Natural Language Engineering **21** (2): 251–269. © Cambridge University Press 2013
doi:10.1017/S1351324913000259

SePaS: Word sense disambiguation by sequential patterns in sentences

MASOUD NAROUEI[1], MANSOUR AHMADI[2] and ASHKAN SAMI[3]

[1] *Young Researchers and Elite Club, Zahedan Branch, Islamic Azad University, Zahedan, Iran*
e-mail: naroee@cse.shirazu.ac.ir
[2] *Young Researchers and Elite Club, Shiraz Branch, Islamic Azad University, Shiraz, Iran*
e-mail: info@mahmadi.com
[3] *Department of Computer Science, Shiraz University, Shiraz, Iran*
e-mail: asami@ieee.org

(*Received 29 December 2011; revised 5 August 2013; accepted 6 August 2013; first published online 6 September 2013*)

Abstract

An open problem in natural language processing is word sense disambiguation (WSD). A word may have several meanings, but WSD is the task of selecting the correct sense of a polysemous word based on its context. Proposed solutions are based on supervised and unsupervised learning methods. The majority of researchers in the area focused on choosing proper size of 'n' in n-gram that is used for WSD problem. In this research, the concept has been taken to a new level by using variable 'n' and variable size window. The concept is based on the iterative patterns extracted from the text. We show that this type of sequential pattern is more effective than many other solutions for WSD. Using regular data mining algorithms on the extracted features, we significantly outperformed most monolingual WSD solutions. The state-of-the-art results were obtained using external knowledge like various translations of the same sentence. Our method improved the accuracy of the multilingual system more than 4 percent, although we were using monolingual features.

1 Introduction

Natural language inherently has ambiguity. There are many words in all languages that have two or more meanings. For example, *palm* is a noun in English, but depending on its context can mean *surface of the hand* or *a type of tree*. Clarifying this ambiguity and selecting the proper meaning based on the context is called Word Sense Disambiguation (WSD). Hence, many people are interested in automating WSD in machine translation process. In addition to machine translation, fields such as information retrieval, search engines technologies, speech processing and speech to text translations may also benefit from this technology. For instance, *Hayat* in Persian means *life* or *yard* with the same pronunciation but different spellings.

Word Sense Disambiguation based on supervised learning methods produce the best results for distinguishing sense of the word in public evaluation (Palmer *et al.* 2001; Snyder and Palmer 2004; Pradhan *et al.* 2007; Zhong and Ng 2010).

https://doi.org/10.1017/S1351324913000259 Published online by Cambridge University Press

Preprocessing and creating effective and efficient feature sets increase their prediction rate and accuracy. Some of the previous works, which are discussed in the related work section, took into account the ordered sequences of words in this context. However, to the best of our knowledge, this is the first work that shows the effectiveness of iterative patterns as a means for WSD algorithms. Iterative patterns are ordered sequences of words in the context that are not necessarily consecutive. Grammatical structure of most languages is based on rules that are iterative; therefore, considering the iterative patterns to correctly identify word senses is essential.

According to Markov assumptions (Jurafsky and Martin 2008), which state that current word does not depend on the entire history of the words in the context but at most on the last few words, sequence of n words, which is a subset of sentence, is investigated to help WSD. The size of 'n' is very important in order to generate effective models. Unfortunately, no consensus exists on the value of 'n' for all WSD problems. Considering a large 'n' for n-grams means increasing the probability of getting the correct sense in WSD problems, but large n-grams may not occur in the training data. In contrast, as the size of 'n' in n-grams decreases, the total number of generated n-grams increases exponentially leading to more computational overhead and even lower results. So we tried to give a new framework to this problem by using a 'variable length size n-gram'.

In this paper, our goal is to extend the definition of n-gram to a new structure by introducing iterative patterns. It means that related words are found dynamically irrespective of where they have occurred in the sentence. We use such patterns to evaluate the effectiveness of frequent pattern-based classification in WSD. The rest of the paper is structured as follows: In Section 2 the related work(s) are described. Section 3 presents the proposed method. Experimental results are presented in Section 4, and finally conclusions and future work wraps up the paper in Section 5.

2 Related work(s)

The first work in WSD was in the late 1940s (Weaver 1949). Researcher Bar-Hillel (1960) knew that WSD was not a simple problem and has many challenges. During the 1970s the problem was attacked with artificial intelligence approaches aiming at language understanding (e.g. Wilks 1975). Since the 1990s, many methods deployed statistics and artificial intelligence for WSD. In the area of artificial intelligence, researchers often use both supervised and unsupervised learning to distinguish the sense of a word.

In WSD, predicting the sense of a word can be seen as a classification problem. Supervised learning methods build a model based on a set of instances converted to feature vectors, called train set, and then the model can predict the meaning of the other unlabeled set, called test sets. Different classifiers were applied to the WSD problem such as Decision Tree (Mooney 1996), Neural Networks (Cottrell 1989; Tsatsaronis *et al.* 2007), Memory-Based (Hoste *et al.* 2002; Decadt *et al.* 2004) and Naive Bayes (Banea and Mihalcea 2011). Other works such as Lefever, Hoste and De Cock (2011) used multiple classifiers which consider both English local context

https://doi.org/10.1017/S1351324913000259 Published online by Cambridge University Press

and a set of generated multilingual bag-of-words from the aligned translations as features and then builds five classifiers by incorporating English and five other languages. The key point in creating an effective model is taking into account influential features. Although many of the similar approaches considered n-grams or n-grams in combination with other features (Stevenson and Guo 2010), considering the sequential order of words in the whole context was not addressed well. Similar approaches used non-contiguous n-grams but they were concerned with the proper size of the window as well as size of 'n'. We believe considering the sequence in the whole context, and a variable length 'n' can be helpful for prediction.

The idea behind using unsupervised approaches in WSD is that many words closely similar to the meaning of each word exist. Word frequencies can be used to cluster, and clusters are used for labeling test data. The disadvantage of unsupervised methods is that they do not achieve accuracy levels of supervised methods. In Navigli (2009) such methods are divided into three categories: context clustering (Ji 2010), word clustering (Lin and Pantel 2002) and co-occurrence graphs (Veronis 2004; Agirre and Edmonds 2006). Graph-based WSD is usually performed on a graph made by senses (nodes) and relations (co-occurrence relations, lexico-semantic etc.) between nodes (edges). The disambiguation process is usually performed by applying a ranking algorithm over the graph and assigning a rank to each sense. These approaches usually have the cost of using additional knowledge, for instance Agirre and Soroa (2009) proposed a graph-based method that uses the knowledge in a lexical knowledge base (LKB; based on WordNet) to perform unsupervised WSD. In another approach, Navigli and Ponzetto (2012) proposed a graph-based approach that uses an immense multilingual lexical knowledge, called BabelNet (Navigli and Ponzetto 2010), and used it for disambiguating in an arbitrary language. Acquiring the sense clue of a word from each language and joining these together can lead us to the target word meaning. Although graphs have a complete structure compared with sequences, the complexity of graphs is more and their efficiency is less in WSD (Veronis 2004; Agirre and Edmonds 2006).

