# Learning Semantic Hierarchies via Word Embeddings

**Ruiji Fu**[†], **Jiang Guo**[†], **Bing Qin**[†], **Wanxiang Che**[†], **Haifeng Wang**[‡], **Ting Liu**[†*]
[†]Research Center for Social Computing and Information Retrieval
Harbin Institute of Technology, China
[‡]Baidu Inc., Beijing, China
{rjfu, jguo, bqin, car, tliu}@ir.hit.edu.cn
wanghaifeng@baidu.com

## Abstract

Semantic hierarchy construction aims to build structures of concepts linked by hypernym–hyponym ("is-a") relations. A major challenge for this task is the automatic discovery of such relations. This paper proposes a novel and effective method for the construction of semantic hierarchies based on word embeddings, which can be used to measure the semantic relationship between words. We identify whether a candidate word pair has hypernym–hyponym relation by using the word-embedding-based semantic projections between words and their hypernyms. Our result, an F-score of 73.74%, outperforms the state-of-the-art methods on a manually labeled test dataset. Moreover, combining our method with a previous manually-built hierarchy extension method can further improve F-score to 80.29%.

## 1 Introduction

Semantic hierarchies are natural ways to organize knowledge. They are the main components of ontologies or semantic thesauri (Miller, 1995; Suchanek et al., 2008). In the WordNet hierarchy, senses are organized according to the "is-a" relations. For example, "dog" and "canine" are connected by a directed edge. Here, "canine" is called a hypernym of "dog." Conversely, "dog" is a hyponym of "canine." As key sources of knowledge, semantic thesauri and ontologies can support many natural language processing applications. However, these semantic resources are limited in its scope and domain, and their manual construction is knowledge intensive and time consuming. Therefore, many researchers have attempted to automatically extract semantic relations or to construct taxonomies.

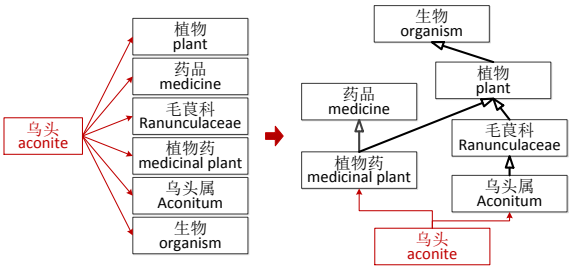


Figure 1: An example of semantic hierarchy construction.

A major challenge for this task is the automatic discovery of hypernym-hyponym relations. Fu et al. (2013) propose a distant supervision method to extract hypernyms for entities from multiple sources. The output of their model is a list of hypernyms for a given enity (left panel, Figure 1). However, there usually also exists hypernym–hyponym relations among these hypernyms. For instance, "植物 (plant)" and "毛茛科 (Ranunculaceae)" are both hypernyms of the entity "乌头 (aconit)," and "植物 (plant)" is also a hypernym of "毛茛科 (Ranunculaceae)." Given a list of hypernyms of an entity, our goal in the present work is to construct a semantic hierarchy of these hypernyms (right panel, Figure 1).[1]

Some previous works extend and refine manually-built semantic hierarchies by using other resources (e.g., Wikipedia) (Suchanek et al., 2008). However, the coverage is limited by the scope of the resources. Several other works relied heavily on lexical patterns, which would suffer from deficiency because such patterns can only cover a small proportion of complex linguistic circumstances (Hearst, 1992; Snow et al., 2005).

*Email correspondence.

[1]In this study, we focus on Chinese semantic hierarchy construction. The proposed method can be easily adapted to other languages.



*Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, pages 1199–1209,
Baltimore, Maryland, USA, June 23-25 2014. 

Besides, distributional similarity methods (Kotlerman et al., 2010; Lenci and Benotto, 2012) are based on the assumption that a term can only be used in contexts where its hypernyms can be used and that a term might be used in any contexts where its hyponyms are used. However, it is not always rational. Our previous method based on web mining (Fu et al., 2013) works well for hypernym extraction of entity names, but it is unsuitable for semantic hierarchy construction which involves many words with broad semantics. Moreover, all of these methods do not use the word semantics effectively.

This paper proposes a novel approach for semantic hierarchy construction based on word embeddings. Word embeddings, also known as distributed word representations, typically represent words with dense, low-dimensional and real-valued vectors. Word embeddings have been empirically shown to preserve linguistic regularities, such as the semantic relationship between words (Mikolov et al., 2013b). For example, $v(\texttt{king}) - v(\texttt{queen}) \approx v(\texttt{man}) - v(\texttt{woman})$, where $v(w)$ is the embedding of the word $w$. We observe that a similar property also applies to the hypernym–hyponym relationship (Section 3.3), which is the main inspiration of the present study.

However, we further observe that hypernym–hyponym relations are more complicated than a single offset can represent. To address this challenge, we propose a more sophisticated and general method — learning a linear projection which maps words to their hypernyms (Section 3.3.1). Furthermore, we propose a piecewise linear projection method based on relation clustering to better model hypernym–hyponym relations (Section 3.3.2). Subsequently, we identify whether an unknown word pair is a hypernym–hyponym relation using the projections (Section 3.4). To the best of our knowledge, we are the first to apply word embeddings to this task.

For evaluation, we manually annotate a dataset containing 418 Chinese entities and their hypernym hierarchies, which is the first dataset for this task as far as we know. The experimental results show that our method achieves an F-score of 73.74% which significantly outperforms the previous state-of-the-art methods. Moreover, combining our method with the manually-built hierarchy extension method proposed by Suchanek et al. (2008) can further improve F-score to 80.29%.

