A Hierarchical Semantics-Aware Distributional Similarity Scheme*

Shuqi Sun[1], Ke Sun[2], Shiqi Zhao[2], Haifeng Wang[2], Muyun Yang[1], and Sheng Li[1]
[1]Harbin Institute of Technology, Harbin, China
{sqsun,ymy}@mtlab.hit.edu.cn, lisheng@hit.edu.cn
[2]Baidu, Beijing, China
{sunke,zhaoshiqi,wanghaifeng}@baidu.com

Abstract

The context type and similarity calculation are two essential features of a distributional similarity scheme (DSS). In this paper, we propose a hierarchical semantic-aware DSS that exploits semantic relation words as extra context information to guide the similarity calculation. First, we define and extract five types of semantic relations, and then develop relation-based similarities from the distributional similarities among the top-ranked relation words. Finally, we integrate various similarities using learning-to-rank technique. Experiments show that semantic relations are beneficial to predicting accurate similarity. On 6904 pairwise similarity comparisons, the predictive accuracy of our approach reaches 83.9%, which significantly outperforms the baseline approaches. We also conduct intrinsic analysis by varying the quality of semantic relations and the usage of individual similarities.

1 Introduction

Distributional similarity is an essential measure of the semantic relatedness between a pair of linguistic objects, including words, phrases, or even sentences. Confident distributional similar objects, are useful in various NLP applications, such as word sense disambiguation (Lin, 1997), lexical substitution (McCarthy and Navigli, 2009), paraphrase ranking (Dinu and Lapata, 2010), text classification (Baker and McCallum, 1998), etc. In this paper, we focus on the semantic similarity between words.

Well-known implementations of word-level distributional similarity scheme (DSS) mainly fall into two categories according to the choice of the context: a) text-window based, and b) dependency path based. The former has the advantages of language-independence and computational efficiency, while the latter captures finer word-word relationships. However, both approaches focus on the usage aspect of words' meanings, which is only indirect indicator of the underlying semantic interactions.

Meanwhile, a great many successful efforts have been made to extract word-level semantic knowledge from the text, including synonyms / antonyms, sibling terms, hypernyms / hyponyms, holonyms / meronyms, etc. Despite of such deliberated studies, there are few considerations about how these semantics-oriented outcomes could contribute to the construction of DSS.

To realize the full potential of such semantic evidences, we propose a semantics-aware DSS by using semantic relation words to guide the similarity calculation. Our motivation is that words having similar semantic relational words, e.g. sibling terms or hypernyms, tend to be more semantically similar. The proposed DSS has a hierarchical layout, in which text-window based distributional similarity is first established from the corpus serving as the basis of the relation-specific similarities for 5 semantic relations. Finally, these similarities are linearly combined using learning-to-rank technique into a single measure, capturing both the context distributions and the semantic relations of the target words.

Our contribution is three-fold. First, by deriving semantic relation based similarities from distributional similarity, we develop a semantics-aware DSS in a hierarchical fashion. The DSS eventually fuses these similarities, and yields significant improvement over several baseline approaches. Second, our design of DSS relies solely on the same corpus where distributional similarity can be derived. It is adaptable to different languages, in

*This work was done when the first author was visiting Baidu.

which distributional similarity and semantic relations are available. Third, our DSS's hierarchical nature allows us to individually replace each component with better implementations to adapt to specific applications or new languages.

2 Related Work

Distributional similarity (a.k.a. contextual similarity) has been elaborately studied to predict semantic similarity, and the type of the context is a main concern. A variety of context types have been proposed to capture the underlying semantic interactions between linguistic objects, including text-window based collocations (Rapp, 2003; Agirre et al., 2009), lexico-syntactic patterns (Turney, 2006; Baroni and Lenci, 2010), grammatical dependencies (Lin, 1998; Padó and Lapata, 2007; Thater et al., 2010), click-through data (Jain and Pennacchiotti, 2010), selectional preferences (Erk and Padó, 2008), synsets in thesaurus (Agirre et al., 2009), and latent topics (Dinu and Lapata, 2010). There are also researches that focus on distribution compositions (Mitchell and Lapata, 2008; Grefenstette et al., 2013) or context constrained similarity calculation (Erk and Padó, 2008).

Extracting sibling or hierarchical semantic relations from corpora forms a different track of research, in which exist ample efforts. Most of them make use of hand-crafted or automatically bootstrapped patterns. Various types of patterns have been tried out, including plain texts (Hearst, 1992; Paşca, 2004), semi-structured HTML tags (Shinzato and Torisawa, 2007), or their combinations (Shi et al., 2010). Bootstrapping approach is shown useful given a number of seeds, which could be either relation instances(Snow et al., 2004; Pantel and Pennacchiotti, 2006), or initial patterns(Pantel et al., 2004). To improve the quality of raw extraction results, some studies also resort to optimizing one relation using other relations (Zhang et al., 2011; Kozareva et al., 2011) or using distributional similarity (Shi et al., 2010).

