

An Agent Approach on Word Sense Disambiguation

Gabriela Șerban and Doina Tătar

Department of Computer Science
University "Babeș-Bolyai"
1, M. Kogalniceanu Street, Cluj-Napoca, Romania
tel: +40.64.405325, fax: +40.64.191.960
{gabis, dtatar@cs.ubbcluj.ro}

Topics: Natural Learning Processing, Machine Learning.

Abstract. The task of disambiguation is to determine which of the senses of an ambiguous word is invoked in a particular use of the word [3]. Starting from the algorithm of Yarowsky [5, 4, 9, 10] and the Naive Bayes Classifier (NBC) algorithm, in this paper we propose an original algorithm which combines their elements. This algorithm preserve the advantage of principles of Yarowsky (*one sense per discourse and one sense per collocation*) with the known high performance of a NBC algorithms. We design an Intelligent Agent, who learns (based on the algorithm mentioned above) to find the correct sense for an ambiguous word in some given contexts.

Keywords: Word sense disambiguation, corpus, agents, learning.

1 Introduction

The word sense disambiguation (WSD) is probably one of the most important open problem and it has now already a long "history" in computational linguistics [2, 1]. WSD problem has direct applications in some fields of text understanding as *information retrieval*, *text summarization*, *machine translation*.

The problem that arises in natural language is that many words (called polysemic), have several meanings or senses. These senses depend on what context they occur. The task of disambiguation is to determine which of the senses of an ambiguous word is invoked in a particular use of the word [3]. WSD is necessary whenever a system's actions depend on the meaning of the text being processed.

The algorithms used in WSD are classified considering whether they involve supervised or unsupervised learning. Unsupervised learning can be viewed as clustering task while supervised learning is usually seen as a classification task. Dictionary based disambiguation, which we will present in the following section can be considered as intermediary between supervised and unsupervised disambiguation [3, 6].

2 Some Known Algorithms for Word Sense Disambiguation

Notational conventions used in the following are:

- w — the word to be disambigued (*target word*);
- s_1, \dots, s_K — possible senses for w ;
- c_1, \dots, c_I — contexts of w in corpus;
- v_1, \dots, v_J — words used as contextual features for disambiguation of w .

Regarding to v_1, \dots, v_J there are two possibilities: they are co-locates or co-occurrences with w . In the first case the contextual features occur in a fixed position near w , in a *window* of fixed length, centered on w . In the second case the contextual features occur together with w , in arbitrarily positions. We will consider the first sense of contextual features.

In [5] (1995), Yarowsky observed that there are constraints between different occurrences of contextual features that can be used for disambiguation. Two such constraints are:

- *One sense per discourse*: the sense of a target word is highly consistent within a given discourse (document) ;
- *One sense per collocation*: the contextual features (nearby words) provide strong clues to the sense of a target word.

In supervised disambiguation a tagged corpus or a semantic annotated corpus is available. Such annotated corpus is used in the on-line product Senseval. The task in this case is to build a classifier which classifies correctly a new context

based on the contextual features occurring in this context. The classifier does no feature selection, but it combines the participation of all contextual features.

A Naive Bayes Classifier realizes the calculus of the sense s' , which for the target word w and a given context c satisfies the relation [3]:

$$\begin{aligned} s' &= \operatorname{argmax}_{s_k} P(s_k | c) = \operatorname{argmax}_{s_k} \frac{P(c | s_k)}{P(c)} P(s_k) \\ &= \operatorname{argmax}_{s_k} P(c | s_k) P(s_k). \end{aligned} \quad (1)$$

The same value for s' is obtained if we consider the logarithm of expression:

$$s' = \operatorname{argmax}_{s_k} (\log P(c | s_k) + \log P(s_k)) \quad (2)$$

The Naive Bayes assumption is that the contextual features are all conditional independent:

$$P(c | s_k) = P(\{v_j | v_j \in c\} | s_k) = \prod_{v_j \in c} P(v_j | s_k). \quad (3)$$

Here v_j represents any word in the context c .