## 2 Background

As main components of ontologies, semantic hierarchies have been studied by many researchers. Some have established concept hierarchies based on manually-built semantic resources such as WordNet (Miller, 1995). Such hierarchies have good structures and high accuracy, but their coverage is limited to fine-grained concepts (e.g., "Ranunculaceae" is not included in WordNet.). We have made similar obsevation that about a half of hypernym–hyponym relations are absent in a Chinese semantic thesaurus. Therefore, a broader range of resources is needed to supplement the manually built resources. In the construction of the famous ontology YAGO, Suchanek et al. (2008) link the categories in Wikipedia onto WordNet. However, the coverage is still limited by the scope of Wikipedia.

Several other methods are based on lexical patterns. They use manually or automatically constructed lexical patterns to mine hypernym–hyponym relations from text corpora. A hierarchy can then be built based on these pairwise relations. The pioneer work by Hearst (1992) has found out that linking two noun phrases (NPs) via certain lexical constructions often implies hypernym relations. For example, $\text{NP}_1$ is a hypernym of $\text{NP}_2$ in the lexical pattern "such $\text{NP}_1$ as $\text{NP}_2$." Snow et al. (2005) propose to automatically extract large numbers of lexico-syntactic patterns and subsequently detect hypernym relations from a large newswire corpus. Their method relies on accurate syntactic parsers, and the quality of the automatically extracted patterns is difficult to guarantee. Generally speaking, these pattern-based methods often suffer from low recall or precision because of the coverage or the quality of the patterns.

The distributional methods assume that the contexts of hypernyms are broader than the ones of their hyponyms. For distributional similarity computing, each word is represented as a semantic vector composed of the pointwise mutual information (PMI) with its contexts. Kotlerman et al. (2010) design a directional distributional measure to infer hypernym–hyponym relations based on the standard IR Average Precision evaluation measure. Lenci and Benotto (2012) propose another measure focusing on the contexts that hypernyms do not share with their hyponyms. However, broader semantics may not always infer broader contexts. For example, for terms "Obama' and

"American people", it is hard to say whose contexts are broader.

Our previous work (Fu et al., 2013) applies a web mining method to discover the hypernyms of Chinese entities from multiple sources. We assume that the hypernyms of an entity co-occur with it frequently. It works well for named entities. But for class names (e.g., singers in Hong Kong, tropical fruits) with wider range of meanings, this assumption may fail.

In this paper, we aim to identify hypernym–hyponym relations using word embeddings, which have been shown to preserve good properties for capturing semantic relationship between words.

## 3 Method

In this section, we first define the task formally. Then we elaborate on our proposed method composed of three major steps, namely, word embedding training, projection learning, and hypernym–hyponym relation identification.

### 3.1 Task Definition

Given a list of hypernyms of an entity, our goal is to construct a semantic hierarchy on it (Figure 1). We represent the hierarchy as a directed graph $G$, in which the nodes denote the words, and the edges denote the hypernym–hyponym relations. Hypernym-hyponym relations are *asymmetric* and *transitive* when words are unambiguous:

- $\forall x, y \in L : x \xrightarrow{H} y \Rightarrow \neg(y \xrightarrow{H} x)$
- $\forall x, y, z \in L : (x \xrightarrow{H} z \wedge z \xrightarrow{H} y) \Rightarrow x \xrightarrow{H} y$

Here, $L$ denotes the list of hypernyms. $x$, $y$ and $z$ denote the hypernyms in $L$. We use $\xrightarrow{H}$ to represent a hypernym–hyponym relation in this paper. Actually, $x$, $y$ and $z$ are unambiguous as the hypernyms of a certain entity. Therefore, $G$ should be a directed acyclic graph (DAG).

### 3.2 Word Embedding Training

Various models for learning word embeddings have been proposed, including neural net language models (Bengio et al., 2003; Mnih and Hinton, 2008; Mikolov et al., 2013b) and spectral models (Dhillon et al., 2011). More recently, Mikolov et al. (2013a) propose two log-linear models, namely the *Skip-gram* and *CBOW* model, to efficiently induce word embeddings. These two models can be trained very efficiently on a large-scale corpus because of their low time complexity.

| No. | Examples |
|---|---|
| 1 | $v$(虾) − $v$(对虾) ≈ $v$(鱼) − $v$(金鱼)<br>$v$(shrimp) − $v$(prawn) ≈ $v$(fish) − $v$(gold fish) |
| 2 | $v$(工人) − $v$(木匠) ≈ $v$(演员) − $v$(小丑)<br>$v$(laborer) − $v$(carpenter) ≈ $v$(actor) − $v$(clown) |
| 3 | $v$(工人) − $v$(木匠) $\not\approx$ $v$(鱼) − $v$(金鱼)<br>$v$(laborer) − $v$(carpenter) $\not\approx$ $v$(fish) − $v$(gold fish) |

Table 1: Embedding offsets on a sample of hypernym–hyponym word pairs.

Additionally, their experiment results have shown that the *Skip-gram* model performs best in identifying semantic relationship among words. Therefore, we employ the *Skip-gram* model for estimating word embeddings in this study.

