2012 IEEE Sixth International Conference on Semantic Computing

A Context Expansion Method for Supervised Word Sense Disambiguation

Francisco Tacoa, Danushka Bollegala and Mitsuru Ishizuka
Graduate School of Information Science and Technology
The University of Tokyo
Tokyo, Japan
Email: tacoa@mi.ci.i.u-tokyo.ac.jp, danushka@iba.t.u-tokyo.ac.jp, ishizuka@i.u-tokyo.ac.jp

***Abstract*—Feature sparseness is one of the main causes for Word Sense Disambiguation (WSD) systems to fail, as it increases the probability of incorrect predictions. In this work, we present a WSD method to overcome this problem by using an automatically-created thesaurus to append related words to a specific context, in order to improve the effectiveness of candidate selection for an ambiguous word. We treat the context as a vector of words taken from sentences, and expand it with words from the thesaurus according to their mutual relatedness. Our results suggest that the method performs disambiguation with high precision.**

I. Introduction

Word Sense Disambiguation (WSD) is a necessary task when we have a text or sentence containing one or several ambiguous words. This is a critical intermediate process for the correct performance of other subsequent tasks such as Machine Translation (MT), Information Retrieval (IR), Question Answering (QA), etc.

Consider the following sentences containing the ambiguous word *bank*: **1)** "I deposited money in the *bank*", and **2)** "He sat on the *bank* of the river". Now, let us assume that *bank* has the senses *finance* and *land*. The WSD task will consist of assigning one of these senses to each word *bank*. One method to solve this problem is to perform word overlapping [1] between the words from the sentences (the context) and the words from a formal definition of the different senses of *bank* (such as the glosses available from WordNet [2]). The correct sense for an ambiguous word will be the one with the greatest number of overlaps. For instance, suppose that we have the following lists of words, one for the *finance* sense: [finance, institution, accept, **deposit**, channel, **money**, lend, activity], and other for the *land* sense: [river, slope, **land**, body, water, watch, current], where texts in boldface denote the overlapping words. The total of overlaps for the different senses will be:

- Sentence 1: $\{\boldsymbol{finance} = 2, land = 0\}$
- Sentence 2: $\{finance = 0, \boldsymbol{land} = 1\}$

In consequence, we assign the *finance* sense in sentence 1 and the *land* sense in sentence 2.

There are some fundamental drawbacks in this method: 1) sentences not always contain words that overlap with those from the lists or vice-versa; 2) instead of words appearing both in the sentence and lists, we could find pairs of related words such as "water" and "river". These cases of low or no overlap are a consequence of feature sparseness. In this work, we propose a method to overcome this sparseness issue by constructing a Sense Sensitive Thesaurus (SST), a lexical resource that contains word lemmas, where each lemma pairs are related according to a measuring score explained later. We use this thesaurus to expand the context information, by adding related words not appearing in the sentences.

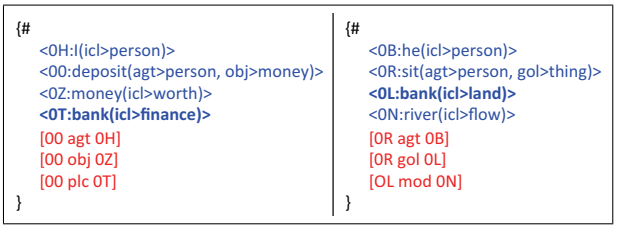

Figure 1. CDL notation for sentences 1 (left) and 2 (right). Blue texts represent entities and red texts represent relation between entities. Text in bold denotes the ambiguous word *bank* with its different senses.

II. Method

Our method requires a sense-annotated corpus. We are using a corpus annotated with the Concept Description Language (CDL) [3]. In CDL notation, entities are unambiguous concepts known as *universal words* (UWs) containing a headword and a constraint in the format HEADWORD(CONSTRAINT). For instance, the UWs for the senses of the word *bank* will be **bank(icl>land)** and **bank(icl>finance)**. *icl* (included in) is a CDL-specific semantic role [4] that indicates that *bank* is a subcategory of both *land* and *finance* in the CDL vocabulary. Figure 1 shows an example of the CDL format.

A. Co-occurrences Matrix

From the labeled corpus we create a co-occurrences matrix **A**, where rows contain the UWs and columns contain POS-tagged lemmas. We are considering co-occurrence at sentence level. To lemmatize and POS-tag the words, we used the Stanford Parser [5].

978-0-7695-4859-3/12 $26.00 © 2012 IEEE
DOI 10.1109/ICSC.2012.27

IEEE computer society

B. Weighted Matrix

Here, we employ an association measure for senses and lemmas, which contains proximity and similarity factors.

1) Proximity Factor: We use the decaying factor proposed in [6], which assigns higher proximity values for closer pairs of words. The decay rate α is set as explained in Section III-B1.

