.

Each lexical feature combines all these components. In most situations, we produce a conjunctive feature for each $K \in \{0, 1, 2\}$. As shown in Table 1, we can get three lexical features from sentence "Basketball player Kobe Bryant was born in Philadelphia, USA." for entity pair ("Kobe Bryant", "Philadelphia").

Syntactic features are intended to represent the syntactic structure of sentences. Dependency path is usually considered which contains a series of directional syntactic relationships (e.g.,"NMOD", "SBJ", "P") and directions ('←','→'). Our syntactic features consist of the conjunction of:

- A dependency path between the entity pair;
- A left or right window of one element that is not part of the above dependency path.

Table 2 illustrates several syntactic features for the sentence "Basketball player Kobe Bryant was born in Philadelphia, USA." and the entity pair ("Kobe Bryant", "Philadelphia").

1.2 Motivation

To visually illustrate the problem existed in the bag-of-words representation, we list some example features derived from a commonly-used dataset [18] in Table 3. We can see that all the lexical features explicitly describe the same relational fact that one location is contained in another location, i.e., relation "contain(location,location)". The only difference between them lies in the linguistic characteristics such as adjectives (e.g., "small", "remote", "holy"), nouns (e.g., "town", "city", "village") and constitutes of sentences (e.g., feature 8). As for syntactic features, both of them express approximately the same structure expect one more "[NMOD]" in feature 2. Actually, the symbol "[NMOD]" represents a modifier of nominal, which often plays a trivial role in mentioning a relation.

Although there exist abundant correlations between features, the bag-of-words representation never takes them into consideration, restricting its capability for relation extraction. We will show through this work that the performance of most state-of-the-art feature-based relation extractors can be improved if feature correlations are effectively identified and utilised. Consider lexical features in Table 3 as an example. If we encounter any of these features (e.g.,

[2] Here, "words" indicate lexical and syntactic features.

feature 1) in a relation mention, all the other highly correlated features (e.g., features 2-8) can be integrated into the feature vector of that mention, enriching the extractor with more information about the corresponding relation and hence improving its performance.

1.3 Challenges

As described above, features differ in their type (i.e., lexical and syntactic), size of the window (i.e., $K \in \{0, 1, 2\}$ in lexical features), etc. The diversity of features poses great challenges in exploring their correlations.

How to deal with different types of features? Lexical features deliver surface information by vocabularies (e.g., “in”, “born”, “president”) while syntactic features describe grammatical structures of relation mentions. Therefore, it is inappropriate to develop a unified method to discover correlations existed in these two types of features. In other words, we need to consider a customised metric for each type of features. Besides, since the number of words between the target entity pair for lexical features can be arbitrary, the developed metric should be independent of the number of words in-between. Moreover, the in-between words have different importance for identifying the desired relation, which should also be considered. Take “LOCATION|, a small town in |LOCATION” as an example. The word “in” is of more importance than “small” for relation “contain(location,location)”.

How to handle lexical features with different sizes of side windows? The lexical features extracted from a relation mention only differ in their window sizes (i.e., $K \in \{0, 1, 2\}$), meaning that these features are highly correlated in extracting the desired relation. Therefore, the developed metric for lexical features should take such correlation into account, rather than simply consider features with the same side window.

How to guarantee the discovered correlations can indeed improve relation extraction? Although there exist a large number of correlations between features, not all of them are beneficial to relation extraction. Some correlations may mislead the relation extractor to an unknown direction, and some may weaken the degree of distinction between features. Consequently, it is indispensable to decide which correlations are informative and remove the rest before incorporating them into relation extractors.

1.4 Contributions

To address the shortcomings of the bag-of-words representation, we explore correlated features from various aspects. To the best of our knowledge, we are the first to directly consider feature correlations in distant supervision for relation extraction. More specifically, our technical contributions in this work are summarised as follows:

a. We apply word embeddings to transform each word in a lexical feature to a continuous, real-valued vector, and the lexical feature is represented by a sequence of word vectors. We then propose average-pooling and max-pooling to obtain a unified representation of these features. In addition, we design self-attention mechanism to assign important words with higher attention weights in the vector representation of lexical features. The correlations between lexical features are computed based on their vector representation.

b. For lexical features with different sizes of side windows, we introduce a way to build generative relation from head features to tail features and estimate the probability of generative relation from a large-scale training data.

c. As syntactic features are composed of some meaningful symbols, we extend an existing kernel method, Mismatch String Kernel, to calculate their correlations.

d. For the obtained correlated feature pairs, we develop a variant of Empirical Conditional Entropy to measure their informativeness and utilise the major voted relation labels of these features to distill the informative correlations.

e. We integrate informative feature correlations with state-of-the-art relation extractors and empirically verify that our approach can effectively discover useful correlations hidden in different features and improve relation extraction.

The remainder of this paper is organised as follows: Section 2 elaborates on our approaches to discovering informative correlations between features; Section 3 reports our experimental results and analysis; Section 4 reviews literature in distant supervision for relation extraction, followed by a brief conclusion in Section 5.

2 METHODOLOGY

The objective of exploring feature correlations is to strengthen these inner-related features, and capture the complex linguistic characteristics of features to improve relation extraction. Assume that we have successfully discovered a set of informative feature correlations denoted as $\{\langle f_{ci}, f_{cj}\rangle\}$. Given a relation mention’s feature vector $\boldsymbol{S_i} : \{f_1, f_2, \cdots, f_n\}$, *if the first element f_{ci} of a feature pair $\langle f_{ci}, f_{cj}\rangle$ exists in $\boldsymbol{S_i}$ while the second element f_{cj} does not (i.e., $f_{ci} \in \boldsymbol{S_i}$ and $f_{cj} \notin \boldsymbol{S_i}$), we will add the feature f_{cj} into $\boldsymbol{S_i}$*. We call it the **“addition process”** hereafter. This process will result in **higher probability of co-occurrence** for feature pair f_{ci} and f_{cj}, in other words, richer information that the relation extractor can utilise to make accurate predictions.