Despite the great progress made in the field of semantic relation extraction, few studies explicitly use semantic relations to guide the similarity calculation. In this paper, we use instance-pattern iteration on a massive corpus to populate semantic relation instances, and derive relation-specific similarities on top of text-window based distributional similarity. Indeed, previous studies did resort to a closed set of lexical patterns that indicate sibling / hypernym / hyponym relations (Baroni and Lenci, 2010; Bansal and Klein, 2012), concept properties (Baroni et al., 2010), and attribute information (Baroni and Lenci, 2010). Compared with these studies, our approach systematically exploits specific semantic relations instead of counting co-occurrence under surface patterns. We also develop a hierarchical similarity fusion architecture, rather than blending the heterogeneous evidences in a single distribution vector (Baroni and Lenci, 2010). It is also notable that sibling term extraction, in which various semantic evidences (e.g. hypernyms) also help, is not in the track of our study. Sibling term extraction focuses on words sharing the same super concept, and does not quantify the pair-wise similarity between them. Nevertheless, sibling terms work fine as an evidence of semantic similarity, as shown in our experiments.

Machine learning based integration of multiple evidences are shown useful in semantic class construction and semantic similarity calculation. Pennacchiotti and Pantel (2009) use gradient boosting decision tree to combine evidences from Web page, query log, Web tablet, and Wikipedia to populate instances of *Actors*, *Athletes*, and *Musicians*. There are also studies that combine distribution and pattern information in lexical entailment (Mirkin et al., 2006) and word clustering (Kaji and Kitsuregawa, 2008). Close to our work, Agirre et al. (2009) train SVM classification models to combine individual similarities derived from dependency path, text-window, and WordNet synsets. The synsets are highly accurate in representing words' meanings. However, the size of the thesaurus is limited, and not equally available in different languages. Although Agirre et al. (2009) tried machine translation techniques to tackle with this issue, abundant named entities and translation errors in the Web corpus still challenge the performance of their approach.

3 Hierarchical Semantics-aware DSS

Our proposed DSS has a four-layer structure, as shown in Figure 1. The bottom layer is the *corpus layer* where a massive Web page repository is preprocessed. Upwards, we build a *distribution layer* to obtain basic text-window based distributional similarity between any pair of target words. The distribution layer provides a distributional similarity database upon which a *semantics layer* takes effect. At this layer we adopt an extraction system

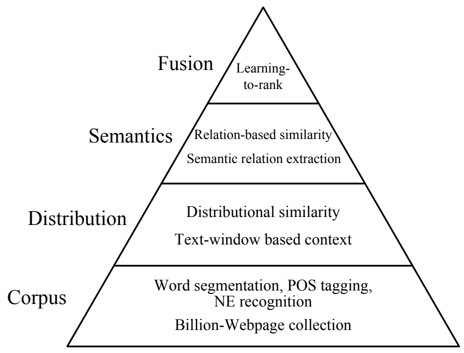
Fusion
Learning-to-rank
Semantics
Relation-based similarity
Semantic relation extraction
Distribution
Distributional similarity
Text-window based context
Corpus
Word segmentation, POS tagging, NE recognition
Billion-Webpage collection

Figure 1: Hierarchical approach to semantics-aware DSS.

that iteratively populates instances for 5 types of semantic relations. Then, for each type of semantic relation, we develop a relation-specific similarity measure. Finally at the *fusion layer*, the similarities at the distribution and semantics layers are integrated linearly using learning-to-rank.

3.1 Corpus Layer

The corpus we work on is a repository of Chinese Web pages collected in 2011. It contains 1.1 billion pages (5.8×10^{11} words in total), and takes 4.7TB of storage. All pages are de-tagged, leaving only their titles and textual content, and then segmented with a word segmentor based on dictionaries plus conditional random field (CRF) models. The segmentor is efficiently implemented, and is able to process 40K words per second with an accuracy around 98%. Based on the segmentation results, two more steps of pre-processing: POS tagging and named entity (NE) recognition are also performed.

3.2 Distribution Layer

The most widely studied types of context for distributional similarity are text-window based context and grammatical context. The construction of the latter requires syntactic or dependency parsing, which is highly language-dependant and may be extremely time-consuming on large corpora.

Therefore, at the distribution layer, we build the distributional similarity database using simple text-window based co-occurrences. Two different lengths of the text-window are experimented: 3 words and 6 words. The window slides one word per step from the beginning of each sentence to the end. Thus, the 3-length and 6-length windows capture two and five words at most on each side of the target word respectively. For each pair of words w and w', four association measures are tried out, covering a range of common practices, including (1) raw number of co-occurrence, (2) point mutual information (PMI), (3) Jaccard index, and (4) local mutual information (LMI) (Evert, 2008). To reduce the amount of computation, we preserve top 5000 context words for each target word. Processing the whole corpus yields ~6.5 million unique target words[1]. For any pair of words w and w' in the vocabulary, the distribution layer provides it a cosine similarity between their context distributions, denoted by $ds(w, w')$.

3.3 Semantics Layer

We deem that semantic relationship is a more direct clue of a word's *meaning* than either its text-window co-occurrences or syntactic dependencies. Therefore, we introduce a semantics layer upon the distribution layer to exploit semantic relations. Specifically, we adopt an extraction system to populate semantic relation instances, and then derive relation-based similarities from the system's output.

