Identification of Word Sense in Twitter Data Based on WordNet Glosses

Apichai Chan-udom[1], Chan Karman[2], and Yoshimi Suzuki[3]

[1] Graduate Faculty of Interdisciplinary Research,
University of Yamanashi, Kofu, Japan
g15dm001@yamanashi.ac.jp
[2] IIJ Innovation Institute Fujimi, Chiyoda-ku, Tokyo, Japan
chan@iij-ii.co.jp
[3] Graduate Faculty of Interdisciplinary Research,
University of Yamanashi, Kofu, Japan
ysuzuki@yamanashi.ac.jp

Abstract. With the exponential growth of information on the Internet, the short text such as search snippets and twitter data are widely available on the Internet. There are some semantic-oriented application for these data need to recognize word senses to detect which words may be similar to each other. To alleviate the sparseness of a sentence extracted from twitter data, we focus on expanding a short sentence with knowledge extracted from the auxiliary tweet corpus. To do this, we applied Word2vec to the tweet corpus and constructing related words of a target word. We identified a sense of the target word by calculating a similarity between two vectors represented, one is a vector of an each gloss(definition of each index word in dictionary) of the target word, and the other is an each gloss of its related words which are obtained by the Word2vec. The result shows that a sense of 30 target words from the technology domain tweet data attained at 83.33% accuracy.

Keywords: Sense identification, Twitter data, Word2vec, WordNet glosses

1 Introduction

Many semantic-oriented application such as Opinion Mining and Question Answering need to recognize which words may be similar to each other. The earliest known approach for word sense identification used corpus-based statistics [2, 3]. The similarity measures based on distributional hypothesis compared a pair of weighted feature vectors that characterize two words. There have been numerous methods that attempt to calculate semantic similarity [1, 5, 8]. Other approaches in sense identification utilized fine-grained and large-scale semantic knowledge like WordNet, COMLEX, EDR dictionary, and Bunrui-Goi-Hyo [6, 7]. A well-known technique in dictionary-based sense identification was the calculating semantic similarity between a context (sentence) of the target word and a gloss representing a sense of the target word in the semantic knowledge.

However, unlike textual corpora such as newspapers and scientific papers, twitter data often consists of short length of text. The following examples show two sentences extracted from BBC news and tweet data including the target word, "drone". The word "drone" has (at least) five noun senses in the WordNet3.1 including radio-controlled aircraft. We can see from the example 1 that the drone is a radio-controlled aircraft as the word occurs 'Unmanned Aerial Vehicles', and 'Remotely Piloted Aerial Systems'. However, it is difficult to identify a sense of drone in the example 2 because it occurs only a few common words.

Example 1. BBC news

> Drones: What are they and how do they work?
> (http://www.bbc.com/news/world-south-asia-10713898)
> "... To the military, they are UAVs (Unmanned Aerial Vehicles) or RPAS (Remotely Piloted Aerial Systems). However, they are more commonly known as drones. ..."

Example 2. Tweet data

> I was on podcast with a great group of tweeps talking tech acquisitions IoT drones

In this paper, we propose a method for identifying word senses in twitter data. To alleviate the sparseness of a sentence extracted from twitter data, we expand a short sentence with knowledge extracted from auxiliary tweet corpus. We applied the Word2vec to the tweet corpus and represented the target word as a vector and use its vector to find its similar words(related words), 40 words. We identified a sense of the target word by calculating a similarity between two vectors, one is a vector of an each glossary of the target word, and another one is a glossary of its related words. The vectors represented a glossary of words that we called "gloss vector". Our experiments used WordNet 3.1 sense inventory.

2 System Outline

Our system consists of five (main) processes as shown in Figure 1. The first process is Tweets Acquisition, tweets acquisition program was developed with C# language and using Tweetinvi C# library, this process collected tweets through Twitter API service. The second process is POS Tagging using the TreeTagger(Windows version) [9]. The third process is Stemming, this process replaced a word with its lemma(or base form). The fourth process determined similar words using word-vectors from Word2vec which result is a set of related words(40 words) of the target word. The last process is Sense Identification, that used our proposed method (two-stages word similarity measure), in this process to find a similar meaning between sense pairs of target words and its related words based on WordNet3.1 Glosses that was represented as a gloss vector of each sense – using Cosine similarity measurement and Jaccard coefficient and Hellinger distance [4] for comparison.