For the rest of this section, we will elaborate on our methods for discovering feature correlations as well as determining the informativeness of each correlation to guarantee that only the informative correlations associated with a certain relation are remained for the addition process. Consider the different types of features, our methods are mainly divided into two parts: lexical correlations and syntactic correlations.

2.1 Correlations between lexical features

There is a flag in each lexical feature reflecting which entity appears first in the sentence (i.e., whether the appearance order of the target entity pair in the sentence is consistent with that in the KB). Intuitively, it is meaningless to compare lexical features with different flags since many relations are directional and asymmetric. Therefore, we only consider correlations between lexical features with the same flag. In addition, lexical features extracted from relation mentions could have different sizes of side windows. We propose two methods, namely *embedding method* and *generative method*, to deal with lexical features with identical and different size of side windows respectively.

2.1.1 Embedding method.

2.1.1 Embedding method. Assume there are two lexical features, Lex_1 and Lex_2, with the same size of side windows. As illustrated

in Table 3, these features could have similar semantic meanings and are capable of identifying the same relation, although their surface forms differ. Therefore, we propose the embedding method to capture such kind of feature correlation. Specifically, we first transform each word in the lexical feature into a continuous, real-valued vector via a pre-trained word2vec tool [12]. The transformed features are denoted as:
Lex_1: $[\boldsymbol{w}_l][\text{CATEGORY}_1][\boldsymbol{w}_m^1,\boldsymbol{w}_m^2,\cdots,\boldsymbol{w}_m^p][\text{CATEGORY}_2][\boldsymbol{w}_r]$
Lex_2: $[\boldsymbol{w}_l'][\text{CATEGORY}_1'][\boldsymbol{w}_m'^1,\boldsymbol{w}_m'^2,\cdots,\boldsymbol{w}_m'^q][\text{CATEGORY}_2'][\boldsymbol{w}_r']$
where the bold letter $\boldsymbol{w}$ represents a vector, the subscript l, m and r denote left part, middle part and right part of the lexical feature respectively, and the superscript p and q are the length of the middle part. Here the two side parts ($\boldsymbol{w}_l$ and $\boldsymbol{w}_l'$, $\boldsymbol{w}_r$ and $\boldsymbol{w}_r'$) have the same length. CATEGORY represents the type of the target entity (e.g., PERSON, ORGANIZATION, LOCATION, NONE). Obviously, it is meaningless to consider feature correlations unless CATEGORY_1 equals $\text{CATEGORY}_1'$ and CATEGORY_2 equals $\text{CATEGORY}_2'$.

The lexical features are divided into three parts (i.e., left, middle and right) which intuitively play different roles in expressing a relation. Hence, we separately compute the semantic similarity between each part of two lexical features. Unlike the side parts, the length of the middle part can be an arbitrary number, making it non-trivial to calculate their similarity. In this paper, we apply three strategies, namely average-pooling, max-pooling, and self-attention, to combine multiple word vectors and obtain a meaningful estimation of the semantic correlation between Lex_1 and Lex_2.

Average-pooling is a moderated method that takes the average value of all the elements. The essence behind this method is to assume that every word has the same importance in describing a relation. We define the pooled representation of three parts (i.e., $l(Lex)$, $m(Lex)$ and $r(Lex)$) for a lexical feature (i.e., Lex) as:

$$Lex: \begin{cases} l(Lex) = \frac{1}{K}\sum_{i=0}^{K} \boldsymbol{w}_l^i & K \in \{1,2\} \\ \\ m(Lex) = \frac{1}{P}\sum_{i=0}^{P} \boldsymbol{w}_m^i & P \in \{1,2,3,4,\cdots\} \\ \\ r(Lex) = \frac{1}{K}\sum_{i=0}^{K} \boldsymbol{w}_r^i & K \in \{1,2\} \end{cases} \tag{1}$$

where K is the length of the side window and P, an arbitrary positive integer, indicates the number of words in the middle part. It is worth noting that the addition and average are element-wise operations (i.e., each dimension is handled separately).

Max-pooling is often utilised in neural models as a layer subsequent to a series of convolutional filters to make the sentence representation independent of its length. In this work, we straightforwardly adopt max-pooling for word embeddings as below:

$$Lex: \begin{cases} l(Lex) = Max(\boldsymbol{w}_l^i) & \boldsymbol{w}_l^i \in \boldsymbol{w}_l \\ \\ m(Lex) = Max(\boldsymbol{w}_m^i) & \boldsymbol{w}_m^i \in \boldsymbol{w}_m \\ \\ r(Lex) = Max(\boldsymbol{w}_r^i) & \boldsymbol{w}_r^i \in \boldsymbol{w}_r \end{cases} \tag{2}$$

where $\boldsymbol{w}_l$, $\boldsymbol{w}_m$ and $\boldsymbol{w}_r$ are sets of vectors in different parts of Lex. Similar to average-pooling, max-pooling is also an element-wise operation. But the final result only considers the maximum value in each unit (i.e., dimension) instead of an equal treatment.