The *Skip-gram* model adopts log-linear classifiers to predict context words given the current word $\mathbf{w}(t)$ as input. First, $\mathbf{w}(t)$ is projected to its embedding. Then, log-linear classifiers are employed, taking the embedding as input and predict $\mathbf{w}(t)$'s context words within a certain range, e.g. $k$ words in the left and $k$ words in the right. After maximizing the log-likelihood over the entire dataset using stochastic gradient descent (SGD), the embeddings are learned.

### 3.3 Projection Learning

Mikolov et al. (2013b) observe that word embeddings preserve interesting linguistic regularities, capturing a considerable amount of syntactic/semantic relations. Looking at the well-known example: $v(\texttt{king}) - v(\texttt{queen}) \approx v(\texttt{man}) - v(\texttt{woman})$, it indicates that the embedding offsets indeed represent the shared semantic relation between the two word pairs.

We observe that the same property also applies to some hypernym–hyponym relations. As a preliminary experiment, we compute the embedding offsets between some randomly sampled hypernym–hyponym word pairs and measure their similarities. The results are shown in Table 1.

The first two examples imply that a word can also be mapped to its hypernym by utilizing word embedding offsets. However, the offset from "carpenter" to "laborer" is distant from the one from "gold fish" to "fish," indicating that hypernym–hyponym relations should be more complicated than a single vector offset can represent. To verify this hypothesis, we compute the embedding offsets over all hypernym–

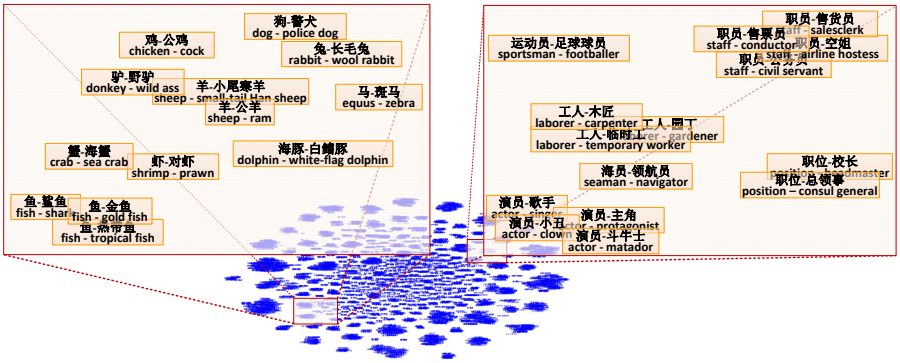


Figure 2: Clusters of the vector offsets in training data. The figure shows that the vector offsets distribute in some clusters. The left cluster shows some hypernym–hyponym relations about animals. The right one shows some relations about people's occupations.

hyponym word pairs in our training data and visualize them.[2] Figure 2 shows that the relations are adequately distributed in the clusters, which implies that hypernym–hyponym relations indeed can be decomposed into more fine-grained relations. Moreover, the relations about animals are spatially close, but separate from the relations about people's occupations.

To address this challenge, we propose to learn the hypernym–hyponym relations using projection matrices.

#### 3.3.1 A Uniform Linear Projection

Intuitively, we assume that all words can be projected to their hypernyms based on a uniform transition matrix. That is, given a word $x$ and its hypernym $y$, there exists a matrix $\Phi$ so that $y = \Phi x$. For simplicity, we use the same symbols as the words to represent the embedding vectors. Obtaining a consistent exact $\Phi$ for the projection of all hypernym–hyponym pairs is difficult. Instead, we can learn an approximate $\Phi$ using Equation 1 on the training data, which minimizes the mean-squared error:

$$\Phi^* = \arg\min_{\Phi} \frac{1}{N} \sum_{(x,y)} \| \Phi x - y \|^2 \qquad (1)$$

where $N$ is the number of $(x, y)$ word pairs in the training data. This is a typical linear regression problem. The only difference is that our predictions are multi-dimensional vectors instead of scalar values. We use SGD for optimization.

[2]Principal Component Analysis (PCA) is applied for dimensionality reduction.

#### 3.3.2 Piecewise Linear Projections

A uniform linear projection may still be underrepresentative for fitting all of the hypernym–hyponym word pairs, because the relations are rather diverse, as shown in Figure 2. To better model the various kinds of hypernym–hyponym relations, we apply the idea of piecewise linear regression (Ritzema, 1994) in this study.

Specifically, the input space is first segmented into several regions. That is, all word pairs $(x, y)$ in the training data are first clustered into several groups, where word pairs in each group are expected to exhibit similar hypernym–hyponym relations. Each word pair $(x, y)$ is represented with their vector offsets: $y - x$ for clustering. The reasons are twofold: (1) Mikolov's work has shown that the vector offsets imply a certain level of semantic relationship. (2) The vector offsets distribute in clusters well, and the word pairs which are close indeed represent similar relations, as shown in Figure 2.

Then we learn a separate projection for each cluster, respectively (Equation 2).

$$\Phi_k^* = \arg\min_{\Phi_k} \frac{1}{N_k} \sum_{(x,y)\in C_k} \| \Phi_k x - y \|^2 \qquad (2)$$

where $N_k$ is the amount of word pairs in the $k^{th}$ cluster $C_k$.

We use the $k$-means algorithm for clustering, where $k$ is tuned on a development dataset.