This assumption has two consequences:

- the structure and order of words in context is ignored;
- the presence of one word in the context doesn't depend on the presence of another.

This is not generally true, but there is a large number of cases in which the algorithm works well.

Concerning the probabilities $P(v_j | s_k)$ and $P(s_k)$, these are calculated from the labeled (annotated) corpus:

$$P(v_j | s_k) = \frac{C(v_j, s_k)}{C(s_k)} \quad P(s_k) = \frac{C(s_k)}{C(w)} \quad (4)$$

where $C(v_j, s_k)$ is the number of occurrences of v_j in the contexts annotated with the sense s_k , $C(s_k)$ is the number of contexts with the sense s_k and $C(w)$ is the total number of occurrences of the word w .

The NBC algorithm is:

Training:

for all senses s_k of w **do**

for all words v_j in corpus **do**

$$P(v_j | s_k) = \frac{C(v_j, s_k)}{C(s_k)}$$

for all senses s_k of w **do**

$$P(s_k) = \frac{C(s_k)}{C(w)}$$

Disambiguation:

for all senses s_k of w **do**

$$score(s_k) = \log P(s_k) + \sum_{v_j \in c} \log P(v_j | s_k)$$

Calculate $s' = \operatorname{argmax}_{s_k} score(s_k)$

In [3] is reported that a disambiguation system based on this algorithm is correct for about 90 percents of cases.

3 Intelligent Agents

The field of *intelligent agents* is in connection with another field of Artificial Intelligence (AI), the field of *machine learning*. *Machine learning* represents the study of system models that, based on a set of data (training data), improve their performance by experiences and by learning some specific experimental knowledge. The attempt of modeling the human reasoning leads to the concept of *intelligent reasoning*. The *reasoning* is the process of conclusion deduction; the *intelligent reasoning* is a kind of reasoning accomplished by humans. Most of the AI systems are deductive ones, able for making inferences (draw conclusions), given their initial or supplied knowledge, without being able for new knowledge acquisition or to generate new knowledge. The learning capability being connected to the intelligent behavior, one of the most important research directions in AI is to implement in the machines the learning capability.

An *agent* [7] is anything that can be viewed as *perceiving* its environment through *sensors* and *acting* upon that environment through *actions*. An *intelligent agent* is an agent with an initial knowledge, having the capability for learning.

4 A Bootstrapping Algorithm Based on the Principles: One Sense Per Discourse and One Sense Per Collocation

The algorithm begins by identifying a small number of training contexts. This could be accomplished by hand tagging with senses the contexts of w for which the sense of w is clear because some *seed collocations* [5, 9, 10] occur in these contexts.

This tagging is made on the base of dictionaries or by using the known on-line dictionary of senses WordNet [13]. This initial set of annotated contexts is used for learning a *naïve bayesian classifier*. This NBC will help in annotating new contexts. By repeating the process, the annotated part of corpus grows. We will stop when the remaining unannotated corpus is empty or any new context can't be annotated.

The notational conventions are as above:

- w is the polysemic word

- $S = \{s_1, s_2, \dots, s_K\}$ are possible senses for w , as in a dictionary, or as obtained with WordNet.
- $C = \{c_1, c_2, \dots, c_I\}$ are contexts (windows) for w , as obtained for w with an on-line corpus tool (for example Cobuild [12]). Each c_i is of the form:

$$c_i = w_1, w_2, \dots, w_t, w, w_{t+1}, \dots, w_z$$

where $w_1, w_2, \dots, w_t, w_{t+1}, \dots, w_z$ are words from the set v_1, \dots, v_J and t and z (usually $z = 2t$) are selected by user.

Let us consider that the words $V = \{v^1, \dots, v^l\} \subset \{v_1, \dots, v_J\}$, where l is small (for example 2) are *surely* associated with the senses for w , such that the occurrence of v^i in the context of w determines the choice of a sense for w (one sense per collocation).