Finally, for identifying the best sense of target word, our system summarizes the similarity score of each sense of target word and decide the best sense of target word according to the domain. In our experiments, domain is technology because our tweet corpus was collected by technology keywords.

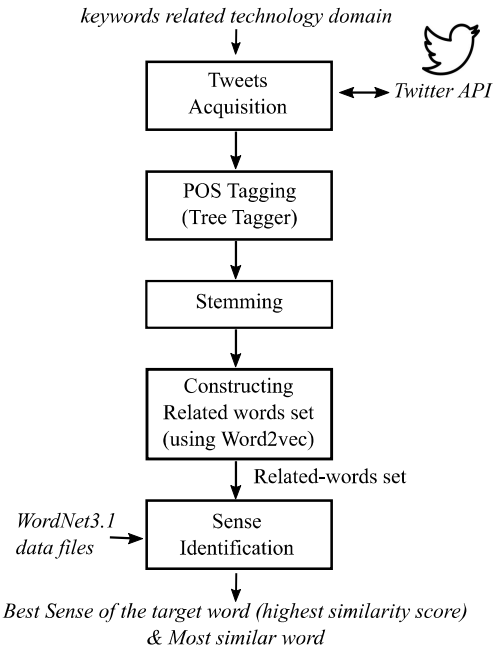

Fig. 1. System outline

3 Proposed Method

From literature review, we found concepts of word similarity measuring can be divided into two main ideas are:

1) The word similarity measure based on the dictionary or using relationship structure of words in the thesaurus. There were collected and presented by Ted Pedersen [13], and Lingling Meng [6].

The word similarity measurements based on dictionary can be measured at the level of word meaning(sense) of each word. However, the performance of the measurement depends on the quality of the relationship structure of words in the dictionary or the thesaurus only, in some case the hierarchy relatedness structure of word senses in WordNet (or dictionaries) not all perfect. Unfortunately, the context information which is important does not use for word similarity measure under this concepts.

2) Another is the measuring the word similarity using context information. Under this concept, the words occur in same contexts frequently we can point that words are similar such as “I read a book” and “I read a magazine” the word “book” and “magazine” occurred in same contexts, that means “book” and “magazine” are similar. Many researchers have published research paper about utilize context information such as D. Hindle, Tomas Mikolov et al, etc.

The important factors effect to the performance of this concept is quantity and quality of text corpus and the computer methodology that can be processed the context information properly such as machine learning, neural network, etc. Presently, the software library is used for constructing a word vector representation using context information e.g., Word2vec, Glove, and FastText.

The Word2vec creates a vector of a word consists of context features. We use the vector representation of words from Word2vec to find a set of similar words of target words, that we called the ”related words”, these are the words that frequently appear in same contexts as the target word. it may be difficult to identify which word of them is most similar to the target word.

For example, “we drink tea”, “we drink coffee”, “we drink beer”, “we drink wine”, the words “tea”, “coffee”, “beer”, and “wine” which occurred in same contexts. If a number of occurring times of each word are equally, what is the word that most similar to the word “tea”? For depth measuring on word similarity used context information only that not enough, we should consider a definition of each word also. Therefore, we have to use the word similarity measure based on the dictionary or thesaurus as the second stage repeatedly.

We proposed the new approach for the word similarity measure that provides better efficiency. It combines two key concepts for measuring the word similarity together. In addition, we apply the new word similarity approach for word sense disambiguation on a specific domain, that is the technology domain based on tweet corpus. The two-stages word similarity measure as shown in Figure 2.

In the two-stages similarity measure, to compute the similarity score between an each gloss vector of the target word and any gloss vector of its related words is defined as follows:

$$sim(tws_i, rws_j) = sim_{s2}(tws_i, rws_j) \times (1 + w2v(tw, rw)) \quad (1)$$

where $sim(tws_i, rws_j)$ is the similarity value between any gloss vector of target word (tws_i) and any gloss vector of its related word (rws_j), and $w2v(tw, rw)$ is the cosine similarity value between the target word (tw) and an each its related word (rw) utilize word vector which obtained from Word2vec, and $sim_{s2}(tws_i, rws_j)$ is the similarity value in the second-stage between a gloss vector of target word and a gloss vector of its related word.

For selecting the most similar word of the target word, our system chooses the word from the highest score of the pairs. To identifying the best sense of target word, our system summarizes the similarity score of each sense of target word and point to which sense has a maximum summation score is the best sense of target word.

