Fundamenta Informaticae 117 (2012) 1–26
DOI 10.3233/FI-2012-690
IOS Press

Feeding Syntactic Versus Semantic Knowledge to a Knowledge-lean Unsupervised Word Sense Disambiguation Algorithm with an Underlying Naïve Bayes Model

Florentina Hristea[*]
Department of Computer Science
Faculty of Mathematics and Computer Science, University of Bucharest, Romania
fhristea@fmi.unibuc.ro

Mihaela Colhon
Department of Computer Science
Faculty of Exact Sciences, University of Craiova, Romania
mcolhon@inf.ucv.ro

Abstract. The present paper concentrates on the issue of feature selection for unsupervised word sense disambiguation (WSD) performed with an underlying Naïve Bayes model. It introduces dependency-based feature selection which, to our knowledge, is used for the first time in conjunction with the Naïve Bayes model acting as clustering technique. Construction of the dependency-based semantic space required for the proposed task is discussed. The resulting disambiguation method, representing an extension of the method introduced in [15], lies at the border between unsupervised and knowledge-based techniques. Syntactic knowledge provided by dependency relations (and exemplified in the case of adjectives) is hereby compared to semantic knowledge offered by the semantic network WordNet (and examined in [15]). Our conclusion is that the Naïve Bayes model reacts well in the presence of syntactic knowledge of this type and that dependency-based feature selection is a reliable alternative to the WordNet-based semantic one.

Keywords: Unsupervised word sense disambiguation, Bayesian classification, The EM algorithm, WordNet, Dependency relations, Dependency-based feature selection

[*] Address for correspondence: Department of Computer Science, Faculty of Mathematics and Computer Science, University of Bucharest, 14, Academiei Str., Bucharest, Sector 1, C.P. 010014, Romania

1. Introduction

Word sense disambiguation (WSD) is a core research problem in computational linguistics and natural language processing (NLP), which was recognized since the beginning of the scientific interest in machine translation, and in artificial intelligence, in general. Finding a solution to the WSD problem is in many cases essential, or even compulsory, either for natural language understanding, or for a wide range of applications such as: information retrieval, machine translation, speech processing, text processing etc.

In the subfield of natural language processing (from the perspective of which we shall approach WSD within the present study), the problem we are discussing here is defined as that of computationally determining which sense of a word is activated by the use of that word in a particular context and represents, essentially, a classification problem.

In spite of the great number of existing disambiguation algorithms, the problem of WSD remains an open one, with three main classes of WSD methods being taken into consideration by the literature: supervised disambiguation, unsupervised disambiguation and knowledge-based disambiguation.

The present paper refers to unsupervised corpus-based methods for WSD. It concentrates on distributional approaches to unsupervised WSD that rely on monolingual corpora, with focus on the usage of the Naïve Bayes model as clustering technique.

Within the framework of the present study, the term “unsupervised” will refer, as in [37], to knowledge-lean methods, that do not rely on external knowledge-sources such as machine-readable dictionaries, concept hierarchies or sense-tagged text. Due to the lack of knowledge they are confronted with, these methods do not assign meanings to words, relative to a pre-existing sense inventory, but make a distinction in meaning based on distributional similarity. While not performing a straightforward WSD, these methods achieve a discrimination among the meanings of a polysemous word. As commented in [1], they have the potential to overcome the knowledge acquisition bottleneck (manual sense-tagging).

The problem we are investigating here could be formulated in the following terms: we are given I sentences that each contain a particular polysemous word; our goal is to divide these I instances of the ambiguous word (the so-called target word) into a specified number of sense groups. These sense groups must be mapped to sense tags in order to evaluate system performance. Let us note that sense tags, as in previous studies [39, 16, 14, 15], will be used only in the evaluation of the sense groups found by the unsupervised learning procedure. The discussed algorithm is automatic and unsupervised in both training and application.