#### 3.3.3 Training Data

To learn the projection matrices, we extract training data from a Chinese semantic thesaurus, Tongyi Cilin (Extended) (CilinE for short) which

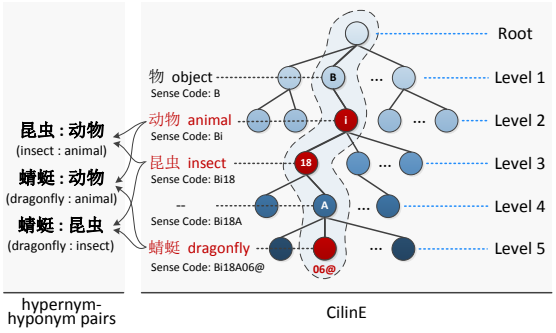


Figure 3: Hierarchy of CilinE and an Example of Training Data Generation

contains 100,093 words (Che et al., 2010).[3] CilinE is organized as a hierarchy of five levels, in which the words are linked by hypernym–hyponym relations (right panel, Figure 3). Each word in CilinE has one or more sense codes (some words are polysemous) that indicate its position in the hierarchy.

The senses of words in the first level, such as "物 (`object`)" and "时间 (`time`)," are very general. The fourth level only has sense codes without real words. Therefore, we extract words in the second, third and fifth levels to constitute hypernym–hyponym pairs (left panel, Figure 3).

Note that mapping one hyponym to multiple hypernyms with the same projection ($\Phi x$ is unique) is difficult. Therefore, the pairs with the same hyponym but different hypernyms are expected to be clustered into separate groups. Figure 3 shows that the word "`dragonfly`" in the fifth level has two hypernyms: "`insect`" in the third level and "`animal`" in the second level. Hence the relations `dragonfly` $\xrightarrow{H}$ `insect` and `dragonfly` $\xrightarrow{H}$ `animal` should fall into different clusters.

In our implementation, we apply this constraint by simply dividing the training data into two categories, namely, *direct* and *indirect*. Hypernym-hyponym word pair $(x, y)$ is classified into the *direct* category, only if there doesn't exist another word $z$ in the training data, which is a hypernym of $x$ and a hyponym of $y$. Otherwise, $(x, y)$ is classified into the *indirect* category. Then, data in these two categories are clustered separately.

[3] `www.ltp-cloud.com/download/`

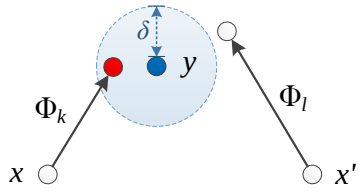


Figure 4: In this example, $\Phi_k x$ is located in the circle with center $y$ and radius $\delta$. So $y$ is considered as a hypernym of $x$. Conversely, $y$ is not a hypernym of $x'$.

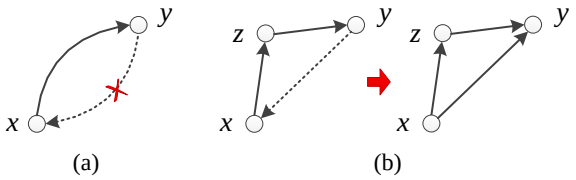


Figure 5: (a) If $d(\Phi_j y, x) > d(\Phi_k x, y)$, we remove the path from $y$ to $x$; (b) if $d(\Phi_j y, x) > d(\Phi_k x, z)$ and $d(\Phi_j y, x) > d(\Phi_i z, y)$, we reverse the path from $y$ to $x$.

### 3.4 Hypernym-hyponym Relation Identification

Upon obtaining the clusters of training data and the corresponding projections, we can identify whether two words have a hypernym–hyponym relation. Given two words $x$ and $y$, we find cluster $C_k$ whose center is closest to the offset $y - x$, and obtain the corresponding projection $\Phi_k$. For $y$ to be considered a hypernym of $x$, one of the two conditions below must hold.

**Condition 1:** The projection $\Phi_k$ puts $\Phi_k x$ close enough to $y$ (Figure 4). Formally, the euclidean distance between $\Phi_k x$ and $y$: $d(\Phi_k x, y)$ must be less than a threshold $\delta$.

$$d(\Phi_k x, y) = \| \Phi_k x - y \|^2 < \delta \qquad (3)$$

**Condition 2:** There exists another word $z$ satisfying $x \xrightarrow{H} z$ and $z \xrightarrow{H} y$. In this case, we use the transitivity of hypernym–hyponym relations.

Besides, the final hierarchy should be a DAG as discussed in Section 3.1. However, the projection method cannot guarantee that theoretically, because the projections are learned from pairwise hypernym–hyponym relations without the whole hierarchy structure. All pairwise hypernym–hyponym relation identification methods would suffer from this problem actually. It is an interesting problem how to construct a globally opti-

mal semantic hierarchy conforming to the form of a DAG. But this is not the focus of this paper. So if some conflicts occur, that is, a relation circle exists, we remove or reverse the weakest path heuristically (Figure 5). If a circle has only two nodes, we remove the weakest path. If a circle has more than two nodes, we reverse the weakest path to form an *indirect* hypernym–hyponym relation.

## 4 Experimental Setup

### 4.1 Experimental Data

In this work, we learn word embeddings from a Chinese encyclopedia corpus named Baidubaike[4], which contains about 30 million sentences (about 780 million words). The Chinese segmentation is provided by the open-source Chinese language processing platform LTP[5] (Che et al., 2010). Then, we employ the *Skip-gram* method (Section 3.2) to train word embeddings. Finally we obtain the embedding vectors of 0.56 million words.