Word Sense Disambiguation approaches, however, can be further classified as knowledge-based (or knowledge-rich) and corpus-based (or knowledge-poor) approaches (Navigli 2009). Knowledge-based approaches depend on the use of external lexical resources, such as dictionaries, ontologies, collocations etc. whereas corpus-based approaches does not make use of any other resource for disambiguation. They usually have lower performance than their supervised alternatives. Ponzetto and Navigli (2010) showed that injecting vast amount of knowledge into WordNet enables simple knowledge-based as well as the highest performing supervised WSD systems to perform. They prove that knowledge-rich disambiguation is a competitive alternative to supervised systems even when relying on a simple algorithm.

A famous feature extraction approach in WSD is based on the Markov model. Shannon (1951) states that a language could be approximated by the nth order Markov model, where 'n' is extended to infinity. Lots of work have been done for calculating the size of 'n' in WSD problems (Brown *et al.* 1992; Iyer, Ostendorf and Meteer 1997; Chen, Beeferman and Rosenfeld 1998). Marti and Bunke (2001) concluded that using a large size for 'n' makes the model more powerful after testing

https://doi.org/10.1017/S1351324913000259 Published online by Cambridge University Press

different sizes for 'n'. Jason and Lethal (2008) investigated that using a combination of lower order n-grams instead of generating a higher order model of n-gram improves WSD for Punjabi language.

In this study we tried to present a general framework, *SePaS*, to the problem of investigating the size of 'n' in the Markov model by using a variable length window and variable sequence size. We use patterns as variable length sequences of words that are not necessarily consecutive. In other words, we believe that the size of 'n' must be decided based on iterative sequences presented in the context, and the variable length window must also be decided dynamically. That is, two words may have a very distinctive characteristic to identify the correct WSD, but may have a low frequency in sentences. In comparison, four or more words with higher frequency may do the same. Thus, the number of words to be considered in each case must be dynamic. Moreover, each sentence length may vary considerably from one to the other.

3 Proposed method

In general, supervised learning methods achieved better results compared with unsupervised approaches (Navigli 2009). The main point in supervised methods is selecting proper features. Most of the related work(s), which were described in the previous section, considered features such as single words or n-grams or combinations with n-grams. In order to build a powerful model, in addition to single words, we also considered dependencies and word order. Without considering the order, 'buildings of computer department are old' can be changed to 'department are computer of old building', which is not the same. Therefore, a greater semantics must exist in the word order. Although n-grams consider the word order, they only consider close sequential proximity. We believe that contiguous sequence of 'n' items from a given context like n-gram is not highly important. In contrast, sequences of words that are not consecutive may be more effective than considering only a close proximity or contiguity of words. To address this proximity limitation, we use frequent iterative patterns because they consider sequential order of the words in the whole context and not just in close proximity.

Previous studies (Cheng *et al.* 2007) show the effectiveness of frequent pattern-based classification, where frequent patterns are used as features for improving the classification step. Follow the following definitions:

Definition 1. ***Itemset***: *Let* $\Gamma = \{I_1, I_2, \ldots, I_m\}$ *be a set of items. A subset of* Γ *is called an itemset. An itemset that contains k items is a k-itemset. The occurrence frequency of an itemset is the number of transactions that contain the itemset. This is also simply known as the frequency of the itemset. Let D be a set of database transactions,* $D = \{T_1, T_2, \ldots, T_m\}$, *where each transaction* T_i *is a non-empty itemset such that* $T \sqsubseteq \Gamma$.

Definition 2. ***Frequent Itemset***: *An itemset* T *is frequent for a transaction dataset D if* $\frac{|D_T|}{|D|} \geq \alpha$, *where* D_T *is any dataset transaction containing* T. $\frac{|D_T|}{|D|}$ *is called the support of* T *in D, written* $s(T)$ *and* α *is the minimum support threshold* $0 \leq \alpha \leq 1$.

https://doi.org/10.1017/S1351324913000259 Published online by Cambridge University Press

A challenging step toward mining frequent itemsets from a large dataset, using a low support, is that such mining often leads to generating a large number of itemsets. This is because if an itemset is frequent, each of its subsets is also frequent. To overcome this difficulty, the concept of closed frequent itemset was introduced.

Definition 3. ***Closed Frequent Itemset***: *An itemset T is closed in a dataset D if there exists no super-itemset Γ such that Γ has the same support count as X in D. An itemset T is a closed frequent itemset in set D if T is both closed and frequent in D.*

For example if $\{A, B\}$ is an itemset that has $support = 5$ and all of its supersets have $support < 5$, then $\{A, B\}$ is a closed itemset. In contrast, if $\{A, B, C\}$ is an itemset that has $support = 5$, then $\{A, B\}$ is not a closed itemset anymore.

Although there are many algorithms (Agrawal and Srikant 1994; Han *et al.* 2004) for finding frequent itemsets, they do not consider the temporal order of elements in the sequence. Nature of sentence is sequential and considering the order helps create influential patterns. One of the most effective algorithms in the area of sequence mining is iterative patterns (Lo and Khoo 2007), which consider closed unique repetitive patterns. This algorithm has shown promising results in the fields such as software fault detection (Lo *et al.* 2009) and dynamic malware detection (Ahmadi *et al.* 2013). In this paper, application of this concept in WSD is presented. In order to understand the closed unique iterative characteristic of such patterns, consider the following definitions:

Definition 4. ***Iterative Pattern Instance***: *Given a pattern $P(e_1, e_2, \ldots, e_n)$, a consecutive series of words $SB = (sb_1, sb_2, \ldots, sb_n)$ in a sentence (word sequence) S in WSD database (WSDDB) is an instance of P iff it is of the following quantified regular expression (QRE):*

$$e_1; [-e_1, \ldots, e_n] * e_2; \ldots; [-e_1, \ldots, e_n] *; e_n$$

QRE resembles the standard regular expression with ';' as the concatenation operator, '[−]' as the exclusion operator (e.g. $[-P, S]$ means any word except P and S), and '*' as the standard Kleene star.