From the wide range of unsupervised learning techniques that could be applied to our problem, we have chosen to use a parametric model in order to assign a sense group to each ambiguous occurrence of the target word. As already mentioned, in each case, we shall assign the most probable group given the context as defined by the Naïve Bayes model, where the parameter estimates are formulated via unsupervised techniques. The theoretical model will be presented and its implementation will be discussed. Special attention will be paid to feature selection, the main issue of the model’s implementation. A novel method of performing knowledge-based feature selection will be presented and discussed.

The idea of the Bayes classifier in the context of word sense disambiguation is that it looks at the words around an ambiguous word within the so-called context window. Each content word contributes potentially useful information about which sense of the ambiguous word is likely to be used with it. The classifier does no feature selection. Instead it combines the evidence coming from all features [24], namely from all content words occurring in the context window of the target. A fact that generates a so-called *bag*[1] *of words* model, based on the Naïve Bayes assumption that all these content words are conditionally independent. The assumption is clearly not true in the case of natural language, but has provided very good disambiguation results, especially in the case of supervised WSD. As commented in [24], there is a surprisingly large number of cases in which the Naïve Bayes assumption does well, "partly because the decisions made can still be optimal even if the probability estimates are inaccurate due to feature dependence"[2]. This is not surprising when viewing the Bayesian model from a cognitive perspective[3], which is an adequate one in the case of a problem concerning natural language processing. And when taking into consideration that, as noted in [8], "without an account of the rationality of the observed input-output relation, the computational level models provide a summary of the observed data, but no rational explanation for the behavior".

When the Naïve Bayes model is applied to supervised disambiguation, the actual words occurring in the context window are usually used as features. This type of framework generates a great number of features and, implicitly, a great number of parameters. This can dramatically decrease the model's performance since the available data is usually insufficient for the estimation of the great number of resulting parameters. A situation that becomes even more drastic in the case of unsupervised disambiguation, where parameters must be estimated in the presence of missing data (the sense labels). In order to overcome this problem, the various existing unsupervised approaches to WSD implicitly or explicitly perform a feature selection. In fact, one can say that discussions concerning the implementation of the Naïve Bayes model for supervised/unsupervised WSD focus almost entirely on the issue of feature selection.

Two early approaches to unsupervised WSD, context group discrimination [41] and McQuitty's Similarity Analysis [38, 39], rely on totally different sets of features and probably still represent the main approaches to feature selection.

As commented in [37], Schütze [41] represents contexts in a high dimensional feature space that is created using a separate large corpus (referred to as the training corpus). While Schütze [41] reduces dimensions by means of LSI/LSA, Pedersen and Bruce [38] define features over a small contextual window (local context) and select them to produce low dimensional event spaces. They make use of a small number of first-order features to create matrices that show the pairwise (dis)similarity between contexts. They rely on local features that include co-occurrence and part of speech information near the target word. Three different feature sets, consisting of various combinations of features of the mentioned types, were defined in [39] for each word and were used to formulate a Naïve Bayes model describing the distribution of sense groups of that word. Unlike Schütze [41], Pedersen and Bruce [39] select features from the same test data that is being discriminated, which, as noted in [37], is a common practice in clustering in general.

More recently, Hristea et al. [16] try to improve the disambiguation results previously obtained when performing unsupervised WSD with an underlying Naïve Bayes model, by using the freely available semantic network WordNet (WN) as knowledge source for feature selection. Their method, initially tested for adjectives only [15], is extended to all main parts of speech and surveyed in [16]. The method makes ample use of the WordNet semantic relations that are typical of each part of speech, which places the disambiguation process at the border between unsupervised and knowledge-based techniques. The semantic network WordNet has been used as unique knowledge source for feature selection. Although

[1] A bag is similar to a set, only it allows repetition.

[2] See also [7].

