Natural Language Engineering **8** (4): 343–358. © 2002 Cambridge University Press
DOI: 10.1017/S1351324902002991 Printed in the United Kingdom

Word sense disambiguation with pattern learning and automatic feature selection

RADA F. MIHALCEA

Department of Computer Science, University of North Texas, Denton, TX 76203-1366, USA
e-mail: rada@cs.unt.edu

(*Received 2 November 2001; revised 25 July 2002*)

Abstract

This paper presents a novel approach for word sense disambiguation. The underlying algorithm has two main components: (1) pattern learning from available sense-tagged corpora (SemCor), from dictionary definitions (WordNet) and from a generated corpus (GenCor); and (2) instance based learning with automatic feature selection, when training data is available for a particular word. The ideas described in this paper were implemented in a system that achieves excellent performance on the data provided during the SENSEVAL-2 evaluation exercise, for both *English all words* and *English lexical sample* tasks.

1 Introduction

Word Sense Disambiguation (WSD) does not need any more an introduction and particularly not in a special issue on WSD evaluation. It is well known that WSD constitutes one of the hardest problems in natural language processing, yet is a necessary step in a large range of applications including machine translation, knowledge acquisition, coreference, information retrieval and others. This fact motivates a continuously increasing number of researchers to develop WSD systems and devote time for finding solutions to this challenging problem.

The system presented here was initially designed for the semantic disambiguation of *all words* in open text. The SENSEVAL competitions provided a good environment for supervised systems, and this fact motivated us to improve our system with the capability of incorporating larger training data sets when available.

There are two important modules in this system. The first one uses pattern learning that relies on machine readable dictionaries and sense-tagged corpora to tag all words in open text. The second module is triggered only for words with large training data, as was the case with the words from the lexical sample tasks. It uses an instance-based learning algorithm with automatic feature selection.

To our knowledge, both pattern learning and automatic feature selection are novel approaches in the WSD field, and they led to very good results during the SENSEVAL-2 evaluation exercise.

https://doi.org/10.1017/S1351324902002991 Published online by Cambridge University Press

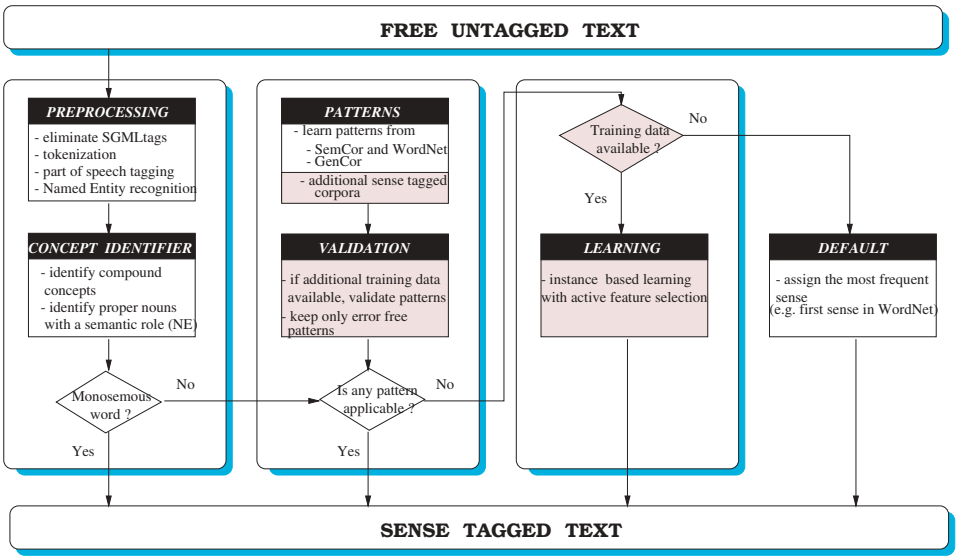


Fig. 1. System architecture.

2 System description

The WSD algorithm used in this system has the capability of tagging words when no specific sense-tagged corpus is available, automatically scaling up to larger training data when provided. Figure 1 shows the system architecture. There are two main components: (1) pattern learning from available sense-tagged corpora and dictionary definitions; and (2) instance-based learning with automatic feature selection. The two modules are preceded by a preprocessing phase that includes compound concept identification, followed by a default phase that assigns the most frequent sense as a last resort, when no other previous methods could be applied.

During the preprocessing stage, SGML tags are eliminated, the text is tokenized, part of speech tags are assigned using Brill tagger (Brill 1995), and Named Entities (NE) are identified with an *in-house* implementation of an NE recognizer. To identify collocations, we determine sequences of words that form compound concepts defined in WordNet. There are two possible problems with this approach. The first concerns subsuming concepts, as in *'United States'* and *'United States of America'*. In such cases, priority is given to the longest sequence of words. The second possible conflict regards overlapping concepts, like the two different compounds *'English Channel'* and *'Channel Tunnel'* found in the text *'English Channel Tunnel'*. Here, we break the tie by keeping the last encountered collocation, with the only reason for this decision being the ease of implementation.

In the second stage, patterns are learned from WordNet, SemCor and GenCor, which is a large sense-tagged corpus automatically built via a set of heuristics (Mihalcea 2002). If additional training data is available, patterns may be filtered through a validation process. Practically, patterns are applied on the sense-tagged data, and they are kept only if no counter-examples are found in the training sets provided.

https://doi.org/10.1017/S1351324902002991 Published online by Cambridge University Press

The third step consists of a learning mechanism with automatic feature selection. This step is initiated only for words with a sufficiently large number of examples, as it was the case with the words in the SENSEVAL lexical sample tasks.