Self-attention mechanism, proposed by [22], is designed to alternate recurrent neural networks (RNN) so that it can incorporate parallel technologies and speed up the efficiency of their entire model. In this work, we also adopt the attention mechanism to combine word vectors. As discussed in Section 1.3, not all words in a sentence contribute equally to expressing a relational fact. Consequently, the equal treatment of each word without any discrimination, as in average-pooling, could be inappropriate in practice. On the other hand, max-pooling is a unit-level pooling method and we argue that it is too fine-grained to disassemble the internal structure of a word. The fact that convolutional filters only slide on word-direction instead of both directions (i.e., word-direction and unit-direction) proves our claim. Hence, we develop a word-level self-attention mechanism to produce the vector representation of lexical features. Specifically, the representation of each part can be formulated as follows:

$$Lex: \begin{cases} l(Lex) = \sum_{i=0}^{K} \alpha_i \cdot \boldsymbol{w}_l^i & K \in \{1,2\} \\ \\ m(Lex) = \sum_{i=0}^{P} \alpha_i \cdot \boldsymbol{w}_m^i & P \in \{1,2,3,4,\cdots\} \\ \\ r(Lex) = \sum_{i=0}^{K} \alpha_i \cdot \boldsymbol{w}_r^i & K \in \{1,2\} \end{cases} \tag{3}$$

where α_i is the attention weight for $\boldsymbol{w}^i$. We further define α_i as:

$$\alpha_i = softmax(\sum_{j=1}^{N} (\boldsymbol{w}^i)^T \boldsymbol{w}^j) = \frac{\exp(\sum_{j=1}^{N} (\boldsymbol{w}^i)^T \boldsymbol{w}^j)}{\sum_{i=1}^{N} \exp(\sum_{q=1}^{N} (\boldsymbol{w}^i)^T \boldsymbol{w}^q)} \tag{4}$$

where $N = P$ or $N = K$ corresponds to different parts of Lex. With the help of Eq. 4, critical words can occupy more proportion in feature representation through higher attention weights while trivial words inversely occupy less proportion.

Now we can obtain the vector representation of lexical features which is independent of their length[3]:
Lex_1: $[l(Lex_1)][\text{CATEGORY}_1][m(Lex_1)][\text{CATEGORY}_2][r(Lex_1)]$
Lex_2: $[l(Lex_2)][\text{CATEGORY}_1'][m(Lex_2)][\text{CATEGORY}_2'][r(Lex_2)]$
The dimension of $l(Lex)$, $m(Lex)$ and $r(Lex)$ equals to the dimension of initial word embeddings. Eventually, the correlation score of two lexical features is computed as:

$$Cor(Lex_1, Lex_2) = \begin{cases} Min\{cos\langle l(Lex_1), l(Lex_2)\rangle, cos\langle m(Lex_1), \\ m(Lex_2)\rangle, cos\langle r(Lex_1), r(Lex_2)\rangle\}, \quad K \in \{1,2\} \\ \\ \cos\langle m(Lex_1), m(Lex_2)\rangle, \quad K = 0 \end{cases} \tag{5}$$

where $cos\langle a, b\rangle$ means the cosine similarity between vectors a and b. When the side window occurs, we take the minimum value of three parts as the final correlation score between two lexical features. In practice, the minimum value often belongs to the middle part, which is consistent with our intuition that the words between entities of interest are more functional in describing their relation.

2.1.2 Generative method.

As introduced in Section 1.1, we will get three lexical features for a relation mention corresponding to different sizes of side windows ($K \in \{0,1,2\}$). We denote them as K-0, K-1, and K-2 respectively. Obviously, it is unreasonable to consider semantic similarity between these features as they are derived from the same sentence. Therefore, we propose a new relationship called *generative relation*. That is, if two lexical features

[3] Here, the length mainly refers to the middle window as the side window of two lexical features is of the same length.

have identical middle part but vary in the size of the side window, then the generative relation that K-0 generates K-1, K-1 generates K-2, or K-0 generates K-2 (the cross-layer generative relation) will be constructed. The generative relation, denoted as $Gen\langle h, t\rangle$, is an ordered relationship without the property of reflexivity, where h is the head feature and t is the tail feature. For instance, given three lexical features:
L_0: Inverse_false[][PERSON][was born in][LOCATION][]
L_1: Inverse_false[player][PERSON][was born in][LOCATION][,]
L_2: Inverse_false[basketball player][PERSON][was born in][LOCATION][,USA]
we will obtain the following generative feature pairs: $Gen\langle L_0,\ L_1\rangle$, $Gen\langle L_1,\ L_2\rangle$ and $Gen\langle L_0,\ L_2\rangle$. After acquiring all the generative feature pairs from a large-scale training data, for each head feature h_i, we gather all the tail features (i.e., $\{t_1, t_2, t_3, \cdots, t_{Ti}\}$) generated by h_i as a tail set for h_i. We then explore feature co-occurrence to calculate the probability of generative relations:

$$Gen\left\langle h_i, t_j\right\rangle = \frac{O(h_i, t_j)}{\sum_{k=1}^{Ti} O(h_i, t_k)} \tag{6}$$

where $O(h_i, t_j)$ is the number of times that h_i and t_j appear in the same relation mention.

2.2 Correlations between syntactic features

In this section, we introduce our method to measure the correlations between syntactic features. As mentioned above, syntactic features consist of a series of directional syntactic relationships (e.g., "SBJ", "ADV", "P") and directions. Hence, we apply kernel method to compute the correlations between syntactic features.