The minimum size of patterns is one single word and the maximum size is the whole context. Any ordered combination of the words in the context can be a pattern. For example, consider the following sentences which were extracted from 'bass.n'[corpus in a two-way ambiguities (TWA) dataset:

Sentence #1 :

Charlie no longer ***played*** *rhythm guitar but stood clutching a mike stand at the edge of the stage, howling at the kids, who pogoed like road drills beside me, Eva gasped and covered her face. Then Charlie was smearing blood over his face and wiping it over the* $<$ *head* $>$ ***bass*** $<$ */head* $>$ ***guitarist.***

Sentence #2 :

Tommy is still obviously the leader and driving force, very much in the Art Blakeymould, and once again he's selected a group of talented young ***players****. For this tour the final line-up had Dave Lewis on Saxes, Les Miller double* $<$ *head* $>$ ***bass*** $<$ */head* $>$*, and* ***guitarist*** *Chris Watson who replaced the previously announced Hammond organist.*

https://doi.org/10.1017/S1351324913000259 Published online by Cambridge University Press

After the stemming step, two words 'played' and 'players' become equal to a single word 'play'. Then we look for instances of the defined pattern. As you can see in the contexts, two instances satisfy the QRE for pattern $< play, bass, guitarist >$ which is of size 3. We highlight the instances in both contexts and show them in a set of triples. For example, in the previous samples, assuming that we do not count hypertext mark-up language (HTML) tags, the set of instances is

$$< 1, 4, 50 >, < 2, 27, 44 >$$

For each triple, the first number is the sequence ID, the second is the start index and the third is the end index.

Definition 5. ***Frequent Iterative Pattern***: *An iterative pattern P is frequent if its instances occur above a certain threshold of a minimum support in WSDDB (which stands for the corpuses of WSD).*

For very long contexts there are a large number of patterns. We only consider the closed and unique patterns.

Definition 6. ***Closed Iterative Pattern***: *A frequent iterative pattern P is closed if there exists no super sequence Q such that:*

(1) *P and Q have the same support.*
(2) *Every instance of P corresponds to a unique instance of Q, denoted as* $Inst(P) \approx Inst(Q)$.

An instance of P $(seqP; startP; endP)$ *corresponds to an instance of* $Q(seqQ; startQ; endQ)$ *iff* $seqP = seqQ$ *and* $startP \geq startQ$ *and* $endP \leq endQ$. *Where 'seq' means each record in WSDD.*

However, even by using closed patterns, we may still generate a large number of iterative patterns. Consider the following transactions:

$$S1 = \{B, C, B, B, C, B, B, C\}$$
$$S2 = \{B, B, B, C, B, B, B, C\}$$

Using minimum absolute support = 2, patterns like $\{B, C\}, \{B, B, C\}$ and $\{B, B, B, C\}$ will be extracted. These patterns have different support values and hence if we mine closed patterns, all three will be extracted. If the sentences are reasonably long and the word B appears very often, the pattern set is likely to explode. To avoid this problem, we mine a compact set of closed patterns that are composed of unique words.

Definition 7. ***Closed Unique Pattern***: *A frequent pattern P is a closed unique pattern if P contains no repeated constituent words, and there exists no super-sequence Q such that:*

(1) *P and Q have the same support.*
(2) *Every instance of P corresponds to a unique instance of Q.*
(3) *Q contains no constituent words that repeat.*

https://doi.org/10.1017/S1351324913000259 Published online by Cambridge University Press

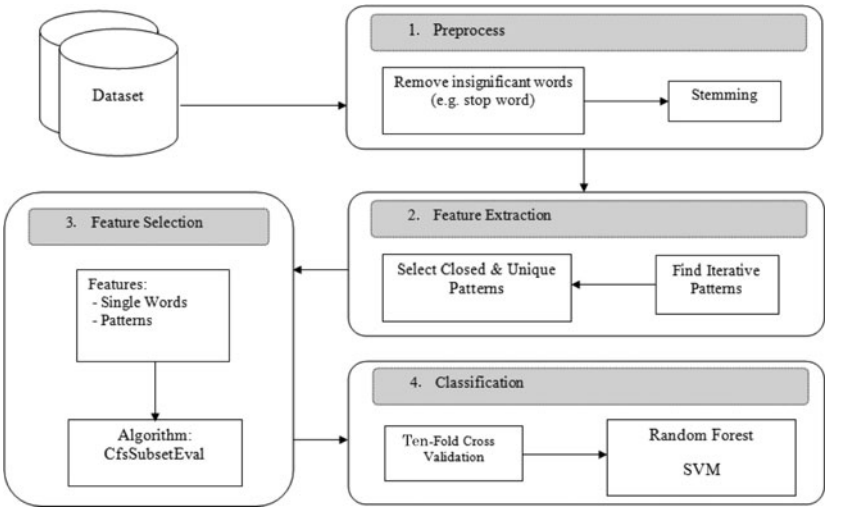


Fig. 1. System overview.

Consider a database with two sequences:

$$S1 = \{A, B, A, B, B, C, E, D, A, B, B\}$$
$$S2 = \{C, E, D, A, B, B, B, B, B\}$$

Assuming minimum support = 2, the pattern $\{A, B\}$ is a closed unique pattern. It contains unique words A and B, and there is no longer unique pattern having the same support as $\{A, B\}$. Consider another pattern $\{C, D\}$ which is unique. This pattern is not closed, as there exists a longer pattern $\{C, E, D\}$ which is also unique and the two patterns have corresponding instances. In the experiments, the set of closed unique patterns is much less than the set of closed patterns, and it is more efficient to mine closed unique patterns. So in addition to single words, we also consider closed unique patterns as features. Because grammatical structure of all languages is rule-based, such features can help to improve the classification step by considering the word's order in the whole corpus.

Proper selection of support value is a challenging problem in SePaS deployment. Very small supports produce too many features that may lead to over-fitting, which reduces the applicability of the algorithm. On the other hand, large supports may cause missing of so many discriminative features. The proper methodology to select proper support value based on each problem is not addressed in this paper. However, the best obtained results are illustrated. This type of contradiction has been addressed extensively in quantitative association rule mining (Srikant and Agrawal 1996) as the selection of proper size of discrimination for maximizing support and confident at the same time.

An overview of the method is shown in Figure 1.

The method is outlined in the following steps:

(1) Preprocessing of the corpus
(2) Looking for frequent sequential patterns in corpus
(3) Using single words and patterns as features

https://doi.org/10.1017/S1351324913000259 Published online by Cambridge University Press

(4) Feature selection and classification

3.1 Preprocessing

Preprocessing plays an important role in the quality of evaluation process. In most practical applications, it takes more than 60 percent of the process. Preprocessing is considered an important step because there are many meaningless words with little information. In this work, we first omit insignificant words, such as stop words,[1] prepositions etc., from every context. Then we use stemming, which pertains to finding the origin of the words and removing prefixes and postfixes. This means, forms like adjectives, nouns and verbs are converted to homological-like words. For instance, both *'travelling'* and *'traveled'* are converted to the same word, *'travel'* . Stemming assures that the two sequences, *'going to drive'* and *'goes to driving'* are seen as a single sequence *'go drive'*. The Porter stemmer (Porter 1980) as a popular open-source application for stemming was used.