The training data for projection learning is collected from CilinE (Section 3.3.3). We obtain 15,247 word pairs of hypernym–hyponym relations (9,288 for *direct* relations and 5,959 for *indirect* relations).

For evaluation, we collect the hypernyms for 418 entities, which are selected randomly from Baidubaike, following Fu et al. (2013). We then ask two annotators to manually label the semantic hierarchies of the correct hypernyms. The final data set contains 655 unique hypernyms and 1,391 hypernym–hyponym relations among them. We randomly split the labeled data into 1/5 for development and 4/5 for testing (Table 2). The hierarchies are represented as relations of pairwise words. We measure the inter-annotator agreement using the kappa coefficient (Siegel and Castellan Jr, 1988). The kappa value is 0.96, which indicates a good strength of agreement.

### 4.2 Evaluation Metrics

We use precision, recall, and F-score as our metrics to evaluate the performances of the methods.

Since hypernym–hyponym relations and its reverse (hyponym–hypernym) have one-to-one correspondence, their performances are equal. For simplicity, we only report the performance of the former in the experiments.

[4]Baidubaike (baike.baidu.com) is one of the largest Chinese encyclopedias containing more than 7.05 million entries as of September, 2013.

[5]www.ltp-cloud.com/demo/

| Relation | # of word pairs | |
|---|---|---|
| | Dev. | Test |
| hypernym–hyponym | 312 | 1,079 |
| hyponym–hypernym* | 312 | 1,079 |
| unrelated | 1,044 | 3,250 |
| Total | 1,668 | 5,408 |

Table 2: The evaluation data. *Since hypernym–hyponym relations and hyponym–hypernym relations have one-to-one correspondence, their numbers are the same.

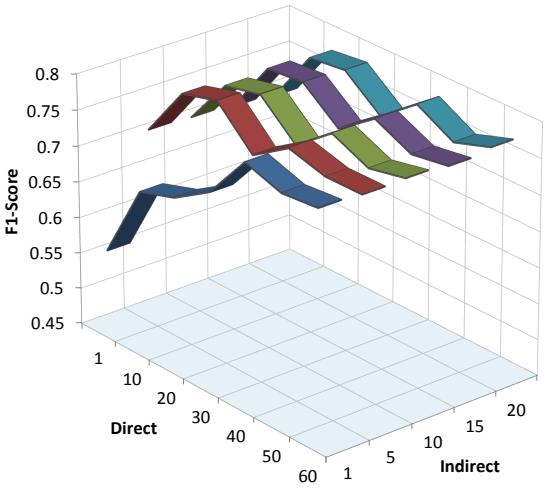


Figure 6: Performance on development data w.r.t. cluster size.

## 5 Results and Analysis

### 5.1 Varying the Amount of Clusters

We first evaluate the effect of different number of clusters based on the development data. We vary the numbers of the clusters both for the *direct* and *indirect* training word pairs.

As shown in Figure 6, the performance of clustering is better than non-clustering (when the cluster number is 1), thus providing evidences that learning piecewise projections based on clustering is reasonable. We finally set the numbers of the clusters of *direct* and *indirect* to 20 and 5, respectively, where the best performances are achieved on the development data.

### 5.2 Comparison with Previous Work

In this section, we compare the proposed method with previous methods, including manually-built hierarchy extension, pairwise relation extraction

| | P(%) | R(%) | F(%) |
|---|---|---|---|
| $M_{Wiki+CilinE}$ | 92.41 | 60.61 | 73.20 |
| $M_{Pattern}$ | 97.47 | 21.41 | 35.11 |
| $M_{Snow}$ | 60.88 | 25.67 | 36.11 |
| $M_{balApinc}$ | 54.96 | 53.38 | 54.16 |
| $M_{invCL}$ | 49.63 | 62.84 | 55.46 |
| $M_{Fu}$ | 87.40 | 48.19 | 62.13 |
| $M_{Emb}$ | 80.54 | 67.99 | **73.74** |
| $M_{Emb+CilinE}$ | 80.59 | 72.42 | 76.29 |
| $M_{Emb+Wiki+CilinE}$ | 79.78 | 80.81 | **80.29** |

Table 3: Comparison of the proposed method with existing methods in the test set.

| Pattern | Translation |
|---|---|
| w 是[一个\|一种] h | w is a [a kind of] h |
| w [、] 等 h | w[,] and other h |
| h [，] 叫[做] w | h[,] called w |
| h [，] [像]如 w | h[,] such as w |
| h [，] 特别是 w | h[,] especially w |

Table 4: Chinese Hearst-style lexical patterns. The contents in square brackets are omissible.

based on patterns, word distributions, and web mining (Section 2). Results are shown in Table 3.

#### 5.2.1 Overall Comparison

$M_{Wiki+CilinE}$ refers to the manually-built hierarchy extension method of Suchanek et al. (2008). In our experiment, we use the category taxonomy of Chinese Wikipedia[6] to extend CilinE. Table 3 shows that this method achieves a high precision but also a low recall, mainly because of the limited scope of Wikipedia.

$M_{Pattern}$ refers to the pattern-based method of Hearst (1992). We extract hypernym–hyponym relations in the Baidubaike corpus, which is also used to train word embeddings (Section 4.1). We use the Chinese Hearst-style patterns (Table 4) proposed by Fu et al. (2013), in which w represents a word, and h represents one of its hypernyms. The result shows that only a small part of the hypernyms can be extracted based on these patterns because only a few hypernym relations are expressed in these fixed patterns, and many are expressed in highly flexible manners.