3 Pattern learning for large vocabulary WSD

Pattern learning and matching is a technique that was successfully used in other NLP tasks, including the disambiguation of confusable word pairs (Brill 2000) and shallow parsing (Argamon, Dagan and Krymolowski 1998).

Within our system, the pattern learning module is intended for solving the semantic ambiguity of *all* words in open text. To this end, we build disambiguation patterns using SemCor, WordNet and GenCor. Several processing steps were required to transform the first two resources into a corpus useful for the task of open text WSD. Moreover, these lexical resources coupled with a set of heuristics are used as seeds for generating a new sense-tagged corpus called GenCor.

SemCor To our knowledge, SemCor (Miller, Leacock, Randee and Bunker 1993) is the only sense-tagged corpus freely available that tags all words in open text. The SENSEVAL-2 English tasks decided to use WordNet 1.7 sense inventory, while SemCor was available only for earlier versions of WordNet. We had therefore to process this corpus and map the WordNet 1.6 senses to their corresponding senses in WordNet 1.7.[1]

WordNet Besides being a large sense inventory, WordNet (Miller 1995) can also be used as a source of examples for the different semantic meanings of a word, through the definitions and examples rendered for each word sense. The main idea in generating a sense-tagged corpus out of WordNet is very simple. It is based on the underlying assumption that each example pertains to a word belonging to the current synset, thereby allowing us to assign the correct sense to at least one word in each example. For instance, the example given for *mother #4* is *'necessity is the mother of invention'*, where the word *mother* can be tagged with its appropriate sense.

GenCor is a generated sense-tagged corpus. More details on how GenCor is generated are presented in Mihalcea (2002). The algorithm underneath *GenCor* combines and extends the approaches proposed by Yarowsky (1995) and Mihalcea and Moldovan (1999) to obtain large collections of sense-tagged examples. Shortly, the generation algorithm is iterative and consists of three main steps:

- **Step 1.** Create a set of seeds, consisting of noun phrases and verb-noun constructs, extracted from:
  - 1.1 SemCor
  - 1.2 Sense-tagged examples in WordNet

[1] SemCor 1.6 is available for download from the WordNet site, http://www.cogsci.princeton.edu/~wn/. SemCor 1.7 can be downloaded from http://www.seas.smu.edu/~rada/semcor

https://doi.org/10.1017/S1351324902002991 Published online by Cambridge University Press

1.3 Sense-tagged examples created with the principles described in Mihalcea and Moldovan (1999). These are examples found on the Web by searching for related unambiguous words. Currently, we only use monosemous synonyms, hypernyms or hyponyms for any given word. For instance, sense-tagged examples for *mother#2* may be obtained by searching for its monosemous hypernym *barm*, and then replace *barm* with *mother* in all the examples that are retrieved.

- **Step 2.** Search the Web for each sequence of words in the seeds set.
- **Step 3.** In the documents that are retrieved, disambiguate words in a small text snippet surrounding the searching seed, using the main ideas of the algorithm in Mihalcea and Moldovan (2000). In this algorithm, words are disambiguated based on their relation with other words in their immediate vicinity. To this end, we rely on the synonymy and hypernymy relations, as defined in WordNet. For example, *mother* and *parent* found close to each other are disambiguated based on their relation *parent#1 is-hypernym mother#1*. Similarly, *car* in the immediate vicinity of *railcar* may be annotated as *car#2*, since *car#2* and *railcar#1* are synonyms in WordNet. Noun phrases and verb noun constructs including the newly disambiguated words form new seeds that are added to the seeds set. Go back to step 2.

Example *'blooming plant#2'* is a noun phrase extracted from SemCor as part of the initial set of seeds. A search on the Web for this construct results in several texts, including *'(2) Florist item means a cut flower, potted plant, blooming plant, inside foliage plant, bedding plant, corsage flower, cut foliage, floral decoration, or live decorative material'*. In this text, we disambiguate all instances of *plant*, and obtain the following new seeds: *potted plant#2*, *foliage plant#2*, *bedding plant#2*. Subsequent searches for these seeds will result in additional texts where new seeds may be extracted. The generation process continues for several iterations, and stops when a certain *a priori* established number of tagged examples is obtained. For instance, the corpus generated for the *English all words* task during SENSEVAL-2 consisted of about 160,000 examples.

Once we create this large corpus with examples of word meanings, we can start to extract patterns. For each semantically tagged word found in the corpus, patterns are constructed including the word itself and its local context. The local context is formed with a window of maximum N words to the left and M words to the right of the word of interest (currently, $M = N = 2$).

Each word in the corpus is represented by its base form, its part of speech, its sense[2], if there is any provided, and its hypernym, again if the sense is known. We have therefore the following format for each pattern word: *baseform/POS/offset/hypernym-offset*. Any of these word components can be unspecified, and therefore denoted with the symbol '*'. A count is also associated with every pattern, indicating the number of times it occurs in the corpus.

