TDSS: A New Word Sense Representation Framework for Information Retrieval

Liwei Chen[1,2], Yansong Feng[1], and Dongyan Zhao[1]

[1] Institute of Computer Science and Technology, Peking University, Beijing, China
[2] Baidu Inc., Beijing, China
chenliwei@baidu.com, fengyansong@pku.edu.cn, zhaodongyan@pku.edu.cn

Abstract. Word sense representation is important in the tasks of information retrieval (IR). Existing lexical databases, e.g., WordNet, and automated word sense representing approaches often use only one view to represent a word, and may not work well in the tasks which are sensitive to the contexts, e.g., query rewriting. In this paper, we propose a new framework to represent a word sense simultaneously in two views, explanation view and context view. We further propose an novel method to automatically learn such representations from large scale of query logs. Experimental results show that our new sense representations can better handle word substitutions in a query rewriting task.

Keywords: word sense induction, graph clustering, query rewriting

1 Introduction

As the amount of the information on the Internet increasing exponentially, there are growing demands in the information retrieval (IR) industry to understand the queries better, so as to provide the users with more accurate and diverse results. Such query understanding tasks require fine modeling of word meanings to capture subtle semantic differences of words. For example, query rewriting, an important IR task in many realistic search engines, should rewrite a query with synonymous or similar words with the ones in the query, so as to obtain more retrieval results. In practice, in order to avoid semantic drift of query meanings, we mainly use synonyms to rewrite the query. Take the query "五道口附近哪里吃早点 (Where can I have breakfast in Wudaokou)" as an example, we would like to use "早餐(breakfast)" to substitute "早点(breakfast)" as a rewriting of the query. However, many words in queries are ambiguous, and we need to use the context to determine which sense a word would use, and the synonymous words of this sense will be used to rewrite the query. This means that the task should exploit two views of a word sense: the explanation view, containing synonyms of the sense, and the context view, representing in what contexts this sense is used. Unfortunately, most existing works on word sense representations are not suitable for the task.

Traditional lexical databases, such as WordNet [11, 10], groups words into sets of synonyms called synsets, providing short definitions and usage examples. It only uses the explanation view to represent the word senses, while the context in the usage examples is rather limited. Furthermore, it requires lots of human efforts to construct and update, thus is difficult to adapt to other domains or languages.

On the other hand, automated word sense induction (WSI) has attracted more and more attention. Previous works on WSI mainly focus on characterizing word meanings by modeling the contexts or descriptions of the ambiguous word, including unsupervised clustering [15, 14, 5], or topic models [2]. Recent research efforts also attempt to build a continuous vector to represent a word or a sense of a word [9, 7, 12, 16, 6, 4, 3], and the models are usually trained on the contexts and/or the textual descriptions of the words. Those approaches often use only one view to represent a word, which makes the two aspects of word sense interact with each other, and may confuse the query rewriting model.

In this paper, we propose a novel framework, two-dimensional semantic space (TDSS), which jointly use two vectors, the explanation vector and the context vector, to represent a word sense. The explanation vector is generated using synonyms, serving as a description of this sense. For convenience, we also call those words as its *explanation words*. The context vector is constructed based on the corresponding context of the sense. We further propose an approach to obtain such TDSS representations for word senses from large scale query logs. The explanation words and context words are extracted from the query paraphrases, and further grouped into multiple senses. Experimental results on 33 Chinese words and a query rewriting case study show that our approach can output reasonable word senses, which can be further used in the task of query rewriting.

2 Related Work

In this section, we briefly review previous studies on traditional context-oriented WSI methods.

Previous studies in WSI are often context-oriented. Those approaches can be divided into three categories: unsupervised clustering approaches, generative approaches and word embedding approaches.

In cluster-based approaches, WSI is treated as a clustering problem. The mentions of a target word are grouped into several clusters according to the similarity of their contexts. Many different clustering algorithms have been used so far, e.g., k-means [14], agglomerative clustering [15], information bottleneck [13].

The graph-based cluster methods can also be used in the task of WSI [8], where words are the nodes and the co-occurrence between words are the edges. Community detection algorithms can be employed to discover word communities in the graph, which is used to represent word senses.