In the same corpus, we apply the method $M_{Snow}$ originally proposed by Snow et al. (2005). The same training data for projections learning from CilinE (Section 3.3.3) is used as seed hypernym–hyponym pairs. Lexico-syntactic patterns are extracted from the Baidubaike corpus by using the seeds. We then develop a logistic regression classifier based on the patterns to recognize hypernym–hyponym relations. This method relies on an accurate syntactic parser, and the quality of the automatically extracted patterns is difficult to guarantee.

We re-implement two previous distributional methods $M_{balApinc}$ (Kotlerman et al., 2010) and $M_{invCL}$ (Lenci and Benotto, 2012) in the Baidubaike corpus. Each word is represented as a feature vector in which each dimension is the PMI value of the word and its context words. We compute a score for each word pair and apply a threshold to identify whether it is a hypernym–hyponym relation.

$M_{Fu}$ refers to our previous web mining method (Fu et al., 2013). This method mines hypernyms of a given word $w$ from multiple sources and returns a ranked list of the hypernyms. We select the hypernyms with scores over a threshold of each word in the test set for evaluation. This method assumes that frequent co-occurrence of a noun or noun phrase $n$ in multiple sources with $w$ indicate possibility of $n$ being a hypernym of $w$. The results presented in Fu et al. (2013) show that the method works well when $w$ is an entity, but not when $w$ is a word with a common semantic concept. The main reason may be that there are relatively more introductory pages about entities than about common words in the Web.

$M_{Emb}$ is the proposed method based on word embeddings. Table 3 shows that the proposed method achieves a better recall and F-score than all of the previous methods do. It can significantly ($p < 0.01$) improve the F-score over the state-of-the-art method $M_{Wiki+CilinE}$.

$M_{Emb}$ and $M_{CilinE}$ can also be combined. The combination strategy is to simply merge all positive results from the two methods together, and then to infer new relations based on the transitivity of hypernym–hyponym relations. The F-score is further improved from 73.74% to 76.29%. Note that, the combined method achieves a 4.43% recall improvement over $M_{Emb}$, but the precision is almost unchanged. The reason is that the inference based on the relations identified automatically may lead to error propagation. For example, the relation $x \xrightarrow{H} y$ is incorrectly identified by $M_{Emb}$.

[6] dumps.wikimedia.org/zhwiki/20131205/

| | P(%) | R(%) | F(%) |
|---|---|---|---|
| $M_{Wiki+CilinE}$ | 80.39 | 19.29 | 31.12 |
| $M_{Emb+CilinE}$ | 71.16 | 52.80 | 60.62 |
| $M_{Emb+Wiki+CilinE}$ | 69.13 | 61.65 | **65.17** |

Table 5: Performance on the out-of-CilinE data in the test set.

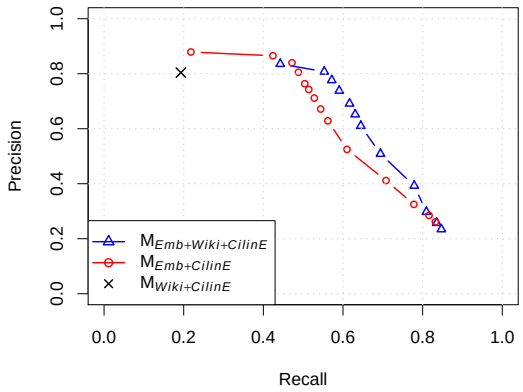


Figure 7: Precision-Recall curves on the out-of-CilinE data in the test set.

When the relation $y \xrightarrow{H} z$ from $M_{CilinE}$ is added, it will cause a new incorrect relation $x \xrightarrow{H} z$.

Combining $M_{Emb}$ with $M_{Wiki+CilinE}$ achieves a 7% F-score improvement over the best baseline $M_{Wiki+CilinE}$. Therefore, the proposed method is complementary to the manually-built hierarchy extension method (Suchanek et al., 2008).

### 5.2.2 Comparison on the Out-of-CilinE Data

We are greatly interested in the practical performance of the proposed method on the hypernym–hyponym relations outside of CilinE. We say a word pair is outside of CilinE, as long as there is one word in the pair not existing in CilinE. In our test data, about 62% word pairs are outside of CilinE. Table 5 shows the performances of the best baseline method and our method on the out-of-CilinE data. The method exploiting the taxonomy in Wikipedia, $M_{Wiki+CilinE}$, achieves the highest precision but has a low recall. By contrast, our method can discover more hypernym–hyponym relations with some loss of precision, thereby achieving a more than 29% F-score improvement. The combination of these two methods achieves a further 4.5% F-score improvement over $M_{Emb+CilinE}$. Generally speaking, the proposed method greatly improves the recall but damages the precision.

Actually, we can get different precisions and recalls by adjusting the threshold $\delta$ (Equation 3). Figure 7 shows that $M_{Emb+CilinE}$ achieves a higher precision than $M_{Wiki+CilinE}$ when their recalls are the same. When they achieve the same precision, the recall of $M_{Emb+CilinE}$ is higher.