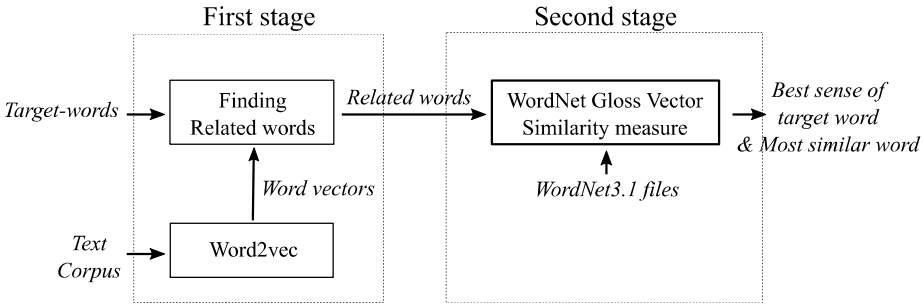

Fig. 2. The two-stages word similarity measure.

4 Similarity Measures

We used several measures to calculate similarity between words. Figure 3 illustrates similarity measures.

Notation
$\boldsymbol{x}, \boldsymbol{y}$: vector of word sense
n: vector size

Similarity Measures

1. Cosine:
   $sim_{cos}(\boldsymbol{x}, \boldsymbol{y}) = \frac{\sum_{i=1}^{n} x_i y_i}{\sqrt{\sum_i^n x_i^2)}\sqrt{(\sum_i^n y_i^2)}}$
2. Jaccard's coefficient:
   $sim_{Jaccard}(\boldsymbol{x}, \boldsymbol{y}) = \frac{\sum_{i=1}^{n} min(x_i, y_i)}{\sum_{i=1}^{n} max(x_i, y_i)}$
3. Hellinger Distance:
   $sim_{Hellinnger}(\boldsymbol{x}, \boldsymbol{y}) = \frac{\sum_{i=1}^{n} \sqrt{x_i y_i}}{(\sum_i^n x_i)(\sum_i^n y_i)}$
   Each vector must be normalized to 1 in L_1 norm:
   $\sum_{i=1}^{n} x_i = 1$ and $\sum_{i=1}^{n} y_i = 1$.

Fig. 3. Similarity measures used for the experiments

5 Experimental Setup

5.1 Twitter Data

Our system collected Tweets every day via the Twitter Search API with "Technology words" as keywords and then stored in the database. And after that, our

system eliminated duplicate tweets and deleted tweets from twitter bots and also eliminate tweets from Re-tweet users who re-tweet the same message more than 3 times (that suspected twitter bots).

The number of tweets in an each data set is 450,000 tweets on the average. The tweets were collected in one week that assigned as a one data set. In our experiments, we used 20 data sets (tweets data). The technology words are used as a keyword such as technology, engineering, vehicle, wifi, robotic, innovation, energy, mobile, program etc., that we used for collecting tweets via Twitter API.

5.2 Customization of Word2vec Parameters

The data sets(tweets data) in the experiment is not large, and the data quality is not well corpus because tweets often include some symbols such as emotion(Emoji/emoticon) symbol(e.g.,☺, ☹,♡). Moreover, an each tweet is limited 140 characters, and its writing style is the uncommonly written format. Therefore, it is necessary to customize parameters of Word2vec for training with tweets data.

For testing to customize Word2vec training parameters, we consider in cosine similarity value between the word "*wireless*" and "*bluetooth*" to compare the effect of Word2vec train parameter.

In the first round of testing, we varied a vector size, *i.e.*, the vector size is step up in 10 from 40 to 220 and compare two training algorithms, hierarchical softmax and negative sampling. As a result, we found that when the vector size is small, cosine similarity value provided the highest value. Moreover, we found that hierarchical softmax training algorithm had a better performance than negative sampling which is shown in Figure 4.

In the second round of testing, we varied the window size parameters. It is searched in steps of 1 from 4 to 10 which is illustrated in Figure 5. We can see from Figure 5. As a result, the obtained cosine similarity value has become to increase when the window size is small.