For example, for the word *plant*, the occurrence in the same context of the word *life* means a sense (let say A), while the occurrence in the same context of the word *manufacturing* means another sense (let say B). These rules can be done generally as a decision list:

$$if\ v^i\ occurs\ in\ a\ context\ of\ w\ (of\ z\ words) \Rightarrow s^i, i = 1, \dots, l \quad (5)$$

So, from the set of contexts obtained as query results with Cobuild, some contexts can be solved. Namely, we marked these contexts with A or B:

- (A)industrial equipment and engineering plant.[p] The company insures
- (A)hard currency. And so we've found a plant, and I have some seeds here from
- (B)the planning and construction of the plant at Rabta near Tripoli and were
- (A)aspect, features and animal and plant life."[p] [p] These were never
- (B)all the allegations. It says the plant produces merely pharmaceuticals.
- (B)d be looking at 75 to 100 jobs and a plant that would produce probably

We start by defining a relation $\delta : WXC$, where W is the set of words and C is the set of contexts (set of array of words). If $w \in W$ is a word and $c \in C$ is a context, we say that $(w, c) \in \delta$ if exists a word $w_1 \in c$ so that the words w and w_1 have the same gramatical root.

In our algorithm, a decision list has the following form:

$$if\ (v^i, c) \in \delta \Rightarrow s^i, i = 1, \dots, l \quad (6)$$

Algorithm

$C_{res} = \Phi$, determine the set $V = \{v^1, \dots, v^l\}$

For each context c in C apply the rules:
if $(v^i, c) \in \delta, \Rightarrow \text{sense } s^i, i = 1, \dots, l, C_{res} = C_{res} \cup \{c\}$
 $C_{rest} = C \setminus C_{res}$
While $C_{rest} \neq \Phi$ **do** :
 Determine a set V^* of words with a maximum frequency in C_{res}
 Define $V = V \cup V^* = \bigcup_{j=1}^l V_{s_j}$,
 where V_{s_j} is the set of words associated with the sense s_j
 (**If** $v \in V^*$, the context c solved with the sense s_j , and $(v, c) \in \delta$,
 then $v \in V_{s_j}$, according with the principle "one sense per discours")
 For each $c_i \in C_{rest}$ apply the BNC algorithm :

$$s_i^* = \operatorname{argmax}_s P(s | c_i) = \operatorname{argmax}_s \frac{P(c_i | s) \times P(s)}{P(c_i)} \quad (7)$$

$$= \operatorname{argmax}_s P(c_i | s) \times P(s)$$

$$\text{where } P(c_i | s) = P(w_1 | s) \cdots P(w_t | s) P(w_{t+1} | s) \cdots P(w_z | s)$$

$$\text{and } P(w_i | s_j) = \begin{cases} 1 & \text{if } (w_i, V_{s_j}) \in \delta \\ \frac{nr.occ.w_i}{nr. total of words} & \text{else} \end{cases}$$

$$C_{res}^* = \{c_i | P(s_i^* | c_i) > N, N \text{ fixed}\}$$

$$C_{res} = C_{res}^* \cup C_{res}$$

$$C_{rest} = C_{rest} \setminus C_{res}$$

5 The Agent for Words' Disambiguation

5.1 General Presentation

The application is written in Visual C++ 6.0 (Figure 1) and implements the behavior of an Intelligent Agent, whose purpose is to find the correct sense for a given word (the target word) in some given contexts (the word sense disambiguation), using the algorithm described in the previous section. In fact it's a kind of semi-supervised learning; the agent starts with an initial knowledge (the senses of the target word and a set of words using as contextual features for the disambiguation) and learns to disambiguate the word in the given contexts.

The environment of this agent consists in some information which the agent reads from an input text file "in.txt":

- the target word(w);
- the possible senses for w ;
- the contexts for w ;
- the words used as contextual features for w 's sense disambiguation.

On the basis of his environment, using the algorithm described in the previous section, the agent learns to find the correct sense of the target word in the given contexts.