3.2 Feature selection

Feature selection is the procedure of choosing a subset of features happening in the training set. This subset contains more discriminative features that can help improve the accuracy in classification. CfsSubsetEval algorithm (Hall 1998) was used for feature selection. CFS is a filtering algorithm that ranks feature subsets according to a correlation-based heuristic evaluation function. The evaluation bias is toward subsets that contain features that are highly correlated with the class and uncorrelated with each other. Irrelevant features should be ignored because they will have low correlation with one or more of the remaining features. The acceptance of a feature depends on the extent to which it predicts classes in areas of instance space not already predicted by other features,

$$M_S = \frac{k\overline{r_{cf}}}{\sqrt{k + k(k-1)\overline{r_{ff}}}} \tag{1}$$

where M_S is the heuristic 'merit' of a feature subset S containing k features, $\overline{r_{cf}}$ is the mean feature-class correlation ($f \in S$) and $\overline{r_{ff}}$ is the average feature–feature inter-correlation. The numerator of (1) can be thought of as providing an indication about the predictiveness of the set of features of the class; and the denominator providing an indication about redundancy in the set of features. Equation forms the core of CFS and imposes a ranking on feature subsets in the search space of all possible feature subsets. The algorithm is implemented in Waikato Environment for Knowledge Analysis (WEKA)[2] platform (Hall *et al.* 2009).

3.3 Classification

Classification is a data analysis technique that extracts models describing important data classes. Such models, called classifiers, predict categorical class labels. For

[1] http://www.ranks.nl/resources/stopwords.html
[2] http://www.cs.waikato.ac.nz/ml/weka/

https://doi.org/10.1017/S1351324913000259 Published online by Cambridge University Press

example, suppose a translator wants to find the proper meaning of the word *'palm'* in the following context extracted from *'palm.n'* corpus in a TWA dataset:

The fats found in your food Saturated Watch out! These are the greatest danger so avoid them: palm kernel oil.

Here a model or classifier must be constructed to predict class (categorical) labels, such as *'hand'* or *'tree'* . Data classification is a two-step process. In the first step, a classification algorithm builds the classifier by analyzing or 'learning from' a training set made up of database tuples and their associated class labels. In the second step, the model is used for classification. A different set, called the test set, is used to evaluate the correctness of the built model. Test data are independent of the training data, meaning that they were not used to construct the model. For the first two experiments, we used WEKA's ten-fold cross validation on the train dataset of the corpus, where the train dataset is divided into ten sub-samples with the same number of instances. Each time, nine of them are used as train data and the remainder is used for testing.

The main matter in classification to improve accuracy is removing irrelevant data and creating influential features. After stemming and finding iterative patterns, we consider both single words and closed unique patterns as features of the dataset. We use frequency for the value of each word and support for the value of the patterns and build a feature vector for each sample. To tune the support of iterative patterns, we examined accuracies with relative supports of between 0.06 to 0.01 for each category and selected the support with the highest accuracy. For the classification step, Random Forest and SVM were applied. First experiments were done using Weka's ten-fold cross validation.

4 Experimental results

This section presents various evaluations of SePaS. For these experiments, we used different benchmark datasets, namely TWA, SemEval-2007 English lexical-sample, SenseEval3 English lexical-sample and SemEval-2007 Word Sense Induction. The first two datasets contain many contexts, which are manually labeled based on the real sense of the ambiguous word. In the incoming sections, each section describes experiments performed on a specific dataset.

4.1 Two-way ambiguities dataset

First dataset, called TWA[3] dataset, has six categories. These categories are actually ambiguous words with two meanings, e.g. bass, crane, motion, palm, plant and tank. These contexts were originally extracted from British National Corpus with nearly one hundred to 200 instances for each ambiguous word. TWA on SePaS was evaluated based on accuracy and Root Mean Square Error (RMSE), which is a quadratic scoring rule that measures the average magnitude of the error.

[3] http://www.cse.unt.edu/rada/downloads.html/#twa

https://doi.org/10.1017/S1351324913000259 Published online by Cambridge University Press

Table 1. *Achieved accuracies on the TWA dataset*

Category	n*	MCS	Mono	Multi	WSC	COL	SP_RF	SP_SVM	RMSE
Bass.n	107	90.6	90.6	92.5	95.6	90.57	97.2	97.2	0.2
Crane.n	95	75.8	75.8	78.9	90.1	76.6	88.4	81.05	0.4
Motion.n	201	70.6	81.1	92.5	84.1	79.1	94.0	88.06	0.3
Palm.n	201	71.1	73.1	89.7	88.1	71.14	92.0	82.9	0.4
Plant.n	187	54.6	79.1	84.0	79.6	54.26	87.2	83.51	0.4
Tank.n	201	62.7	77.6	78.6	73.8	64.68	83.0	77.61	0.4
average		70.9	79.55	86.03	85.22	72.73	90.3	84.92	

*Number of instances

Comparison of results on TWA dataset with two previous works (Ji 2010; Banea and Mihalcea 2011) is presented in Table 1. Ji (2010) proposed an unsupervised method for WSD. They clustered contexts, instead of words, and considered more knowledge regarding relationship between words by applying Google n-gram (n = 5) corpus Version II,[4] other than only WordNet (Fellbaum 1998) relatedness measure for contexts. This method used extra knowledge that is not presented in the monolingual dataset. They used the consequent clusters. Clustering 1.2 billion 5-grams is also a time-consuming task. Banea and Mihalcea (2011) showed that creating features based on a multilingual translation can improve the performance of the WSD system by a significant margin as compared with other systems that use only monolingual features. They used all the translations of text in German, French and Spanish, in addition to English text. The permutation of these four languages was used to build fifteen new cases, four of which are monolingual, six are bilingual, four trilingual and one four-lingual. The average accuracy of the classifier on each group was considered based on binary features representing each word. The results are shown in Table 1. The Most Common Sense (MCS) column shows the accuracies achieved from the MCS method (Banea and Mihalcea 2011). MCS used WEKA's ZeroR algorithm. *Mono* is the result obtained by applying multi-nominal Naive Bayes classifier on monolingual features, where the features are individual English words, and the existence of a word in the context is shown with a binary value as the weight (Banea and Mihalcea 2011). *Multi* is the maximum of accuracies, obtained by applying multi-nominal Naive Bayes classifier on different combinations of English and three other languages (Banea and Mihalcea 2011). Ji (2010) applied unsupervised learners and web-scale phrases for clustering TWA, which are shown in WSC. Although some accuracies of this method are better than multilingual classification, overall average is lower (Table 1). *COL* shows the results obtained by applying SVM classifier on collocation features. Collocations, which are formally described in Section 4.5, are similar to iterative patterns, but they are not ordered. We used a minimum support of 0.03 for extracting collocations which showed the best performance as described in Section 4.5.

[4] http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC2006T13

https://doi.org/10.1017/S1351324913000259 Published online by Cambridge University Press

The acquired accuracies and RMSE of SePaS are shown in the last three columns by *SP_RF* using Random Forest and *SP_SVM* by applying SVM respectively. RMSE shows the error obtained from SVM. As Table 1 indicates, in addition to improving monolingual classification results with significant margin, our proposed method also improves multilingual classifications, even though we are not using the information that the multilingual has used. On average, SePaS based on Random Forest classification improved the accuracy by 4.27 percent higher than multilingual classifiers. We want to emphasize that SePaS uses only English translation of contexts. Both Banea and Mihalcea (2011) and Ji (2010) considered a large amount of data for classification. As stated, SePaS uses only train dataset, i.e. less data, and achieves better results, so the results are statistically significant.