From our experiments result, we found that CBOW model has good performance than skip-gram model. And finally, we can design our training command for train Word2vec with tweets data as below:

```
./word2vec -train tweets.txt -output vectorFile -cbow 1
-window 4 -size 80 -negative 0 -hs 1 -sample 1e-5 -threads
20 -binary 1 -iter 25 -min-count 6
```

5.3 Evaluation of Similarity Measurement

Firstly, we assume our tweets corpus based on technology domain because we collected tweets via Twitter API using technology keywords. For evaluation of our proposed method, we had designed 30 words(testing words), they are ambiguous words that often appear in technology domain i.e., AI, application, cloud, cookie, desktop, device, display, drone, energy, hacker, memory, method, mouse, notebook, ontology, os, phone, platform, processor, program, protocol, router, screen,

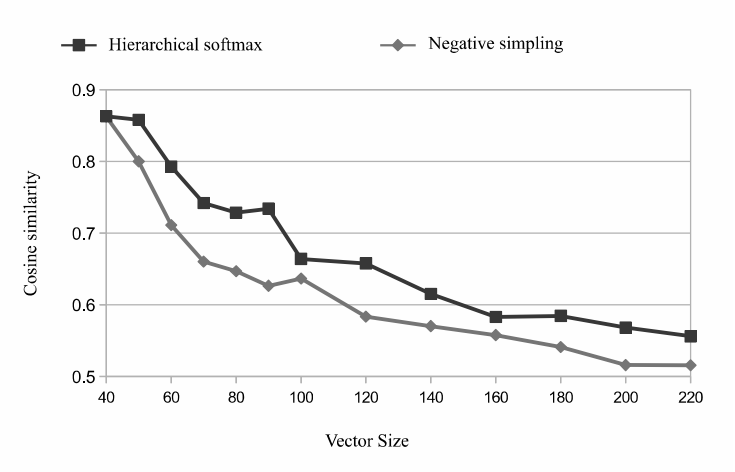

Fig. 4. Comparison between hierarchical softmax and negative sampling training algorithm in Word2vec

signal, spam, speaker, vehicle, web, window, worm, because our tweets corpus was technology context. In comparison, we compared our proposed method result with first sense heuristic method result are shown in Table 3.

6 Experimental Results

6.1 Similarity Measurement Comparison

We examined the effect of each similarity measurement utilizes the gloss vector for computing similarity score between sense pairs. The vector represented a glossary(definition) of a word, that glossary obtained from WordNet 3.1 data files. To evaluate similarity measurement using 30 testing words(mentioned above) which were assigned the best sense according to technology context. We did experiments with 20 data sets(tweets data) for every similarity measure methods and compared by accuracy percentage on average as shown in Table 1. As the result, the best performance was obtained from the Hellinger distance with TF-IDF weighting. Moreover, we found that removing stop words from a gloss vector that can improve overall performance.

For customizing a gloss vector, we examined gloss vectors consist of (1) synset-words (2) hypernym gloss, and (3) hypernym synset-words. Table 2 shows the results.

In Table 2, Gl means the method using Glosses of WordNet, RS indicates without stop words, AS refers to Adding synset-words, AHG means Adding Hypernym Glosses and AHS indicates Adding Hypernym synset-words. In each column, "$\surd$" and "-" indicate with and without each information. We obtained

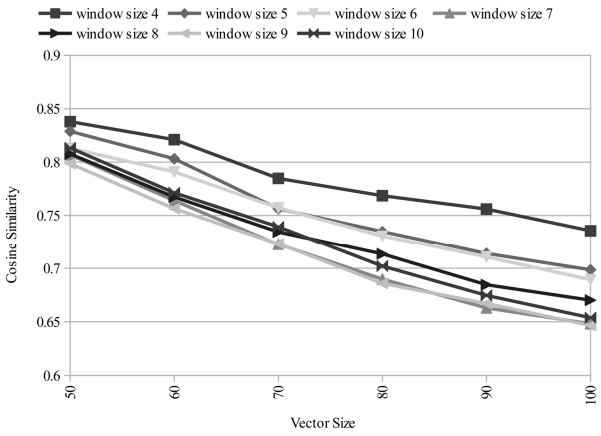

Fig. 5. Cosine similarity for varying window size and vector size in Word2vec

Table 1. Similarity improvement for six methods

Methods	without stop words	with stop words	Improvement rate
Hellinger Distance (TF-IDF)	71.04	48.02	23.02%
Cosine (Binary)	69.97	43.30	26.67%
Cosine (TF-IDF)	69.55	51.98	17.57%
Hellinger Distance (term freq.)	69.44	38.89	30.55%
Jaccard Coefficient	69.31	38.33	30.98%
Cosine(term freq.)	68.44	37.29	31.15%

the highest accuracy with the gloss vector added synset-words and hypernym synset-words.

