BiRRE: Learning Bidirectional Residual Relation Embeddings for Supervised Hypernymy Detection

Chengyu Wang[1,2], **Xiaofeng He**[3*]
[1] School of Software Engineering, East China Normal University [2] Alibaba Group
[3] School of Computer Science and Technology, East China Normal University
chywang2013@gmail.com, hexf@cs.ecnu.edu.cn

Abstract

The hypernymy detection task has been addressed under various frameworks. Previously, the design of unsupervised hypernymy scores has been extensively studied. In contrast, supervised classifiers, especially distributional models, leverage the global contexts of terms to make predictions, but are more likely to suffer from "lexical memorization". In this work, we revisit supervised distributional models for hypernymy detection. Rather than taking embeddings of two terms as classification inputs, we introduce a representation learning framework named Bidirectional Residual Relation Embeddings (BiRRE). In this model, a term pair is represented by a BiRRE vector as features for hypernymy classification, which models the possibility of a term being mapped to another in the embedding space by hypernymy relations. A Latent Projection Model with Negative Regularization (LPMNR) is proposed to simulate how hypernyms and hyponyms are generated by neural language models, and to generate BiRRE vectors based on bidirectional residuals of projections. Experiments verify BiRRE outperforms strong baselines over various evaluation frameworks.

1 Introduction

As a type of linguistic resources, hypernymy relations refer to "is-a" relations between terms. Such relations are frequently exploited in a wide range of NLP tasks, including taxonomy induction (Mao et al., 2018), lexical entailment (Vulic et al., 2017) and Web query understanding (Wang et al., 2015).

In the NLP community, the task of hypernymy detection has been studied under various frameworks, e.g., unsupervised hypernym discovery (Roller et al., 2018; Chen et al., 2018; Chang et al., 2018), supervised hypernymy classification (Shwartz et al., 2016; Nguyen et al., 2017), graded lexical entailment (Vulic et al., 2017). To address unsupervised hypernym discovery, pattern-based and distributional approaches are two mainstream types of methods. Pattern-based approaches use Hearst patterns (Hearst, 1992) and their variants to extract hypernymy relations from texts (Kozareva and Hovy, 2010; Roller and Erk, 2016). Distributional methods employ hypernymy measures (or called scores) to predict hypernymy based on distributional vectors (Santus et al., 2014, 2017), alleviating the pattern sparsity issue. Le et al. (2019) combine Hearst patterns and hyperbolic embeddings for unsupervised hypernym detection.

Compared to unsupervised tasks, the supervised hypernymy detection task is formulated more directly, classifying a term pair as hypernymy or non-hypernymy based on two terms' representations (Yu et al., 2015; Anke et al., 2016; Nguyen et al., 2017). Although this task definition is more straightforward, the corresponding methods receive criticism because they may suffer from "lexical memorization" (Levy et al., 2015), referring to the phenomenon that they only learn whether a term is a "prototypical hypernym", rather than the actual relations between two terms. To address the problem, several methods combine other signals as inputs for hypernymy classifiers, such as dependency paths (Shwartz et al., 2016) and the WordNet concept hierarchy (Nguyen et al., 2017). Nonetheless, it is worth studying whether supervised classifiers can learn hypernymy relations purely based on distributional representations.

In this paper, we revisit supervised distributional models for hypernymy detection, and propose a representation learning framework named Bidirectional Residual Relation Embeddings (BiRRE). To handle "lexical memorization" (Levy et al., 2015), we learn a BiRRE vector for each term pair as features for the classifier, avoiding using the two terms' embeddings directly. The BiRRE vector

* Corresponding author.

models the possibility of a term being mapped to another in the embedding space by hypernymy relations, learned via existing neural language models and supervised signals of the training set. Specifically, we introduce the Latent Projection Model with Negative Regularization (LPMNR) to simulate how hypernyms and hyponyms are generated in the the embedding space. The BiRRE vectors are generated based on bidirectional residuals of projection results of LPMNR. Experiments over multiple public datasets and various evaluation frameworks prove that BiRRE outperforms strong baselines.

The rest of this paper is organized as follows. Section 2 summarizes the related work. The BiRRE framework is elaborated in Section 3, with experiments shown in Section 4. Finally, we conclude our paper and discuss the future work in Section 5.

2 Related Work

In this section, we overview related work on various tasks related to hypernymy detection. Due to space limitation, we focus on recent advances and refer readers to Wang et al. (2017a) for earlier work.

Pattern-based approaches date back to Hearst (1992), utilizing handcrafted patterns in English for text matching. An example of Hearst patterns is "[...] such as [...]". They are employed to build large-scale taxonomies (Wu et al., 2012; Faralli et al., 2019). Although Hearst patterns are fairly simple, recent studies show they are highly useful for designing hypernymy measures (Roller et al., 2018; Le et al., 2019). Other approaches aim at improving the coverage of generalized Hearst patterns by automatic pattern expansion (Kozareva and Hovy, 2010; Roller and Erk, 2016), or considering other context-rich representations (such as Heterogeneous Information Networks (Shi et al., 2019)). A potential drawback of pattern-based methods is that the recall of extraction results over specific domains is limited (Alfarone and Davis, 2015), as textual patterns are naturally sparse in the corpus.

To overcome the sparsity issue, **distributional hypernymy measures** model the degree of hypernymy within a term pair. A majority of these hypernymy measures are based on Distributional Inclusion Hypothesis (DIH) (Weeds et al., 2004), meaning that a hypernym covers a broader spectrum of contexts, compared to its hyponyms. The improvements and variants of DIH include (Santus et al., 2014; Chen et al., 2018; Chang et al., 2018) and many others. A comprehensive overview of distributional hypernymy measures can be found in Santus et al. (2017). Recently, Le et al. (2019) combine Hearst patterns and distributional vectors for hypernym detection. Additionally, the work of graded lexical entailment (Vulic et al., 2017) and cross-lingual graded lexical entailment (Vulic et al., 2019) aims at computing numerical scores, indicating the degree of hypernymy of a term pair.

For **supervised hypernymy classification**, traditional approaches employ distributional vectors of two terms as features, such as the Concat model, the Diff model, the SimDiff model (Turney and Mohammad, 2015). Recently, several approaches are proposed to learn hypernymy embeddings, considering the semantic hierarchies of concepts (Yu et al., 2015; Luu et al., 2016; Nguyen et al., 2017; Chang et al., 2018; Nickel and Kiela, 2018; Ganea et al., 2018; Rei et al., 2018; Chen et al., 2018). For example, Yu et al. (2015) learn hypernym and hyponym embeddings for a term by max-margin neural network. Nguyen et al. (2017) propose hierarchical embeddings for hypernymy classification, jointly trained over texts and the WordNet concept hierarchy. Rei et al. (2018) propose a directional similarity neural network based on word embeddings to predict the degree of hypernymy between two terms. Yet a number of models encode terms in the hyperbolic space, such as the hyperbolic Lorentz Model (Nickel and Kiela, 2018), Hyperbolic Entailment Cones (Ganea et al., 2018), and others (Le et al., 2019; Aly et al., 2019). The hyperbolic geometry is more capable of modeling the transitivity property of hypernymy. Additionally, patterns and distributional vectors can also be combined for supervised hypernymy prediction, as in Shwartz et al. (2016); Held and Habash (2019) and several systems submitted to SemEval 2018 Task 9 (Camacho-Collados et al., 2018).