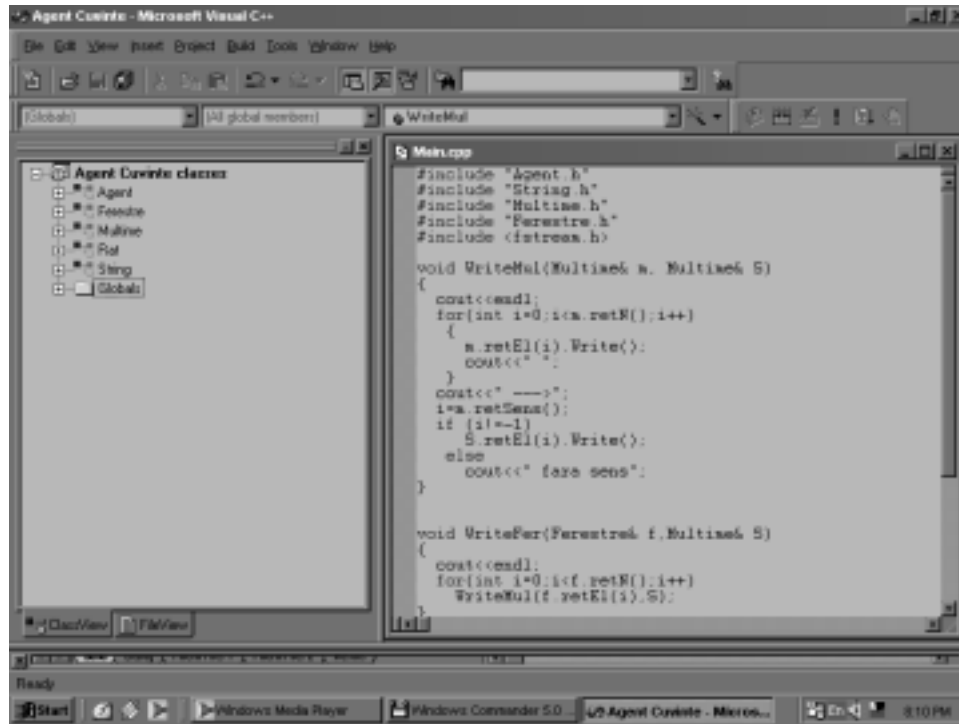


Fig. 1. The Agent

5.2 The Agent's Design

The basis classes used for implementing the agent's behavior are the following:

- **String**: defines the type *String* (array of characters), having methods for:
 - adding a char in a *String*;
 - accessing the length and the characters of a *String*;
 - displaying, comparing, concatenating *Strings*.
- **Set**: defines the type *Array* of strings (corresponding to a context which contains the target word *w*), associated with a sense of *w*. The main methods of this class are for:
 - adding a *String* in an *Array*;
 - accessing the number of elements and the strings of an *Array*;
 - testing the membership of a string in the *Array*;
 - setting the corresponding sense for *w*;
 - finding the reunion of two *Arrays*.

- **Contexts:** defines the type *Set* of arrays of strings (array of contexts), representing the contexts for which we want to associate a sense corresponding to *w*. The main methods of this class are for:
 - adding an element in the *Set*;
 - accessing the number of elements and the elements of a *Set*;
 - testing the membership of an array in the *Set*;
 - finding the difference of two *Sets*.
- **Agent:** the main class of the application, which implements the agent behavior and the learning algorithm (Figure 2).