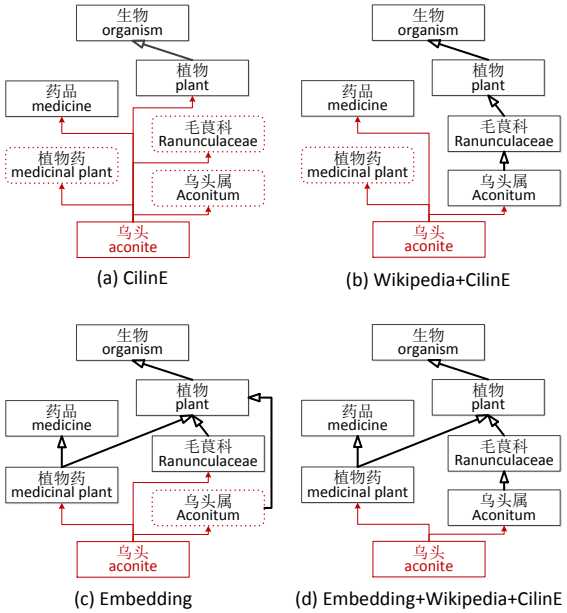


Figure 8: An example for error analysis. The red paths refer to the relations between the named entity and its hypernyms extracted using the web mining method (Fu et al., 2013). The black paths with hollow arrows denote the relations identified by the different methods. The boxes with dotted borders refer to the concepts which are not linked to correct positions.

## 5.3 Error Analysis and Discussion

We analyze error cases after experiments. Some cases are shown in Figure 8. We can see that there is only one general relation "植物 (plant)" $\xrightarrow{H}$ "生物 (organism)" existing in CilinE. Some fine-grained relations exist in Wikipedia, but the coverage is limited. Our method based on word embeddings can discover more hypernym–hyponym relations than the previous methods can. When we combine the methods together, we get the correct hierarchy.

Figure 8 shows that our method loses the relation "乌头属 (Aconitum)" $\xrightarrow{H}$ "毛茛科 (Ranunculaceae)." It is because they are very semantically similar (their cosine similarity is 0.9038). Their representations are so close to each other in the embedding space that we have not find projections suitable for these pairs. The

error statistics show that when the cosine similarities of word pairs are greater than 0.8, the recall is only 9.5%. This kind of error accounted for about 10.9% among all the errors in our test set. One possible solution may be adding more data of this kind to the training set.

## 6 Related Work

In addition to the works mentioned in Section 2, we introduce another set of related studies in this section.

Evans (2004), Ortega-Mendoza et al. (2007), and Sang (2007) consider web data as a large corpus and use search engines to identify hypernyms based on the lexical patterns of Hearst (1992). However, the low quality of the sentences in the search results negatively influence the precision of hypernym extraction.

Following the method for discovering patterns automatically (Snow et al., 2005), McNamee et al. (2008) apply the same method to extract hypernyms of entities in order to improve the performance of a question answering system. Ritter et al. (2009) propose a method based on patterns to find hypernyms on arbitrary noun phrases. They use a support vector machine classifier to identify the correct hypernyms from the candidates that match the patterns. As our experiments show, pattern-based methods suffer from low recall because of the low coverage of patterns.

Besides Kotlerman et al. (2010) and Lenci and Benotto (2012), other researchers also propose directional distributional similarity methods (Weeds et al., 2004; Geffet and Dagan, 2005; Bhagat et al., 2007; Szpektor et al., 2007; Clarke, 2009). However, their basic assumption that a hyponym can only be used in contexts where its hypernyms can be used and that a hypernym might be used in all of the contexts where its hyponyms are used may not always rational.

Snow et al. (2006) provides a global optimization scheme for extending WordNet, which is different from the above-mentioned pairwise relationships identification methods.

Word embeddings have been successfully applied in many applications, such as in sentiment analysis (Socher et al., 2011b), paraphrase detection (Socher et al., 2011a), chunking, and named entity recognition (Turian et al., 2010; Collobert et al., 2011). These applications mainly utilize the representing power of word embeddings to alleviate the problem of data sparsity. Mikolov et al. (2013a) and Mikolov et al. (2013b) further observe that the semantic relationship of words can be induced by performing simple algebraic operations with word vectors. Their work indicates that word embeddings preserve some interesting linguistic regularities, which might provide support for many applications. In this paper, we improve on their work by learning multiple linear projections in the embedding space, to model hypernym–hyponym relationships within different clusters.

## 7 Conclusion and Future Work

This paper proposes a novel method for semantic hierarchy construction based on word embeddings, which are trained using a large-scale corpus. Using the word embeddings, we learn the hypernym–hyponym relationship by estimating projection matrices which map words to their hypernyms. Further improvements are made using a cluster-based approach in order to model the more fine-grained relations. Then we propose a few simple criteria to identity whether a new word pair is a hypernym–hyponym relation. Based on the pairwise hypernym–hyponym relations, we build semantic hierarchies automatically.

In our experiments, the proposed method significantly outperforms state-of-the-art methods and achieves the best F1-score of 73.74% on a manually labeled test dataset. Further experiments show that our method is complementary to the previous manually-built hierarchy extension methods.

For future work, we aim to improve word embedding learning under the guidance of hypernym–hyponym relations. By including the hypernym–hyponym relation constraints while training word embeddings, we expect to improve the embeddings such that they become more suitable for this task.

## Acknowledgments

This work was supported by National Natural Science Foundation of China (NSFC) via grant 61133012, 61273321 and the National 863 Leading Technology Research Project via grant 2012AA011102. Special thanks to Shiqi Zhao, Zhenghua Li, Wei Song and the anonymous reviewers for insightful comments and suggestions. We also thank Xinwei Geng and Hongbo Cai for their help in the experiments.