3.3.1 Relation Extraction

Here, we present a fully-featured, yet lightweight semantic relation extraction system that is capable to conduct in-depth mining in massive corpora. The system follows the line of instance-pattern iteration on a massive corpus, and can be substituted by any implementation of such fashion.

We define and extract five types of semantic relations, as listed in Table 1. The extraction starts in the first iteration with a number of seed relation instances. A relation instance is defined as a triple (w, r, w'), which means words w and w' have the relation r.

r	Relation	Description
r_1	w [*Sibling*]$_{is}$ w'	w, w' are sibling terms
r_2	w [*Hyponym*]$_{is}$ w'	w' "is a" w w' is a "Instance-of" w
r_3	w [*Hypernym*]$_{is}$ w'	w "is a" w' w is a "Instance-of" w'
r_4	w [*Meronym*]$_{is}$ w'	w' is a "Part-of" w w' is a "Member-of" w w' is a "Substance-of" w
r_5	w [*Holonym*]$_{is}$ w'	w is a "Part-of" w' w is a "Member-of" w' w is a "Substance-of" w'

Table 1: List of semantic relations.

In a nutshell, during a full iteration, the system

[1]This is much larger than the number of typical Chinese words, which is mainly caused by the huge amount of NEs in the Web corpus, plus typos and word segmentation errors.

first uses seed instances to populate initial patterns that match them from the corpus, and assigns a unique semantic relation to each pattern according to the seeds it matches. Then, the system uses these patterns to extract new word pairs, and assigns relations to the word pairs according to the patterns that extracted them. The whole procedure is described in detail as the following four steps:

(1) Pattern initialization Find all sentences that contain the words w and w' in any seed relation instance (w, r, w'). Each sentence is then split into *prefix*, *infix*, and *suffix* by w and w'. The total length limit of these three parts is 10 words. Within this limitation, the system exhaustively enumerates all possible *prefixes* and *suffixes* (the *infix* remains unchanged). For example, in the sentence "A B w C w' D E", the *prefixes* can be $\{$'A', 'B', 'A B', ' '$\}$ and the *suffixes* can be $\{$'D', 'E', 'D E', ' '$\}$. Then, each combination of "*prefix SLOT*$_1$ *infix SLOT*$_2$ *suffix*" forms the word level of a pattern. Plus the POS and NE tags, a pattern finally contains three levels of information. In addition, to increase the recall of the patterns, named entities at the word level are replaced by their NE tags. This means at these positions, the pattern would match an arbitrary word as long as the word's NE tags are matched. For instance, say the following sentence matches a seed (苹果(Apple), [*Sibling*]$_{is}$, 三星(Samsung)):

近日，[苹果]和[三星]在美国进行了专利诉讼。
(Recently, [Apple] and [Samsung] conducted patent litigation in the U.S.)

A pattern derived from this sentence could be:

Word:	Recently 近日	$\{SLOT_1\}$	and 和	$\{SLOT_1\}$	in 在	U.S. LOC
POS:	t	nz	c	nz	p	ns
NE:	NOR	BRD	NOR	BRD	NOR	LOC

where BRD, LOC stand for brand and location, and NOR means the word is not a NE.

(2) Pattern-relation mapping For pattern p, consider all seed instances (w, r, w') it matched. For each relation r in those seeds, count it w.r.t. p. Then, r is scored by $tfidf$, where tf is r's count, and df is the number of relations that have a non-zero count. Finally, map p to the relation r_{max} with the highest score s_{max}, and assign s_{max} to p as its score.

(3) Instance extraction For each semantic relation r, extract word pairs using top scored 1,000 patterns that are mapped to r. For each sentence, if it matches a pattern p's word sequence and meets the POS / NE tag constraints at *SLOT*$_1$ and *SLOT*$_2$ in p, then the words falling into the two slots are extracted.

Note that different patterns (even those with different relations) may extract the same word pair. To determine the final relation of a word pair $<w$, $w'>$, the system traverses all the patterns that can extract it. The patterns are then grouped by the relations they map to. Within each group, the patterns' scores are added up. The relation $r^\star$ whose group has the highest sum score S is selected. Then, with $r^\star$, a new instance $(w, r^\star, w')$ is generated, with S as its weight. The system will also generates a reversed instance $(w', r^{\star-1}, w)$. E.g., for (Intel, [*Sibling*]$_{is}$ AMD) and (Intel, [*Hypernym*]$_{is}$ Company) , the system also generates (AMD, [*Sibling*]$_{is}$ Intel) and (Company, [*Hyponym*]$_{is}$ Intel) respectively.

(4) New seed generation Add the top-weighted relation instances obtained in step (3) into the seed set. In the current setting, each relation's top-500 weighted instances are added as new seeds in each iteration.

In practice, the seed set used is small ontology of totally $222k$ instances of the five relations The system produces $27M$ instances after 2 iterations, which are used in our experiments. For each word w, we define its relation words as the words appearing in the slot "w [*relation*]$_{is}$ ␣". E.g., w's [*Hypernym*]$_{is}$ relation words are its hypernyms.