[2] The sense is specified through the synset offset. The benefit of this notation is that we enable synonym matches, e.g. *mother#1* and *female_parent#1* both have the offset 08284239.

https://doi.org/10.1017/S1351324902002991 Published online by Cambridge University Press

Additionally, a set of constraints is applied to filter out meaningless patterns. For instance, based on the observation that patterns like ⟨the/DT/*/* rest/NN/11411361/*⟩ (obtained for $N=1, M=0$) usually lack meaningful information, we filter out all patterns consisting of a DT-NN sequence. These constraints are basically indicators of what word combinations are not allowed in the patterns set. In addition to DT-NN, patterns may not consist of a modal followed by a verb (MD-VB), a noun followed by a conjunction (NN-CC), and others.

When trying to disambiguate a word, we first search for all available patterns that would match the current context. A pattern is said to match the current context if: (1) all words in the pattern are retrieved in the local context in the same order and at the same relative distance with respect to the target word; and (2) each pattern word has a complete or partial match with its corresponding context word. A complete match is obtained when all specified word components are matched. If only a subset of the pattern word components find a match among the corresponding context word components, then we have a partial match, and a smaller score is assigned, as shown in the *PatternMatching* function below. If more than one pattern is available, then the decision of which pattern to apply is based on the pattern *strength*. The strength of a pattern is evaluated in terms of (1) the number of specified components, (2) the number of occurrences and (3) the length of the pattern. For example, ⟨the/DT/*/* modal/JJ/01551759/* age/NN/*/* at/IN/*/*⟩ is considered to be stronger than ⟨modal/JJ/01551759/* age/NN/*/*⟩. Also, ⟨clear/JJ/00406603/* water/NN/12281250/*⟩ is stronger than ⟨clear/JJ/*/* water/NN/12281250/*⟩. The hypernym is also provided for the purpose of allowing generalizations. For instance, ⟨*/NN/*/03507584 door/NN/02746251/*⟩ matches 'kitchen door' as well as 'bedroom door' (03507584 is the offset for *room #1*). The *PatternMatching* function below illustrates the main steps performed during pattern matching.

```
function PatternMatching(word W)
  find patterns P_i containing W
  if at least one pattern found
  then
   for each pattern P_i
    Score(P_i) = 0
    for each word W_Pi in the pattern, W_Pi ≠ W
     Score (P_i) += ScorePatternWord (W_Pi)
    if Score (P_i) ≠ 0 for at least one pattern
     then return sense for W from pattern P_i with Score (P_i) = MAX
     else
      for all senses s of W
       if ∃ hypernym (W_s)
        if PatternMatching (W_s) ≠ 0
         then return s
    return 0
```

https://doi.org/10.1017/S1351324902002991 Published online by Cambridge University Press

function *ScorePatternWord(word* W_{P_i}*)*
try to match pattern word W_{P_i} *on current context*
if *complete match* **then return** *4*
if *word + POS match* **then return** *3*
if *offset + POS match* **then return** *3*
if *hypernym-offset match* **then return** *2*
if *POS match* **then return** *1*

Another important step performed during the all words disambiguation task is sense propagation. The patterns do not guarantee a complete coverage of all words in input text, and therefore additional methods are required. We use a cache-like procedure that relies on the 'one sense per discourse' paradigm to assign to each ambiguous word the sense of its closest occurrence, if any exists. The words left ambiguous at this point are assigned by default the first sense in WordNet.

4 Learning with automatic feature selection

Learning mechanisms for disambiguating word sense have a long tradition in the WSD field, including a large range of algorithms and feature types. Most of the efforts in the WSD field have been concentrated so far towards *supervised* learning algorithms, and these are the methods that achieve the best performance at the cost of low recall (they address only few, pre-selected words). Each sense-tagged occurrence of a particular word is transformed into a feature vector, suitable for an automatic learning process. Two main decisions need to be taken when designing such a system: the set of features to be used and the learning algorithm. Commonly used features include surrounding words and their part of speech, context keywords (Ng and Lee 1996) or context bigrams (Pedersen 2001), and various syntactic properties (Fellbaum, Palmer, Dang, Delfs and Wolf 2001), etc. As for the learning methodology, a large range of algorithms have been used, including neural networks (Leacock, Chodorow and Miller 1998), decision trees (Pedersen 2001), decision lists (Yarowsky 2000), memory-based learning (Veenstra, van den Bosch, Buchholz, Daelemans and Zavrel 2000), and others. An experimental comparison of seven learning algorithms used to disambiguate the meaning of the word *line* is presented in Mooney (1996). See also Yarowsky and Florian (2002) in this issue.

For our system, we have decided for an instance based algorithm with information gain feature weighting. The reasons for this decision are threefold. First, it has been advocated that forgetting exceptions is harmful for language learning applications (Daelemans, van den Bosch and Zavrel 1999), and instance-based algorithms are known for their property of taking into consideration every single training example when making a classification decision. Secondly, instance-based learning algorithms have been successfully used in WSD applications (Veenstra *et al.* 2000). Finally, this type of algorithms is efficient in terms of training and testing time. We initially used

https://doi.org/10.1017/S1351324902002991 Published online by Cambridge University Press

the MLC_{++}[3] implementation, and later on switched to Timbl (Daelemans, Zavrel, ven der Sloot and van den Bosch 2001).

