Word Relatives in Context for Word Sense Disambiguation

David Martinez
Computer Science and
Sofware Engineering
University of Melbourne
Victoria 3010 Australia
davidm@csse.unimelb.edu.au

Eneko Agirre
IXA NLP Group
Univ. of the Basque Country
Donostia, Basque Country
e.agirre@ehu.es

Xinglong Wang
School of Informatics
University of Edinburgh
EH8 9LW, Edinburgh, UK
xwang@inf.ed.ac.uk

Abstract

The current situation for Word Sense Disambiguation (WSD) is somewhat stuck due to lack of training data. We present in this paper a novel disambiguation algorithm that improves previous systems based on acquisition of examples by incorporating local context information. With a basic configuration, our method is able to obtain state-of-the-art performance. We complemented this work by evaluating other well-known methods in the same dataset, and analysing the comparative results per word. We observed that each algorithm performed better for different types of words, and each of them failed for some particular words. We proposed then a simple unsupervised voting scheme that improved significantly over single systems, achieving the best unsupervised performance on both the Senseval 2 and Senseval 3 lexical sample datasets.

1 Introduction

Word Sense Disambiguation (WSD) is an intermediate task that potentially can benefit many other NLP systems, from machine translation to indexing of biomedical texts. The goal of WSD is to ground the meaning of words in certain contexts into concepts as defined in some dictionary or lexical repository.

Since 1998, the Senseval challenges have been serving as showcases for the state-of-the-art WSD systems. In each competition, Senseval has been growing in participants, labelling tasks, and target languages. The most recent Senseval workshop (Mihalcea et al., 2004) has again shown clear superiority in performance of supervised systems, which rely on hand-tagged data, over other kinds of techniques (knowledge-based and unsupervised). However, supervised systems use large amounts of accurately sense-annotated data to yield good results, and such resources are very costly to produce and adapt for specific domains. This is the so-called knowledge acquisition bottleneck, and it has to be tackled in order to produce technology that can be integrated in real applications. The challenge is to make systems that disambiguate all the words in the context, as opposed to techniques that work for a handful of words.

As shown in the all-words tasks in Senseval-3 (B. Snyder and M. Palmer, 2004), the current WSD techniques are only able to exceed the most frequent sense baseline by a small margin. We believe the main reason for that is the lack of large amounts of training material for English words (not to mention words in other languages). Unfortunately developing such resources is difficult and sometimes not feasible, which has been motivating us to explore unsupervised techniques to open up the knowledge acquisition bottleneck in WSD.

The unsupervised systems that we will apply on this paper require raw corpora and a thesaurus with relations between word senses and words. Although these resources are not available for all languages, there is a growing number of WordNets in different languages that can be used[1]. Other approach would be to apply methods based on distributional similarity to build a thesaurus automatically from raw corpora (Lin, 1998). The relations can then be applied in our algorithm. In this paper we have focused on the results we can obtain for

[1] http://www.globalwordnet.org/gwa/wordnet_table.htm

English, relying on WordNet as thesaurus (Fellbaum, 1998).

A well known approach for unsupervised WSD consists of the automatic acquisition of training data by means of monosemous relatives (Leacock et al., 1998). This technique roughly follows these steps: (i) select a set of monosemous words that are related to the different senses of the target word, (ii) query the Internet to obtain examples for each relative, (iii) create a collection of training examples for each sense, and (iv) use an ML algorithm trained on the acquired collections to tag the test instances. This method has been used to bootstrap large sense-tagged corpora (Mihalcea, 2002; Agirre and Martinez, 2004).

Two important shortcomings of this method are the lack of monosemous relatives for some senses of the target words, and the noise introduced by some distant relatives. In this paper we directly address those problems by developing a new method that makes use of polysemous relatives and relies on the context of the target word to reduce the presence of noisy examples.

The remaining of the paper is organised as follows. In Section 2 we describe related work in this area. Section 3 briefly introduces the monosemous relatives algorithm, and our novel method is explained in Section 4. Section 5 presents our experimental setting, and in Section 6 we report the performance of our technique and the improvement over the monosemous relatives method. Section 7 is devoted to compare our system to other unsupervised techniques and analyse the prospects for system combination. Finally, we conclude and discuss future work in Section 8.