Another type of supervised models can be categorized as **projection-based approaches**, which model how to map embeddings of a term to those of its hypernyms. Fu et al. (2014) is most influential, followed by a number of variants. Biemann et al. (2017); Wang et al. (2017b, 2019b) improve projection learning by considering explicit negative samples. The usage of orthogonal matrices is exploited in Wang et al. (2019a). One advantage is that they do not perform classification on two terms' embeddings directly, alleviating "lexical memoriza-

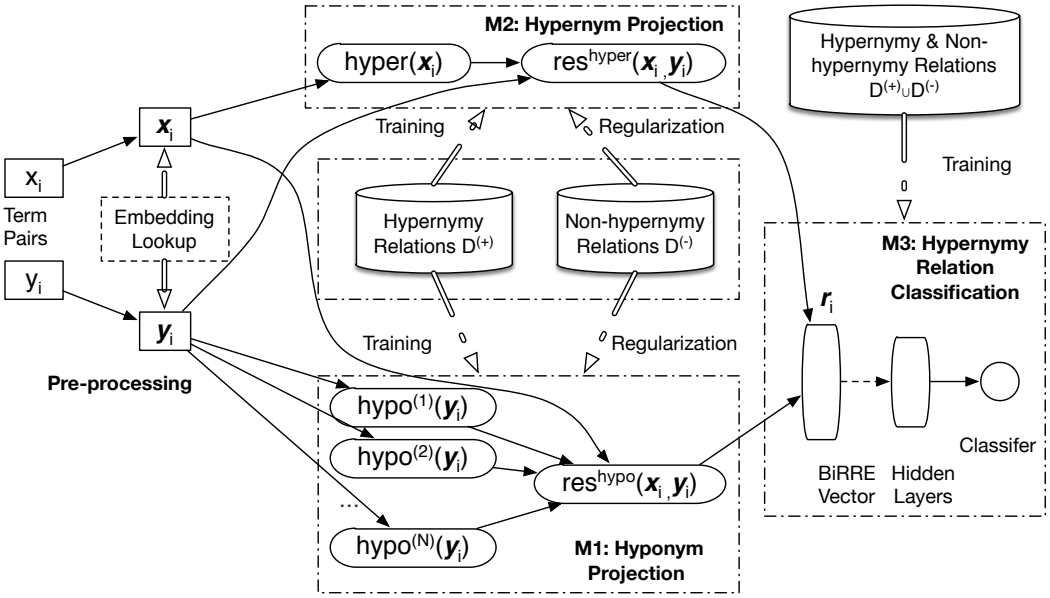


Figure 1: The BiRRE framework for supervised hypernymy detection.

tion" (Levy et al., 2015). Compared to previous work, BiRRE is supervised, but does not minimize the classification error firstly. It uses LPMNR to learn hypernym/hyponym generation process by projection learning. Hence, it takes advantages of both traditional classification and projection-based approaches.

3 The BiRRE Framework

In this section, we first introduce the task description and the BiRRE framework. The detailed steps and justifications are elaborated subsequently.

3.1 Task Description

Given two sets of term pairs: the training sets of hypernymy $D^{(+)} = \{(x_i, y_i)\}$ and non-hypernymy relations $D^{(-)} = \{(x_i, y_i)\}$, the task is to learn a classifier f to distinguish hypernymy vs. non-hypernymy relations. Particularly, y_i is a hypernym of x_i if $(x_i, y_i) \in D^{(+)}$. For non-hypernym relations, the relation types between two terms x_i and y_i in $D^{(-)}$ can be reverse-hypernymy, synonymy, antonymy, or unrelated, depending on the respective task and dataset settings.

3.2 General Framework

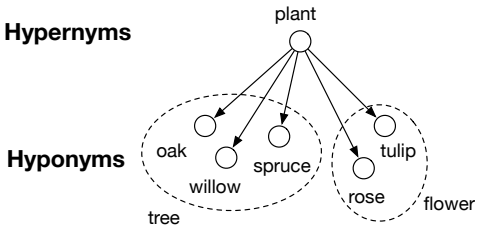
The BiRRE framework is shown in Figure 1, consisting of pre-processing and three major modules.

Pre-processing: The pre-processing step of the BiRRE framework requires minimal computation. For each term pair $(x_i, y_i) \in D^{(+)} \cup D^{(-)}$, we retrieve the corresponding embedding vectors from any neural language models (e.g., Word2Vec, GloVe), without fine-tuning. Denote normalized embeddings of x_i and y_i as $\mathbf{x}_i$ and $\mathbf{y}_i$, respectively.

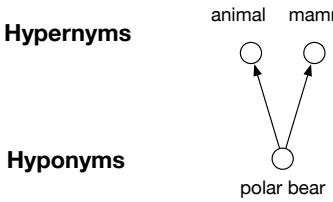
M1: The hyponym projection module learns how to map embeddings of a hypernym to those of its hyponyms. Consider the example in Figure 2. There are usually one-to-many mappings (in semantics) from hypernyms to hyponyms. Hence, we map a hypernym to its N semantically diverse hyponyms by LPMNR. We denote the N hyponym embeddings w.r.t. y_i as $hypo^{(1)}(\mathbf{y}_i), \cdots, hypo^{(N)}(\mathbf{y}_i)$[1]. Based on the difference between the true hyponym embeddings $\mathbf{x}_i$ and the N predicted hyponym embeddings , we compute the hyponym residual vector $res^{hypo}(\mathbf{x}_i, \mathbf{y}_i)$ to measure the "goodness" of mapping from $\mathbf{y}_i$ to $\mathbf{x}_i$. As shown in Biemann et al. (2017), the explicit usage of negative samples (i.e., non-hypernymy relations) improves the performance of projection learning. In this module, we take $D^{(+)}$ as the training set and $D^{(-)}$ for regularization purposes.

M2: The hypernym projection module learns how to map embeddings of a hyponym to those of its hypernyms. Based on Figure 2, such mappings tend to be simpler. Hence, we only learn one mapping model from a hyponym to embeddings of its hypernym. We denote the hypernym embeddings

[1]Because the training process is completed in the embedding space, our model learns to associate low-density hypernym regions with multiple numbers of high-density hyponym regions. Here, $\mathbf{M}^{(1)}\mathbf{y}_i, \cdots, \mathbf{M}^{(N)}\mathbf{y}_i$ may refer to the distributions of word embeddings of hyponyms, with no guarantee that they refers to actual word embeddings.