In NLP, many efforts have been invested in developing various kinds of kernel functions to boost the performance of classification tasks [3, 5, 9, 15]. Unlike them, kernel function in this work serves as a correlation criterion. Since syntactic features used in relation extraction are a type of shallow parsing, we utilise one of the most commonly used kernel methods, Mismatch String Kernel (MSK) [11], to calculate feature correlation. We denote the mapping for syntactic feature Syn as:

$$\phi_u^{p,m}(Syn) = |\{(v_1, v_2) : s = v_1 v v_2 : |u| = |v| = p, d(u,v) \leq m\}| \tag{7}$$

where v_1, v_2, v, s and u are conjunctions of directed syntactic relationships, p is the length of u and v, and $d(u,v) \leq m$ counts the number of directed syntactic relationships in u that differ from v by at most m. We then formulate the associated kernel between two syntactic features $Syn1$ and $Syn2$ by:

$$\begin{aligned} k_{p,m}(Syn_1, Syn_2) &= \langle \phi^{p,m}(Syn_1), \phi^{p,m}(Syn_2)\rangle \\ &= \sum_{u \in \sum^p} \phi_u^{p,m}(Syn_1)\phi_u^{p,m}(Syn_2) \end{aligned} \tag{8}$$

where $\sum^p$ is an arbitrary conjunction of directed syntactic relationships with length of p. To decrease the complexity of calculating Eq. 8, following [9], we use a trie-based computing strategy. In this way, the computational complexity of Eq. 8 is reduced to $O(p^{m+1}|\sum|^m(|Syn1| + |Syn2|))$.

Furthermore, we measure the similarity between syntactic features $Syn1$ and $Syn2$ through an adjustment of MSK as below:

$$Sim\langle Syn_1, Syn_2\rangle = \frac{K_{p,m}\langle Syn_1, Syn_2\rangle}{K_{p,m}\langle Syn_1, Syn_1\rangle + K_{p,m}\langle Syn_2, Syn_2\rangle} \tag{9}$$

The denominator of Eq. 9 is a normalisation factor that makes $Sim\langle Syn1, Syn2\rangle \in [0,1]$. Note that this equation makes sense only when the syntactic features have the identical location (i.e., left and right) of windows and the same type of entity pairs.

2.3 Distillation of correlations

After applying the above approaches, we will obtain a great number of related feature pairs: $\langle f_{c1}, f_{c2}\rangle, \langle f_{c3}, f_{c4}\rangle, \cdots, \langle f_{ci}, f_{cj}\rangle \cdots$ with their correlation values. We pre-define thresholds λ_1 and λ_2 for lexical features and syntactic features respectively for an initial filtering of weakly-correlated feature pairs. For each retained pair $\langle f_{ci}, f_{cj}\rangle$, we believe they play approximately the same role in expressing a certain relation.

In order to guarantee the effectiveness of these correlated feature pairs for relation extraction, we need to make sure that these pairs are associated with a certain relation. Feature pairs obtained through the generative method naturally satisfy this criterion.[4] However, the embedding method and MSK have not built an intrinsic link between the phases of discovering feature correlations with the task of relation extraction. That is, the embedding method and MSK are mainly established on semantic level without any consideration of the distribution of features belonging to a correlated feature pair and their corresponding coincident relation labels in training data. Therefore, not all newly-discovered feature pairs are beneficial to the relation extractor. We argue that the informative feature pairs should have the following two properties: *(1) the two features in a feature pair must have semantic relatedness or intrinsic relatedness; (2) The added feature f_{cj} will appear more frequently in its major anchored relation label.*

Since we obtain feature pairs by using the embedding method and MSK, the first property has already been met. The second property is proposed to make features, which themselves have the same effect but are regarded as completely different by the bag-of-words representation, linked more closely through **higher probability of co-occurrence**. Furthermore, we intend to group features into a useful cluster with reference to the specific relation label. During training process, the extractor discerns the cluster and assigns higher weights to features belonging to the cluster. Then the cluster of features becomes the critical clues for the extractor to predict whether a sentence expresses the specific relation label or not, and will be explicitly distinguished from other features by the relation extractor. Figure 1 visualises the intention of our proposed method.

2.3.1 *Variant of empirical conditional entropy.*

Based on the above analysis, we develop a variant of empirical conditional entropy to select informative feature pairs from the original set. The existing

[4] The distillation of feature pairs does not include the pairs achieved from the generative method as they are statistics of the training data.

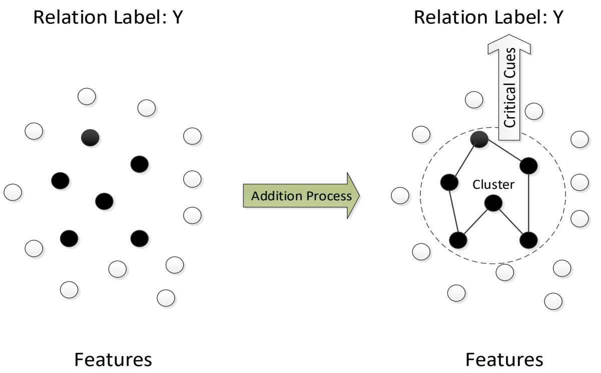


Figure 1: The degree of tightness between features before and after the addition process

empirical conditional entropy $H(Y|X)$ can be denoted as:

$$H(Y|X) = \sum_{i=1}^{n} p(x_i)H(Y|x_i) = -\sum_{i=1}^{n}\sum_{j=1}^{m} p(x_i)p(y_j|x_i)\log p(y_j|x_i) \tag{10}$$

where x_i is a possible value of feature X and y_j is a possible value of label Y. In the bag-of-words representation, X is a binary variable (i.e., $X \in$\{0, 1\}). Due to the severe sparsity in training data, in most cases, X takes the value 0, which will make $p(X = 0)$ approximately equals to 1. On the other hand, we mainly focus on relation labels where X is active (i.e., $X = 1$). Hence, we design a variant of empirical conditional entropy $H_{va}(Y|X)$ as below:

$$H_{va}(Y|X) = -\sum_{j=1}^{m} p(y_j|X=1)\log p(y_j|X=1) \tag{11}$$

Given a related feature pair $\langle f_{ci}, f_{cj}\rangle$, if we use the feature pair to conduct the addition process, the value of $H_{va}(Y|X)$ for feature f_{cj} might be changed. Here, we use $H_{va}^{bef}(Y|f_{cj})$, $H_{va}^{aft}(Y|f_{cj})$ to represent $H_{va}(Y|f_{cj})$ before and after the addition process, respectively. According to the second property for informative feature pairs, f_{cj} needs to appear more frequently in its major voted relation label, which can be formulated as $H_{va}^{bef}(Y|f_{cj}) \geq H_{va}^{aft}(Y|f_{cj})$.