4.2 SemEval-2007 English lexical-sample dataset

The second standard dataset that was used for evaluation is SemEval-2007 English lexical-sample dataset[5] (Pradhan *et al.* 2007), consisting of one hundred sections, including thirty-five nouns and sixty-five verbs. The samples for each word have been extracted from Brown Corpus, Penn Tree Bank and OntoNotes sense tags (Hovy *et al.* 2006). The number of contexts in each section is considerably diverse for each ambiguous word. To make comparison, only thirty-one sections used in Banea and Mihalcea (2011) were analyzed.

The analysis results are presented in Table 2. The MCS column shows the accuracies (Banea and Mihalcea 2011) achieved from the MCS method. *Mono* presents the results of multi-nominal Naive Bayes classifier on monolingual features (Banea and Mihalcea 2011). *Multi* presents the maximum of accuracies (Banea and Mihalcea 2011) based on the combinations of English and three other languages. *COL* shows the results of applying SVM classifier on collocation features with minimum support of 0.03, which provided the best result. Finally, accuracies and RMSE of SePaS are shown in the last three columns by *SP_RF* using Random Forest and *SP_SVM* by applying SVM respectively. RMSE shows the error obtained from SVM.

SePaS, a monolingual WSD, on average outperforms even multilingual WSD presented by Banea and Mihalcea (2011). It is worth to note that generating multilingual dataset is a time-consuming task and requires extra knowledge that may not be present all the time. Since SePaS uses only English translation of the corpus, the difference of 0.34 is empirically significant. Figure 2 summarizes the obtained results of previous experiments. The left-hand side in the figure shows the average accuracies on TWA dataset and the right-hand side shows the average accuracies on SemEval dataset. The columns correspond to the columns in Tables 1 and 2. Although SePaS based on SVM achieved high results, in total and compared with multilingual classifier, SePaS based on Random Forest produces the highest results and outperforms the previous works.

[5] http://nlp.cs.swarthmore.edu/semeval/tasks/task17/data.shtml

https://doi.org/10.1017/S1351324913000259 Published online by Cambridge University Press

Table 2. *Achieved accuracies on the SemEval dataset*

Category	n	s*	MCS	Mono	Multi	COL	SP_RF	SP_SVM	RMSE
approve.v	53	2	94.34	94.34	96.23	94.34	92.45	94.34	0.23
ask.v	348	6	64.94	68.39	75.00	64.94	67.24	67.24	0.33
bill.n	404	3	65.10	88.12	92.82	66.3	95.05	91.58	0.18
buy.v	164	5	78.66	78.66	77.64	78.66	88.41	80.49	0.25
capital.n	278	4	92.81	92.81	93.53	92.81	95.32	94.24	0.15
care.v	69	3	78.26	78.26	86.47	78.26	82.61	78.26	0.38
effect.n	178	3	82.02	82.02	86.33	83.15	88.20	83.15	0.33
exchange.n	363	5	71.90	73.83	85.95	73.83	84.02	80.99	0.27
explain.v	85	2	88.24	88.24	88.24	88.24	94.12	91.76	0.28
feel.v	347	3	82.13	82.13	82.04	82.13	88.76	82.13	0.34
grant.v	19	2	63.16	73.68	78.95	63.16	78.95	84.21	0.39
hold.v	129	8	34.88	45.74	43.41	34.88	37.21	37.98	0.35
hour.n	187	4	84.49	84.49	84.49	84.49	91.98	91.58	0.18
job.n	188	3	74.47	74.47	84.04	74.47	82.45	74.47	0.41
part.n	481	4	81.91	81.91	85.45	81.91	83.99	81.7	0.27
people.n	754	4	91.11	91.11	93.37	91.11	92.57	91.51	0.18
point.n	469	9	71.64	73.99	84.22	73.99	79.74	80.6	0.16
position.n	268	7	27.61	60.82	68.91	58.58	58.58	58.58	0.34
power.n	251	3	47.81	84.46	83.27	82.87	83.27	82.87	0.33
president.n	879	3	86.23	89.87	90.79	89.87	95.34	95.22	0.17
promise.v	50	2	88.00	88.00	88.00	88.00	94.00	92.00	0.28
propose.v	34	2	85.29	85.29	87.25	85.29	94.12	88.24	0.34
rate.n	1,009	2	84.64	86.92	89.30	86.92	92.27	89.49	0.32
remember.v	121	2	99.17	99.17	99.17	99.17	99.17	99.17	0.09
rush.v	28	2	92.86	92.86	92.86	92.86	89.29	92.86	0.26
say.v	2,161	5	97.78	97.78	97.78	97.78	98.06	97.78	0.09
see.v	158	6	44.94	47.47	52.53	44.94	50.63	51.27	0.4
state.n	617	3	83.14	83.95	85.74	83.14	90.92	83.95	0.32
system.n	450	5	55.56	72.44	75.78	78	74.89	78	0.29
value.n	335	3	89.25	89.25	89.85	89.85	89.85	89.85	0.26
work.v	230	7	64.78	65.65	68.99	64.78	65.65	65.65	0.31
average			75.71	80.52	83.50	78.99	83.84	82.3	

*Number of the senses

4.3 SenseEval-3 lexical-sample dataset

SenseEval-3 lexical-sample task[6] is used to compare results of SePaS and the state-of-the-art supervised method, IMS (Zhong and Ng 2010). In the preprocessing step, IMS uses WordNet to find the lemma form of each token. These use part-of-speech (POS) tags of surrounding words, surrounding words and local collocations as three different kinds of features. Then they apply classifiers such as SVM for prediction and achieve strong results.

[6] http://www.senseval.org/

https://doi.org/10.1017/S1351324913000259 Published online by Cambridge University Press

Table 3. *WSD accuracies on SenseEval3 lexical-sample task*

System	Accuracy
IMS	72.6
SePaS	67.2
MFS	55.2

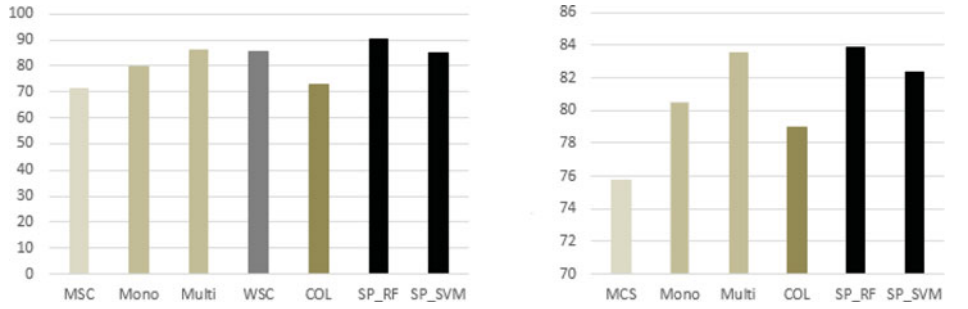


Fig. 2. (Colour online) Comparison among average accuracies of different methods and SePaS, the left on TWA dataset and the right on SemEval dataset.