(a) Hypernym–to–hyponym mappings



(b) Hypnoym–to–hypernym mappings

Figure 2: Two types of hypernymy relation mappings.

as $hyper(\mathbf{x}_i)$. This step is learned by a simplified version of LPMNR. Similarly, we denote the hypernym residual vector as $res^{hyper}(\mathbf{x}_i, \mathbf{y}_i)$, measuring the "goodness" of mapping from $\mathbf{x}_i$ to $\mathbf{y}_i$. In this module, we also take $D^{(+)}$ as the training set and $D^{(-)}$ for regularization.

M3: Finally, the BiRRE vector (denoted as $\mathbf{r}_i$) w.r.t. (x_i, y_i) is computed by concatenating $res^{hypo}(\mathbf{x}_i, \mathbf{y}_i)$ and $res^{hyper}(\mathbf{x}_i, \mathbf{y}_i)$. A feed-forward neural network is trained over $D^{(+)}$ and $D^{(-)}$ for hypernymy relation classification. The parameters of **M3** are learned by back propagation, with parameters of **M1** and **M2** fixed in this step.

3.3 Hyponym Projection (M1)

Previously, several approaches (Fu et al., 2014; Yamane et al., 2016) assume there is a $d \times d$ projection matrix $\mathbf{M}$ such that $\mathbf{M}\mathbf{x}_i \approx \mathbf{y}_i$ where d is the dimension of word embeddings for $(x_i, y_i) \in D^{(+)}$. According to Wang et al. (2019a), the usage of orthogonal matrices has better performance for hypernymy prediction, as the the cosine similarity of $\mathbf{M}\mathbf{x}_i$ and $\mathbf{y}_i$ can be maximized when $\mathbf{M}\mathbf{x}_i$ and $\mathbf{y}_i$ are normalized.

Let $\mathcal{M} = \{\mathbf{M}^{(1)}, \cdots, \mathbf{M}^{(N)}\}$ be the parameter collection of our hyponym projection model (i.e., N $d \times d$ orthogonal projection matrices). For each hypernym y_i, these N projection matrices map $\mathbf{y}_i$ to the embeddings of N semantically diverse hyponyms $\mathbf{M}^{(1)}\mathbf{y}_i, \cdots, \mathbf{M}^{(N)}\mathbf{y}_i$. The major challenge is that the explicit semantics of N projections are unknown, and may vary across different datasets. To derive a unified solution for all scenarios, we introduce a latent variable $\theta_i^{(p)} \in (0, 1)$ to represent the weight of $(x_i, y_i) \in D^{(+)}$ w.r.t. the projection matrix $\mathbf{M}^{(p)}$ ($p \in \{1, \cdots, N\}$, $\sum_{(x_i,y_i)\in D^{(+)}} \theta_i^{(p)} = 1$). The objective of hyponym projection is as follows:[2]

$$\min_{\mathcal{M}} \sum_{(x_i,y_i)\in D^{(+)}} \sum_{p=1}^{N} \theta_{i,j}^{(p)} \|\mathbf{M}^{(p)}\mathbf{y}_i - \mathbf{x}_i\|^2 \quad \text{s. t.} \quad {\mathbf{M}^{(p)}}^T\mathbf{M}^{(p)} = \mathbf{I}_d, p \in \{1, \cdots, N\} \tag{1}$$

where $\mathbf{I}_d$ is the $d \times d$ identity matrix.

A potential drawback of Eq. (1) is that it only considers hypernymy relations $D^{(+)}$. The relation classification objective is not optimized. As Biemann et al. (2017) suggest, negative samples can of help for learning projection regularizers. The regularizers push the projected hyponym embeddings of a term further away from its non-hyponyms, making hypernymy and non-hypernymy relations more separable. Hence, we reformulate Eq. (1) as:

$$\min_{\mathcal{M}} \frac{1}{|D^{(+)}|} \sum_{(x_i,y_i)\in D^{(+)}} \sum_{p=1}^{N} \theta_i^{(p)} \|\mathbf{M}^{(p)}\mathbf{y}_i - \mathbf{x}_i\|^2 + \frac{\lambda}{|D^{(-)}|} \sum_{(x_i,y_i)\in D^{(-)}} \sum_{p=1}^{N} \phi_i^{(p)} (\mathbf{M}^{(p)}\mathbf{y}_i)^T \cdot \mathbf{x}_i \quad \text{s. t.} \quad {\mathbf{M}^{(p)}}^T\mathbf{M}^{(p)} = \mathbf{I}_d, p \in \{1, \cdots, N\} \tag{2}$$

where $\lambda > 0$ is the regularization balancing factor. The latent variable $\phi_i^{(p)} \in (0, 1)$ is the weight of the negative sample $(x_i, y_i) \in D^{(-)}$ w.r.t. $\mathbf{M}^{(p)}$. The constraint $\sum_{(x_i,y_i)\in D^{(-)}} \phi_i^{(p)} = 1$ also holds.

To the best of our knowledge, there is no standard off-the-shelf solution to Eq. (2). We slightly change the regularization term of Eq. (2). The objective function is changed as follows, which we refer as the Latent Projection Model with Negative Regularization (LPMNR):

$$\min_{\mathcal{M}} \frac{1}{|D^{(+)}|} \sum_{(x_i,y_i)\in D^{(+)}} \sum_{p=1}^{N} \theta_i^{(p)} \|\mathbf{M}^{(p)}\mathbf{y}_i - \mathbf{x}_i\|^2 - \frac{\lambda}{|D^{(-)}|} \sum_{(x_i,y_i)\in D^{(-)}} \sum_{p=1}^{N} \phi_i^{(p)} \|\mathbf{M}^{(p)}\mathbf{y}_i - \mathbf{x}_i\|^2 \quad \text{s. t.} \quad {\mathbf{M}^{(p)}}^T\mathbf{M}^{(p)} = \mathbf{I}_d, p \in \{1, \cdots, N\} \tag{3}$$

[2]For simplicity, we omit the constraints of latent variables in the objective functions in this paper.

Optimizing Eq. (3) is non-trivial due to the existence of the unknown weights $\theta_i^{(p)}$ and $\phi_i^{(p)}$. In this paper, we present a dual-iterative algorithm to solve the problem. All values of $\theta_i^{(p)}$ and $\phi_i^{(p)}$ are randomly initialized (with $\theta_i^{(p)}, \phi_i^{(p)} \in (0,1)$, $\sum_{(x_i,y_i)\in D^{(+)}} \theta_i^{(p)} = 1$ and $\sum_{(x_i,y_i)\in D^{(-)}} \phi_i^{(p)} = 1$). In each iteration, we update the values of $\theta_i^{(p)}$, $\phi_i^{(p)}$ and $\mathbf{M}^{(p)}$. When all the values of $\theta_i^{(p)}$ and $\phi_i^{(p)}$ are fixed, Eq. (3) can be regarded as a variant of the Multi-Wahba problem (Wang et al., 2019a). For simplicity, let $\alpha = \frac{\lambda|D^{(+)}|}{|D^{(-)}|}$. We extend their work and give an SVD based closed-form solution to Eq. (3) in Algorithm 1.