Generative approaches assume that different senses have different lexical distributions. For example, [2] utilizes a parametric Bayesian model, i.e., LDA, to solve the WSI task. The word senses are characterized as distributions over words and an ambiguous word is then drawn from a distribution of senses. In order to automatically decide the number of senses, instead of LDA, [17] propose to use a nonparametric Bayesian model, called Hierarchical Dirichlet Process (HDP).

Recently word embedding approaches have attracted more and more attention In those approaches, a word or the sense of a word is often represented by a continuous vector which is built by neural network algorithms [9, 7, 12, 16, 6, 4, 3]. Most approaches use the contexts and/or the textual descriptions of the words to train those models.

3 A New Word Sense Representation Framework

In TDSS, we represent the sense of a word in two views, explanation view and context view. We use synonyms of a word to generate its explanation view. Take the word "看(see, look)" as an example, the explanation view of one of its senses may consist words such as "阅读(read)" and "浏览(browse)".

For the context view, we extract words from the contexts where this meaning is used. In the example of the word "看(see, look)", we can use the words which often exist in its contexts, such as "书(book)", "报纸(newspaper)", etc., to generate the context view.

Formally, given a word w with k senses, its ith sense can be represented as a tuple $S_i =< \boldsymbol{E}_i, \boldsymbol{C}_i >$, where $\boldsymbol{E}_i$ is a vector representing the explanation of this sense, and $\boldsymbol{C}_i$ is a vector representing the context of this sense. We also assign a popularity for $\boldsymbol{E}_i$, which indicates how popular $\boldsymbol{E}_i$ is among all sense explanations of the target word. We restrict that the popularity of all sense explanations of a word should sum up to 1:

$$\sum_{i=1}^{k} Pop(\boldsymbol{E}_i) = 1 \tag{1}$$

Generally speaking, the senses used in more common and diverse contexts are often more popular. We calculate the popularity of a sense based on the probability of the sense's contexts:

$$Pop(\boldsymbol{S}_i) = \frac{P(\boldsymbol{C_i})}{\sum\limits_{j=1}^{k} P(\boldsymbol{C_j})} \tag{2}$$

In this paper we construct the vectors as a simple bag of words (BoW) model, where each dimension of the vector is a word in the sense explanations or contexts. It is possible to use more sophisticated models such as topic models or word embedding, which will be our future work.

4 TDSS Sense Extraction

In order to automatically obtain the TDSS representations of word senses, we first extracting paraphrases from the query logs, and detect word alignments between those paraphrase pairs. Then we extract substitution pairs as well as the corresponding contexts based on the alignments. The substitutions of a word will be considered as its explanation words. Finally, for a given word, we collect all its substitutions as well as the corresponding contexts, adopt a clustering algorithm to group the substitutions into different senses and further obtain the TDSS representation from each sense.

4.1 Explanation Words and Context Extraction

Generating Paraphrases Given the query logs, we adopt the approach in [18] to generate sentential paraphrases from them, which will be further used as the sources of mining explanation words and contexts of word meanings.

Word Alignment For each pair of sentential paraphrases, we align the words in one paraphrase to their corresponding substitution words in the other. We adopt a rule-based aligning strategy for the word alignment. Two words are aligned if they are the same, synonyms (according to an existing thesaurus), or the words inside a window around them (window size is set to 1) are the same or synonyms.

Substitution and Context Extraction Generally speaking, two aligned words a and b in a paraphrase pair can be synonyms, co-hyponyms or hypernyms, etc. Thus, we can use a word's substitutions to form the explanation views of its senses. For each sense, we also need a context vector $\boldsymbol{C}$. The words inside a fixed window size (set to 3 empirically) of the query paraphrases are extracted to generate the context vectors.

Now for each substitution pair, we will compute the substitution probability approximately based on all paraphrases. Given a word a and its substitution b, the probability of a is substituted by b can be calculated as:

$$p(a \rightarrow b) = \frac{freq(a \rightarrow b)}{\sum\limits_{b_i \in B} freq(a \rightarrow b_i)} \tag{3}$$

where $freq(a \rightarrow b)$ is the times a is substituted by b in total, B is all words which can substitute a.