3.3.2 Similarity

Recall that at the distribution layer, each pair of words has a distributional similarity, denoted by$ds(\cdot,\cdot)$. On top of this, we individually develop a relation-based similarity $rs_i(\cdot,\cdot)$ for each semantic relation $r_i \in \{r_1, r_2, \ldots, r_5\}$. For two words w and w', $rs_i(w, w')$ is defined as the average of non-zero distributional similarities between their top-N (at most) relation words under r_i (denoted by $r_i^N(\cdot)$):

$$rs_i(w, w') = \frac{1}{|\{(u,v)|ds(u,v) > 0\}|} \sum_{\substack{u \in r_i^N(w) \\ v \in r_i^N(w')}} ds(u, v) \tag{1}$$

In our experiments, we universally set N to 10.

One alternate practice is to directly calculate the traditional cosine similarity between the relation word distributions of w and w'. We do not take this approach because such manner suffers from data sparseness. In particular, sometimes the relation words of w and w' are quite similar, but none of them are shared by w and w' (this means the cosine similarity will be 0). For instance:

- *hand* and *head* may not share any meronym, e.g. *hand* only has meronym *finger* while *head* has *eye*, *nose*, $\ldots$;
- *Carmel* (a small city in IN, U.S.) may only have the hypernym $\{$*small city*$\}$ while *New York* may have $\{$*city*, *big city*$\}$.

In our approach, owing to the non-zero $ds(\cdot,\cdot)$ between *finger* / *eye*, or between *small city* / *big city*, the two pairs of words will have positive relation-based similarities.

3.4 Fusion Layer: Learning-to-rank

Eventually, we fuse $ds(\cdot,\cdot)$ and $rs_i(\cdot,\cdot)$ together to get the final similarity prediction of each pair of words. We choose a straightforward manner by linearly combining $ds(\cdot,\cdot)$ and $rs_i(\cdot,\cdot)$:

$$\text{FUSE}(w,w') = \alpha_d \cdot ds(w,w') + \sum_i \alpha_i \cdot rs_i(w,w') \quad (2)$$

Note that the relation-based similarities are built upon $ds(\cdot,\cdot)$ (Eq. 1), so FUSE is essentially a hierarchical combination of $ds(\cdot,\cdot)$ guided by semantic relations. Linear combination is simple, but turns out to be effective through the experiments. More elaborated fusion method may be invested in future studies.

To get the weights α_d and α_i, we adopt pair-wise learning-to-rank technique rather than regression. This is because it is difficult to assign an absolute score of semantic similarity to a pair of words, especially when seasoned linguists are not available. On the other hand, given two word pairs <*A, B*> and <*A,C*>, it is relatively easier to tell whether <*A, B*> is more similar than <*A,C*>, or vice versa.

We use the ranking option of SVM^{light} v6.02 (Joachims, 1999) with linear kernel to optimize the weights against human judgements. The goal of the learning process is to minimize the number of wrong pair-wise similarity comparisons. In the testing phase, the model assigns to each testing sample a real-value prediction, which is exactly a linear fusion of the corresponding sub-similarities. As for the technical details in SVM^{light}, each word pair <*X, Y*> yields a sample. If the human judgement suggests <*A, B*> is more similar than <*A, C*>, then the corresponding samples will be assigned to an unique sample group, with the target values 1 and 0 respectively. If a sub-similarity value does not exist due to out-of-vocabulary issue, the corresponding feature is set to "missing".

The judgement is obtained from a Chinese thesaurus (HIT-SCIR, 2006), containing 77,458 words that are manually grouped according to a five-level category hierarchy. Words grouped together at the lowest(fifth) level include both synonyms (e.g. *sea* / *ocean*) and comparable terms (e.g. *Germany* / *France*). The lower the level two words appear in together, the more semantically related they are. We directly use this clue to determine the semantic similarity between words. For instance, words that appear together in a level-3 category but not in any level-4 category have a similarity of 3. Words do no appear together in any category have a zero-similarity.

After a pilot study, we found that words with similarities 0-2 are indistinguishably dissimilar. So we merged these similarity levels together as zero-similarity. Moreover, to further distinguish semantically-similar and comparable words, we set the similarity between synonyms to 6 instead of 5 for comparable terms. Finally, we got five similarity levels: 0, 3, 4, 5, and 6. To make the experiment manageable, we randomly sample 200 nouns from the thesaurus and extract their similar words at every level, and arrange them as a serial of similarity judgements like $sim(w,w') > sim(w,w'')$. The whole dataset contains 2,204 words and 6,904 judgements. To avoid the randomness in data, we adopt five-fold cross-validation on it. In each fold, we use 3 parts of the data to train the model, and tune / test it using the other two parts.

4 Evaluation

4.1 Experiment Settings

We compare our fused similarity with three baselines. The first one is classical text-window based distributional similarity $ds(\cdot,\cdot)$. The other two baseline approaches are listed as follows:

Lin's similarity (Lin, 1998) (LIN98). LIN98 combines PMI values from different distributions linearly. The formula uses dependency paths, and we extend it to semantic relations extracted as in Section 3.3.1. As a by-product in the extraction phase, words' co-occurrence counts under each semantic relation are acquired to compute the PMI values. The text-window based distribution is also included in the combination.