2 Related Work

The construction of unsupervised WSD systems applicable to all words in context has been the goal of many research initiatives, as can be seen in special journals and devoted books - see for instance (Agirre and Edmonds, 2006) for a recent book. We will now describe different trends that are being explored.

Some recent techniques seek to alleviate the knowledge acquisition bottleneck by combining training data from different words. Kohomban and Lee (2005) build semantic classifiers by merging data from words in the same semantic class. Once the class is selected, simple heuristics are applied to obtain the fine-grained sense. The classifier follows memory-based learning, and the examples are weighted according to their semantic similarity to the target word. Niu et al. (2005) use all-words training data to build a word-independent model to compute the similarity between two contexts. A maximum entropy algorithm is trained with the all-words corpus, and the model is used for clustering the instances of a given target word. One of the problems of clustering algorithms for WSD is evaluation, and in this case they map the clusters to Senseval-3 lexical-sample data by looking at 10% of the examples in training data. One of the drawbacks of these systems is that they still require hand-tagged data.

Parallel corpora have also been widely used to avoid the need of hand-tagged data. Recently Chan and Ng (2005) built a classifier from English-Chinese parallel corpora. They grouped senses that share the same Chinese translation, and then the occurrences of the word on the English side of the parallel corpora were considered to have been disambiguated and “sense tagged” by the appropriate Chinese translations. The system was successfully evaluated in the all-words task of Senseval-2. However, parallel corpora is an expensive resource to obtain for all target words. A related approach is to use monolingual corpora in a second language and use bilingual dictionaries to translate the training data (Wang and Carroll, 2005). Instead of using bilingual dictionaries, Wang and Martinez (2006) tried to apply machine translation on translating text snippets in foreign languages back into English and achieved good results on English WSD.

Regarding portability, methods to automatically rank the senses of a word given a raw corpus, such as (McCarthy et al., 2004), have shown good flexibility to adapt to different domains, which is a desirable feature of all-words systems. We will compare the performance of the latter two systems and our approach in Section 7.

3 Monosemous Relatives method

The “monosemous relatives” approach is a technique to acquire training examples automatically and then feed them to a Machine Learning (ML) method. This algorithm is based on (Leacock et al., 1998), and follows these steps: (i) select a set of monosemous words that are related to the different senses of the target word, (ii) query the Internet to obtain examples for each relative, (iii) create a collection of training examples for each sense, and (iv) use an ML algorithm trained on the acquired collections to tag the test instances. This method has been applied in different works (Mihalcea, 2002; Agirre and Martinez, 2004). We describe here the approach by Agirre and Martinez (2004), which we will apply to the same datasets as the novel method described in Section 4.

In this implementation, the monosemous relatives are obtained using WordNet, and different relevance weights are assigned to these words depending on the distance to the target word (synonyms are the closest, followed by immediate hypernyms and hyponyms). These weights are used to determine an order of preference to construct the training corpus from the queries, and 1,000 examples are then retrieved for each query. As explained in (Agirre and Martinez, 2004), the number of examples taken for each sense has a big impact in the performance, and information on the expected distribution of senses influences the results. They obtain this information using different means, such as hand-tagged data distribution (from Semcor), or a prior algorithm like (McCarthy et al., 2004). In this paper we present the results of the basic approach that uses all the retrieved examples per sense, which is the best standalone unsupervised alternative.

The ML technique Agirre and Martinez (2004) applied is Decision Lists (Yarowsky, 1994). In this method, the sense s_k with the highest weighted feature f_i is selected, according to its log-likelihood (see Formula 1). For this implementation, they used a simple smoothing method: the cases where the denominator is zero are smoothed by the constant 0.1.

$$weight(s_k, f_i) = \log(\frac{Pr(s_k|f_i)}{\sum_{j \neq k} Pr(s_j|f_i)}) \qquad (1)$$

The feature set consisted of local collocations (bigrams and trigrams), bag-of-words features (unigrams and salient bigrams), and domain features from the WordNet Domains resource (Magnini and Cavagliá, 2000). The Decision List algorithm showed good comparative performance with the monosemous relatives method, and it had the advantage of allowing hands-on analysis of the different features.