6.2 Results of Word Sense Identification

Our system determines similarity score of all gloss-vector pairs between target words and its related words. The results were shown in Table 3. The results obtained by our proposed method and the method with the first sense heuristic.

The method attained at 83.33% accuracy in word sense identification. This shows the effectiveness of the method.

In the failure case, the word "cookie" was not identified to "a short line of text that a website puts on your computer's hard drive when you access the website", but become "any of various small flat sweet cakes", while our tweets data was collected by technology keywords. In the tweets data, we found that "cookie" often co-occurred with words about kind of "desserts" or "foods" e.g., *cooking machine*, jam, chocolate, and so on.

Table 2. Accuracy for a gloss vector customization

Glossary Customization					Accuracy
Gl	RS	AS	AHS	AHG	rate
√	√	√	√	-	72.87
√	√	√	√	√	71.30
√	√	-	√	-	70.14
√	√	√	-	√	69.58
√	√	-	√	√	68.98
√	√	√	-	-	68.52
√	√	-	-	√	68.06
√	√	-	-	-	67.55
√	-	-	-	-	40.28

Table 3. Results of word sense identification

Method	Accuracy
First sense of WordNet	23.33% (7/30)
Proposed method	83.33% (25/30)

7 Conclusion

In this paper, we proposed a method for identifying word senses in twitter data. The result showed the effectiveness of the method. For future work, we will conduct additional experiments using other new words in twitter data.

Acknowledgments

The authors would like to thank anonymous reviewers for their valuable comments. This work was supported by the Grant-in-aid for the Japan Society for the Promotion of Science (JSPS), No.26330247.

References

1. A. Mehwish and R. Muhanmmad: *Sentence based semantic similarity measure for blog-posts*. In Digital Content, Multimedai Technology and its applications(IDC), 6th, International Conference (2010)
2. D. Hindle: *Noun classification from predicate argument structures*. In Proc. of 28th Annual Meeting of the Association for Computational Linguistics, pp. 268-275 (1990)
3. D. Lin: *Automatic retrieval and clustering of similar words*. In Proc. of 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, pp. 768-773 (1998)
4. W. Dingding, S. Sahar, and L. Tao: *Update summarization using semi-supervised learning based on hellinger distance*. In CIKM'15 Proceedings of the 24th ACM International on Conference on Information and Knowledge Management, pp. 1907-1910 (2015)

5. M. J. Kusner, Y. Sun, N. I. Kolkin, and K. Q.Weinberger: *From word embeddings to document distances.* In Proc. of the 32nd International Conference on Machine Learning, pp. 957-966 (2015)
6. L. Meng, R. Huang, and J. Gu: *A review of semantic similarity measures in wordnet.* International Journal of Hybrid Information Technology, volume 6, No.1. (2013)
7. M. A. Rodriguez and M. J. Egenhofer: *Determining semantic similarity among entity classes from different ontologies.* IEEE Trans. on Knowledge and Data Engineering, volume 15, No.2. (2003)
8. G. Salton, and C. Buckley: *The smart retrieval system experiments in automatic text retrieval.* In Information processing & management, volume 24(5), pp. 513-523 (1988)
9. H. Schmid: *Probabilistic part-of-speech tagging using decision trees.* In Proceedings of International Conference on New Methods in Language Processing, Manchester (1994)
10. C. Fellbaum: *WordNet: An Electronic Lexical Database.* MIT Press (1998)
11. T. Mikolov, I. Sutskever, K. Chen, G. Corrado and J. Dean: *Distributed Representations of Words and Phrases and their Compositionality.* Advances in Neural Information Processing Systems 26th, Curran Associates, pp. 3111-3119 (2013)
12. S. Patwardhan and T. Pedersen: *Using WordNet-based Context Vectors to Estimate the Semantic Relatedness of Concepts.* In Proc. of 11th conference of the European Chapter of the Association for Computational Linguistics(EACL-2006), Trento, Italy (2006)
13. Ted Pedersen, Siddharth Patwardhan, and Jason Michelizzi: *Wordnet::Similarity-Measuring the Relatedness of Concepts.* In Daniel MarcuSusan Dumais and Salim Roukos, editors, HLT-NAACL, Association for computational Linguistics, pages 38-41, Boston, Massachusetts, USA, May 2- May 7 (2004)
14. V. Jana: TF-IDF and Cosine similarity. [Online]. (2013, Oct.) Available: https://janav.wordpress.com/2013/10/27/tf-idf-and-cosine-similarity/