2) Similarity Factor: We use $\mathbf{A}$ to compute the Pointwise Mutual Information (PMI) [7] between sense and lemma pairs, according to the following equation:

$$\mathrm{PMI}'(s,l) = \mathrm{PMI}(s,l) \times df(s,l) \quad (1)$$

where $df(s,l)$ is the discounting factor proposed in [8].

Finally, we calculate the similarity of senses and lemmas to create a weighted matrix $\mathbf{W}$ through the following equation:

$$sim(s,l) = \mathrm{PMI}'(s,l) \times D(w_s, w_l) \quad (2)$$

where $D(w_s, w_l)$ is the decaying factor from [6], applied to the words corresponding to the target sense and lemma. Each row of $\mathbf{W}$ becomes a vector representation for a sense, i.e., the ***sense vectors***.

C. Context Vectors

We construct feature vectors that we call the ***context vectors***, to represent each sentence by adding lemmas of the words found in the sentence as features and assigning "1" as their values, except for the target ambiguous word for which we assign "0". This process indicates what word we will disambiguate in the sentence. We are working only with nouns and verbs. For the previous example sentences, the context vectors will be:

- Sentence 1: [deposit.v: 1, money.n: 1, bank.n: 0]
- Sentence 2: [sit.v: 1, bank.n: 0, river.n: 1]

D. Score for Sense Vectors

Let us denote a context vector as $\mathbf{c}$ and a sense vector as $\mathbf{s}$. In order to determine the best candidate for a word sense, we score each $\mathbf{s}$ corresponding to the senses of the target ambiguous word by computing the inner product as follows:

$$score(\mathbf{s}) = \sum_{i=1}^{n} c_i s_i. \quad (3)$$

The sense of the vector $\mathbf{s}$ with the highest score will be considered as the correct one for the target ambiguous word.

E. Sense Sensitive Thesaurus (SST)

The context may not provide enough information to disambiguate words properly. Therefore, we build an **SST** to add related words to the context. We take each pair of lemmas from the matrix $\mathbf{W}$, and compute the relatedness $\tau(l_a, l_b)$ as follows:

$$\tau(l_a, l_b) = \frac{\sum_{s \in \Gamma(l_b)} sim(s, l_a)}{\sum_{s \in \Gamma(l_b)} sim(s, l_b)} \quad (4)$$

where $\Gamma(l_b) = \{s_x | sim(s_x, l_b) > 0\}$, i.e., the set of senses that co-occur with l_b. Note that l_a is constrained by the information available for l_b. The minimum possible value for relatedness is 0, which indicates that no senses co-occur with both lemmas and, therefore, they have no relatedness. This measure is asymmetric, i.e., $\tau(l_a, l_b)$ is not necessarily equal to $\tau(l_b, l_a)$. In the SST, for each lemma l (referred to as ***base entry***) we have a list of other co-occurring lemmas (referred to as ***neighbors*** of l). The total of base entries for the SST is the total of columns found in $\mathbf{W}$.

F. Context Expansion

We expand $\mathbf{c}$ through the following procedure:

1) For each $c \in \mathbf{c}$, we use the SST to find the base entries for which c is a neighbor, and create a list B that contains all the base entries sorted by their relatedness score in descending order.
2) We use the top N elements from the list B to expand the original context, and assign $1/r$ as their values. We use $1/r$ instead of the score given by $\tau(l_a, l_b)$ because these values can be very small in practice, and the absolute differences between scores are not important.

As consequence, we create an ***expanded context vector*** $\mathbf{c}'$ that contains the elements $[c_1, \ldots, c_M, b_1, \ldots, b_N]$. We use $\mathbf{c}'$ in Equation 3 to calculate the score of each sense vector $\mathbf{s}$.

G. Candidate Classification

We use the candidate classification algorithm implemented in Classias[1] to learn the word sense candidate classification. This algorithm uses a maximum entropy model to learn the confidence score of a candidate word sense. We created different training instances by using the following **feature approaches**: Additive-Prefixed (**AP**), Additive-Non-Prefixed (**AN**), Product-Prefixed (**PP**), Product-Non-Prefixed (**PN**). Examples of these formats can be seen in Table I. Here, "CONTEXT=" and "SENSE=" indicate features from the context and sense vectors, respectively; "BASE=" indicates features taken from the thesaurus; "<**score**>" is the feature values in the vectors (additive) or the product of the values of the combined features (product).

The purpose of prefixes is to distinguish whether the features belong to the sense vector or the context vector, making the classifier to learn different weights for them. We always distinguish the features used for expansion (the base entries) from the others, in order to learn specific weights that indicate how much the base entries are useful to expand a context vector. The *additive* and *product* approaches allow us to generate training instances with small and big amount of features, respectively.

[1] http://www.chokkan.org/software/classias/

Table I
EXAMPLES OF FEATURES FOR TRAINING INSTANCES.