4 Relatives in Context

The goal of this new approach is to use the WordNet relatives and the contexts of the target words to overcome some of the limitations found in the "monosemous relatives" technique. One of the main problems is the lack of close monosemous relatives for some senses of the target word. This forces the system to rely on distant relatives whose meaning is far from the intended one. Another problem is that by querying only with the relative word we do not put any restrictions on the sentences we retrieve. Even if we are using words that are listed as monosemous in WordNet, we can find different usages of them in a big corpus such as Internet (e.g. Named Entities, see example below). Including real contexts of the target word in the queries could alleviate the problem.

For instance, let us assume that we want to classify *church* with one of the 3 senses it has in Senseval-2: (1) Group of Christians, (2) Church building, or (3) Church service. When querying the Internet directly with monosemous relatives of these senses, we find the following problems:

- Metaphors: the relative *cathedral* (2nd sense) appears in very different collocations that are not related to any sense of *church*, e.g. *the cathedral of football*.
- Named entities: the relative *kirk* (2nd sense), which is a name for a Scottish church, will retrieve sentences that use Kirk as a proper noun.
- Frequent words as relatives: relatives like *hebraism* (1st sense) could provide useful examples, but if the query is not restricted can also be the source of many noisy examples.

The idea behind the "relatives in context" method is to combine local contexts of the target word with the pool of relatives in order to obtain a better set of examples per sense. Using this approach, we only gather those examples that have a close similarity with the target contexts, defined by a set of pre-defined features. We will illustrate this with the following example from the Senseval-2 dataset, where the goal is to disambiguate the word *church*:

> *The* ***church*** *was rebuilt in the 13th century and further modifications and restoration were carried out in the 15th century.*

We can extract different features from this context, for instance using a dependency parser. We can obtain that there is a object-verb relation between *church* and *rebuild*. Then we can incorporate this knowledge to the relative-based query and obtain training examples that are closer to our target sentence. In order to implement this approach with rich features we require tools that allow for linguistic queries, such as the linguist's engine (Resnik and Elkiss, 2005), but other approach would be to use simple features, such as strings of words, in order to benefit directly from the examples coming from search engines in the Internet. In this paper we decided to explore the latter technique to observe the performance we can achieve with simple features. Thus, in the example above, we query the Internet with snippets such as "The *cathedral* was rebuilt" to retrieve training examples. We will go back to the example at the end of this section.

With this method we can obtain a separate training set starting from each test instance and the pool of relatives for each sense. Then, a ML algorithm can be trained with the acquired examples. Alternatively, we can just rank the different queries according to the following factors:

- Length of the query: the longer the match, the more similar the new sentence will be to the target.
- Distance of the relative to the target word: examples that are obtained with synonyms will normally be closer to the original meaning.
- Number of hits: the more common the snippet we query, the more reliable.

We observed a similar performance in preliminary experiments when using a ML method or applying an heuristic on the above factors. For this paper we devised a simple algorithm to rank queries according to the three factors, but we plan to apply other techniques in the acquired training data in the future.

Thus, we build a disambiguation algorithm that can be explained in the following four steps:

1. Obtain pool of relatives: for each sense of the target word we gather its synonyms, hyponyms, and hypernyms. We also take polysemous nouns, as we expect that in similar local contexts the relative will keep its related meaning.

2. Construct queries: first we tokenise each target sentence, then we apply sliding windows of different sizes (up to 6 tokens) that include the target word. For each window and each relative in the pool, we substitute the target word for the relative and query the Internet. Then we store the number of hits for each query. The algorithm stops augmenting the window for the relative when one of its substrings returns zero hits.

3. Ranking of queries: we devised a simple heuristic to rank the queries according to our intuition on the relevant parameters. We chose these three factors (in decreasing order of relevance):

- Number of tokens of the query.
- Type of relative: preference order: (1) synonyms, (2) immediate hyponyms, (3) immediate hypernyms, and (4) distant relatives.
- Number of hits: we choose the query with most hits. For normalisation we divide by the number of hits of the relative alone, which penalises frequent and polysemous relatives.

We plan to improve this ranking approach in the future, by learning the best parameter set on a development corpus. We also would like to gather a training corpus from the returned documents and apply a ML classifier.

4. Assign the sense of the highest ranked query: another alternative that we will explore in the future is to vote among the k highest ranked queries.

We will show how the algorithm works with the example for the target word *church* presented above. Using the relatives (synonyms, hypernyms, and hyponyms) of each sense and the local context we query the Internet. The list of the longest matches that have at least 2 hits is given in Table 1. In this case the second sense would be chosen because the words *nave*, *abbey*, and *cathedral* indicate this sense. In cases where the longest match corresponds to more than one sense the closest relative is chosen; if there is still a tie the number of hits (divided by the number of hits of the relative for normalisation) is used.