Algorithm 1 Closed-form Solution to Eq. (3)

1: **for** $p = 1$ to N **do**
2: $\quad \mathbf{B}^{(p)} = \sum_{(x_i,y_i)\in D^{(+)}} \theta_i^{(p)} \mathbf{x}_i \mathbf{y}_i^T - \alpha \cdot \sum_{(x_i,y_i)\in D^{(-)}} \phi_i^{(p)} \mathbf{x}_i \mathbf{y}_i^T$;
3: $\quad \mathbf{U}^{(p)} \mathbf{\Sigma}^{(p)} \mathbf{V}^{(p)T} = SVD(\mathbf{B}^{(p)})$;
4: $\quad \mathbf{R}^{(p)} = \text{diag}(\underbrace{1, \ldots, 1}_{d-1}, \det(\mathbf{U}^{(p)})\det(\mathbf{V}^{(p)}))$;
5: $\quad \mathbf{M}^{(p)} = \mathbf{U}^{(p)} \mathbf{R}^{(p)} \mathbf{V}^{(p)T}$;
6: **end for**

Proof: It is trivial to see that the optimal values of each matrix is independent from each other. Hence, we only need to optimize:

$$\min_{\mathcal{M}} \frac{1}{|D^{(+)}|} \sum_{(x_i,y_i)\in D^{(+)}} \theta_i^{(p)} \|\mathbf{M}^{(p)}\mathbf{y}_i - \mathbf{x}_i\|^2 - \frac{\lambda}{|D^{(-)}|} \sum_{(x_i,y_i)\in D^{(-)}} \phi_i^{(p)} \|\mathbf{M}^{(p)}\mathbf{y}_i - \mathbf{x}_i\|^2$$
$$\text{s. t.} \quad {\mathbf{M}^{(p)}}^T \mathbf{M}^{(p)} = \mathbf{I}_d$$

For simplicity, let $\alpha = \frac{\lambda|D^{(+)}|}{|D^{(-)}|}$, with the superscript (p) omitted. The problem can be transformed as:

$$J(\mathbf{M}) = \sum_{(x_i,y_i)\in D^{(+)}} \theta_i \|\mathbf{M}\mathbf{y}_i - \mathbf{x}_i\|^2 - \alpha \cdot \sum_{(x_i,y_i)\in D^{(-)}} \phi_i \|\mathbf{M}\mathbf{y}_i - \mathbf{x}_i\|^2$$
$$\text{s. t.} \quad \mathbf{M}^T\mathbf{M} = \mathbf{I}_d$$

Define the matrix $\mathbf{B} = \sum_{(x_i,y_i)\in D^{(+)}} \theta_i \mathbf{x}_i \mathbf{y}_i^T - \alpha \cdot \sum_{(x_i,y_i)\in D^{(-)}} \phi_i \mathbf{x}_i \mathbf{y}_i^T$. We re-write the objective function as: $J(\mathbf{M}) = 1 - \text{tr}(\mathbf{M}\mathbf{B}^T)$. Hence, we have transformed the problem into the Multi-Wahba problem (Wang et al., 2019a). $J(\mathbf{M})$ is minimized when the optimal value of $\mathbf{M}$ is:

$$\mathbf{M}^* = \mathbf{U}\text{diag}(\underbrace{1, \ldots, 1}_{d-1}, \det(\mathbf{U})\det(\mathbf{V}))\mathbf{V}^T$$

with $\mathbf{U}\mathbf{\Sigma}\mathbf{V}^T = SVD(\mathbf{B})$. ■

After optimal values of $\mathbf{M}^{(p)}$ are computed, the values of all $\|\mathbf{M}^{(p)}\mathbf{y}_i - \mathbf{x}_i\|^2$ are known. In this condition, we fix the values of $\mathbf{M}^{(p)}$ and update all $\theta_i^{(p)}$ and $\phi_i^{(p)}$. We turn the problem of minimizing Eq. (3) into the following problems:

$$\min_{\theta_i^{(p)}} \sum_{(x_i,y_i)\in D^{(+)}} \|\mathbf{M}^{(p)}\mathbf{y}_i - \mathbf{x}_i\|^2 \cdot \theta_i^{(p)} \quad (4)$$

$$\max_{\phi_i^{(p)}} \sum_{(x_i,y_i)\in D^{(-)}} \|\mathbf{M}^{(p)}\mathbf{y}_i - \mathbf{x}_i\|^2 \cdot \phi_i^{(p)} \quad (5)$$

We update $\theta_i^{(p)}$ and $\phi_i^{(p)}$ by constrained gradient descent where the updating formulas are:

$$\theta_i^{(p)*} = \theta_i^{(p)} - \eta \cdot \sum_{(x_i,y_i)\in D^{(+)}} \|\mathbf{M}^{(p)}\mathbf{y}_i - \mathbf{x}_i\|^2 \quad (6)$$

$$\phi_i^{(p)*} = \phi_i^{(p)} + \eta \cdot \sum_{(x_i,y_i)\in D^{(-)}} \|\mathbf{M}^{(p)}\mathbf{y}_i - \mathbf{x}_i\|^2 \quad (7)$$

where $\eta > 0$ is the learning rate (a small decimal). $\theta_i^{(p)*}$ and $\phi_i^{(p)*}$ are updated values of $\theta_i^{(p)}$ and $\phi_i^{(p)}$ for the new iteration, respectively. After the update of all weights, we normalize the weights to satisfy: $\sum_{(x_i,y_i)\in D^{(+)}} \theta_i^{(p)} = 1$ and $\sum_{(x_i,y_i)\in D^{(-)}} \phi_i^{(p)} = 1$. The iterative procedure continues until convergence, with the algorithm summarized in Algorithm 2.

After training, given $\mathbf{x}_j$, $\mathbf{M1}$ outputs N hyponym embeddings: $hypo^{(1)}(\mathbf{y}_i) = \mathbf{M}^{(1)}\mathbf{y}_i, \cdots, hypo^{(N)}(\mathbf{y}_i) = \mathbf{M}^{(N)}\mathbf{y}_i$. We define the hyponym residual vector $res^{hypo}(\mathbf{x}_i, \mathbf{y}_i)$ as follows:

$$res^{hypo}(\mathbf{x}_i, \mathbf{y}_i)) = \mathbf{x}_i - \mathbf{M}^{(\tilde{p})}\mathbf{y}_i$$

where $\tilde{p}$ is the index of the selected projection matrix that best fits for $(x_i, y_i) \in D^{(+)}$. We set $\tilde{p}$ empirically as: $\tilde{p} = \text{argmin}_p \|\mathbf{x}_i - \mathbf{M}^{(p)}\mathbf{y}_i\|_2$.