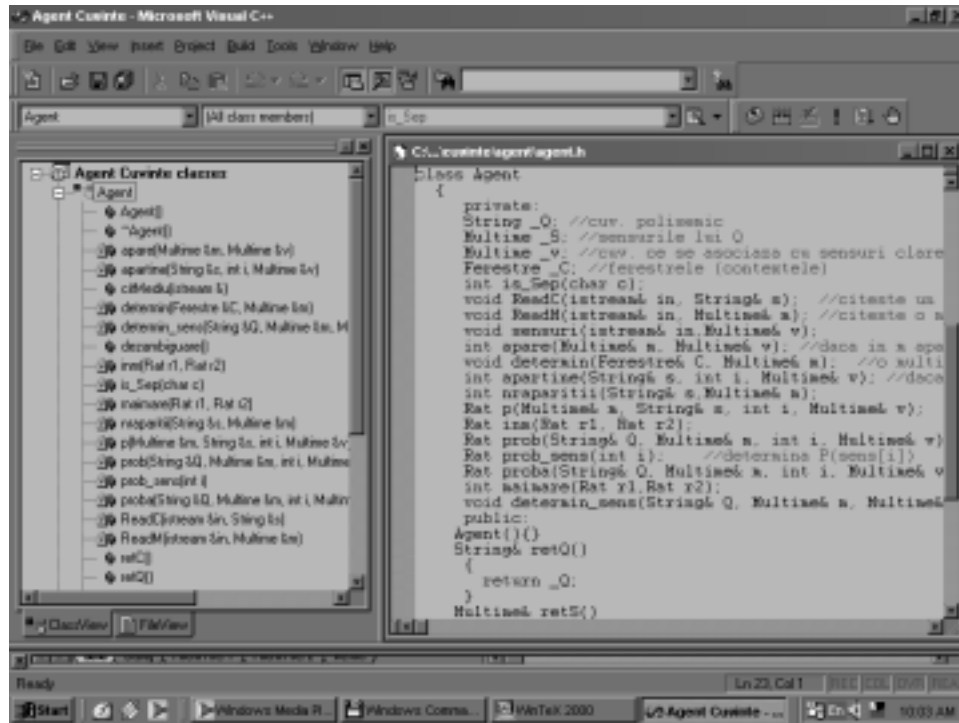


Fig. 2. The main class of the Agent

The private member data of this class are:

- **Q**: the target word;
- **S**: the set of senses for the target word;
- **v**: the set of words used as contextual features for *Q*'s disambiguation;

- **C**: the contexts for the target word.

The public methods of the agent are the followings:

- **readEnvironment**: reads the information about the environment from an input stream ;
- **disambiguation**: the main (learning) algorithm of the agent used to find the correct senses of the target word in the environment's contexts;
- **retQ**: returns the target word (Q);
- **retS**: returns the set of senses of the target word (S);
- **retV**: returns the member data v ;
- **retC**: returns the contexts for the target word (C);

Besides the public methods, the agent has some private methods used in the method **disambiguation**.

We notice that all the representations of data structures are linked, which means that there are no limitations for the structures' length (number of contexts, number of words in a context).

5.3 Experiment

Our aim is to solve some contexts in which appear the word *band*, from the set of contexts obtained as query results with Cobuild [12]. Using the application we accomplish the training of the agent in the following environment (given in the text file "band.txt").

The file "band.txt":

- the target word

band

- the senses of the target word

set music ring strip

- the words used as contextual features for Q 's disambiguation and the indexes of the corresponding sense of the target word

**song 2 sing 2 paper 4 dance 2 club 2 release 2 jazz 2 member 1
jewel 3 sound 2 rock 2**

- the contexts of the target word

1. going to happen, we're not that kinda *band*. [p] I don't write 15 songs in a
2. fiber, more dust, a broken rubber *band*, a paper clip, a penny, more dust, a
3. Hickman conducts an all-woman's *band* and choir, the next she sings
4. olde worlde part of town where the *band* are staying. All hideous new Europe

5. studio-dusty shrouds. Finally, the *band* that had me dancing 'til
6. to reunite musicians of a famous soul *band* who have not played for 30 years.
7. fan club show. It's a rowdy night-the *band* first played here in '87 with the
8. and 'You Love Us'-are the best the *band* have released so far, claim Jeff and
9. the more esoteric brands of big-*band* jazz in favour of a lively
10. The Commitments I've never been in a *band*, know nothing about it. [p] Adams:
11. Maker, the signatures of each *band* member, lovingly inscribed in non-
12. Cert 15 [p] 15 (18) RESERVOIR DOGS: A *band* of foiled jewel robbers reassemble
13. it right up here. (Fade down) FX *BAND* SIX: MUSIC. 'STYLE Fade up. Fade
14. to none - a recent reviewer said: The *band* sounds learner than before, the
15. a new idea they cannot, like a young *band*, simply book the games equivalent of
16. Radiation put together a rockabilly *band*, The Tearjerkers, while Panter
17. Before Us is by The Albion Dance *Band*-with the emphasis on 'dance'. Live,
18. Tonight the *band* plays at the Hotel,
19. present a famous soul *band*, with more than 10 albums,

After the agent reads the information from the environment, he applies the disambiguation algorithm for the given contexts. The result is shown below (each context is followed by the sense for the target word - found by the agent after the disambiguation).