2.3.2 Validation of major voted relation label. In fact, the change of the value for $H_{va}(Y|f_{cj})$ can only guarantee that the feature f_{cj} will concentrate more on one relation label. But it does not mean the concentrated label is its originally major voted one. For example, if f_{ci} occurs much more frequently than f_{cj} in the training data, the distribution of f_{cj} will dramatically shift to the distribution of f_{ci} after the addition process. In such a circumstance, the effect of f_{cj} will deviate from its initial purpose if the major voted labels of f_{ci} and f_{cj} are different.

To avoid the deviation, we collect statistics on the initial training data about the most possible label sets (i.e., $Lab(f_{ci})$, $Lab(f_{cj})$) for f_{ci} and f_{cj}, respectively. Since there exists multi-label problem in distantly supervised relation extraction (i.e., an entity pair has multiple relation labels in the KB, for example, capital(France, Paris), contain(France, Paris), etc.), some relation labels may co-occur in these aligned sentences (e.g., the sentence mentioning (France, Paris)). We first remove these co-occur labels from $Lab(f_{ci})$ and $Lab(f_{cj})$, and denote the remaining parts as: $Lab^r(f_{ci})$ and $Lab^r(f_{cj})$. According to the second property of informative pairs, $Lab^r(f_{ci})$ should be included by $Lab^r(f_{cj})$ (i.e., $Lab^r(f_{ci}) \subset Lab^r(f_{cj})$).

Algorithm 1 Distillation of correlations

```
Input the set of correlated feature pairs $FP : \{\langle f_{c1}, f_{c2}\rangle, \langle f_{c3}, f_{c4}\rangle, \langle f_{ci}, f_{cj}\rangle \cdots\}$, the set of all the mentions' feature vectors with their labels $D : \{(Sen^q, Lable^q)|q = 1, ..., N\}$, the set of the most possible relation labels for each feature $Lab : \{Lab(f_{c1}), Lab(f_{c2}), Lab(f_{c3}), \cdots\}$.
1:  initialize $O = \{\}$
2:  do randomly select a feature pair $\langle fc_i, fc_{i+1}\rangle$ from $FP$ and delete the pair from $FP$
3:      initialize $D' = \{\}$
4:      compute $H_{va}^{bef}(Y|f_{cj})$ according to $D$
5:      for $q = 1, \ldots, N$ do
6:          if $f_{ci} \in Sen^q$ and $f_{cj} \notin Sen^q$
7:              put $f_{cj}$ to $Sen^q$
8:          end if
9:          put $(Sen^q, Label^q)$ to $D'$
10:     end for
11:     compute $H_{va}^{aft}(Y|f_{cj})$ according to $D'$
12:     get $Lab^r(f_{ci})$ and $Lab^r(f_{cj})$ from $Lab(f_{ci})$ and $Lab(f_{cj})$ by removing co-occured labels in training data
13:     if $H_{va}^{bef}(Y|f_{cj}) \geq H_{va}^{aft}(Y|f_{cj})$ and $Lab^r(f_{ci}) \subset Lab^r(f_{cj})$
14:         put $(\langle f_{ci}, f_{cj}\rangle$, 'informative') to $O$
15:     else
16:         put $(\langle f_{ci}, f_{cj}\rangle$, 'non-informative') to $O$
17:     end if
18: repeat step 2-16 until $FP = \emptyset$
Output $O$
```

In summary, the informative feature pair $\langle f_{ci}, f_{cj}\rangle$ should meet the following two criteria:

(1) : $H_{va}^{bef}(Y|f_{cj}) \geq H_{va}^{aft}(Y|f_{cj})$
(2) : $Lab^r(f_{ci}) \subset Lab^r(f_{cj})$

Algorithm 1 shows the distillation process of these correlations. The algorithm is executed for correlated feature pairs obtained by word embeddings and MSK. Since feature pairs acquired by generative relation are the statistical results of the training data, it is unreasonable to adopt Algorithm 1 to handle them.

3 EXPERIMENTS

In this section, we empirically evaluate our approach compared with several state-of-the-art relation extractors.

Dataset

We consider the most commonly used dataset which was developed by [18]. The data was generated by aligning Freebase relations with New York Times (NYT) corpora. To find entity mentions in texts, they used Stanford named entity recognizer [5] [7] and treated consecutive mentions with the same category as a single entity

[5] Available at http://nlp.stanford.edu/software/CRF-NER.shtml

Table 4: P@N scores for relation extraction

P@N(%)	100	200	300	Mean
Mintz	52.29	49.28	46.28	49.28
Mintz+Cor	59.63	56.94	52.43	**56.33**
Multir	59.00	65.50	60.67	61.72
Multir+Cor	71.29	66.17	61.13	**66.19**
RankRE	55.00	56.50	50.33	53.94
RankRE+Cor	61.00	57.5	52.33	**56.94**

mention. The association between Freebase and New York Times was built by performing a string match between entity mention phrases and the canonical names of entities in Freebase. The Freebase relations were divided into two parts, one for training and the other for testing. Then the former was aligned to NYT in the year 2005-2006 and the latter to NYT in the year 2007. After that, for each entity pair, they used openNLP POS tagger [6] and MaltParser [7] to obtain lexical and syntactic features respectively.