5 Experimental setting

For our experiments we relied on the lexical-sample datasets of both Senseval-2 (Kilgarriff, 2001) and Senseval-3 (Mihalcea et al., 2004). We

Query	Sense
The *nave* was rebuilt in the 13th century	2
The *abbey* was rebuilt in the 13th century	2
The *cathedral* was rebuilt in the 13th century	2
The *Catholic Church* was rebuilt in	1
The *Christian church* was rebuilt	1
The *church service* was	3
The *religious service* was	3

Table 1: Longest matches for relative words of *church* in the Senseval-2 example "*The **church** was rebuilt in the 13th century and further modifications and restoration were carried out in the 15th century.*".

will refer to these sets as S2LS and S3LS respectively. This approach will give us the chance to measure the performance on different sets of words, and compare our results to the state of the art. We will focus on nouns in this work, in order to better study the specific problems to be analysed in the error analysis. The test sets consist on 29 nouns in S2LS, and 20 nouns in S3LS. The sense inventory in S2LS corresponds to WordNet 1.7 (pre-release), while for S3LS the senses belong to WordNet 1.7.1.

Our main goal is to build all-words WSD systems, and this preliminary test on lexical-sample datasets will give us a better idea of the performance we can expect. The same algorithms can be used for extending the evaluation to all the words in context by considering each target word separately. We plan to carry out this evaluation in the near future.

Regarding evaluation, we used the scoring software provided by the Senseval organisation to measure the precision and recall of the systems. Precision refers to the ratio of correct answers to the total number of answers given by the system, and recall indicates the ratio of correct answers to the total number of instances. All our algorithms have full coverage (that is, they always provide an answer), and therefore precision equals recall. In some cases we may present the results per sense, and then the precision will refer to the ratio of correct answers to the number of answers given to the sense; recall will be the ratio of correct answers to the number of test instances linked to the sense.

6 Results

The results of applying the "monosemous relatives" (MR) and the "relatives in context" (RC) algorithm are shown in Table 2. The micro-averaged results show that the new method clearly outperforms the monosemous relatives in this dataset. However, we can also notice that this improvement does not happen for all the words in the set. One of the problems of unsupervised systems is that they are not able to perform robustly for all words, as supervised can do because of the valuable information contained in the hand-tagged corpora. Thus, we normally see different performances depending on the type of words in the target set, which suggest that the best way to raise unsupervised performance is the combination of algorithms, as we will see in Section 7.

S2LS			S3LS		
Word	MR	RC	Word	MR	RC
art	**61.1**	40.3	argument	24.7	**38.7**
authority	22.0	**45.1**	arm	10.2	**27.1**
bar	**52.1**	16.9	atmosphere	**31.3**	24.7
bum	18.8	**72.5**	audience	**51.8**	34.0
chair	**62.9**	54.8	bank	32.3	**60.6**
channel	**28.7**	27.9	degree	39.3	**43.8**
child	1.6	**46.8**	difference	**26.4**	23.7
church	**62.1**	58.1	difficulty	13.0	**43.5**
circuit	**52.8**	47.2	disc	**52.2**	45.0
day	2.2	**36.7**	image	4.1	**23.0**
detention	16.7	**62.5**	interest	**26.8**	23.1
dyke	**89.3**	85.7	judgment	20.6	**25.0**
facility	26.8	**50.0**	organization	**71.4**	69.6
fatigue	**73.8**	67.5	paper	25.6	**42.7**
feeling	**51.0**	49.0	party	**67.5**	67.2
grip	8.0	**26.0**	performance	20.5	**33.3**
hearth	37.5	**40.6**	plan	**78.0**	76.2
holiday	7.4	**74.1**	shelter	36.2	**44.9**
lady	**79.3**	8.7	sort	13.5	**65.6**
material	50.8	50.8	source	22.4	**53.1**
mouth	41.2	**43.9**			
nation	**80.6**	36.1			
nature	**44.4**	26.7			
post	**47.4**	36.2			
restraint	9.1	**22.7**			
sense	18.6	**48.8**			
spade	**66.1**	32.3			
stress	**52.6**	21.1			
yew	**85.2**	55.6			
Avg S2	39.9	**41.5**	Avg S3	34.2	**43.2**
Avg S2-S3	36.8	**42.4**			

Table 2: Recall of the "Monosemous Relatives" method (MR) and the "Relatives in Context" (RC) technique in the two Senseval datasets. Best results per word in bold.