Context 1 — *music*
 Context 2 — *strip*
 Context 3 — *music*
 Context 4 — *set*
 Context 5 — *music*
 Context 6 — *music*
 Context 7 — *music*
 Context 8 — *music*
 Context 9 — *music*
 Context 10 — *set*
 Context 11 — *set*

Context 12 — *ring*
Context 13 — *music*
Context 14 — *music*
Context 15 — *set*
Context 16 — *music*
Context 17 — *music*
Context 18 — *music*
Context 19 — *music*

We observe that the Agent learns to find the correct sense of the word *band* in the contexts 4, 5, 10, 12, 13, 15, 16, 18, 19. The sense of the word in the other contexts is deduced from the set of words used as contextual features for the disambiguation. In our experiment, the learning rate is 100%. For example, from the context 7, the agent learns to associate the word *play* with the sense *music*, and from context 6 the agent learns to associate the word *soul* with the sense *music*.

6 Conclusions and Further Work

If the Agent (described above) starts with a substantial initial knowledge (number of senses of the target word, set of words used as contextual attributes for the disambiguation) and if the environment consists in a big number of contexts, the the disambiguation (learning) algorithm works very well (the number of senses of the target word learned by the agent grows).

Further work is planned to be done in the following directions:

- We plan to establish a better evaluation for our Agent, working with some standard ambiguous words and a more impressive amount of contexts from different corpora (as BNC <http://sara.natcorp.ox.ac.uk/lookup.html>);
- We will compare the results with those obtained with SENSEVAL's , two recent pilot applications in WSD;
- As input of our agent we plan to use SEMCOR [11], a manually sense tagged corpus, in which all words have been tagged with WordNet senses;
- At University of Bucharest is in construction a WordNet for Romanian language, and we will use that as input for our Agent;
- The trained output can be used in a multilingual sense task.

References

1. J. Allen : " Natural language understanding", Benjamin/Cummings Publ. , 2nd ed., 1995
2. D. Jurafsky, J. Martin: " Speech and language processing", Prentice Hall, 2000
3. C. Manning, H. Schutze: " Foundation of statistical natural language processing", MIT, 1999

4. P. Resnik, D. Yarowsky : " Distinguishing Systems and Distinguishing sense: new evaluation methods for WSD ", Natural Language Engineering, 1 , nr 1, 1998
5. D. Yarowsky: "Hierarchical Decision Lists for WSD", Kluwer Academic Publishers, 1999
6. F. Sebastiani: " A tutorial on Automated Text Categorization", pp 1-25, ESSLLI 2001
7. Russell, S.J., Norvig, P.: Artificial intelligence. A modern approach. Prentice-Hall International, 1995
8. D. Tatar: "Inteligența artificială: demonstrare automată de teoreme, prelucrarea limbajului natural", Editura Microinformatica, 2001
9. David Yarowsky: "Unsupervised Word Sense Disambiguation Rivaling Supervised Methods", Proceedings of ACL'95, pp 189-196
10. David Yarowsky: "WordSense Disambiguation Using Statistical Models of Rognet's Categories Trained on Large Corpora ", [http:// citeseer.nj.nec.com](http://citeseer.nj.nec.com)
11. Adam Kilgarriff: "Gold Standard Datasets for Evaluating Word Sense Disambiguation Programs", [http:// citeseer.nj.nec.com](http://citeseer.nj.nec.com)
12. <http://titania.cobuild.collins.co.uk/form.html>
13. [http://www.cogsci.princeton.edu/~ wn/](http://www.cogsci.princeton.edu/~wn/)