Implementation details

For the embedding method, we employ the popular word2vec[8] tool to achieve meaningful vector representations of words. We use three-fold validation on the training data to decide values of the parameters. The default parameter settings are as follows:

- Word dimension: 50
- Threshold for lexical features λ_1: 0.9
- Threshold for syntactic features λ_2: 0.95

As our proposed method can be applied to any feature-based models for relation extraction[9], we select three typical methods to empirically evaluate the effectiveness of our approach:

- ***Mintz*** [14] is a traditional feature-based method and is the first to match a large KB with free texts to provide training data for relation extraction.
- ***Multir*** [8] is a representative approach of alleviating multi-instance multi-label problem in distant supervision.
- ***RankRE*** [26] applies learning-to-rank to relation extraction so that the model can aggregate inter-sentence information for predictions.

The discovered correlations are incorporated into the training and testing data of these methods through the addition process. Specifically, the correlations obtained by the embedding method and MSK are used to change the feature space of both training and testing data. Correlations from the generative method, however, are only applied to the testing set since they are obtained through statistic analysis of the training data. In addition, the newly-added feature f_{cj} from the embedding method and MSK receives the same initial weight as f_{ci} (i.e., $weight(f_{cj}) = 1$), while the feature weight from the generative method will be initialised as the probability of the generative relation (i.e., $weight(f_{cj}) = Gen\langle f_{ci}, f_{cj}\rangle$). In this way, we obtain three corresponding approaches: ***Mintz+Cor***, ***Multir+Cor*** and ***RankRE+Cor*** respectively to be compared with their corresponding baselines.[10]

Precision and recall curves

Following [8, 14, 26], we conduct the held-out evaluation which generates precision and recall by comparing predictions with the relational facts in KB. Figure 2 depicts the PR curves for each pair of comparative methods. From this figure, we can see that: (1) Our proposed methods (i.e., *Mintz+Cor*, *Multir+Cor* and *RankRE+Cor*) substantially and constantly achieve higher precision than their corresponding baselines with the same recall. This result proves that the informative correlations can indeed tighten the sparse feature space and resolve the drawbacks of the bag-of-words representation in relation extraction. (2) When recall is low, the performance improvements after using correlations are more prominent. We believe that the correlated pairs are able to make the extraction models aware of the critical clues of the specific relation labels so that the models have more confidence in making accurate predictions. (3) For *Multir+Cor*, there exists a dramatic drop around the recall point of 1.2%. We take a deep inspection for the predictions located in this region, and find that most of the predictions are actually correct but are not included in the KB due to its incompleteness (i.e., false negative problem).

P@N scores

We rank the predictions in decreasing order of confidence scores, fetch the top N predictions and check their accuracy. The results are shown in Table 4. From this table, we find that in P@100, P@200 and P@300, as expected, our methods boost the performance of their baselines. Eventually, in term of mean scores, *Mintz+Cor* gets the precision that is 7.05% higher than *Mintz*. Similarly, *Multir+Cor* and *RankRE+Cor* have improvements of 4.47% and 3% compared to their corresponding baselines. It indicates that the bag-of-words representation is unable to capture the complex linguistic characteristics of features, and confuses the extraction models. In contrast, the proposed correlations alleviate this problem, and can be beneficial to these relation extraction models which employ shallow feature representation.

Effectiveness of different types of correlations

In the above experiment, we have overall results of using all the correlations. Similar to the categories of features, our correlations can be mainly grouped into two parts: lexical correlations and syntactic correlations. Here, we are intended to check the effectiveness of each type of correlations separately, which is shown in Figure 3. In this figure, we have the following observations: (1) Both *RankRE+Cor(Lex)* and *RankRE+Cor(Syn)* bring better performance than *RankRE.* It demonstrates that the embedding method, together with the generative method is effective in exploring inner correlations between lexical features while MSK is useful in the area of syntactic features. It is inappropriate to compare *RankRE+Cor(Lex)* with *RankRE+Cor(Syn)* because they are for different purposes. (2) In most regions of the curves, *Rank+Cor* often achieves the highest precision among all the counterparts. It indicates that considering

[6] Available at http://opennlp.sourceforge.net/

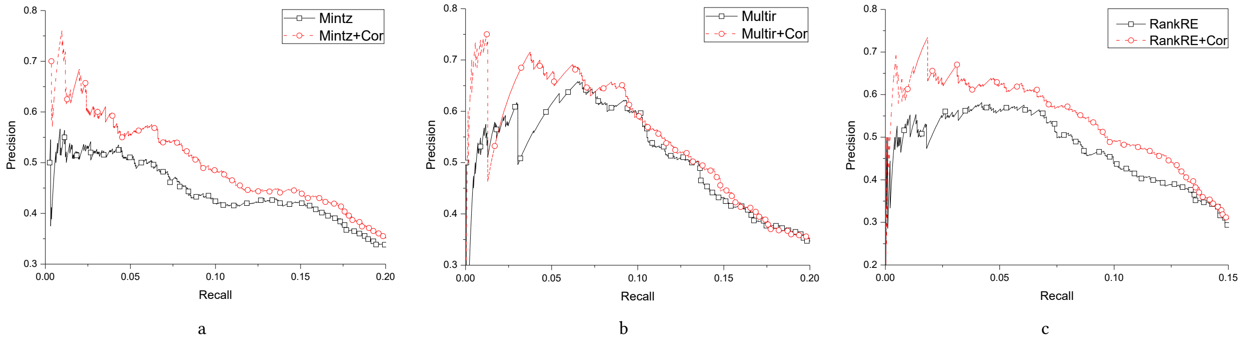
[7] Available at http://www.maltparser.org/

[8] Available at https://code.google.com/p/word2vec/

[9] These methods often adopt the bag-of-words representation to vectorise sentence.