Joint cosine similarity (JCS). There is also previous work that uses pattern-constrained context information as extra clue of semantic similarity (Baroni and Lenci, 2010). Different from LIN98, words' co-occurrence counts under each semantic relation are replaced by the number of patterns that extract them (Baroni and Lenci, 2010). Here, the text-window and the relation based distributions are mingled into a single distribution, and cosine similarity is obtained. Baroni and Lenci (2010) uses LMI in relation based distributions, but we found PMI achieves better performance.

The text-window distribution used in both LIN98 and JCS is based on 3-length window and PMI, since this configuration shows the best performance in our experiments. For the relation based distribution in LIN98 and JCS, we initially use the whole (noisy) relation extraction result, and make further analysis by varying the amount (and quality) of the relation instances. LIN98 and JCS also generate integrated similarities based on multiple evidences, we will compare their effectiveness with our approach.

Two evaluations metrics are used:

Accuracy of comparison (*Acc.*). We say a system makes a correct comparison if it returns $S(A,B) > S(A,C)$ that coincides with human judgement. The overall accuracy is defined as the percentage of correct comparisons over the whole dataset.

Spearman's ρ. For each word pair <*A*, *B*>, we count the number of word pairs <*A*, x> that are judged less similar than <*A*, *B*>, and use it as an absolute score of similarity between *A* and *B*. This allows us to compare similarity predictions with such scores globally, and get the ρ coefficient.

To get meaningful conclusions, we use approximate randomization (Noreen, 1989) to test the significance of *Acc.* comparison, and Steiger's Z-test (Steiger, 1980) for Spearman's ρ comparison.

4.2 $ds(\cdot,\cdot)$ Configurations

Distributional similarity $ds(\cdot,\cdot)$ is an important baseline. Moreover, by substituting Eq. 1 into Eq. 2, one will find that $ds(\cdot,\cdot)$ is also the basic building block of the fused similarity. With the multiple choices of the text-window lengths (3 and 6) and association measures (Raw co-occurrence, PMI, Jaccard, and LMI) listed in subsection 3.2, we now try to find out an optimal configuration of $ds(\cdot,\cdot)$. The results are obtained based on the whole dataset, as shown in Table 2. For the $ds(\cdot,\cdot)$ configurations (the first 8 rows), the subscripts are the text-window's length and the superscripts are the association measures used. Performance of LIN98 and JCS is also included (rows 9∼10).

	Acc.	ρ
$ds_{3wd}^{Raw\ cooc}(\cdot,\cdot)$	77.0	0.458
$ds_{6wd}^{Raw\ cooc}(\cdot,\cdot)$	75.2	0.427
$ds_{3wd}^{PMI}(\cdot,\cdot)$	**80.8**	0.522
$ds_{6wd}^{PMI}(\cdot,\cdot)$	77.4	0.438
$ds_{3wd}^{Jac.}(\cdot,\cdot)$	80.1	0.527
$ds_{6wd}^{Jac.}(\cdot,\cdot)$	79.0	0.501
$ds_{3wd}^{LMI}(\cdot,\cdot)$	80.0	**0.544**
$ds_{6wd}^{LMI}(\cdot,\cdot)$	78.2	0.497
LIN98	79.4	0.496
JCS	**82.2**	**0.553**

Table 2: Performance of $ds(\cdot,\cdot)$, LIN98, and JCS

In both *Acc.* and ρ, $ds(\cdot,\cdot)$ with the 3-length window significantly ($p < 0.01$) outperforms that with the 6-length window, except when using Jaccard ($p = 0.12$ for *Acc.*). Although the window length is easy to choose, it remains unclear which association measure is the most appropriate. With the 3-length window, the performance of PMI, Jaccard, and LMI are comparable. Thus, we will have to try out all PMI, LMI and Jaccard in the fusion phase.

As for the other two baseline approaches, JCS significantly outperforms all $ds(\cdot,\cdot)$ configurations in both *Acc.* ($p < 0.05$) and ρ ($p < 0.07$) as shown in bold font, but LIN98 does not.

4.3 Similarity Fusion

In similarity fusion (Eq. 2), for the sake of conciseness, we use the same $ds(\cdot,\cdot)$ configuration to compute both the distributional similarity and the relation-based similarities (Eq. 1). The *Acc.* and ρ of the fused similarity (denoted by FUSE) using different $ds(\cdot,\cdot)$ configurations are shown in Table 3. Recall that because of the indistinguishable *Acc.*, three configurations need to be examined: $ds_{3wd}^{PMI}(\cdot,\cdot)$, $ds_{3wd}^{LMI}(\cdot,\cdot)$, and $ds_{3wd}^{Jac.}(\cdot,\cdot)$.

	$ds(\cdot,\cdot)$ configuration used		
	$ds_{3wd}^{PMI}(\cdot,\cdot)$	$ds_{3wd}^{LMI}(\cdot,\cdot)$	$ds_{3wd}^{Jac.}(\cdot,\cdot)$
Acc.	**83.9**	81.9	78.9
ρ	**0.591**	0.558	0.500

Table 3: Performance of our proposed fused similarity (FUSE) using different $ds(\cdot,\cdot)$ configurations.