Even more important than the choice of learning methodology is the selection of features to be employed during the learning process. Our intuition was that different sets of features have different effects depending on the ambiguous word considered. Usually, features are weighted using weighting schemes that are based on information gain, gain ratio, chi-squared or other information content measures. Still, weights are computed independently for each feature and therefore this strategy does not always guarantee to provide the best results. Sometimes it is better to leave features out than assign them even a small weight (Daelemans *et al.* 2001). We need therefore to identify efficient criteria for feature selection.

Feature selection is a technique that has been successfully used in other Artificial Intelligence applications. Cardie (1996) proposes a linguistic and cognitive biased approach for relative pronoun resolution. In Aha and Bankert's (1994) system, features are selected using searching algorithms, with increased performance obtained in the problem of cloud types classification. In all these applications, performing feature selection prior to the learning phase was found to be a helpful factor towards increased performance.

For our system, features are automatically selected using a forward search algorithm. The classic approach used so far in WSD was to build word experts via a learning process that determines the values for a pre-selected set of features. Instead, we first learn the set of features that would best model the word characteristics, and therefore exploit at maximum the idiosyncratic nature of words. It is only at a second stage that we actually create the word experts by determining the values for the set of features previously selected.

Using this approach, we combine the advantages of instance based learning mechanisms that have the useful property of *'not forgetting exceptions'*, with an optimized feature selection scheme. One could argue that decision trees have the capability of selecting relevant features, but it has been shown (Almuallim and Dietterich 1991) that irrelevant features significantly affect the performance of decision trees as well.

The algorithm for automatic feature selection is sketched below.

---

function *Automatic Feature Selection*
  generate a pool of features $PF = \{F_i\}$
  initialize the set of selected features with the empty set $SF = \{\emptyset\}$
  extract training and testing corpora for the given target ambiguous word.
  loop: for *each feature* $F_i \in PF$
    run a 10-fold cross validation on the training set; each example in the training set contains the features in SF **and** *the feature* F_i.
    determine the feature F_i *leading to the best accuracy*
    remove F_i *from* PF *and add it to* SF
  goto loop *until no improvements are obtained*

---

[3] Machine Learning library available at http://www.sgi.com/tech/mlc.

https://doi.org/10.1017/S1351324902002991 Published online by Cambridge University Press

5 Features that are good indicators of word sense

Three types of features are distinguished:

1. *0-param* features, which can be used or not, without any parameters to set. For example, the part of speech of a surrounding word is a *0-param* feature, since a learning example can either contain or omit this feature, without having to indicate a specific parameter.
2. *1-param* features, which, once selected, have one variable parameter that can be set to a specific value (alternatively, this parameter may be left with its default value). As an example, consider the *context* feature (CF), which adds as attributes the words in a surrounding window of length K. Deciding the value for K implicitly means setting *one parameter* for this feature.
3. *2-param* features with two parameters associated. For example, one can select MX keywords as representative for the context of an ambiguous word, where a keyword is defined as a word occurring at least MN times. Therefore, *two parameters* have to be set for this feature, MX and MN.

The features that we employed so far are presented below. They form the *pool of features* PF from which features are selected using the algorithm described in section 4. In the following, the ambiguous word is denoted with *AW*:

CW *Current word* (*0-param*) The word *AW* itself, exactly as it occurs in the text. *Notation: CW*

CP *Current part of speech* (*0-param*) The part of speech of the word *AW*. *Notation: CP*

CF *Contextual features* (*1-param*) The words and parts of speech of the K words surrounding *AW* (Bruce and Wiebe 1999). *Notation: CF[=K], default-3*

COL *Collocations* (*1-param*) Collocations formed with maximum K words surrounding *AW* (Ng and Lee 1996). *Notation: COL[=K], default = 3*

HNP *Head of noun phrase 2* (*0-param*) The head of the noun phrase to which *AW* belongs, if any. *Notation: HNP*

SK *Sense specific keywords* (*2-param*) Maximum MX keywords occurring at least MN times are determined for each sense of the ambiguous word. The value of this feature is either 0 or 1, depending whether the current example contains one of the determined keywords or not (Ng and Lee 1996). *Notation: SK[=MN,MX], default = 5,5*

B *Bigrams* (*2-param*) Maximum MX bigrams occurring at least MN times are determined for all training examples. The value of this feature is either 0 or 1, depending if the current example contains one of the determined bigrams or not. Bigrams are ordered using the Dice coefficient, which gives a measure of association among two words in a corpus. Pedersen (2001) gives several alternatives for measures used in bigrams selection. *Notation: B[=MN,MX], default = 5,20*

Other In addition, we have a set of eleven other features that refer to surrounding words with a given part of speech: *Verb before* (VB *0-param*), *Verb after* (VA *0-param*), *Noun before* (NB *0-param*), *Noun after* (NA *0-param*), *Named*

https://doi.org/10.1017/S1351324902002991 Published online by Cambridge University Press

Entity before (NEB *0-param*), *Named Entity after* (NEA *0-param*), *Preposition before* (PB *0-param*), *Preposition after* (PA *0-param*), *Pronoun before* (PRB *0-param*), *Pronoun after* (PRA *0-param*), *Determiner before* (DT *0-param*).

New features can be easily added to the pool, with no changes required in the main algorithm. We have initially tested the system on the SENSEVAL-1 data, in an non-competitional environment, when there was enough time to parse the data. Two additional features were considered at that time to help towards performance. We decided not to use them in the current experiments, mainly for time considerations, since parsing is a highly computationally intensive task.