It is also important to estimate the probability of the target word w substituted by the explanation word e given the context word c. Formally, $p(w \rightarrow e|c)$ can be calculated as:

$$p(w \rightarrow e|c) = \frac{freq(w \rightarrow e|c)}{\sum\limits_{e_i \in E_c} freq(w \rightarrow e_i|c)} \tag{4}$$

where $freq(w \rightarrow e|c)$ is the times that w is substituted by e given the context word c, E_c is the set of all words that can substitute w under the context word c.

4.2 Sense Graph Construction

In our approach, the senses of a word are captured from the sense graph of this word. The graph is constructed based on the substitution pairs and the context words extracted in the previous subsection. For a given word, we collect all its substitution words, which can be seen as its explanation words. Then we connect the obtained explanation words by their pairwise relatedness estimated based on the substitution probabilities and context words obtained in the previous subsection.

Graph Pruning Since there are lots of noises in the generated substitution pairs, we need to prune the sense graph to reduce the noises. The following strategies are used to prune the graph.

The first strategy is simple: we prune any explanation words with low substitution probabilities or low substitution frequencies. The pruning thresholds are set empirically for this strategy.

The second strategy utilizes triangle-like substitutions to prune unreliable graph nodes and edges. The assumption is that a triangle-like substitution structure are more stable and reliable, and can help us to prune the noises from the graph. Suppose we have a node a in the sense graph of a word w, which means w can be substituted by a. If we can find another node b in the graph satisfying a can be replaced by b, then we reserve both a and b. Otherwise, a will be pruned from the graph. Figure 1 illustrates a triangle-like substitution structure, where 去(go on a trip) is the word w, 游览(travel) is the word a and 旅行(take a trip) is the word b.

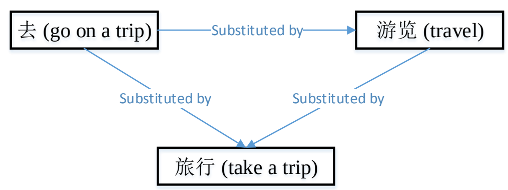

Fig. 1. Illustration of the triangle-like substitution structure.

Edge Weighting The weights of the edges in the sense graph indicate the relatedness between the explanation words. Given the sense graph of a word w, and two nodes a and b in this graph, the edge e_{ab} between the two nodes is weighted as:

$$weight(e_{ab}) = \alpha sub(a, b) + (1 - \alpha)sim(a, b) \quad (5)$$

$sub(a, b)$ is the substitution relatedness between a and b, which is calculated as the average value of $p(a \to b)$ and $p(b \to a)$. $sim(a, b)$ is the context similarity (cosine similarity) between a and b, calculated based on the contexts in which w is substituted by a and b respectively. The context word c is weighted by the conditional probability $p(a \to b|c)$. α is a parameter used to adjust the importance of the two components. It will be optimized on a held out set.

After the pruning and edge weighting step, for each target word, we obtain a sense graph from all remaining substitutions as well as their pairwise weightings. Figure 2 illustrates an abridged version of a sense graph for the word 看. The meanings of 看 in Chinese includes 看望(visit), 治疗(cure) and 观察(observe), etc.

4.3 Sense Generation and Weighting

After we construct the sense graph for a word, we adopt a clustering method to group the explanation words into several clusters, and further generate the TDSS representation for the senses of this word.