	Additive
Prefixed	CONTEXT=money.n:<**score**> SENSE=money.n:<**score**> BASE=currency.n:<**score**>
Non-Prefixed	money.n:<**score**> BASE=currency.n:<**score**>
	Product
Prefixed	CONTEXT=money.n+SENSE=money.n:<**score**> BASE=currency.n+SENSE=money.n:<**score**>
Non-Prefixed	money.n+cash.n:<**score**> BASE=currency.n+money.n:<**score**>

III. EXPERIMENTS

A. Dataset

We extracted sentences from 43 Wikipedia articles and created a corpus annotated with CDL tags. There are in total 3340 annotated sentences for which we run the evaluation using 5-fold cross-validation.

B. Parameters

1) Decay rate: We set $\alpha = \{0, 0.25, 0.5, 0.75, 1\}$ to evaluate the importance of the proximity factor when creating the weighted matrix. If $\alpha = 0$ then $D(w_s, w_l) = 1$, which means that $sim(s, l) = \mathrm{PMI}'(s, l)$ in Equation 2. The most consistent results for expanded context vectors were obtained by using $\alpha = 0.25$.

2) L2 regularization: This parameter affects the performance of our trained candidate classifier. We set different $L2$ values for each training set.

3) Total elements in sense vector: Product approaches generate $M \times N$ features for training instances used to learn the candidate classifier. Therefore, we restrict the total elements from the sense vector to a considerably lower value to make the process less computationally intensive. In preliminary experiments, we found that using the top 20 elements for the sense vector is sufficient for the method to achieve almost the same performance. We do not restrict the sense vectors in the additive approaches since the total features for training instances is $M + N$.

C. Results

In our experiments we evaluate the effectiveness of the expanded context vectors over the original context vectors. We created test sets with different feature representations and different number of expansions. Figure 2 shows the results. Note that "0" expansions is equivalent to using the original context vector. Our best results were for the Product-Prefixed (PP) approach, reaching 75.14% and 75.13% for context vectors with 50 and 100 expansions, respectively. We compare the performance of our method against two baselines: Most Frequent Sense (MFS) and Random Sense (RS). For the MFS, we use **A** to compute the frequency of each sense from the training set, and assign the sense with the highest frequency to each ambiguous word. This baseline reached 72.78% of accuracy, and the RS baseline reached 40.31%.

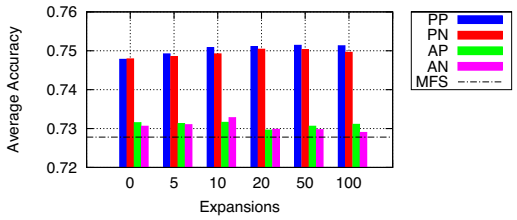

Figure 2. Performance of the different feature approaches compared to the upper and lower baselines ("MFS": Most Frequent Sense). Random Sense baseline (not shown here) reached 40.31% of accuracy

IV. CONCLUSIONS AND FUTURE WORK

We presented a context expansion method based on an SST, that can be used to improve the performance of WSD over labeled data. The results showed that in general, WSD using contexts expanded with related words achieves better performance than WSD using the original context. As a future work, the experiments can be extended to incorporate adjectives and adverbs, in order to run an evaluation on all types of content words.

REFERENCES

[1] M. Lesk, "Automatic sense disambiguation using machine readable dictionaries: how to tell a pine cone from an ice cream cone," in *Proceedings of the SIGDOC '86*. USA, 1986, pp. 24–26.

[2] G. A. Miller, "Wordnet: a lexical database for english," *Communications of the ACM*, vol. 38, pp. 39–41, 1995.

[3] T. Yokoi, H. Yasuhara, H. Uchida, M. Zhu, and K. Hasida, "CDL (Concept Description Language): A common language for semantic computing," in *Online Proc. WWW2005 Workshop on the SeC2005, Japan*, 2005.

[4] H. Uchida, M. Zhu, and T. Della Senta, *The Universal Networking Language*, 2nd ed. UNDL Foundation, 2005.

[5] D. Klein and C. D. Manning, "Accurate unlexicalized parsing," in *Proceedings of the ACL*. USA, 2003, pp. 423–430.

[6] J. Gao, M. Zhou, J.-Y. Nie, H. He, and W. Chen, "Resolving query translation ambiguity using a decaying co-occurrence model and syntactic dependence relations," in *Proceedings of the ACM SIGIR*. USA, 2002, pp. 183–190.

[7] P. D. Turney, "Mining the web for synonyms: PMI-IR versus LSA on TOEFL," in *Proceedings of the 12th EMCL*. UK, 2001, pp. 491–502.

[8] P. Pantel and D. Ravichandran, "Automatically labeling semantic classes," in *HLT/NAACL-04*, D. M. Susan Dumais and S. Roukos, Eds., vol. 4. USA, 2004, pp. 321–328.