PPT *Parse path* (*1-param*) Maximum K parsing labels found on the path to the top of the parse tree (sentence top). *Notation: PPT[=K], default=10*. For instance, given the parse tree *(S(NP(JJ big) (NN house)))*, the value for this feature for the noun *house* is *NN, NP, S*.

SPC *Same parse phrase components* (*1-param*) Maximum K parse components found in the same phrase as *AW*. *Notation: SPC[=K], default=3*. For the example above, this feature would be set to *JJ, NN*.

6 Results on SENSEVAL-2 data

The overall performance of the system on the data provided during the *English all words* task was 69% for fine-grained scoring, and 69.8% for coarse-grained scoring. On the *English lexical sample* data, we obtain 63.8% for fine-grained scoring, and 71.2% for coarse-grained scoring. These results rank this system as the best performing one in the ranking made before the deadline. See Edmonds and Cotton (2002) for details on SENSEVAL-2.

Tables 1–3 present the results obtained during the *lexical sample* task, for 73 ambiguous words, including 29 nouns, 15 adjectives and 29 verbs. For each word, the table shows: the number of examples in the training and test sets; the features automatically selected as a result of applying the algorithm in section 4; the 10-fold cross validation precision obtained on training data with the selected features set; the precision for fine-grained and coarse-grained scoring as computed by the SENSEVAL-2 organizers. Collocations are identified since the preprocessing stage and the learning process is applied separately on each concept[4]; due to space limitations, the table shows only features and results obtained for single words.

For the *1-param* and *2-param* features, there is a range of values allowed for their parameters: [1–5] for the *1-param* features, and [1–10] for the *2-param* features. This means that, for instance, CF can be set to CF = 1, CF = 2, CF = 3, CF = 4 or CF = 5. The selection of the best value is performed empirically using the same cross-validation algorithm.

[4] Training and testing corpora are extracted for each ambiguous word. This means that examples pertaining to the multiword *'dress down'* are separated from the examples for the single word *'dress'*.

https://doi.org/10.1017/S1351324902002991 Published online by Cambridge University Press

Table 1. *Training and test sizes, optimal feature sets and precisions (10-fold on training data, fine-grained and coarse-grained on test data) for 29 nouns*

Word.pos	Size		Features	10-fold valid.	SENSEVAL score	
	Train	Test			Fine	Coarse
art.n	194	98	CF = 1 HNP B = 2,5 VB NB	60.6%	71.4%	74.5%
authority.n	183	92	CW CP COL = 1 VB NB	62.2%	70.7%	91.3%
bar.n	264	151	CW CP CF = 1 COL = 1 B = 5,3 VB NB NEA	60.8%	62.3%	74.5%
bum.n	80	45	CW NA NEA	86.2%	77.8%	80.0%
chair.n	137	69	CW	92.3%	85.5%	88.4%
channel.n	138	73	CP NB	43.0%	46.6%	56.2%
child.n	129	64	CW CP CF = 1 COL = 1 B = 5,3 NB NEB DT	76.1%	68.8%	68.8%
church.n	128	64	CW CP CF = 2 COL = 1 B = 5,1	64.4%	56.2%	56.2%
circuit.n	169	85	CP CF = 3 B = 5,1 VB	51.6%	58.8%	62.4%
day.n	289	145	CP CF = 2 HNP NEB PB	78.0%	76.1%	77.3%
detention.n	63	32	any	94.0%	87.5%	87.5%
dyke.n	58	28	CW CF = 2 SK = 5,2	91.4%	89.3%	89.3%
facility.n	114	58	CP COL = 1 VB PRB	74.5%	79.3%	98.3%
fatigue.n	76	43	CP B = 5,3 NB	86.6%	88.4%	90.7%
feeling.n	102	51	CP CF = 1 COL = 3 HNP NEA	64.0%	74.5%	74.5%
grip.n	100	51	CP CF = 3 COL = 2 PB DT	60.0%	41.2%	58.8%
hearth.n	64	32	CP CF = 1 HNP	66.7%	75.0%	87.5%
holiday.n	62	31	CP	96.0%	93.5%	96.8%
lady.n	103	53	CW HNP	84.0%	88.7%	94.3%
material.n	140	69	CW CP COL = 1 B = 2,5 VA NEA	53.3%	56.5%	60.9%
mouth.n	118	60	CP COL = 1 VB NB PB	65.7%	65.0%	93.3%
nation.n	75	37	CP	80.0%	54.1%	54.1%
nature.n	92	46	CP DT	58.0%	69.6%	80.4%
post.n	150	79	CW CP CF = 1 COL = 2	74.6%	64.6%	68.4%
restraint.n	91	45	CP COL = 2 HNP B = 2,5 VB NB PA	67.3%	62.2%	71.1%
sense.n	107	53	CP CF = 1 B = 3,3 NEB PB	74.5%	75.5%	74.4%
spade.n	64	33	CP CF = 1 COL = 2	94.0%	97.0%	97.0%
stress.n	78	39	CP COL = 2 B = 5,2	68.0%	64.1%	89.7%
yew.n	57	28	CF = 1	94.0%	89.3%	100.0%
TOTAL.N	3,523	1,759	–	–	69.5%	76.6%

When no training data is provided (as was the case with the SENSEVAL-2 verb *'keep going'*), the first sense is applied by default. Also, when the training set size is smaller than 15 examples, we do not use the automatic feature selection algorithm; we use instead a default set of features (CW CP CF = 1 COL = 1).