Even if an all-words approach gives a better idea of the performance of different techniques, the Senseval lexical-sample dataset tries to include words with different degrees of polysemy and frequency in order to provide a balanced evaluation. We also show in Section 7 the performance of other techniques previously described in Sec-

S.	Definition
1	beginning, origin, root, rootage - the place where something begins
2	informant - a person who supplies information
3	reference - a publication (or a passage from a publication) that is referred to
4	document (or organization) from which information is obtained
5	facility where something is available
6	seed, germ - anything that provides inspiration for later work
7	generator, author - someone who originates or causes or initiates something

Table 3: Sense inventory for *source* in WordNet 1.7.1.

tion 2.

Sometimes it is worth to "eyeball" the real examples in order to get insight on the algorithms. For that, we chose the word *source* in the S3LS dataset, which clearly improves its performance with the new method. This word has 7 senses in WordNet 1.7.1, shown in Table 3. The Senseval grouping provided by the organisation joins senses 3 and 4, leaving each of the others as separate groups. The coarse inventory of senses has been seen as an alternative to fine-grained WSD (Ciaramita and Altun, 2006).

For this word, we see that the "monosemous relatives" approach achieves a low recall of 22.4%. Analysing the results per sense, we observed that the precision is good for sense 1 (90%), but the recall is as low as 4.7%, which indicates that the algorithm misses many of the instances. The drop in performance seems due to the following reasons: (i) close monosemous relatives found for sense 1 are rare (direct hyponyms such as "headspring" or "provenance" are used), and (ii) far and highly productive relatives are used for senses 2 and 7, which introduce noise (e.g. the related multiword "new edition" for sense 7). In the case of the "relatives in context" algorithm, even if we have a similar set of relatives per each sense, the local context seems to help disambiguate better, achieving a higher recall of 53.1%. In this case the first sense, which is the most frequent in the test set (with 65% of the instances), is better represented and this allows for improved recall.

Following with the target word *source*, we picked a real example from the test set to see the behaviour of the algorithms. This sentence was hand-tagged with sense 1, and we show here a fragment containing the target word:

> *...tax will have been deducted at* **source***, and this will enable you to sign a Certificate of Deduction...*

The monosemous relatives method is not able to find good collocations in the noisy training data, and it has to rely in bag-of-word features to make its decision. These are not usually as precise as local collocations, and in the example they point to senses 1, 2, and 4. The scorer gives only 1/3 credit to the algorithm in this case. Notice that one of the advantages of using Decision Lists is that it allows us to have a closer look to the features that are applied in each decision. Regarding the "relatives in context" method, in this example it is able to find the correct sense relying in collocations such as *deducted at origin* and *deducted at beginning*.

7 Comparison with other systems

In this section we compared our results with some of the state-of-the-art systems described in Section 2 for this dataset. We chose the Automatic Ranking of Senses by (McCarthy et al., 2004), and the Machine Translation approach by (Wang and Martinez, 2006). These unsupervised systems were selected for a number of reasons: they have been tested in Senseval data with good performance, the techniques are based on different knowledge sources, and the results on Senseval data were available to us. We also devised a simple unsupervised heuristic that would always choose the sense that had a higher number of close relatives in WordNet, picking randomly when there was a tie. We tested this approach in previous work (Wang and Martinez, 2006) and it showed to work well for discarding rare senses. We applied it here as a standalone system. We do not include the results of supervised systems because they can benefit strongly from ready-made hand-tagged data, which is not normally available in a real setting.

The performance of the three systems, together with the previous two, is given in Table 4. We can see that overall the Automatic ranking approach (RK) gives the best performance, with almost the same score as our Relatives in Context (RC) approach. The Machine Translation (MT) method performs 2 points lower overall, but its recall is balanced in the two different datasets. Surprisingly, the simple Number of Relatives (NR) heuristic does better than the Monosemous Relatives (MR), performing specially well in the S3LS

Algorithm	Avg S2LS	Avg S3LS	Avg S2LS-S3LS
RK	39.0	**45.5**	**42.5**
MT	40.8	40.7	40.7
NR	33.6	43.0	38.7
RC	**41.5**	43.2	42.4
MR	39.9	34.2	36.8

Table 4: Recall of different algorithms on Senseval datasets. Best results per column in bold. RK: Automatic Ranking of Senses, MT: Translation-based, NR: Number of Relatives heuristic, RC: Relatives in Context, MR: Monosemous Relatives.

dataset.