[10] The lexical features directly use self-attention to produce the embedding representation. We will analyse other strategies: average-pooling and max-pooling later.

Figure 2: Comparison with the existing feature-based methods



a

b

c

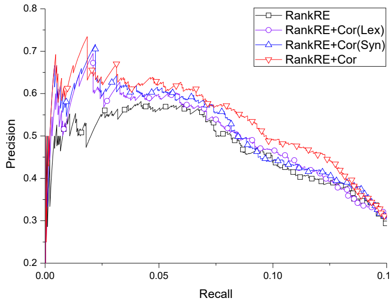


Figure 3: PR-curves for different sets of correlations

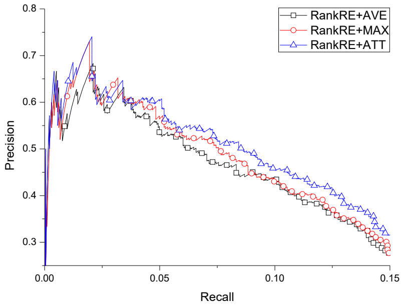


Figure 4: PR-curves for different pooling strategies

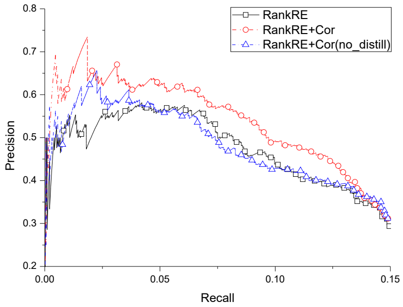


Figure 5: Importance of distillation process

the combination of both lexical and syntactic correlations is superior to any individual part, although either of these correlations can make an improvement.

Effectiveness of attention mechanism

We propose three optional strategies (i.e., average-pooling, max-pooling and self-attention) to make lexical features' representation independent of their length. Here, we check the effectiveness of each strategy. The results are depicted in Figure 4. We can see that: (1) *RankRE+AVE* is inferior to *RankRE+MAX*. It means that the average treatment attenuates the key components of lexical features and results in an ineffective representation of them. (2) *RankRE+ATT* offers better performance compared to *RankRE+MAX*. We believe that the unit-level (i.e., dimension) operations destroys the semantic structures of words. On the contrary, our self-attention mechanism retains the interior structure of words and focuses on word-level attention so that the important words occupy more proportion in feature representation.

Effectiveness of distillation process

Figure 5 shows the comparison results of whether to use the distillation process or not (namely *RankRE+Cor* and *RankRE+Cor(no_distill)*). In this figure, *RankRE+Cor(no_distill)* performs much worse than *RankRE+Cor*, even worse than *RankRE* when recall is between 5% and 10%. In order to find the reason, we analyse the proportion of positive and negative (NA) mentions in training data and the correlated feature pairs reserved by *RankRE+Cor(no_distill)*. The fact is that in training data, the number of NA mentions is almost five times greater than the number of positive mentions. Hence, most of the extracted features appear in the NA label. The undistilled feature pairs will shift the feature space to NA. However, NA label is not part of the label-set in the precision/recall curves reported in the research community. Therefore, the results demonstrate the necessity of the distillation process, and our distillation strategy equips the model with the ability to filter out useless feature pairs so that the distilled correlations can indeed benefit relation extraction.

4 RELATED WORK

The idea of distant supervision was originally introduced in information extraction by [4], which generated training data for a naive-Bayes extractor by matching Yeast Protein Database with PubMed abstracts. Analogously, [1] used a database of BibTeX records and research paper citations to learn a linear-chain CRF. The KYLIN system [23] trained an extractor to create new infoboxes autonomously by employing training examples from classes of pages (Wikipedia) with similar infoboxes. Later, researchers paid more attention to big data. In 2009, [14] aligned an existing knowledge base (Freebase) with Wikipedia articles to produce training data for large-scale relation extraction.

However, all the previous methods generated training set using a strong assumption: if an entity pair participates in a relation, then all the sentences that mention the entity pair will express the relation, which will obviously bring noisy data problem. In order to address this issue, [18] regarded the wrongly labeled data as multi-instance problem and relaxed the hypothesis: at least one sentence that mentions the entity pair might express the relation. In accordance with their assumption, they used a factor graph to make decisions and applied constraint-driven semi-supervision to train the model. [8] further relaxed the assumption by adopting multi-instance and multi-label learning models. Another fact, some researchers aimed to reduce false positive examples in training data [20], while others solved the false negative problem in training set induced by incompleteness of knowledge base [13]. Additionally, [26] developed a ranking-based approach so that the model was able to leverage inter-sentence information.

Recently, neural networks have been successfully applied to many NLP tasks, including sentiment analysis [6], text classification [25] and so on. Researchers attempted to use neural networks for relation extraction. Among them, [19] used recursive neural networks (RNN) in relation extraction while [24] utilised convolutional neural networks (CNN). Although these methods have improved the performance of relation extractors, neural networks are often deemed as black-box, and are unable to explain which feature plays a key role in the prediction. In contrast to them, here, we retain the original features (lexical and syntactic) and directly consider the correlations between features to improve the performance of feature-based methods.