In both *Acc.* and ρ, $ds_{3wd}^{PMI}(\cdot,\cdot)$ based FUSE has significantly ($p < 0.005$) superior performance (shown in bold font). In both metrics, it also significantly ($p < 0.01$) outperforms all baseline approaches, including all $ds(\cdot,\cdot)$ configurations, LIN98, and JCS. The results suggest that on our dataset, the most suitable $ds(\cdot,\cdot)$ to use in FUSE is $ds_{3wd}^{PMI}(\cdot,\cdot)$, which achieves 83.9% accuracy in predicting whether a word pair <A, B> is more similar than <A, C>.

As a global comparison, we have the following performance rankings:

$$Acc.: \text{LIN98} <_{0.05} ds_{3wd}^{PMI}(\cdot,\cdot) <_{0.05} \text{JCS} <_{0.01} \text{FUSE}$$
$$\rho: \text{LIN98} <_{0.01} ds_{3wd}^{LMI}(\cdot,\cdot) <_{0.01} \text{JCS} <_{0.01} \text{FUSE}$$

where the subscripts show the significance level. JCS outperforms the best $ds(\cdot,\cdot)$ configurations in both $Acc.$ and ρ, confirming the contribution of the semantic evidences obtained by the in-depth mining in the corpus. Moreover, FUSE achieves even better performance, showing the effectiveness of the design of relation-based similarity (Eq. 1) and the linear combination mechanism (Eq. 2).

Ideally, an effective fusion should have worked for all $ds(\cdot,\cdot)$ configurations. However, FUSE using $ds_{3wd}^{Jac.}(\cdot,\cdot)$ yields bad performance. Through intrinsic analysis we found that $ds_{3wd}^{Jac.}(\cdot,\cdot)$ is more sensitive to the noise in the relation data than $ds_{3wd}^{PMI}(\cdot,\cdot)$.

4.4 Quality of Semantic Relations

Initially, we use all of the extracted relation instances in the experiments without threshold based filtering. Without doubt, there is much noise in the bottom of the extraction results. Through controlling the weight threshold of the relation instances, we now shrink the global extraction results to top ∼5%, ∼10%, ∼30%, and ∼60% subsets to see how their quality and coverage change, and how they affect the performance of FUSE, LIN98, and JCS.

FUSE uses top 10 relation words to calculate the relation-based similarity. Thus, instead of examining the global extraction results, we focus on the top 10 relation words of the target words in our dataset, because FUSE's performance is our main concern.

The full evaluation is expensive. There are totally 2,204 target words in the dataset, involving 70,000 relation words. So we randomly sample 200 words from the 2,204 words, and evaluate the accuracy of their relation words by varying the amount of the global extraction results. The results are summarized in Table 4. While the amount

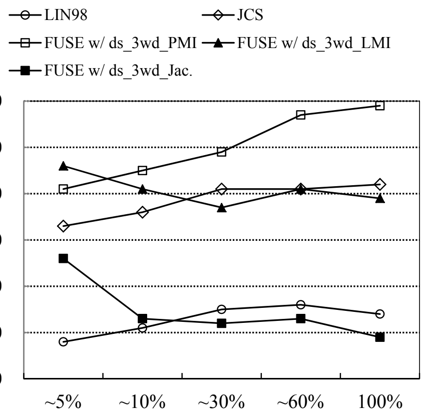

Figure 2: Accuracy of LIN98, JCS, and FUSE varying the amount of the global extraction results.

of global extraction results shrink, low-weight relation words are gradually removed, and the coverage of the relation words decreases. Only the top ∼5% results have acceptable accuracies. Intrinsic study shows that holonyms and meronyms concentrate to location names due to the bias in the seeds. This causes a significantly low quality and coverage for these two relations.

The low-quality extraction results pose an austere challenge. Here, we re-examine LIN98, JCS, and FUSE (5-fold CV using $ds_{3wd}^{PMI}(\cdot,\cdot)$, $ds_{3wd}^{LMI}(\cdot,\cdot)$, or $ds_{3wd}^{Jac.}(\cdot,\cdot)$) based on the four subsets of the global extraction results, and assemble the performance figures in Figure 2. For the sake of space limit, we only include the $Acc.$ results. Spearman's ρ shows a similar trend.

The results further confirms the ranking listed in subsection 4.3. A common finding is that the bottom 40% of the global extraction results are hardly useful. FUSE based on $ds_{3wd}^{PMI}(\cdot,\cdot)$ handles the noise in the data quite well. FUSE based on $ds_{3wd}^{LMI}(\cdot,\cdot)$, or $ds_{3wd}^{Jac.}(\cdot,\cdot)$ seems partial to high-quality data. Similar to $ds_{3wd}^{PMI}(\cdot,\cdot)$ based FUSE, performance of LIN98 and JCS drops while shrinking the number of relation instances. This indicates they prefer recall to precision of the extraction results.

4.5 Feature Analysis

We have shown that the fusion of $ds_{3wd}^{PMI}(\cdot,\cdot)$ and $rs_i(\cdot,\cdot)$ shows superior performance, yet each sub-similarity's contribution remains unclear. Using 100% global extraction results, we incrementally add relation-based similarities to $ds_{3wd}^{PMI}(\cdot,\cdot)$, and report the fusion's cross-validation performance in Table 5. The order of addition is coincide to the quality of the relations (see Table 4).