As illustrated in Table 3, SePaS performs significantly better than MFS, but compared with IMS it shows lower performance. Although IMS uses a complex preprocessing and also multiple knowledge-based features, the main reason lies in feature extraction. SePaS needs a perfect relative support to perform well. Lower supports cause generating features that appear rarely, which decrease the classification accuracy due to 'overfitting' issue – features are not representatives. They also cause extracting many features, thus slowing down the model learning process, and even worse the classification accuracy deteriorates (another kind of 'overfitting' issue – features are too many). On the other hand, a high support results in extracting a subset of discriminating features. Although the problem with low supports is handled using feature selection, finding the perfect support is not addressed. The perfect support depends on the size of the corpus. Tasks such as SensEval include various categories, each of which having a different size, so applying a general support to the whole corpus yields decrease in the total performance of the system. For this comparison, the best results were obtained from relative support of 0.03 using SVM, but as mentioned, applying this support to the whole task decreases the performance.

4.4 Comparison with n-grams

n-grams[7] are a general approach for feature extraction in WSD. As mentioned, these approaches are investigated after Markov assumptions. However, they have the drawback of finding the proper size of 'n'. Iterative patterns is a similar approach

[7] Contiguous sequence of 'n' items.

https://doi.org/10.1017/S1351324913000259 Published online by Cambridge University Press

Table 4. *Comparison between n-gram features and iterative patterns*

Word	2-gram	RMSE*	3-gram	RMSE	4-gram	RMSE	IT_P†	RMSE
approve.v	94.34	0.23	94.34	0.23	94.34	0.23	94.34	0.23
bill.n	91.33	0.24	81.43	0.35	76.98	0.3	92.07	0.17
buy.v	81.7	0.24	80.48	0.25	78.65	0.26	82.31	0.24
effect.n	85.39	0.31	84.83	0.31	83.14	0.33	84.83	0.31
grant.v	78.94	0.45	73.68	0.51	73.68	0.51	78.94	0.45
hold.v	37.2	0.35	31.78	0.36	36.43	0.35	37.98	0.35
hour.n	89.83	0.22	84.49	0.27	84.49	0.27	86.63	0.25
job.n	73.93	0.41	74.46	0.41	74.46	0.41	74.46	0.41
power.n	66.13	0.47	56.57	0.53	48.6	0.58	84.46	0.32
president.n	90.67	0.24	86.23	0.3	86.23	0.3	95.1	0.81
remember.v	98.34	0.12	99.17	0.09	89.25	0.32	99.17	0.09
state.n	84.27	0.32	83.14	0.33	83.14	0.33	84.032	0.32
value.n	89.25	0.26	89.25	0.26	89.25	0.26	89.25	0.26
Average	81.73		78.08		76.47		83.22	

*Root mean squared error
†Iterative patterns

which uses variable 'n' and a variable size window, hence can be a general n-gram approach. To have a comparison, we performed additional experiments on SemEval-2007 English lexical-sample dataset. For each category, 2-grams, 3-grams, 4-grams and iterative patterns are extracted. We did not include single words in the feature vector. To evaluate the performance, we performed ten-fold cross validation on the train dataset. The results are shown in Table 4. To select the classifier, different classifiers are applied such as Naive Bayes, SVM and Random Forest. SePaS based on Random Forest performed better as indicated in the first experiment. No feature selection was applied. The results retain the same pattern for most categories as discussed in the following, so we selected a portion of the results for comparison. We consider the relative support of 0.02 for both iterative patterns and n-grams, which also improved the results of n-grams. In Table 4, columns 2 to 7 show the accuracy of n-grams and columns 8 and 9 show the results of iterative patterns.

As Table 4 shows, usually lower n-grams have a higher accuracy, but in some cases the higher n-grams have a better accuracy (e.g. 'job.n'). Because we need the highest accuracy, we must evaluate many higher n-grams and not just lower n-grams. In other words, the proper size of 'n' must be determined according to the context. The results of iterative patterns are the same or even better than the results of the best n-grams. This pattern of results was the same among most of the categories that we evaluated. It is obvious from the results that the iterative patterns can be a general n-gram approach which does not need a predefined size. There is no statistical difference between the results because as we have indicated before, iterative patterns are a general n-gram solution where the best size for 'n' in terms of performance is chosen.

https://doi.org/10.1017/S1351324913000259 Published online by Cambridge University Press

Table 5. *Comparison among SePaS and HRG-based system*

System	Accuracy
HRG	87.6
SePaS	85.9
MFS	80.9

4.5 Comparison with collocations

A collocation is an arrangement or juxtaposition of two or more words or other elements, especially those that co-occur commonly. Collocations and iterative patterns are almost similar but are different from aspects of size and order. The length of collocations must be defined whereas iterative patterns can have various lengths based on their frequency, and opposite to iterative patterns collocations are not ordered. Two sets of experiments for comparison of collocations and iterative patterns are performed.

The first experiment was performed on the framework of SemEval-2007 WSI task (SWSI) (Agirre and Soroa 2007) under the second evaluation setting, i.e. supervised evaluation. The corpus[8] consists of texts of the *Wall Street Journal* corpus and is hand-tagged with OntoNotes senses. We focused on all thirty-five nouns and partitioned them according to train and test-separated in *key files* in the corpus.

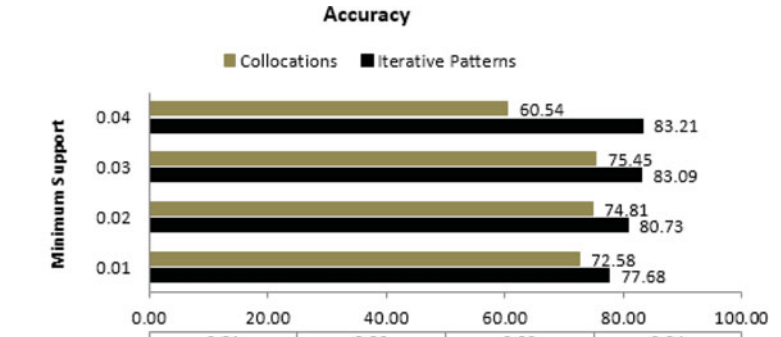
After training the model using provided senses with SVM, we tagged instances of the test set and evaluated the performance with the official scoring software of the task. We used the standard measures of recall and precision in order to produce their harmonic mean (F-score). The results are compared with the state-of-the-art WSD system that used collocation, Hierarchical Random Graphs (HRG) (Klapaftis and Manandhar 2010) . The HRG system presented a hierarchical grouping (binary tree) of the senses of ambiguous word using collocations of two words. The referred tree uses a graph in which the contexts of the polysemous word are represented as vertices and the contexts' similarities are considered as edges. The similarity is based on the collocational and bag-of-word weights. The results are shown in Table 5. As indicated in the table, the HRG system achieved better result than SePaS in the current evaluation settings.