5 CONCLUSION

In this paper, we propose a novel approach to utilise correlations between features for distantly supervised relation extraction. We develop the embedding method, the generative method and MSK to explore different correlations of lexical and syntactic features. After that, a distillation process is conducted to further select informative correlations obtained by the proposed method. Finally, we incorporate the captured correlations into several existing extraction models and the results demonstrate that our method is able to find the underlying correlated information between features, which effectively relaxes the deficiency of the bag-of-words representation and improves the performance of the state-of-the-art approaches.

6 ACKNOWLEDGEMENTS

This research is partially supported by Natural Science Foundation of China (Grant No. 61772356) and the Australian Research Council (Grants No. DP170101172)

REFERENCES

[1] Kedar Bellare and Andrew McCallum. 2007. Learning Extractors from Unlabeled Text using Relevant Databases. In *Sixth International Workshop on Information Integration on the Web*.

[2] Razvan C. Bunescu and Raymond J. Mooney. 2005. Subsequence kernels for relation extraction. In *Advances in Neural Information Processing Systems 18 (NIPS 2005)*. MIT Press, 171–178.

[3] Michael Collins and Nigel Duffy. 2002. New Ranking Algorithms for Parsing and Tagging: Kernels over Discrete Structures, and the Voted Perceptron. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 263–270.

[4] Mark Craven and Johan Kumlien. 1999. Learning to extract relations from the web using minimal supervision. In *Proceedings of the Sixteenth National Conference on Artificial Intelligence and Eleventh Conference on Innovative Applications of Artificial Intelligence*. AAAI, 77–86.

[5] Danilo Croce, Alessandro Moschitti, and Roberto Basili. 2011. Structured Lexical Similarity via Convolution Kernels on Dependency Trees. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 1034–1046.

[6] Cıcero Nogueira dos Santos and Maıra Gatti. 2014. Deep Convolutional Neural Networks for Sentiment Analysis of Short Texts. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics*. Association for Computational Linguistics, 69–78.

[7] Jenny Rose Finkel, Trond Grenager, and Christopher Manning. 2005. Incorporating Non-local Information into Information Extraction Systems by Gibbs Sampling. In *43rd Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 363–370.

[8] Raphael Hoffmann, Congle Zhang, Xiao Ling, Luke Zettlemoyer, and Daniel S. Weld. 2011. Knowledge-BasedWeak Supervision for Information Extraction of Overlapping Relations. In *The 49th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 541–550.

[9] Shawe-Taylor J and Cristianini N. 2004. . Cambridge university press.

[10] Nanda Kambhatla. 2004. Combining Lexical, Syntactic, and Semantic Features with Maximum Entropy Models for Extracting Relations. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics.

[11] Christina Leslie, Eleazar Eskin, Jason Weston, and William Stafford Noble. 2002. Mismatch String Kernels for SVM Protein Classification. In *Advances in Neural Information Processing Systems 15*. MIT Press, 1441–1448.

[12] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2014. Distributed Representations ofWords and Phrases and their Compositionality. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems*. Association for Computational Linguistics, 3111–3119.

[13] Bonan Min, Ralph Grishman, Li Wan, Chang Wang, and David Gondek. 2013. Distant Supervision for Relation Extraction with an Incomplete Knowledge Base. In *The 2013 Conference of the North American Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, 777–782.

[14] Mike Mintz, Steven Bills, and Rion Snow andDan Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the 47th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 1003–1011.

[15] Alessandro Moschitti. 2006. Efficient convolution kernels for dependency and constituent syntactic trees. In *17th European Conference on Machine Learning*. Springer, 318–329.

[16] Ndapandula Nakashole, Gerhard Weikum, and Fabian Suchanek. 2012. Discovering and Exploring Relations on the Web. In *Proceedings of the VLDB Endowment*. VLDB Endowment Inc., 1982–1985.

[17] Ndapandula Nakashole, Gerhard Weikum, and Fabian Suchanek. 2013. Discovering Semantic Relations from the Web and Organizing them with PATTY. In *ACM Conference on Management of Data*. ACM, 29–34.

[18] Sebastian Riedel, Limin Yao, and Andrew McCallum. 2010. Modeling Relations and Their Mentions without Labeled Text. In *Proceedings of the 2014 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*. Springer Berlin Heidelberg, 148–163.

[19] Richard Socher, Brody Huval, Christopher D. Manning, and Andrew Y. Ng. 2012. Semantic Compositionality through Recursive Matrix-Vector Spaces. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*. Association for Computational Linguistics, 1201–1211.

[20] Shingo Takamatsu, Issei Sato, and Hiroshi Nakagawa. 2012. Reducing Wrong Labels in Distant Supervision for Relation Extraction. In *The 50th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 721–729.

[21] Duyu Tang, Furu Wei, Nan Yang, Ming Zhou, Ting Liu, and Bing Qin. 2014. Learning Sentiment-SpecificWord Embedding for Twitter Sentiment Classification. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 23–25.

[22] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, and Łukasz Kaiser. 2017. Attention Is All You Need. In *Proceedings of Neural Information Processing Systems*. Association for the Advancement of Artificial Intelligence, 5998–6008.

[23] Fei Wu and Dan Weld. 2007. Autonomously semantifying Wikipedia. In *Proceedings of the Sixteenth ACM Conference on Information and Knowledge Management*. ACM, 41–50.

[24] Daojian Zeng, Kang Liu, Yubo Chen, and Jun Zhao. 2015. Distant Supervision for Relation Extraction via Piecewise Convolutional Neural Networks. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 1753–1762.

[25] Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems 28*. MIT Press, 649–657.

[26] Hao Zheng, Zhoujun Li, Senzhang Wang, Zhao Yan, and Jianshe Zhou. 2016. Aggregating Inter-Sentence Information to Enhance Relation Extraction. In *Proceedings of AAAI*. Association for the Advancement of Artificial Intelligence, 3108–3114.