We can now analyse the performance of the five systems per word. The results are given in Table 5. We can see that the choice of the target word heavily affects the results of the algorithms, with most of them having very low results for a handful of words, with recall below 20% and even 10%. These very low results make a difference when compared to supervised systems, which do degrade gracefully. None of the algorithms is robust enough to achieve an acceptable performance for all words.

Measuring the agreement of the different algorithms is a way to know if a combined system would improve the results. We calculated the kappa statistic, which has been widely used in NLP tasks (Carletta, 1996), to measure the agreement on the answers of the algorithms in S2LS and S3LS (cf. Table 6). The table shows the averaged results per word of the S2LS dataset in the upper-right side, and the S3LS values in the bottom-left side. We can see that all the results are closer to 0 than 1, indicating that they tend to disagree, and suggesting that the systems offer good prospects for combination. The highest agreement is attained between methods RK and NR in both datasets, and the lowest between RC and MT.

In order to study the potential for combination, we tried the simplest method, that of one system one vote, where each system returns a single vote for the winning sense, and the sense getting most votes wins. In case of ties, all the senses getting the same number of votes are returned. Note that the Senseval scorer penalises systems returning multiple senses (unless all of them are correct).

The results of the ensemble and also of leaving one system out in turn are given in Table 7. The table shows that the best combination (in bold) for each of the datasets varies, which is natural given the variance of each of the single systems, and that the combination of all 5 systems attains very good performance on both datasets.

In the lower lines, Table 7 shows a number of reference systems: the best unsupervised system that took part in each of the S2LS and S3LS com-

Words	Algorithms				
	MR	RC	MT	RK	NR
art	**61.1**	40.3	47.2	**61.1**	**61.1**
authority	22.0	**45.1**	17.6	37.4	37.4
bar	**52.1**	16.9	44.1	14.4	14.4
bum	18.8	72.5	80.0	**85.0**	7.5
chair	62.9	54.8	85.5	**88.7**	**88.7**
channel	28.7	27.9	**47.1**	10.3	2.9
child	1.6	46.8	35.5	43.5	**56.5**
church	**62.1**	58.1	32.3	40.3	40.3
circuit	52.8	47.2	**54.7**	43.4	43.4
day	2.2	**36.7**	32.4	1.4	4.3
detention	16.7	62.5	79.2	**87.5**	12.5
dyke	**89.3**	85.7	67.9	**89.3**	**89.3**
facility	26.8	**50.0**	17.9	26.8	26.8
fatigue	73.8	67.5	67.5	**82.5**	**82.5**
feeling	51.0	49.0	16.3	**59.2**	**59.2**
grip	8.0	**26.0**	14.0	16.0	8.0
hearth	37.5	40.6	56.2	**75.0**	**75.0**
holiday	7.4	74.1	11.1	7.4	**96.3**
lady	**79.3**	8.7	45.7	10.9	10.9
material	**50.8**	**50.8**	19.5	15.3	15.3
mouth	41.2	43.9	41.2	**56.1**	**56.1**
nation	**80.6**	36.1	37.5	**80.6**	19.4
nature	**44.4**	26.7	22.2	17.8	21.1
post	**47.4**	36.2	37.9	43.1	43.1
restraint	9.1	**22.7**	13.6	18.2	9.1
sense	18.6	**48.8**	37.2	11.6	11.6
spade	66.1	32.3	**67.7**	**67.7**	3.2
stress	52.6	21.1	**55.3**	50.0	2.6
yew	**85.2**	55.6	**85.2**	81.5	81.5
argument	24.7	38.7	45.9	**51.4**	21.6
arm	10.2	27.1	71.4	**82.0**	44.0
atmosphere	31.3	24.7	45.7	**66.7**	**66.7**
audience	51.8	34.0	57.0	**67.0**	**67.0**
bank	32.3	60.6	37.1	**67.4**	**67.4**
degree	39.3	**43.8**	41.4	22.7	16.4
difference	26.4	23.7	32.5	**40.4**	16.7
difficulty	13.0	**43.5**	26.1	34.8	34.8
disc	52.2	45.0	**58.0**	27.0	27.0
image	4.1	23.0	21.6	**36.5**	**36.5**
interest	26.8	23.1	31.2	**41.9**	11.8
judgment	20.6	25.0	**40.6**	28.1	28.1
organization	71.4	69.6	19.6	**73.2**	**73.2**
paper	25.6	**42.7**	30.8	23.1	25.6
party	**67.5**	67.2	52.6	6.9	62.1
performance	20.5	33.3	**46.0**	24.1	26.4
plan	78.0	76.2	29.8	**82.1**	**82.1**
shelter	36.2	**44.9**	39.8	33.7	**44.9**
sort	13.5	**65.6**	20.8	**65.6**	**65.6**
source	22.4	53.1	9.4	0.0	**65.6**
Wins	11	12	8	**22**	18
Average	36.8	42.4	40.7	**42.5**	38.7