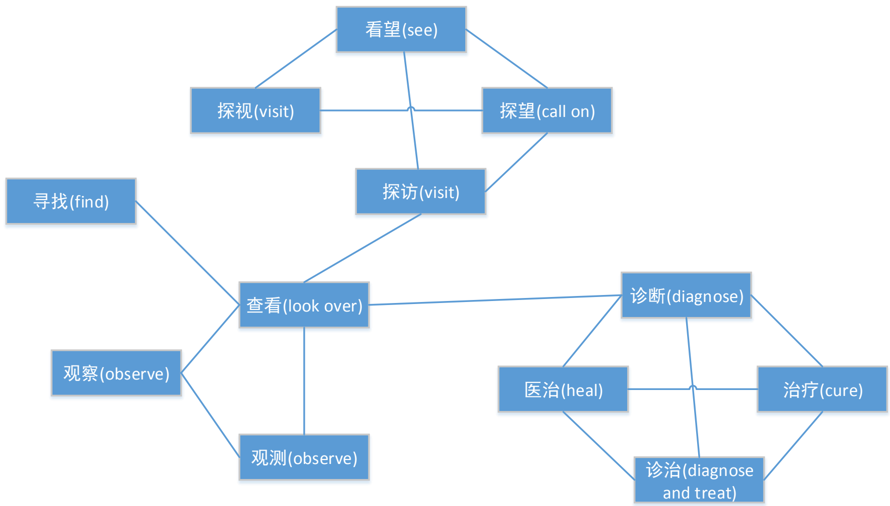

Fig. 2. An abridged example of the semantic graph. The weights of the edges are also omitted for the sake of brevity.

Graph Clustering Now we will cluster the explanation words and context words into different senses. Many algorithms can be used for this task, such as KMeans, HAC, and density-based clustering algorithms. It is more suitable to use the algorithms which do not need to determine the number of clusters in advance, since it is difficult for us to know the exact numbers of the senses of each word.

We choose a label propagation algorithm to cluster the graph. All nodes of the graph will be initialized with a different sense. In each iteration, for each node of the graph, we collect all its neighbors, and the score of each sense is calculated by summing the weights of all neighbors which support it. The senses whose scores are higher than a threshold will be the senses of this node. If the senses of less than 10% nodes are modified in an iteration, the procedure will stop.

Sense Generation After the clustering, we can obtain the TDSS representations for senses of the target word w. Each cluster can be used to construct the explanation vector of one sense. The value of each dimension in an explanation vector $\boldsymbol{E}$ is the substitution probability of the corresponding explanation word:

$$\boldsymbol{E} = (p(w \to e_1), p(w \to e_2), ..., p(w \to e_m)) \tag{6}$$

where e_i is the explanation words in the cluster. For each sense, we aggregate all its sense words' contexts together, and filter the ones whose frequencies are too low. The remaining words form the $\boldsymbol{C}$ vector of this sense. The value of the elements in $\boldsymbol{C}$ is the conditional probability that the target word w is substituted by the explanation words of the current sense given the corresponding context word c. Thus, $\boldsymbol{C}$ is computed as:

$$\boldsymbol{C} = (p(w \to E|c_1), p(w \to E|c_2), ..., p(w \to E|c_n)) \tag{7}$$

where E is the set of the explanation words of the corresponding sense, and $p(w \to E|c_i)$ is calculated as:

$$p(w \to E|c_i) = \sum_{e_j \in E} p(w \to e_j|c_i) \tag{8}$$

According to the definition of the sense popularity in Section 3, given a word w which has k senses, the popularity of S_i can be computed as:

$$Pop(S_i) = \frac{\sum\limits_{c \in C_i} freq(c)}{\sum\limits_{j=1}^{k} \sum\limits_{c \in C_j} freq(c)} \tag{9}$$

where $freq(c)$ is the frequency of the context word c, C_i is the set of context words of S_i. Since the popular senses are usually more important for the users, we collect the senses with the top n popularity as our results. Table 1 shows some samples of the generated senses for the word 打(beat).

Table 1. Samples for Induced Word Senses

Word	Sense word cluster S_i and context cluster C_i
打(beat)	$Sense_1$: 打针(injection), 注射(injection) $Context_1$:美白(whitening), 乙肝 (hepatitis B), 麻药(Anesthetic)
	$Sense_2$: 打仗(to war), 攻略(strategy), 过关(pass game level), 进攻(attack) $Context_2$:剑灵(Blade Soul), 装备 (equipment), 外挂(plug-in), 技能(skills)
	$Sense_3$: 标出(mark), 写成(written), 键入(keyboard input), 打出(show) $Context_3$:WPS, 括号(parentheses), 平方(square), 文档(document)
	$Sense_4$: 毒打(beat cruelly), 挨打(be beaten), 掌掴(Slap), 殴打(beat up) $Context_4$:苏醒(wake up), 婴儿(baby), 老公(husband), 保安(guard)

5 Experiments

We construct a dataset with 33 ambiguous Chinese words to evaluate the performance of our model. Those words are manually selected from the ambiguous words in the Chinese query logs, and the dataset is designed to be representative for different kinds of words, including nouns, verbs, adjectives, adverbs, words with only one character, and words with multiple POS tags. Resolving the ambiguity of those words is highly important for the task of query understanding in the industry. The words we use are listed in Table 2: Each word's explanation words are collected and clustered into senses manually by three volunteers. We integrate their annotations by majority vote. This dataset will be our gold standard for evaluating the explanation view. We do not manually generate the context views because they usually contain too many words. The context views will be evaluated implicitly in the case study.