Based on the objective in Eq. (3), if $(x_i, y_i) \in D^{(+)}$, $\|res^{hypo}(\mathbf{x}_i, \mathbf{y}_i)\|_2$ tends to be small. Otherwise, $\|res^{hypo}(\mathbf{x}_i, \mathbf{y}_i)\|_2$ would be large. Hence, it is discriminative for hypernymy classification.

Algorithm 2 Optimization Algorithm for Eq. (3)

Randomly initialize all $\theta_i^{(p)}$ and $\phi_i^{(p)}$;
while Eq. (3) does not converge **do**
  Compute $\mathcal{M}$ by Algorithm 1;
  for $p = 1$ to N **do**
    while Eq. (4) does not converge **do**
      Update and normalize $\theta_{i,j}^{(p)}$ by Eq. (6);
    end while
    while Eq. (5) does not converge **do**
      Update and normalize $\phi_{i,j}^{(p)}$ by Eq. (7);
    end while
  end for
end while

Algorithm 3 Training Algorithm of BiRRE

Learn N hyponym projection matrices $\mathcal{M}$;
Learn hypernym projection matrix $\mathbf{Q}$;
for each $(x_i, y_i) \in D^{(+)} \cup D^{(-)}$ **do**
  Compute the BiRRE vector $\mathbf{r}_i$ by Eq. (8);
end for
Train the hypernymy classifier f over $D^{(+)} \cup D^{(-)}$ using $\mathbf{r}_i$ as features;

3.4 Hypernym Projection (M2)

The hypernym projection module can be regarded as a simplified version of the previous module. Denote $\mathbf{Q}$ as the $d \times d$ projection matrix. The objective of hypernym projection is formulated as follows:

$$\min_{\mathbf{Q}} \frac{1}{|D^{(-)}|} \sum_{(x_i,y_i)\in D^{(-)}} \|\mathbf{Q}\mathbf{x}_i - \mathbf{y}_i\|^2 - \frac{\lambda}{|D^{(+)}|} \sum_{(x_i,y_i)\in D^{(+)}} \|\mathbf{Q}\mathbf{x}_i - \mathbf{y}_i\|^2 \text{ s. t. } \mathbf{Q}^T\mathbf{Q} = \mathbf{I}_d$$

It can be solved by Algorithm 1 with weights reduced and $N = 1$. Similar to hyponym projection, we compute the hypernym residual vector $res^{hyper}(\mathbf{x}_i, \mathbf{y}_i)$ as follows:

$$res^{hyper}(\mathbf{x}_i) = \mathbf{Q}\mathbf{x}_i - \mathbf{y}_i$$

3.5 Hypernymy Relation Classification (M3)

For each pair $(x_i, y_i) \in D^{(+)} \cup D^{(-)}$, we generate the BiRRE vector $\mathbf{r}_i$ via the concatenation of two residual vectors:

$$\mathbf{r}_i = res^{hypo}(\mathbf{x}_i, \mathbf{y}_i) \oplus res^{hyper}(\mathbf{x}_i, \mathbf{y}_i) \quad (8)$$

A feed forward neural network is trained for hypernymy vs. non-hypernymy classification over $D^{(+)}$ and $D^{(-)}$ using $\mathbf{r}_i$ as features. To this end, we summarize the high-level training process of BiRRE, as shown in Algorithm 3. There can be zero, one or multiple hidden layers in the neural network. The detailed study of network structures will be discussed in the experiments.

3.6 Discussion

Orthogonal projections have been applied to predict various types of word relations (Ethayarajh, 2019). However, the mechanisms behind orthogonal projections in the embedding space for predicting such relations can not be fully explained by NLP researchers. In BiRRE, we use different numbers of matrices in **M1** and **M2**, in order to capture the mappings between hypernyms and hyponyms. Due to the complicated nature of linguistics, such projections are not 100% correct. Hence, we learn the residual vectors and train a classifier (in **M3**) to decide which dimensions learned by **M1** and **M2** are best predictors for hypernymy relations. Therefore, the performance of BiRRE can be improved.

4 Experiments

In this section, we conduct extensive experiments to evaluate the BiRRE model over various benchmarks. We also compare it with state-of-the-art to show its effectiveness.

4.1 Experimental Settings

The default word embeddings used by our model are pre-trained by the fastText model (Bojanowski et al., 2017) over the English Wikipedia corpus of version December 2019. We train the model by ourselves using their original codes. The embedding size is set as $d = 300$, according to their paper. In the implementation, the parameters η and N are set to 10^{-3} and $\max\{1, \lfloor \lg |D^{(+)}| \rfloor\}$ (an empirical formula), respectively. We also tune the model parameters in subsequent experiments. The neural network in M3 is fully connected and trained via the Adam algorithm with the dropout rate to be 0.1.

4.2 Experiment 1: Effectiveness of BiRRE

We use the largest hypernymy relation dataset (to our knowledge) from Shwartz et al. (2016) to test the effectiveness of BiRRE. It is created from various resources: WordNet, DBPedia, Wikidata and YAGO, and divided into random split and lexical split. Especially, the lexical split forces training, testing and validation sets contain distinct vocabularies, disabling "lexical memorization" (Levy

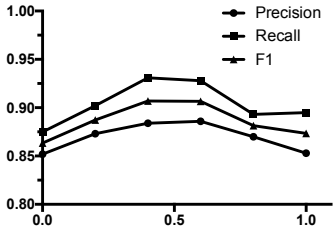
Method	Precision	Recall	F1	Precision	Recall	F1
	Random Split			Lexical Split		
Roller and Erk (2016)	0.926	0.850	0.886	0.700	**0.964**	0.811
Shwartz et al. (2016)	0.913	0.890	0.901	0.809	0.617	0.700
Glavas and Ponzetto (2017)	0.933	0.826	0.876	0.705	0.785	0.743
Rei et al. (2018)	0.928	0.887	0.907	0.826	0.860	0.842
BiRRE	**0.945**	**0.932**	**0.938**	**0.880**	0.918	**0.898**

Table 1: Performance of different approaches over the dataset (Shwartz et al., 2016).

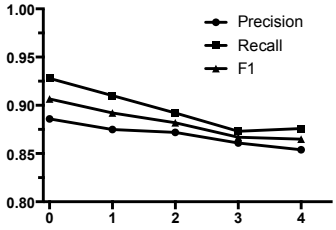
et al., 2015). We follow the same evaluation steps of Shwartz et al. (2016); Rei et al. (2018) and report the results in Table 1. The network structure and parameters are tuned over the validation set.

Based on the results, BiRRE consistently outperforms state-of-the-art by 3.1% and 5.6% in terms of F1. Additionally, the performance gap between lexical and random splits has been narrowed down from 6.5% (Rei et al., 2018) to 4.0% (BiRRE). It shows that BiRRE alleviates "lexical memorization", compared to other distributional models. We also conduct pairwise statistical tests between Rei et al. (2018) and our outputs. It shows that BiRRE outperforms the approach significantly.