Table 5: Recall of the 5 algorithms per word and in average, the best results per word are given in bold. The top rows show the S2LS words, and the bottom rows the S3LS words.

Algorithm	MR	RC	MT	RK	NR
MR	-	0.23	0.28	0.41	0.43
RC	0.13	-	**0.13**	0.31	0.30
MT	0.11	**0.09**	-	0.23	0.35
RK	0.33	0.25	0.26	-	**0.45**
NR	0.23	0.15	0.28	**0.36**	-

Table 6: Averaged kappa agreement between pairs of algorithms. Results on the S2LS dataset are given in the upper-right side, and the S3LS values in the bottom-left side.

System	S2LS	S3LS
All	42.3	51.0
Leave MR out	41.7	**52.1**
Leave RC out	40.3	48.3
Leave MT out	40.0	47.5
Leave RK out	44.5	46.7
Leave NR out	**45.9**	49.9
Best Senseval unsup	35.8	47.5
Best single system	41.5	45.5
Oracle	80.4	84.3

Table 7: Voting systems, best unsupervised systems, best single systems, and oracle on S2LS and S3LS.

petitions, and the best single system in each of the datasets[2]. The combination of the 5 systems is able to beat all of them in both datasets, showing that the simple voting system was effective to improve the single systems and attain the best totally unsupervised system in this dataset. We can also see that the novel technique described in this paper (RC) contributes to improve the ensemble in both datasets. This does not happen for the monosemous relatives approach, which degrades performance in S3LS.

As an upperbound, we also include the oracle combination in Table 7, which determines that an instance has been correctly tagged if any of the algorithms has got it right. This oracle shows that the union of the 5 systems cover 80.4% and 84.3% of the correct solutions for each of the datasets, and that there is ample room for more sophisticated combination strategies.

8 Conclusions and Future work

The current situation for WSD is somewhat stuck due to lack of training data. We present in this paper a novel disambiguation algorithm that improves previous systems based on the acquisition of examples by incorporating local context information. With a basic configuration, our method is able to obtain state-of-the-art performance.

We complemented this work by evaluating other well-known methods in the same dataset, and analysing the comparative results per word. We observed that each algorithm performed better for different types of words, and each of them failed for some particular words. We then proposed a simple unsupervised voting scheme that improved significantly over single systems, achieving the best performance on both the Senseval 2 and Senseval 3 lexical sample datasets.

We have also shown that there is ample room for improvement, as the oracle combination sets an upperbound of around 80% for a perfect combination. This work naturally leads to explore more sophisticated combination strategies, using meta-learning to try to understand which features of each word make a certain WSD system succeed (or fail). We would also like to widen the range of systems, either using existing unsupervised off-the-shelf WSD systems and/or reimplementing them.

Regarding the “Relatives in Context” method, there are different avenues to explore. We plan to use this approach to acquire automatic sense tagged data for training, instead of relying on rules. We also would like to study the use of richer features than the local strings to acquire examples that have similar linguistic structures.

Finally, we want to test the new technique on an all-words corpus. A simple approach would be to process each instance of each word separately as in the lexical sample. However, we could also try to disambiguate all words in the context together, by substituting the target words with their relatives jointly. We are comparing our unsupervised systems in the testbeds where supervised systems are comfortable (lexical-sample tasks). We think that unsupervised systems can have the winning hand in more realistic settings like those posed by Senseval all-words tasks.