The HRG system used an external corpus (British National Corpus) for finding the distribution of nouns in the contexts of the base corpus whereas SePaS used only frequency in the base corpus. The HRG also created graphs and extracted binary trees from them which has more computational complexity than manipulating the sequences used in our approach.

The second experiment is a comprehensive analysis on TWA dataset between iterative patterns and collocations. This comparison is performed to create more justifiable setting between iterative patterns and collocations. We considered only nouns in the context and extracted iterative patterns and collocations with different

[8] http://nlp.cs.swarthmore.edu/semeval/tasks/task02/data.shtml

https://doi.org/10.1017/S1351324913000259 Published online by Cambridge University Press



	0.01	0.02	0.03	0.04
Collocations	72.58	74.81	75.45	60.54
Iterative Patterns	77.68	80.73	83.09	83.21

Fig. 3. (Colour online) Comparison among accuracy of iterative patterns and collocations using different supports.

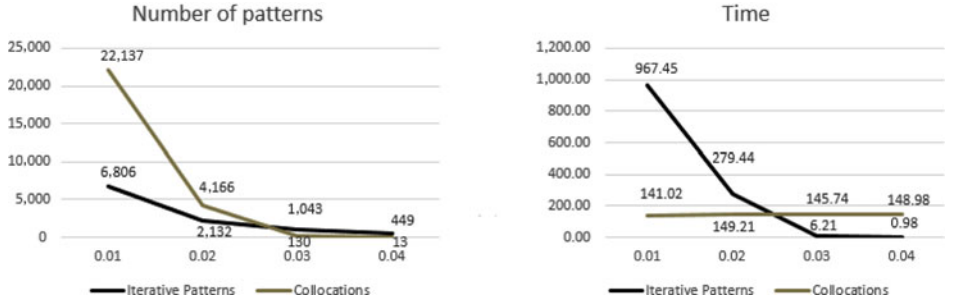


Fig. 4. (Colour online) Comparison of time(second) of pattern extraction and number of generated patterns between iterative patterns and collocations.

minimum relative supports. The evaluation results are presented from three different aspects: pattern extraction time, number of patterns and accuracy. As shown in Figure 3, by applying ten-fold cross validation using Random Forest, iterative patterns achieved higher accuracies compared with collocations. The trend between the number of generated patterns and the time of pattern extraction is shown in Figure 4. The best accuracy is obtained using a relative support of 0.04 and 0.03 for iterative patterns and collocations respectively. For these supports, Figure 4 shows that iterative patterns need less than a second for extraction, while collocations need 145.74 seconds. By using minimum supports of less than 0.02, iterative patterns take much more processing time for extraction, while collocations extraction takes almost the same amount of time. On the other hand, for lower supports, the number of extracted collocations increases substantially whereas iterative patterns do not increase very radically and stay almost the same. In conclusion, based on proper supports, by using iterative patterns more meaningful results can be obtained in lesser processing time.

5 Conclusions and future work

This paper presents SePaS, a new approach toward word sense disambiguation based on iterative patterns. These patterns are sequences of words that are not necessarily

https://doi.org/10.1017/S1351324913000259 Published online by Cambridge University Press

consecutive, but the order of words in the context is preserved. We evaluated framework SePaS with different evaluations and achieved high accuracies compared with the current state-of-the-art WSD systems. Our method is also more efficient than many of the previous methods by considering less information to correctly classify the proper WSD. We compared iterative patterns with n-gram features and showed that iterative patterns can be a general solution to the problem of identifying the proper size of 'n' in n-gram approaches. We also made a comprehensive analysis between iterative patterns and collocations and showed that using the proper minimum supports, iterative patterns perform much better in terms of accuracy and time of pattern extraction. In conclusion, iterative patterns show promising results in various tasks of WSD and require more attention. In the future, we are interested in proposing solutions for finding the best relative support. We will also attempt to replace synonyms in the corpus that may lead to better patterns. For a better comparison with the HRG system, we are planning to propose an unsupervised system based on iterative patterns, so that the setting could become more justifiable. Another future work may consider multilingual iterative patterns for building a model. Other values except frequency or presence in the dataset may also improve the results.

Acknowledgments

We would like to appreciate Rada Mihalcea and Carmen Banea for their valuable comments. We would also like to thank anonymous reviewers for their useful comments.

References

Agirre, E., and Edmonds, P. 2006. *Word Sense Disambiguation: Algorithms and Applications*. New York, NY: Springer.

Agirre, E., and Soroa, A. 2007. Semeval-2007 task 02: evaluating word sense induction and discrimination systems. In *Proceedings of SemEval-2007*, Prague, Czech Republic, pp. 7–12.

Agirre, E., and Soroa, A. 2009. Personalizing PageRank for word sense disambiguation. In *Proceedings of the 12th Conference of the European Chapter of the ACL*, Athens, Greece, pp. 33–41.

Agrawal, R., and Srikant, R. 1994. Fast algorithms for mining association rules in large databases. In *Proceedings of the 20th International Conference on Very Large Databases (VLDB)*, Santiago de Chile, Chile, pp. 487–99.

Ahmadi, M., Sami, A., Rahimi, H., and Yadegari, B. 2013. Malware detection by behavioral sequential patterns. *Computer Fraud and Security* **2013**(8): 11–9 (Elsevier).

Banea, C., and Mihalcea, R. 2011. Word sense disambiguation with multilingual features. In *Proceedings of the 9th International Conference on Computational Semantics*, Oxford, UK, pp. 25–34.

Bar-Hillel, Y. 1960. The present status of automatic translation of languages. *Advances in Computers* **1**: 91–163 (Academic Press, New York).

Brown, P. F., Della Pietra, V. J., Stephan. A. Della Pietra., Mercer, R.L., and Lai, J. C. 1992. An estimate of an upper bound for the entropy of English. *Computational Linguistics* **18**(1): 31–40 (MIT Press).

https://doi.org/10.1017/S1351324913000259 Published online by Cambridge University Press

Chen, S. F., Beeferman. D., and Rosenfeld, R. 1998. Evaluation metrics for language models. In *Proceedings of the DARPA Broadcast News Transcription and Understanding Workshop*, Virginia, USA, pp. 275–80.

Cheng, H., Yan, X., Han, J., and Hsu, C. 2007. Discriminative frequent pattern analysis for effective classification. In *Proceedings of the IEEE 23rd International Conference on Data Engineering (ICDE 07)*, Istanbul, Turkey, pp. 716–25.

Cottrell, G. W. 1989. *A Connectionist Approach to Word Sense Disambiguation*. London: Pitman.