6.1 Discussion

The *all words* task owes its performance to SemCor, WordNet, GenCor, the pattern learning procedure, the cache-like sense propagation algorithm and the simple 'most

https://doi.org/10.1017/S1351324902002991 Published online by Cambridge University Press

Table 2. *Training and test sizes, optimal feature sets and precisions (10-fold on training data, fine-grained and coarse-grained on test data) for 15 adjectives*

Word.pos	Size		Features	10-fold valid.	SENSEVAL score	
	Train	Test			Fine	Coarse
blind.a	105	55	HNP	70.0%	85.5%	85.5%
colourless.a	68	35	CW CP CF = 1 COL = 1 SK = 3,3	85.7%	48.6%	48.6%
cool.a	103	52	CF = 1 COL = 2 HNP VB PB PRB DT	56.1%	51.9%	51.9%
faithful.a	47	23	CW	68.0%	87.0%	87.0%
fine.a	139	70	CP CF = 2 HNP B = 5,1 NA	46.0%	54.3%	54.3%
fit.a	57	29	CF = 1 B = 3,3 VB NA	85.0%	82.8%	82.8%
free.a	165	82	CP CF = 1 COL = 2	65.0%	58.5%	58.5%
graceful.a	56	29	CW	87.0%	79.3%	79.3%
green.a	190	94	CP VA	80.0%	79.8%	79.8%
local.a	75	38	CP NA	88.0%	81.6%	81.6%
natural.a	205	103	CP CF = 1 HNP VB NB NEB PRA	50.0%	56.3%	56.3%
oblique.a	56	29	CW CP CF = 1 COL = 4 B = 3,3	84.0%	86.2%	86.2%
simple.a	130	66	CP CF = 1 COL = 2 HNP NA PB PRA DT	53.3%	53.0%	53.0%
solemn.a	52	25	CP COL = 1 DT	92.8%	96.0%	96.0%
vital.a	74	38	CW CP NB	88.7%	94.7%	94.7%
TOTAL.A	1,535	768	–	–	68.8%	68.8%

frequent sense' heuristic. We address all open class words in open text, and therefore a recall of 100% is obtained on this data. From this, a coverage of 40.23% is due to pattern learning, 7.84% to sense propagation, and the rest of 51.93% is attained by tagging words with their most frequent sense. If only the last procedure is applied on the entire data set, the overall precision drops to 63.89%, which may be considered as a baseline for this task.

To determine the contribution of the various knowledge sources, and find the raise in precision brought by the use of GenCor, we performed two comparative experiments: one where only SemCor and WordNet were employed as sources of tagged data, and a second one where GenCor was used in addition to these two resources. The overall precision obtained during the first experiment was 65.1%, while the second experiment led to a precision of 69.3%, therefore more than 4% precision are gained due to GenCor.

The disambiguation of the words in the *lexical sample* task relies mainly on the SENSEVAL training data and the instance based learning algorithm with automatic feature selection, which provides complete coverage of the test data. Table 4 lists the number of times each feature was used in the semantic disambiguation of nouns, verbs and adjectives. The most often used features turn out to be CW, CP, CF and COL, which are also the features most frequently mentioned in the literature. Almost all words took advantage of the current part of speech (CP) feature. This is in agreement with Stevenson and Wilks (2001), who have emphasized the major

https://doi.org/10.1017/S1351324902002991 Published online by Cambridge University Press

Table 3. *Training and test sizes, optimal feature sets and precisions (10-fold on training data, fine-grained and coarse-grained on test data) for 29 verbs*

Word.pos	Size		Features	10-fold valid.	SENSEVAL score	
	Train	Test			Fine	Coarse
begin.v	557	280	CF = 1 NA	80.40%	87.5%	87.5%
call.v	132	66	CF = 1 COL = 2 VB NB DT	70.00%	40.9%	66.7%
carry.v	132	66	CW CP COL = 1 NB	35.00%	39.4%	50.0%
collaborate.v	57	30	CW CP CF = 1	95.80%	90.0%	90.0%
develop.v	133	69	CW CP B = 2,5 NA PB	22.50%	36.2%	49.3%
draw.v	82	41	CF = 2 COL = 2 NEB	11.00%	31.7%	43.9%
dress.v	119	59	CP CF = 1 NB NA PB	57.50%	57.6%	86.4%
drift.v	63	32	CW CP CF = 2 COL = 3 HNP NEB PA	22.00%	59.4%	62.5%
drive.v	84	42	CW CP CF = 2 PRA DT	45.00%	52.4%	69.0%
face.v	186	93	CP	84.00%	81.7%	90.3%
ferret.v	2	1	any	–	100.0%	100.0%
find.v	132	68	CP CF = 2 SK = 5,2	10.00%	29.4%	39.7%
keep.v	133	67	CP B = 3,3	38.00%	44.8%	46.3%
leave.v	132	66	CP CF = 1 COL = 3 NEA	28.90%	47.0%	53.0%
live.v	129	67	CP NA	63.00%	67.2%	68.7%
match.v	86	42	CW CP HNP SK = 5,5 NA	26.40%	40.5%	59.5%
play.v	129	66	CW CP CF = 4 COL = 4 VB NA	21.00%	50.0%	51.5%
pull.v	122	60	CP COL = 1 HNP B = 2,10 SK = 5,5	23.00%	48.3%	68.3%
replace.v	86	45	CP COL = 3 SK = 5,1 B = 3,2	54.00%	44.4%	88.9%
see.v	131	69	CW CP CF = 2 SK = 4,4 PB	23.00%	37.7%	42.0%
serve.v	100	51	CP CF = 4 HNP B = 5,5 VA NEB PRB PRA	36.00%	49.0%	54.9%
strike.v	104	54	CW CP CF = 2 NEB	23.00%	38.9%	51.9%
train.v	125	63	CW CP CF = 2 COL = 4 NA PB PA DT	34.00%	41.3%	52.4%
treat.v	88	44	CP CF = 3 COL = 2 VB NEA PA PRB PRA	36.00%	63.6%	79.5%
turn.v	131	67	CP CF = 2 VB NA PA PRB	30.70%	35.8%	53.7%
use.v	147	76	CW CP NA VA PRB	65.00%	72.4%	84.2%
wander.v	100	50	CP PA	81.00%	74.0%	90.0%
wash.v	25	12	CW CP CF = 2 COL = 2 SK = 3,5 NEA	32.00%	66.7%	83.3%
work.v	119	60	CW CP CF = 2 COL = 2 B = 3,3 NA PA	42.00%	43.3%	58.3%
TOTAL.v	3,673	1,857	–	–	56.4%	67.0%