		∼5%		∼10%		∼30%		∼60%		100%	
	Relation words	# wd.	*Acc.*	# wd.	*Acc.*	# wd.	*Acc.*	# wd.	*Acc.*	# wd.	*Acc.*
r_1	w [*Sibling*]$_{is}$ ␣	1,115	96.6	1,325	88.3	1,532	80.9	1,615	78.7	1,648	77.9
r_2	w [*Hyponym*]$_{is}$ ␣	263	77.6	627	38.8	1,018	27.8	1,227	23.9	1,378	21.8
r_3	w [*Hypernym*]$_{is}$ ␣	608	73.8	1,059	52.0	1,376	44.6	1,488	42.7	1,561	40.9
r_4	w [*Meronym*]$_{is}$ ␣	266	41.7	416	29.6	586	22.7	741	19.4	972	15.1
r_5	w [*Holonym*]$_{is}$ ␣	141	55.3	161	50.9	197	42.6	388	23.2	462	20.1

Table 4: Quantity and quality analysis of the 200 sampled words' relation words.

Feature set	*Acc.*	ρ
ds_{3wd}^{PMI}	80.8	0.522
$ds_{3wd}^{PMI} + rs_1$	83.1	0.559
$ds_{3wd}^{PMI} + rs_1 + rs_3$	83.4	0.587
$ds_{3wd}^{PMI} + rs_1 + rs_3 + rs_2$	83.6	0.587
$ds_{3wd}^{PMI} + rs_1 + rs_3 + rs_2 + rs_5$	83.7	0.590
$ds_{3wd}^{PMI} + rs_{1\sim5}$	83.9	0.591
$ds_{3wd}^{PMI} + rs_2$	81.3	0.522
$ds_{3wd}^{PMI} + rs_3$	82.3	0.574
$ds_{3wd}^{PMI} + rs_4$	81.2	0.532
$ds_{3wd}^{PMI} + rs_5$	81.1	0.550

Table 5: FUSE's performance on sub feature sets.

Unsurprisingly, $rs_1(\cdot,\cdot)$, i.e. [*Sibling*]$_{is}$ based similarity is the most effective, owing to its high quality. $rs_3(\cdot,\cdot)$ ([*Hypernym*]$_{is}$) dominates the rest of the performance improvement, and adding it alone to $ds_{3wd}^{PMI}(\cdot,\cdot)$ also largely improves the performance. It is reasonable since comparing the sibling terms or hypernyms (i.e. "what is it") are natural ways to compare words' meanings. Though "masked" by [*Sibling*]$_{is}$ and [*Hypernym*]$_{is}$, other relations also show their contribution (yet small) when added to $ds_{3wd}^{PMI}(\cdot,\cdot)$ alone. $rs_2(\cdot,\cdot)$ ([*Hyponym*]$_{is}$) is an exception, and its weight is also negative in the trained models. This indicates that hyponyms may not be an adequate evidence for semantic similarity.

Given the bad quality of [*Meronym*]$_{is}$ and [*Holonym*]$_{is}$ relations, their effectiveness seems bizarre. In fact, though a great number of relation words are not correct, they can be considered as special context words. Owing to the design of the relation-based similarity (Eq. 1), the distributional similarities of those words still contribute to the target words' similarity calculation. This finding allows us to relax the quality restriction of semantic relation extraction. Our hierarchical approach to semantics-aware distributional similarity would work on the basis of noisy relation databases.

5 Conclusion

In this paper, we propose a hierarchical semantics-aware distributional similarity scheme (DSS). We introduce a semantic layer over the classical distribution layer by employing a semantic relation extraction system and a mechanism that computes words' relation-specific similarities based on simple distributional similarity. Finally, the fusion of the distributional and relation-based similarities is completed by learning-to-rank.

Experiments show that the in-depth mining in the corpus provides effective evidences for semantic similarity. On our dataset, the fused similarity significantly improves distributional similarity, and also outperforms the baseline approaches that blend the heterogeneous evidences in a single vector. Additionally, intrinsic analysis shows that [*Sibling*]$_{is}$ and [*Hypernym*]$_{is}$ relations are the most effective semantic clues.

In future studies, we will experiment on more elaborated combination similarity fusion mechanisms other that linear combination. We will also explore more types of semantic evidences, e.g. synonym, antonym, semantic attribute, or thematic relations such as agent / patient relations.

Acknowledgments

This work was supported by (1) the National High Technology Research and Development Program of China (863 Program, No. 2011AA01A207), (2) the Natural Science Foundation of China (No. 61272384 & 61105072), (3) China Postdoctoral Science Foundation (No. 2012M510220), and (4) Beijing Postdoctoral Research Foundation (No. 2012ZZ-99).

References

Eneko Agirre, Enrique Alfonseca, Keith Hall, Jana Kravalova, Marius Paşca, and Aitor Soroa. 2009. A study on similarity and relatedness using distributional and wordnet-based approaches. In *NAACL '09*, pages 19–27.

L. Douglas Baker and Andrew Kachites McCallum. 1998. Distributional clustering of words for text classification. In *SIGIR '98*, pages 96–103.

Mohit Bansal and Dan Klein. 2012. Coreference semantics from web features. In *ACL '12*, pages 389–398.