We tune the value of λ from 0.0 to 1.0 using the development set. The results over the lexical spilt of the dataset (Shwartz et al., 2016) are shown in Figure 3(a). Bigger λ means a larger effect of negative regularization. As seen, the usage of negative regularization improves the prediction performance by a large margin. A suitable choice of λ is generally around 0.4 to 0.6. As for the neural network structures, the number of hidden nodes does not have a large impact on the model performance. Hence, we only report the results when we use the same number of nodes in hidden layers as the dimension of word embeddings d, shown in Figure 3(b). Our results are consistent with previous research, which show that adding more hidden layers can decrease the prediction accuracy, leading to model overfitting.



(a) Tuning λ



(b) Varying network depth

Figure 3: Tuning parameter λ and the depth of the neural network classifier for BiRRE.

4.3 Experiment 2: Supervised Hypernymy Classification

We evaluate BiRRE over two benchmark datasets: BLESS (Baroni and Lenci, 2011) and ENTAILMENT (Baroni et al., 2012), consisting of 14,547 and 2,770 labeled term pairs, respectively. For evaluation, we follow the same "leave-one-out" evaluation protocols as used in previous research (Yu et al., 2015; Luu et al., 2016; Nguyen et al., 2017). All the experimental results are reported in terms of averaged accuracy. Because the two datasets do not have separate validation sets, we take the dataset (Shwartz et al., 2016) to tune parameters of BiRRE. To prevent "data leakage", we exclude all the data of the validation set that also appear in the test set for parameter tuning. We compare BiRRE against several previous supervised models (Mikolov et al., 2013; Yu et al., 2015; Luu et al., 2016; Nguyen et al., 2017; Wang et al., 2019a). [3]

The averaged accuracy scores of all these methods are shown in Table 2. From the results, we can see that our model outperforms all previous baseline approaches, having the averaged accuracy of 98% and 93%, respectively. We also conduct the paired t-test, which shows that BiRRE sig-

[3] We have also considered SemEval 2018 Task 9 (Camacho-Collados et al., 2018) for evaluation. However, this task focuses on the complete process of retrieving (or discovering) hypernyms for input terms from specific corpora. Hence, it is not suitable to evaluate BiRRE directly.

Method	BLESS	ENT.
Mikolov et al. (2013)	0.84	0.83
Yu et al. (2015)	0.90	0.87
Luu et al. (2016)	0.93	0.91
Nguyen et al. (2017)	0.94	0.91
Wang et al. (2019a)	0.97	0.92
BiRRE	**0.98**	**0.93**

Table 2: Performance comparison for supervised hypernymy classification in terms of averaged accuracy. ENT. stands for ENTAILMENT.

Feature Set	BLESS	ENT.	Shwartz
$\mathbf{x}_i + \mathbf{y}_i$	0.76	0.77	0.72
$\mathbf{x}_i - \mathbf{y}_i$	0.79	0.74	0.73
$\mathbf{x}_i \oplus \mathbf{y}_i$	0.81	0.80	0.77
$res^{hypo}(\mathbf{x}_i, \mathbf{y}_i)$	0.92	0.87	0.84
$res^{hyper}(\mathbf{x}_i, \mathbf{y}_i)$	0.89	0.84	0.82
$\mathbf{r}_i$ (i.e., BiRRE)	**0.99**	**0.93**	**0.88**

Table 3: Ablation study results of BiRRE in terms of averaged accuracy. ENT. stands for ENTAILMENT.

nificantly outperforms classical models (Mikolov et al., 2013). Compared to the strongest competitor (Wang et al., 2019a), the accuracy of our model is also higher by 1%.

4.4 Experiment 3: Ablation Study of BiRRE

We further study the effectiveness of individual residual vectors for hypernymy classification and conduct the following ablation study. Each time, we only use a unidirectional residual vector as features (i.e., $res^{hypo}(\mathbf{x}_i, \mathbf{y}_i)$ and $res^{hyper}(\mathbf{x}_i, \mathbf{y}_i)$). Additionally, we follow several previous papers (Yu et al., 2015; Luu et al., 2016; Nguyen et al., 2017), using the addition, offset and concatenation of embedding vectors as features (i.e., $\mathbf{x}_i + \mathbf{y}_i$, $\mathbf{x}_i - \mathbf{y}_i$ and $\mathbf{x}_i \oplus \mathbf{y}_i$ to train the neural networks for hypernymy classification. These three models are treated as naive baselines. The experimental settings are the same as in Experiments 1 and 2.

The experimental results over BLESS (Baroni and Lenci, 2011), ENTAILMENT (Baroni et al., 2012) and the lexical split of the dataset (Shwartz et al., 2016) are illustrated in Table 3. We have the following three observations. i) Traditional models using $\mathbf{x}_i + \mathbf{y}_i$, $\mathbf{x}_i - \mathbf{y}_i$ and $\mathbf{x}_i \oplus \mathbf{y}_i$ as features do not yield satisfactory results. The most likely cause is that they suffer from the "lexical memorization" problem. ii) The hyponym residual vector $res^{hypo}(\mathbf{x}_i, \mathbf{y}_i)$ is slightly more effective than the hypernym residual vector $res^{hyper}(\mathbf{x}_i, \mathbf{y}_i)$. It means that the more complicated hyponym generation process is more precise and suitable for our task. iii) By combining $res^{hypo}(\mathbf{x}_i, \mathbf{y}_i)$ and $res^{hyper}(\mathbf{x}_i, \mathbf{y}_i)$, the proposed approach is more effective and outperforms previous methods.

4.5 Experiment 4: Hypernym Discovery

Yet another widely used evaluation framework is hypernym discovery, including three subtasks: i) ranked hyernym detection, ii) hyernymy direction classification and iii) graded lexical entailment, as presented in Nguyen et al. (2017); Roller et al. (2018); Le et al. (2019) and many others. These subtasks require algorithms to output unsupervised scores (or measures), indicating the level of hypernymy within a term pair. Therefore, this framework is not directly applicable to evaluate BiRRE.

We evaluate BiRRE on hypernym discovery by external supervision. For ranked hyernym detection, following Roller et al. (2018); Le et al. (2019), we consider five test sets: BLESS (Baroni and Lenci, 2011), EVAL (Santus et al., 2015), LEDS (Baroni et al., 2012), SHWARTZ (Shwartz et al., 2016) and WBLESS (Weeds et al., 2014). For each test set, we use the remaining four datasets (excluding all term pairs in the current test set) to train and tune the BiRRE model. For each term in the test set, we create a ranked list of candidate hypernyms by placing positive predictions over negative. Next, for candidate hypernyms with the same relation label, we rank them by norms of BiRRE vectors to produce the final ranked list.