[2] For nouns the best scores of the competing systems were obtained by dictionary-based systems in both S2LS (Litkowski, 2001), and S3LS (Pedersen, 2004).

References

E. Agirre and P. Edmonds. 2006. *Word Sense Disambiguation*. Kluwer Academic Publishers.

E. Agirre and D. Martinez. 2004. Unsupervised wsd based on automatically retrieved examples: The importance of bias. In *Proceedings of EMNLP 2004*, Barcelona, Spain.

B. Snyder and M. Palmer. 2004. The English all-words task. In *Proceedings of the 3rd ACL workshop on the Evaluation of Systems for the Semantic Analysis of Text (SENSEVAL)*, Barcelona, Spain.

J. Carletta. 1996. Assessing agreement on classification tasks: the kappa statistic. In *Computational Linguistics 22(2)*.

Y.S. Chan and H.T. Ng. 2005. Scaling up word sense disambiguation via parallel texts. In *Proceedings of the 20th National Conference on Artificial Intelligence (AAAI 2005)*, Pittsburgh, Pennsylvania, USA.

M. Ciaramita and Y. Altun. 2006. Broad-coverage sense disambiguation and information extraction with a supersense sequence tagger. In *Proceedings of EMNLP 2006*.

C. Fellbaum. 1998. *WordNet: An Electronic Lexical Database*. MIT Press.

A. Kilgarriff. 2001. English Lexical Sample Task Description. In *Proceedings of the Second International Workshop on evaluating Word Sense Disambiguation Systems*, Toulouse, France.

U. Kohomban and W.S. Lee. 2005. Learning Semantic Classes for Word Sense Disambiguation. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*.

C. Leacock, M. Chodorow, and G. A. Miller. 1998. Using corpus statistics and WordNet relations for sense identification. In *Computational Linguistics*, volume 24, pages 147–165.

D. Lin. 1998. Automatic retrieval and clustering of similar words. In *In Proceedings of COLING-ACL*, Montreal, Canada.

K. Litkowski. 2001. Use of Machine-Readable Dictionaries in Word-Sense Disambiguation for Senseval-2. In *Proceedings of the Second International Workshop on evaluating Word Sense Disambiguation Systems*, Toulouse, France.

B. Magnini and G. Cavagliá. 2000. Integrating subject field codes into WordNet. In *Proceedings of the Second International LREC Conference*, Athens, Greece.

D. McCarthy, R. Koeling, J. Weeds, and J. Carroll. 2004. Finding Predominant Word Senses in Untagged Text. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL)*, Barcelona, Spain.

R. Mihalcea, T. Chklovski, and Adam Killgariff. 2004. The Senseval-3 English lexical sample task. In *Proceedings of the 3rd ACL workshop on the Evaluation of Systems for the Semantic Analysis of Text (SENSEVAL)*, Barcelona, Spain.

R. Mihalcea. 2002. Bootstrapping large sense tagged corpora. In *Proceedings of the 3rd International Conference on Languages Resources and Evaluations (LREC 2002)*, Las Palmas, Spain.

C. Niu, W. Li, R.K. Srihari, and H. Li. 2005. Word independent context pair classification model for word sense disambiguation. In *Proceedings of the Ninth Conference on Computational Natural Language Learning (CoNLL-2005)*.

T. Pedersen. 2004. The duluth lexical sample systems in senseval-3. In *Proceedings of the 3rd ACL workshop on the Evaluation of Systems for the Semantic Analysis of Text (SENSEVAL)*, Barcelona, Spain.

P. Resnik and A. Elkiss. 2005. The linguist's search engine: An overview. In *Proceedings of ACL 2005 (Demonstration Section)*.

X. Wang and J. Carroll. 2005. Word sense disambiguation using sense examples automatically acquired from a second language. In *Proceedings of HLT/EMNLP*, Vancouver, Canada.

X. Wang and D. Martinez. 2006. Word sense disambiguation using automatically translated sense examples. In *Proceedings of EACL 2006 Workshop on Cross Language Knowledge Induction*, Trento, Italy.

D. Yarowsky. 1994. Decision Lists for Lexical Ambiguity Resolution: Application to Accent Restoration in Spanish and French. In *Proceedings of the 32nd Annual Meeting of the Association for Computational Linguistics*, Las Cruces, NM.