Decadt, B., Hoste, V., Daelemans, W., and Van Den Bosch, A. 2004. GAMBL, genetic algorithm optimization of memory-based WSD. In *Proceedings of Senseval-3: 3rd International Workshop on the Evaluation of Systems for the Semantic Analysis of Text*, Barcelona, Spain, pp. 108–12. Stroudsburg, PA: Association for Computational Linguistics.

Fellbaum. C. 1998. *WordNet: An Electronic Lexical Database*. Cambridge, MA: MIT Press (a Bradford book).

Hall, M. A. 1998. *Correlation-Based Feature Subset Selection for Machine Learning*. Hamilton, New Zealand: University of Waikato.

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., and Witten, I. H. 2009. The WEKA data mining software: an update. *ACM SIGKDD Explorations Newsletter* **11**(1): 10–8.

Han, J., Pei, J., Yin, Y., and Mao, R. 2004. Mining frequent patterns without candidate generation. *Data Mining and Knowledge Discovery* **8**(1): 53–87 (Kluwer Academic, Netherlands.

Hoste, V., Hendrickx, I., Daelemans, W., and Van Den Bosch, A. 2002. Parameter optimization for machine-learning of word sense disambiguation. *Natural Language Engineering* **8**(4): 311–25 (Cambridge University Press, Cambridge, UK).

Hovy, E., Marcus, M., Palmer, M., Ramshaw, L., and Weischedel, R. 2006. OntoNotes: the 90% solution. In *Proceedings of HLTNAACL*, New York, USA companion volume: short papers, pp. 57–60.

Iyer, R., Ostendorf, M., and Meteer, M. 1997. Analyzing and predicting language model improvements. In *Proceedings of the IEEE Workshop on Automatic Speech Recognition and Understanding*, Santa Barbara, CA, pp. 254–61.

Jason, G. S., and Lethal, G. S. 2008. Size of N for word sense disambiguation using N-gram model for Punjabi language. *International Journal of Translation* **20**(1–2): 47–56.

Ji, H. 2010. One sense per context: improving word sense disambiguation using web-scale phrase clustering. In *4th International Universal Communication Symposium (IUCS)*, Beijing, China, pp. 181–4.

Jurafsky, D., and martin, J. 2008. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*, 2nd ed. New Jersey: Prentice-Hall.

Klapaftis, I. P., and Manandhar, S. 2010. Word sense induction & disambiguation using hierarchical random graphs. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pp. 745–55. Cambridge, MA: Association for Computational Linguistics.

Lefever, E., Hoste, V., and De Cock, M. 2011. ParaSense or how to use parallel corpora for word sense disambiguation. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics (ACL-11)*, Oregon, USA, pp. 317–22.

Lin, D., and Pantel, P. 2002. Discovering word senses from text. In *8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Canada, pp. 613–9.

Lo, D., Cheng, H., Han, J., Khoo, S.-C., and Sun, C. 2009. Classification of software behaviors for failure detection: a discriminative pattern mining approach. In *Proceedings of the 15th ACM SIGDD International Conference on Knowledge Discovery and Data Mining*, New York, pp. 557–66.

https://doi.org/10.1017/S1351324913000259 Published online by Cambridge University Press

Lo, D., and Khoo, S.-C. 2007. Efficient mining of iterative patterns for software specification discovery. In *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, California, pp. 460–9.

Marti, U. V., and Bunke, H. 2001. On the influence of vocabulary size and language models in unconstrained handwritten text recognition. In *Proceedings of 6th International Conference on Document Analysis and Recognition*, Seattle, WA, pp. 260–5.

Mooney, R. J. 1996. Comparative experiments on disambiguating word senses: an illustration of the role of bias in machine learning. In *Proceedings of the 1996 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Philadelphia, PA, pp. 82–91.

Navigli, R. 2009. Word sense disambiguation: a survey. *ACM Computing Surveys (CSUR)* **41**(2): 1–69.

Navigli, R., and Ponzetto, S. P. 2010. BabelNet: building a very large multilingual semantic network. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, Uppsala, Sweden, pp. 216–25.

Navigli, R., and Ponzetto, S. P. 2012. Joining forces pays off: multilingual joint word sense disambiguation. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, Jeju Island, South Korea, pp. 1399–410. Stroudsburg, PA: ACL.

Palmer, M., Fellbaum, C., Cotton, S. Delfs, L., and Dang, H. T. 2001. English tasks: all-words and verb lexical sample. In *Proceedings of SENSEVAL-2: 2nd International Workshop on Evaluating Word Sense Disambiguation Systems*, Tolouse, France, pp. 21–4.

Ponzetto, S. P. and Navigli, R. 2010. Knowledge-rich word sense disambiguation rivaling supervised systems. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, Uppsala, Sweden, pp. 1522–32.

Porter, M. F. 1980. An algorithm for suffix stripping. *Program: Electronic Library and Information Systems* **14**(3): 130–7 (MCB UP, West Yorkshire, UK).

Pradhan, S., Loper, E., Dligach, D., and Palmer, M. 2007. Semeval-2007 task-17: English lexical sample, SRL and all words. In *Proceedings of the 4th International Workshop on Semantic Evaluations*, Prague, Czech Republic, pp. 87–92.

Shannon, C. E. 1951. Prediction and entropy of printed English. *Bell System Technical Journal* **30**: 50–64.

Snyder, B., and Palmer, M. 2004. The English all-words task. In *ACL 2004 Senseval-3 Workshop*, Barcelona, Spain, pp. 41–3.

Srikant, R., and Agrawal, R. 1996. Mining quantitative association rules in large relational tables. In *Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data*, Montreal, Canada, pp. 1–12.

Stevenson, M., and Guo, Y. 2010. Disambiguation in the biomedical domain: the role of ambiguity type. *Journal of Biomedical Informatics* **43**(6): 972–81 (Elsevier).

Tsatsaronis, G., Vazirgiannis, M., and Androutsopoulos, I. 2007. Word sense disambiguation with spreading activation networks generated from thesauri. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI)*, Hyderabad, India, pp. 1725–30.

Veronis, J. 2004. Hyperlex: lexical cartography for information retrieval. *Computer Speech and Language* **18**(3): 223–52 (Elsevier).

Weaver, W. 1949. Translation. In William N. Locke and A. Donald Booth (eds.), *Machine Translation of Languages: Fourteen Essays (written in 1949, published in 1955)*, pp. 15–23. New York, NY: John Wiley.

Wilks, Y. 1975. Preference semantics. In E. L. Keenan (ed.), *Formal Semantics of Natural Language*, pp. 329–48. Cambridge, UK: Cambridge University Press.

Zhong, Z., and Ng, H. T. 2010. It makes sense: a wide-coverage word sense disambiguation system for free text. In *Proceedings of the ACL 2010 System Demonstrations*, Uppsala, Sweden, pp. 78–83.

https://doi.org/10.1017/S1351324913000259 Published online by Cambridge University Press