For the hypernymy direction classification subtask, we use three test sets: BLESS (Baroni and Lenci, 2011), WBLESS (Weeds et al., 2014) and BIBLESS (Kiela et al., 2015). Because this subtask is directly evaluated in terms of accuracy, we train the supervised BiRRE model using the external dataset (Shwartz et al., 2016) (also excluding term overlaps) and report the performance. Another subtask evaluated in Roller et al. (2018); Le et al. (2019) is graded lexical entailment (Vulic et al., 2017). Because BiRRE only produces discrete outputs, how BiRRE can be adapted for graded lexical entailment is left as future work.

The experimental results are summarized in Table 4. For comparison, we take three recent models (Nguyen et al., 2017; Roller et al., 2018; Le et al., 2019) as strong baselines. Due to space limitation, for Roller et al. (2018), we only list the

Method	BLESS	EVAL	LEDS	SHWARTZ	WBLESS
Task: Ranked Hyernym Detection (Average Precision)					
Nguyen et al. (2017)	0.45	0.54	-	-	0.85
Roller et al. (2018)	0.76	0.48	0.84	0.44	0.96
Le et al. (2019)	0.81	0.50	**0.89**	0.50	**0.98**
BiRRE	**0.87**	**0.56**	0.88	**0.56**	**0.98**
Method	**BLESS**	**WBLESS**	**BIBLESS**		
Task: Hyernymy Direction Classification (Accuracy)					
Nguyen et al. (2017)	0.92	0.87	0.81		
Roller et al. (2018)	0.96	0.87	0.85		
Le et al. (2019)	0.94	0.90	0.87		
BiRRE	**0.98**	**0.95**	**0.92**		

Table 4: Experimental results of ranked hyernym detection and hyernymy direction classification.

scores generated by "spmi(x, y)" due to its superiority. We can see that BiRRE consistently outperforms baselines over most of the datasets. As for LEDS and WBLESS, the results of BiRRE and the state-of-the-art (Le et al., 2019) are comparable. Hence, our supervised distributional model BiRRE can also address hypernym discovery, previously addressed by unsupervised hypernymy scores.

We need to claim that models in Table 4 use different knowledge sources (either patterns or distributional vectors) for parameter learning. Strictly speaking, the gaps of scores in this set of tasks do not necessarily reflect which method is better in all situations. It still remains an open question that how to evaluate all types of methods related to hypernymy detection in a unified framework.

4.6 Experiment 5: Choice of Different Word Embeddings

We also test our model using other types of word embeddings. We consider two other types of traditional word embeddings: Word2Vec (Mikolov et al., 2013) and GloVe (Pennington et al., 2014), as well as BERT (Devlin et al., 2019) representations without contexts [4]. Experiments are conducted over the same datasets as used in Experiment 3. The results are shown in Table 5, in terms of accuracy. As shown, the effect of fastText (Bojanowski et al., 2017) is slightly better than Word2Vec and GloVe. The representations of BERT do not yield satisfactory performance, probably due to the fact that the dimensionality of BERT is higher than other models, making the number of parameters in BiRRE too large to be learned. Note that the study of deep neural language models is beyond the scope of this paper, which can be explored in the future.

[4]The dimensions of Word2Vec and GloVe are the same as fastText. The pre-trained BERT model we use is Google's base model, released at https://github.com/google-research/bert.

Word Embed.	BLESS	ENT.	Shwartz
Word2Vec	0.94	0.90	0.82
GloVe	0.96	0.88	0.83
BERT	0.87	0.85	0.77

Table 5: The performance of BiRRE using other word embeddings. ENT. stands for ENTAILMENT.

5 Conclusion and Future Work

In this paper, we present the BiRRE model for supervised hypernymy detection. It employs two projection-based hypernym and hyponym generation modules based on word embeddings to learn BiRRE vectors for hypernymy classification. Experimental results show that BiRRE outperforms state-of-the-arts over various benchmark datasets.

Future work includes i) improving projection learning to model complicated linguistic properties of hypernymy; ii) extending our model to address other tasks, such as graded lexical entailment (Vulic et al., 2017) and cross-lingual graded lexical entailment (Vulic et al., 2019); and iii) exploring how deep neural language models (such as BERT (Devlin et al., 2019), Transformer-XL (Dai et al., 2019), XLNet (Yang et al., 2019)) can improve the performance of hypernymy detection.

Acknowledgements

We would like to thank anonymous reviewers for their valuable comments. This work is supported by the National Key Research and Development Program of China under Grant No. 2016YFB1000904.

References

Daniele Alfarone and Jesse Davis. 2015. Unsupervised learning of an IS-A taxonomy from a limited domain-specific corpus. In *IJCAI*, pages 1434–1441.

Rami Aly, Shantanu Acharya, Alexander Ossa, Arne Köhn, Chris Biemann, and Alexander Panchenko. 2019. Every child should have parents: A taxonomy refinement algorithm based on hyperbolic term embeddings. In *ACL*, pages 4811–4817.

Luis Espinosa Anke, José Camacho-Collados, Claudio Delli Bovi, and Horacio Saggion. 2016. Supervised distributional hypernym discovery via domain adaptation. In *EMNLP*, pages 424–435.

Marco Baroni, Raffaella Bernardi, Ngoc-Quynh Do, and Chung-chieh Shan. 2012. Entailment above the word level in distributional semantics. In *EACL*, pages 23–32.

Marco Baroni and Alessandro Lenci. 2011. How we blessed distributional semantic evaluation. In *GEMS*, pages 1–10.

Chris Biemann, Dmitry Ustalov, Alexander Panchenko, and Nikolay Arefyev. 2017. Negative sampling improves hypernymy extraction based on projection learning. In *EACL*, pages 543–550.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *TACL*, 5:135–146.

José Camacho-Collados, Claudio Delli Bovi, Luis Espinosa Anke, Sergio Oramas, Tommaso Pasini, Enrico Santus, Vered Shwartz, Roberto Navigli, and Horacio Saggion. 2018. Semeval-2018 task 9: Hypernym discovery. In *SemEval@NAACL-HLT*, pages 712–724.

Haw-Shiuan Chang, ZiYun Wang, Luke Vilnis, and Andrew McCallum. 2018. Distributional inclusion vector embedding for unsupervised hypernymy detection. In *NAACL*, pages 485–495.

Hong-You Chen, Cheng-Syuan Lee, Keng-Te Liao, and Shou-de Lin. 2018. Word relation autoencoder for unseen hypernym extraction using word embeddings. In *EMNLP*, pages 4834–4839.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime G. Carbonell, Quoc Viet Le, and Ruslan Salakhutdinov. 2019. Transformer-xl: Attentive language models beyond a fixed-length context. In *ACL*, pages 2978–2988.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *NAACL*, pages 4171–4186.

Kawin Ethayarajh. 2019. Rotate king to get queen: Word relationships as orthogonal transformations in embedding space. In *EMNLP-IJCNLP*, pages 3501–3506.

Stefano Faralli, Irene Finocchi, Simone Paolo Ponzetto, and Paola Velardi. 2019. Webisagraph: A very large hypernymy graph from a web corpus. In *CLIC-IT*.