role played by part of speech in WSD. It is interesting to observe that in terms of words in context, bigrams seem to be more effective than simple keywords. Also, the best setting for the CF feature was found to be a one or two word window.

In terms of average number of features, the semantic disambiguation of nouns requires the smallest number of features (3.7), followed by adjectives (4.4) and verbs

https://doi.org/10.1017/S1351324902002991 Published online by Cambridge University Press

Table 4. *Feature distribution for nouns, verbs, adjectives*

	Part of speech			
	Noun	Verb	Adjective	Total
Words	29	29	15	73
Features				
CW	10	13	9	32
CP	22	25	14	61
CF	14	18	8	40
COL	13	12	6	31
HNP	6	4	5	15
SK	1	6	3	10
B	10	6	3	19
VB	7	4	3	14
VA	1	2	1	4
NB	8	3	2	13
NA	1	10	4	15
NEB	3	4	1	8
NEA	4	3	0	7
PB	4	4	2	10
PA	1	6	0	7
PRB	1	4	1	6
PRA	0	3	2	5
DT	3	3	3	9
Total	109	130	67	306

(4.5). These statistics are not yet conclusive, since they are computed for a small number of words, but they are indicative of the complexity of the task for various parts of speech. Further investigations and larger amounts of data will eventually confirm this preliminary conclusion.

Several interesting cases were encountered in the SENSEVAL-2 data, justifying our approach of using *automatic* feature selection. The influence of a feature greatly depends on the target word: a feature can increase the precision for a word, while making things worse for another word. For instance, a word such as *free* does not benefit from the SK feature, whereas *colourless* gains almost 7% in precision when this feature is used.

$$\begin{aligned} \textit{free.a}[\text{CW CP CF=1 SK=3,3}] &\rightarrow 57.85\% \\ \textit{free.a}[\text{CW CP CF=1}] &\rightarrow 63.57\% \\ \textit{colorless.a}[\text{CW CP CF=1}] &\rightarrow 78.57\% \\ \textit{colorless.a}[\text{CW CP CF=1 SK=3,3}] &\rightarrow 85.71\% \end{aligned}$$

Another interesting example is the noun *chair*, which was disambiguated with high precision by simply using the Current Word (CW) feature. This is explained by

https://doi.org/10.1017/S1351324902002991 Published online by Cambridge University Press

the fact that the most frequent senses are *Chair* meaning *person* and *chair* meaning *furniture*, and therefore the distinction between lower and upper case spellings makes the distinction among the different meanings of this word.

The noun *detention* has the same precision computed during the 10-fold cross validation runs, independent of the feature or combination of features used. This is because out of its two senses, one sense occurs in 97% of the examples, and hence it statistically dominates the other sense. There were several other interesting cases, including the adjective *local* which gained 20% in precision by simply using the feature NA, the word *faithful* which is best disambiguated with the CW feature, and others.

The system was also tested on the SENSEVAL-1 data (Kilgarriff and Palmer 2000), where the disambiguation task was performed with respect to Hector dictionary. The overall result achieved on this data was comparable to the one reported by the best performing system. Besides proving the validity of our approach, this fact also proves that our system is not tight in any ways to the sense inventory or data format. Going from SENSEVAL-1 to SENSEVAL-2 required only minimal changes in the system, mainly in the preprocessing phase (to accept as input the new data format) and in the postprocessing phase (to output the answer sense keys in the format required).

7 Conclusion

Pattern learning and automatic feature selection are new approaches in the WSD field. They have been implemented in a system that was evaluated on the SENSEVAL-2 data, with an excellent performance in both *English all words* and *English lexical sample* tasks.

Patterns represent a great way of capturing contexts representative for a word meaning. The usage of hypernyms as one of the pattern components gives us the means for generalization beyond words explicitly expressed in text.

In supervised learning algorithms, instance based learning with feature weighting provides a performance comparable with the best results achieved so far in word sense disambiguation. Its performance is greatly increased if coupled with an algorithm for automatic feature selection. This process is completely automated and it practically creates a classifier tailored to the behaviour of each ambiguous word.