## References


Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. 2003. A neural probabilistic language model. *The Journal of Machine Learning Research*, 3:1137–1155.

Rahul Bhagat, Patrick Pantel, Eduard H Hovy, and Marina Rey. 2007. Ledir: An unsupervised algorithm for learning directionality of inference rules. In *EMNLP-CoNLL*, pages 161–170.

Wanxiang Che, Zhenghua Li, and Ting Liu. 2010. Ltp: A chinese language technology platform. In *Coling 2010: Demonstrations*, pages 13–16, Beijing, China, August.

Daoud Clarke. 2009. Context-theoretic semantics for natural language: an overview. In *Proceedings of the Workshop on Geometrical Models of Natural Language Semantics*, pages 112–119. Association for Computational Linguistics.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *The Journal of Machine Learning Research*, 12:2493–2537.

Paramveer Dhillon, Dean P Foster, and Lyle H Ungar. 2011. Multi-view learning of word embeddings via cca. In *Advances in Neural Information Processing Systems*, pages 199–207.

Richard Evans. 2004. A framework for named entity recognition in the open domain. *Recent Advances in Natural Language Processing III: Selected Papers from RANLP 2003*, 260:267–274.

Ruiji Fu, Bing Qin, and Ting Liu. 2013. Exploiting multiple sources for open-domain hypernym discovery. In *EMNLP*, pages 1224–1234.

Maayan Geffet and Ido Dagan. 2005. The distributional inclusion hypotheses and lexical entailment. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 107–114. Association for Computational Linguistics.

Marti A Hearst. 1992. Automatic acquisition of hyponyms from large text corpora. In *Proceedings of the 14th conference on Computational linguistics-Volume 2*, pages 539–545. Association for Computational Linguistics.

Lili Kotlerman, Ido Dagan, Idan Szpektor, and Maayan Zhitomirsky-Geffet. 2010. Directional distributional similarity for lexical inference. *Natural Language Engineering*, 16(4):359–389.

Alessandro Lenci and Giulia Benotto. 2012. Identifying hypernyms in distributional semantic spaces. In *Proceedings of the Sixth International Workshop on Semantic Evaluation*, pages 75–79. Association for Computational Linguistics.

Paul McNamee, Rion Snow, Patrick Schone, and James Mayfield. 2008. Learning named entity hyponyms for question answering. In *Proceedings of the Third International Joint Conference on Natural Language Processing*, pages 799–804.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. 2013b. Linguistic regularities in continuous space word representations. In *Proceedings of NAACL-HLT*, pages 746–751.

George A Miller. 1995. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.

Andriy Mnih and Geoffrey E Hinton. 2008. A scalable hierarchical distributed language model. In *Advances in neural information processing systems*, pages 1081–1088.

Rosa M Ortega-Mendoza, Luis Villaseñor-Pineda, and Manuel Montes-y Gómez. 2007. Using lexical patterns for extracting hyponyms from the web. In *MICAI 2007: Advances in Artificial Intelligence*, pages 904–911. Springer.

Alan Ritter, Stephen Soderland, and Oren Etzioni. 2009. What is this, anyway: Automatic hypernym discovery. In *Proceedings of the 2009 AAAI Spring Symposium on Learning by Reading and Learning to Read*, pages 88–93.

HP Ritzema. 1994. *Drainage principles and applications.*

Erik Tjong Kim Sang. 2007. Extracting hypernym pairs from the web. In *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*, pages 165–168. Association for Computational Linguistics.

Sidney Siegel and N John Castellan Jr. 1988. Nonparametric statistics for the behavioral sciences. McGraw-Hill, New York.

Rion Snow, Daniel Jurafsky, and Andrew Y. Ng. 2005. Learning syntactic patterns for automatic hypernym discovery. In Lawrence K. Saul, Yair Weiss, and Léon Bottou, editors, *Advances in Neural Information Processing Systems 17*, pages 1297–1304. MIT Press, Cambridge, MA.

Rion Snow, Daniel Jurafsky, and Andrew Y. Ng. 2006. Semantic taxonomy induction from heterogenous evidence. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 801–808, Sydney, Australia, July. Association for Computational Linguistics.

Richard Socher, Eric H Huang, Jeffrey Pennin, Christopher D Manning, and Andrew Ng. 2011a. Dynamic pooling and unfolding recursive autoencoders for paraphrase detection. In *Advances in Neural Information Processing Systems*, pages 801–809.

Richard Socher, Jeffrey Pennington, Eric H Huang, Andrew Y Ng, and Christopher D Manning. 2011b. Semi-supervised recursive autoencoders for predicting sentiment distributions. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 151–161. Association for Computational Linguistics.

Fabian M Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2008. Yago: A large ontology from wikipedia and wordnet. *Web Semantics: Science, Services and Agents on the World Wide Web*, 6(3):203–217.

Idan Szpektor, Eyal Shnarch, and Ido Dagan. 2007. Instance-based evaluation of entailment rule acquisition. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 456–463, Prague, Czech Republic, June. Association for Computational Linguistics.

Joseph Turian, Lev Ratinov, and Yoshua Bengio. 2010. Word representations: a simple and general method for semi-supervised learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 384–394. Association for Computational Linguistics.

Julie Weeds, David Weir, and Diana McCarthy. 2004. Characterising measures of lexical distributional similarity. In *Proceedings of the 20th international conference on Computational Linguistics*, page 1015. Association for Computational Linguistics.