Table 2. The words used in our evaluation. The English words afterwards illustrate two representative meanings of the Chinese words.

戒(ring, give up, etc.)	完(whole, finish, etc.)
淡薄(weak, thin, etc.)	新鲜(fresh, novel, etc.)
黄色(yellow, adult, etc.)	信(letter, trust, etc.)
门槛(sill, threshold, etc.)	包(wrap, bag, etc.)
加油(fuel charging, cheer up, etc.)	直(straight, directly, etc.)
去(leave, remove, etc.)	放(put, release, etc.)
热(hot, popular, etc.)	代表(represent, representative, etc.)
光(light, smooth, etc.)	好(good, like, etc.)
透(through, completely, etc.)	便宜(cheap, interest, etc.)
挂钩(hook, link up with, etc.)	算账(reckoning, get even with, etc.)
水分(humidity, exaggeration, etc.)	同志(comrade, gay, etc.)
看(look, read, etc.)	口(mouth, opening, etc.)
宽(wide, relieve, etc.)	结实(solid, strong, etc.)
打(hit, from, etc.)	分(branch, divide, etc.)
赶(rush for, catch up with, etc.)	花(flower, spend, etc.)
早点(breakfast, a bit earlier, etc.)	疙瘩(pimple, knot, etc.)
负(negative, burden, etc.)	

The query logs we use are from one popular Chinese search engine. We collect the queries from the year 2012 and 2013, producing about 230,000,000 paraphrase pairs.

5.1 Evaluating Explanation Word Extraction

It is almost impossible to compute the exact recall of explanation words extraction because we can only collect the ones which are common. Thus, we only consider the precision as well as the recall with respect to common explanation words against human annotations. The results show that our explanation word extraction can achieve *a precision of 62.9%, and a recall of 79.1%*. In order to study how the incorrect extractions are generated, we randomly select 200 incorrect substitutions, and manually investigate the paraphrases from which they are extracted. We summarize the major reasons with the top 3 proportion are: incorrect paraphrase mappings, incorrect word segmentations and typos or grammatical errors in the queries. The proportion of each reason is listed in Table 3.

Table 3. The major reasons of incorrect substitution extraction.

Reasons	**Percentage(%)**
Incorrect paraphrase mappings	37.0
Incorrect word segmentations	32.0
Typos or grammatical errors	35.0

The top 1 reason is incorrect paraphrase mappings. Since the paraphrases are automatically generated from query logs, they inevitably contain some noises and incorrect mappings. Sometimes a user may click a title which is highly related but not the same as the query. For example, the query is “如何看qq空间照片”(How to see the pictures in the QZone), but the title which the user actually clicked is “如何上传qq空间照片”(How to upload the pictures to the QZone). This will make “看”(see) be aligned to “上传”(upload), and afterwards an incorrect substitution is extracted.

Second, incorrect word segmentation results may also invite incorrect substitutions. In Chinese natural language processing, word segmentation is an important and necessary preprocessing step. The errors in word segmentations would certainly effect the task of word alignments, and further bring noises to substitution extraction. For instance, in the paraphrases “有一个人让我好想念”(There is a person I miss so much) and “有一个女孩让我好想念”(There is a girl I miss so much), the correct segmentation of the last three characters should be “好(so much) 想念(miss)”, but the second sentence is wrongly segmented as “好想(miss so much) 念(miss)”. It makes “好(so much)” be aligned to “好想(miss so much)”, which does not make sense.