Acknowledgments

The author would like to thank the anonymous reviewers for their feedback and useful comments, which were very helpful in preparing the final version of this paper.

References

Aha, D. W. and Bankert, R. L. (1994) Feature selection for case-based classification of cloud types: An empirical comparison. *Proceedings AAAI'94 Workshop on CaseBased Reasoning*, pp. 106–112. Seattle, WA.

https://doi.org/10.1017/S1351324902002991 Published online by Cambridge University Press

Almuallim, H. and Dietterich, T. G. (1991) Learning with many irrelevant features. *Proceedings 9th National Conference on Artificial Intelligence (AAAI-91)* **2**: 547–552. Anaheim, CA.

Argamon, S., Dagan, I. and Krymolowski, Y. (1998) A memory-based approach to learning shallow natural language patterns. *Proceedings 17th International Conference on Computational Linguistics (COLING-ACL-98)*, Montreal, Canada.

Brill, E. (1995) Transformation-based error driven learning and natural language processing: A case study in part-of-speech tagging. *Computational Linguistics* **21**(4): 543–566.

Brill, E. (2000) Pattern-based disambiguation for natural language processing. *Proceedings Conference on Empirical Methods in Natural Language Processing EMNLP*, Hong Kong.

Bruce, R. and Wiebe, J. (1999) Decomposable modeling in natural language processing. *Computational Linguistics* **25**(2): 195–207.

Cardie, C. (1996) Automating feature set selection for case-based learning of linguistic knowledge. *Proceedings Conference on Empirical Methods in Natural Language Processing EMNLP*, pp. 113–126. Somerset, NJ.

Daelemans, W., van den Bosch, A. and Zavrel, J. (1999) Forgetting exceptions is harmful in language learning. *Machine Learning* **34**(1–3): 11–34.

Daelemans, W., Zavrel, J., van der Sloot, K. and van den Bosch, A. (2001) Timbl: Tilburg memory based learner, version 4.0, reference guide. Technical report, University of Antwerp.

Edmonds, P. and Cotton, S. (2002) Senseval-2: Overview. *Proceedings Senseval-2 Workshop, Association of Computational Linguistics*, pp. 1–6. Toulouse, France.

Fellbaum, C., Palmer, M., Dang, H. T., Delfs, L. and Wolf, S. (2001) Manual and automatic semantic annotation with WordNet. *WordNet and Other lexical resources: NAACL 2001 Workshop*, pp. 3–10. Pittsburgh, PA.

Kilgarriff, A. and Palmer, M. (eds.) (2000) Special issue: SENSEVAL. Evaluating Word Sense Disambiguation programs. *Comput. and the Humanities* **34**.

Leacock, C., Chodorow, M. and Miller, G. A. (1998) Using corpus statistics and WordNet relations for sense identification. *Computational Linguistics* **24**(1): 147–165.

Mihalcea, R. and Moldovan, D. I. (1999) An automatic method for generating sense tagged corpora. *Proceedings AAAI-99*, pp. 461–466. Orlando, FL.

Mihalcea, R. and Moldovan, D. I. (2000) An iterative approach to word sense disambiguation. *Proceedings FLAIRS-2000*, pp. 219–223. Orlando, FL.

Mihalcea, R. (2002) Bootstrapping large sense tagged corpora. *Proceedings 3rd International Conference on Language Resources and Evaluation LREC 2002*, pp. 1407–1411. Canary Islands, Spain.

Miller, G., Leacock, C., Randee, T. and Bunker, R. (1993) A semantic concordance. *Proceedings 3rd DARPA Workshop on Human Language Technology*, pp. 303–308. Plainsboro, NJ.

Miller, G. (1995) Wordnet: A lexical database. *Comm. ACM* **38**(11): 39–41.

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

Ng, H. T. and Lee, H. B. (1996) Integrating multiple knowledge sources to disambiguate word sense: An examplar-based approach. *Proceedings 34th Annual Meeting of the ACL (ACL-96)*, Santa Cruz.

Pedersen, T. (2001) A decision tree of bigrams is an accurate predictor of word sense. *Proceedings North American Chapter of the Association for Computational Linguistics, NAACL 2001*, pp. 79–86. Pittsburg, PA.

Stevenson, M. and Wilks, Y. (2001) The interaction of knowledge sources in word sense disambiguation. *Computational Linguistics* **27**(3): 321–351.

Veenstra, J., van den Bosch, A., Buchholz, S., Daelemans, W. and Zavrel, J. (2000) Memory-based word sense disambiguation. *Comput. and the Humanities* **34**: 171–177.

https://doi.org/10.1017/S1351324902002991 Published online by Cambridge University Press

Yarowsky, D. and Florian, R. (2002) Evaluating sense disambiguation across diverse parameter spaces. *Natural Lang. Eng.* Special Issue on 'Evaluating Word Sense Disambiguation' **8**(4) (this issue).

Yarowsky, D. (1995) Unsupervised word sense disambiguation rivaling supervised methods. *Proceedings 33rd Annual Meeting of the ACLs (ACL-95)*, pp. 189–196. Cambridge, MA.

Yarowsky, D. (2000) Hierarchical decision lists for word sense disambiguation. *Comput. and the Humanities* **34**: 179–186.

https://doi.org/10.1017/S1351324902002991 Published online by Cambridge University Press