Marco Baroni and Alessandro Lenci. 2010. Distributional memory: A general framework for corpus-based semantics. *Comput. Linguist.*, 36(4):673–721.

Marco Baroni, Brian Murphy, Eduard Barbu, and Massimo Poesio. 2010. Strudel: A corpus-based semantic model based on properties and types. *Cognitive Science*, 34(2):222–254.

Georgiana Dinu and Mirella Lapata. 2010. Measuring distributional similarity in context. In *EMNLP '10*, pages 1162–1172.

Katrin Erk and Sebastian Padó. 2008. A structured vector space model for word meaning in context. In *EMNLP '08*, pages 897–906.

Stefan Evert. 2008. Corpora and collocations. In A. Lüeling and M. Kytö, editors, *Corpus Linguistics. An International Handbook*. Mouton de Gruyter.

Edward Grefenstette, Georgiana Dinu, Yao-Zhong Zhang, Mehrnoosh Sadrzadeh, and Marco Baroni. 2013. Multi-step regression learning for compositional distributional semantics. *CoRR*, abs/1301.6939.

Marti A. Hearst. 1992. Automatic acquisition of hyponyms from large text corpora. In *COLING '92*, pages 539–545.

HIT-SCIR. 2006. Retrieved from: `http://ir.hit.edu.cn/phpwebsite/index.php?module=pagemaster&PAGE_user_op=view_page&PAGE_id=162`.

Alpa Jain and Marco Pennacchiotti. 2010. Open entity extraction from web search query logs. In *COLING '10*, pages 510–518.

Thorsten Joachims. 1999. Advances in kernel methods. chapter Making large-scale support vector machine learning practical, pages 169–184. MIT Press.

Nobuhiro Kaji and Masaru Kitsuregawa. 2008. Using hidden markov random fields to combine distributional and pattern-based word clustering. In *COLING '08*, pages 401–408.

Zornitsa Kozareva, Konstantin Voevodski, and Shang-Hua Teng. 2011. Class label enhancement via related instances. In *EMNLP '11*, pages 118–128.

Dekang Lin. 1997. Using syntactic dependency as local context to resolve word sense ambiguity. In *ACL '97*, pages 64–71.

Dekang Lin. 1998. Automatic retrieval and clustering of similar words. In *ACL '98*, pages 768–774.

Diana McCarthy and Roberto Navigli. 2009. The english lexical substitution task. *Language Resources and Evaluation*, 43(2):139–159.

Shachar Mirkin, Ido Dagan, and Maayan Geffet. 2006. Integrating pattern-based and distributional similarity methods for lexical entailment acquisition. In *COLING-ACL '06*, pages 579–586.

Jeff Mitchell and Mirella Lapata. 2008. Vector-based models of semantic composition. In *Proceedings of ACL-08: HLT*, pages 236–244.

Eric W. Noreen. 1989. *Computer-intensive methods for testing hypotheses*. A Wiley-Interscience publication. Wiley, New York, NY [u.a.].

Marius Paşca. 2004. Acquisition of categorized named entities for web search. In *CIKM '04*, pages 137–145.

Sebastian Padó and Mirella Lapata. 2007. Dependency-based construction of semantic space models. *Comput. Linguist.*, 33(2):161–199.

Patrick Pantel and Marco Pennacchiotti. 2006. Espresso: leveraging generic patterns for automatically harvesting semantic relations. In *ACL '06*, pages 113–120.

Patrick Pantel, Deepak Ravichandran, and Eduard Hovy. 2004. Towards terascale knowledge acquisition. In *COLING '04*.

Marco Pennacchiotti and Patrick Pantel. 2009. Entity extraction via ensemble semantics. In *EMNLP '09*, pages 238–247.

Reinhard Rapp. 2003. Word sense discovery based on sense descriptor dissimilarity. In *Proceedings of the Ninth Machine Translation Summit*, pages 315–322.

Shuming Shi, Huibin Zhang, Xiaojie Yuan, and Ji-Rong Wen. 2010. Corpus-based semantic class mining: distributional vs. pattern-based approaches. In *COLING '10*, pages 993–1001.

Keiji Shinzato and Kentaro Torisawa. 2007. A Simple WWW-based Method for Semantic Word Class Acquisition. In *RANLP 2005*, volume 292 of *Current Issues in Linguistic Theory*, pages 207–216. John Benjamins.

Rion Snow, Daniel Jurafsky, and Andrew Y. Ng. 2004. Learning syntactic patterns for automatic hypernym discovery. In *NIPS*.

James H. Steiger. 1980. Tests for comparing elements of a correlation matrix. *Psychological Bulletin*, 87(2):245–251, March.

Stefan Thater, Hagen Fürstenau, and Manfred Pinkal. 2010. Contextualizing semantic representations using syntactically enriched vector models. In *ACL '10*, pages 948–957.

Peter D. Turney. 2006. Similarity of semantic relations. *Comput. Linguist.*, 32(3):379–416, September.

Fan Zhang, Shuming Shi, Jing Liu, Shuqi Sun, and Chin-Yew Lin. 2011. Nonlinear evidence fusion and propagation for hyponymy relation mining. In *ACL '11*, pages 1159–1168.