Finally, the queries input by the users, and the titles of the clicked webpages, are often not well normalized. Some of them may contain wrongly written or mispronounced characters, some of them may even contain serious grammatical errors. Under these circumstances, a word may be aligned to another word which is fully unrelated with it. Take the paraphrases “怎么在战网上完魔兽”(How to complete Warcraft on Battle.net) and “怎么在战网上打魔兽”(How to play Warcraft on Battle.net) as an example, the word “ 完”(complete) in the first sentence should actually be the word “玩”(play), which is a synonym of the word “打”(play) in this context. Because “完”(complete) and “玩”(play) pronounce the same in Chinese, sometimes the users may use the first word by mistake. This will generate an incorrect substitution between “完”(complete) to “打”(play).

We can observe from Table 3 that the sum of the proportions of the three reasons is larger than 100%. This is because some incorrect substitutions are extracted for multiple reasons.

5.2 Evaluating Word Sense Generation

We use the B-cubed criterion [1] to evaluate the performance of the explanation word clustering. We also manually evaluate the performance of generated word senses according to the explanation view. The results of the top 20 senses are are listed in Table 4.

Table 4. The performance of explanation words clustering.

	P(%)	R(%)	FMeasure(%)
Clustering	89.4	49.3	60.1
Word Sense	67.0	59.7	63.1

We can observe that the senses we generated achieve a precision of 67%. The correct explanation views can represent the sense of a word well. For example, one sense of the word “看(see)” consists of explanation words “治疗(treat a disease)”,“ 治(cure)” and “医治(heal)”. We can also see that the clustering precision is much higher than the recall. This is because we tune the parameters to make the clusters smaller but more accurate, so as to decrease the effect of the noises in the explanation words. Furthermore, in most tasks of query understanding, a smaller but more accurate cluster would be better than a larger cluster with noises.

5.3 Case Study: Query Rewriting

Now we will evaluate how the TDSS representations of word senses perform in the query rewriting task compared with two strong continuous word representation approaches, CBOW and skip-gram [9]. We randomly collect 921 queries containing the 33 Chinese words, and manually labeled the words which can substitute the target word.

For CBOW and skip-gram baselines, we first selected the top 100 similar words from the vocabulary as candidates. Then we compute the similarity between the candidates and the contexts of the target words in the queries, and the ones above a threshold will be collected as the rewriting words. We use the extracted query paraphrases which contain the 33 words to train the CBOW and skip-gram model.

For our TDSS word senses, we select the best sense for the target word based on the similarities between the query and the context views, and the explanation words of the selected sense will be used as the rewriting words. As for the evaluate criterion, we use the precision of the generated rewriting words. Since it is difficult to obtain the exact recall of the results, we use the average number of correct rewriting words of all the queries in the dataset instead. The results are listed in Table 5.

Table 5. The performance of TDSS and the baselines in query rewriting.

Approach	**Precision(%)**	**Average Number of Correct Rewriting Words**
CBOW	19.0	1.8
Skip-Gram	6.4	**3.2**
TDSS	**51.2**	2.9

From the results we can observe that our TDSS representations can obtain the best precision. This is mainly because that it separates the essentials of a word sense, explanations and contexts, into two views, while the existing approaches, i.e., CBOW and skip-gram, combines them into one view, which may blur the differences of the two types of information. Skip-Gram can obtain a little more correct rewriting words than TDSS, but in practice the precision of the words are more important for us since a incorrect substitution would invite incorrect retrieval results, which may decrease the user experience. The results also imply that the context view works well in query rewriting, which implicitly proves that our approach can generate reliable context view for word senses.

6 Conclusions

In this paper, we describe a novel word sense representation framework, which captures the sense of a word in two separate views, explanation and context, and further propose an approach to extract such representations from large scale of query logs, without replying on much human involvement. Experimental results on a Chinese dataset show that our new word sense representation framework can help better handle information retrieval tasks, such as query rewriting, where fine modelling of word meanings is desired. For further work, we will look for more robust representations for word meanings to better represent the two views, and will also attempt to apply the obtained word senses to more retrieval applications.

Acknowledgement.We would like to thank Ben Xu, Wensong He, Shuaixiang Dai, Xiaozhao Zhao, Qiannan Lv, and the anonymous reviewers for their helpful feedback. This work is supported by National High Technology R&D Program of China (Grant No. 2015AA015403, 2014AA015102) and Natural Science Foundation of China (Grant No. 61202233, 61272344, 61370055). For any correspondence, please contact Liwei Chen.