Ruiji Fu, Jiang Guo, Bing Qin, Wanxiang Che, Haifeng Wang, and Ting Liu. 2014. Learning semantic hierarchies via word embeddings. In *ACL*, pages 1199–1209.

Octavian-Eugen Ganea, Gary Bécigneul, and Thomas Hofmann. 2018. Hyperbolic entailment cones for learning hierarchical embeddings. In *ICML*, pages 1632–1641.

Goran Glavas and Simone Paolo Ponzetto. 2017. Dual tensor model for detecting asymmetric lexico-semantic relations. In *EMNLP*, pages 1757–1767.

Marti A. Hearst. 1992. Automatic acquisition of hyponyms from large text corpora. In *COLING*, pages 539–545.

William Held and Nizar Habash. 2019. The effectiveness of simple hybrid systems for hypernym discovery. In *ACL*, pages 3362–3367.

Douwe Kiela, Laura Rimell, Ivan Vulic, and Stephen Clark. 2015. Exploiting image generality for lexical entailment detection. In *ACL-IJCNLP*, pages 119–124.

Zornitsa Kozareva and Eduard H. Hovy. 2010. A semi-supervised method to learn and construct taxonomies using the web. In *EMNLP*, pages 1110–1118.

Matt Le, Stephen Roller, Laetitia Papaxanthos, Douwe Kiela, and Maximilian Nickel. 2019. Inferring concept hierarchies from text corpora via hyperbolic embeddings. In *ACL*, pages 3231–3241.

Omer Levy, Steffen Remus, Chris Biemann, and Ido Dagan. 2015. Do supervised distributional methods really learn lexical inference relations? In *NAACL*, pages 970–976.

Anh Tuan Luu, Yi Tay, Siu Cheung Hui, and See-Kiong Ng. 2016. Learning term embeddings for taxonomic relation identification using dynamic weighting neural network. In *EMNLP*, pages 403–413.

Yuning Mao, Xiang Ren, Jiaming Shen, Xiaotao Gu, and Jiawei Han. 2018. End-to-end reinforcement learning for automatic taxonomy induction. In *ACL*, pages 2462–2472.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. In *ICLR*.

Kim Anh Nguyen, Maximilian Köper, Sabine Schulte im Walde, and Ngoc Thang Vu. 2017. Hierarchical embeddings for hypernymy detection and directionality. In *EMNLP*, pages 233–243.

Maximilian Nickel and Douwe Kiela. 2018. Learning continuous hierarchies in the lorentz model of hyperbolic geometry. In *ICML*, pages 3776–3785.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *EMNLP*, pages 1532–1543.

Marek Rei, Daniela Gerz, and Ivan Vulic. 2018. Scoring lexical entailment with a supervised directional similarity network. In *ACL*, pages 638–643.

Stephen Roller and Katrin Erk. 2016. Relations such as hypernymy: Identifying and exploiting hearst patterns in distributional vectors for lexical entailment. In *EMNLP*, pages 2163–2172.

Stephen Roller, Douwe Kiela, and Maximilian Nickel. 2018. Hearst patterns revisited: Automatic hypernym detection from large text corpora. In *ACL*, pages 358–363.

Enrico Santus, Alessandro Lenci, Qin Lu, and Sabine Schulte im Walde. 2014. Chasing hypernyms in vector spaces with entropy. In *EACL*, pages 38–42.

Enrico Santus, Vered Shwartz, and Dominik Schlechtweg. 2017. Hypernyms under siege: Linguistically-motivated artillery for hypernymy detection. In *EACL*, pages 65–75.

Enrico Santus, Frances Yung, Alessandro Lenci, and Chu-Ren Huang. 2015. Evalution 1.0: an evolving semantic dataset for training and evaluation of distributional semantic models. In *LDL@IJCNLP*, pages 64–69.

Yu Shi, Jiaming Shen, Yuchen Li, Naijing Zhang, Xinwei He, Zhengzhi Lou, Qi Zhu, Matthew Walker, Myunghwan Kim, and Jiawei Han. 2019. Discovering hypernymy in text-rich heterogeneous information network by exploiting context granularity. In *CIKM*, pages 599–608.

Vered Shwartz, Yoav Goldberg, and Ido Dagan. 2016. Improving hypernymy detection with an integrated path-based and distributional method. In *ACL*, pages 2389–2398.

Peter D. Turney and Saif M. Mohammad. 2015. Experiments with three approaches to recognizing lexical entailment. *NLE*, 21(3):437–476.

Ivan Vulic, Daniela Gerz, Douwe Kiela, Felix Hill, and Anna Korhonen. 2017. Hyperlex: A large-scale evaluation of graded lexical entailment. *Computational Linguistics*, 43(4).

Ivan Vulic, Simone Paolo Ponzetto, and Goran Glavas. 2019. Multilingual and cross-lingual graded lexical entailment. In *ACL*, pages 4963–4974.

Chengyu Wang, Yan Fan, Xiaofeng He, and Aoying Zhou. 2019a. A family of fuzzy orthogonal projection models for monolingual and cross-lingual hypernymy prediction. In *WWW*, pages 1965–1976.

Chengyu Wang, Xiaofeng He, and Aoying Zhou. 2017a. A short survey on taxonomy learning from text corpora: Issues, resources and recent advances. In *EMNLP*, pages 1190–1203.

Chengyu Wang, Xiaofeng He, and Aoying Zhou. 2019b. Improving hypernymy prediction via taxonomy enhanced adversarial learning. In *AAAI*, pages 7128–7135.

Chengyu Wang, Junchi Yan, Aoying Zhou, and Xiaofeng He. 2017b. Transductive non-linear learning for chinese hypernym prediction. In *ACL*, pages 1394–1404.

Zhongyuan Wang, Kejun Zhao, Haixun Wang, Xiaofeng Meng, and Ji-Rong Wen. 2015. Query understanding through knowledge-based conceptualization. In *IJCAI*, pages 3264–3270.

Julie Weeds, Daoud Clarke, Jeremy Reffin, David J. Weir, and Bill Keller. 2014. Learning to distinguish hypernyms and co-hyponyms. In *COLING*, pages 2249–2259.

Julie Weeds, David J. Weir, and Diana McCarthy. 2004. Characterising measures of lexical distributional similarity. In *COLING*, pages 1015–1021.

Wentao Wu, Hongsong Li, Haixun Wang, and Kenny Qili Zhu. 2012. Probase: a probabilistic taxonomy for text understanding. In *SIGMOD*, pages 481–492.

Josuke Yamane, Tomoya Takatani, Hitoshi Yamada, Makoto Miwa, and Yutaka Sasaki. 2016. Distributional hypernym generation by jointly learning clusters and projections. In *COLING*, pages 1871–1879.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *NeurIPS*, pages 5754–5764.

Zheng Yu, Haixun Wang, Xuemin Lin, and Min Wang. 2015. Learning term embeddings for hypernymy identification. In *IJCAI*, pages 1390–1397.