References

1. Bagga, A., Baldwin, B.: Entity-based cross-document coreferencing using the vector space model. International Conference on Computational Linguistics-volume pp. 79–85 (1998)
2. Brody, S., Lapata, M.: Bayesian word sense induction. In: Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics. pp. 103–111. Association for Computational Linguistics (2009)
3. Chen, T., Xu, R., He, Y., Wang, X.: Improving distributed representation of word sense via wordnet gloss composition and context clustering. In: Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers). pp. 15–20. Association for Computational Linguistics, Beijing, China (July 2015), `http://www.aclweb.org/anthology/P15-2003`
4. Chen, X., Liu, Z., Sun, M.: A unified model for word sense representation and disambiguation. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). pp. 1025–1035. Association for Computational Linguistics, Doha, Qatar (October 2014), `http://www.aclweb.org/anthology/D14-1110`
5. Dorow, B., Widdows, D.: Discovering corpus-specific word senses. In: Proceedings of the tenth conference on European chapter of the Association for Computational Linguistics-Volume 2. pp. 79–82. Association for Computational Linguistics (2003)
6. Guo, J., Che, W., Wang, H., Liu, T.: Learning sense-specific word embeddings by exploiting bilingual resources. In: Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers. pp. 497–507. Dublin City University and Association for Computational Linguistics, Dublin, Ireland (August 2014), `http://www.aclweb.org/anthology/C14-1048`
7. Huang, E., Socher, R., Manning, C., Ng, A.: Improving word representations via global context and multiple word prototypes. In: Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). pp. 873–882. Association for Computational Linguistics, Jeju Island, Korea (July 2012), `http://www.aclweb.org/anthology/P12-1092`

8. Jurgens, D.: Word sense induction by community detection. In: Proceedings of TextGraphs-6: Graph-based Methods for Natural Language Processing. pp. 24–28. Association for Computational Linguistics (2011)
9. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. CoRR abs/1301.3781 (2013), http://dblp.uni-trier.de/db/journals/corr/corr1301.html#abs-1301-3781
10. Miller, G.A.: Wordnet: a lexical database for english. Communications of the ACM 38(11), 39–41 (1995)
11. Miller, G.A., Beckwith, R., Fellbaum, C., Gross, D., Miller, K.J.: Introduction to wordnet: An on-line lexical database*. International journal of lexicography 3(4), 235–244 (1990)
12. Neelakantan, A., Shankar, J., Passos, A., McCallum, A.: Efficient non-parametric estimation of multiple embeddings per word in vector space. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). pp. 1059–1069. Association for Computational Linguistics, Doha, Qatar (October 2014), http://www.aclweb.org/anthology/D14-1113
13. Niu, Z.Y., Ji, D.H., Tan, C.L.: I2r: Three systems for word sense discrimination, chinese word sense disambiguation, and english word sense disambiguation. In: Proceedings of the 4th International Workshop on Semantic Evaluations. pp. 177–182. Association for Computational Linguistics (2007)
14. Purandare, A., Pedersen, T.: Word sense discrimination by clustering contexts in vector and similarity spaces. In: Proceedings of the Conference on Computational Natural Language Learning. vol. 72. Boston (2004)
15. Schütze, H.: Automatic word sense discrimination. Computational linguistics 24(1), 97–123 (1998)
16. Tian, F., Dai, H., Bian, J., Gao, B., Zhang, R., Chen, E., Liu, T.Y.: A probabilistic model for learning multi-prototype word embeddings. In: Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers. pp. 151–160. Dublin City University and Association for Computational Linguistics, Dublin, Ireland (August 2014), http://www.aclweb.org/anthology/C14-1016
17. Yao, X., Van Durme, B.: Nonparametric bayesian word sense induction. In: Proceedings of TextGraphs-6: Graph-based Methods for Natural Language Processing. pp. 10–14. Association for Computational Linguistics (2011)
18. Zhao, S., Wang, H., Liu, T.: Paraphrasing with search engine query logs. In: Proceedings of the 23rd International Conference on Computational Linguistics. pp. 1317–1325. Association for Computational Linguistics (2010)