[3] For a cognitive point of view on the Naïve Bayes model see the comprehensive study [8].

not totally knowledge-lean, the full presentation of the method [16] has once again reinforced the benefits of combining the unsupervised approach to the WSD problem with a knowledge source of type WordNet. Especially since we must keep in mind that knowledge-lean methods as the one proposed in [39] can also require information that is not always available. Such knowledge-lean methods can equally have difficulties when asking for information like part of speech, for instance, especially if a part-of-speech tagger does not exist for the language under investigation.

The disambiguation results obtained in [15, 16] were compared to those of Pedersen and Bruce [39] since both disambiguation methods use an algorithm of the same type i.e. unsupervised and based on an underlying Naïve Bayes model. However, the two compared algorithms perform feature selection in a completely different manner, as already specified. While Pedersen and Bruce [39] use a restricted set of local features that include co-occurrence and part of speech information near the target word, Hristea et al. [15, 16] make use of WordNet for feature selection. The latter approach implements a Naïve Bayes model that uses as features *the actual words* occurring in the context window of the target and decreases the existing number of features by selecting a restricted number of such words, as indicated by WordNet. The size of the feature set is therefore reduced by performing knowledge-based feature selection. The Naïve Bayes model is fed semantic knowledge. In the case of all parts of speech test results have shown [14, 15, 16] that feature selection using a knowledge source of type WordNet is more effective in sense disambiguation than local type features are.

The present paper focuses on an entirely different way of performing feature selection, that is equally knowledge-based. The disambiguation process will once again take place at the border between unsupervised and knowledge-based techniques as a result of feeding knowledge of a totally different nature to the same underlying Naïve Bayes model. The paper investigates the usage of syntactic features provided by dependency relations, as defined by the classical Dependency Grammar formalism[4]. Although dependency-based semantic space models have been studied and discussed by several authors (e.g. [35, 36]), to our knowledge, grammatical dependencies have not yet been used in unsupervised WSD with an underlying Naïve Bayes model.

The semantic space proposed in this paper is based on syntactic knowledge, more precisely on dependency relations, extracted from natural language texts via a syntactic parser. The syntactic dependency relations extracted by the parser will indicate those words (features) which should be part of the disambiguation vocabulary when trying to decrease the number of parameters for unsupervised WSD.

Dependency relations are considered a linguistically rich representation where fixed word order is not required, argument structure differences can be captured, different types of contexts can be selected and words do not have to co-occur within a small, fixed context window [36]. Such properties have recommended dependency relations as appropriate for feeding syntactic knowledge to the Naïve Bayes model.

The present paper introduces dependency-based feature selection in the case of adjectives and compares test results with those obtained when using the disambiguation vocabulary previously generated [15] by WordNet. We ultimately compare totally different ways of feeding knowledge of various types to a knowledge-lean algorithm for unsupervised WSD based on an underlying Naïve Bayes model. The discussed method will once again prove that a basic, simple knowledge-lean disambiguation algorithm, hereby represented by the Naïve Bayes model, can perform quite well when provided knowledge in an appropriate way, a remark also made by [40].

[4] Dependency grammar (DG) is a class of syntactic theories developed by Lucien Tesnière [44]. Within this theory, syntactic structure is determined by the grammatical relations existing between a word (a *head*) and its *dependents*.

Specifically, the Naïve Bayes model needs to be fed knowledge in order to perform well as clustering technique for unsupervised WSD. This knowledge can be fed in various ways and can be of various natures. The present paper examines syntactic knowledge provided by dependency relations (and exemplified in the case of adjectives) as compared to semantic knowledge offered by the semantic network WordNet. We hereby hope to initiate an open discussion concerning the *type of knowledge* that is best suited for the Naïve Bayes model when performing the task of unsupervised WSD.

2. The Naïve Bayes model in the context of word sense disambiguation

The algorithm for word sense disambiguation under study here exemplifies an important theoretical approach in statistical language processing: Bayesian classification [11]. As already mentioned, the idea of the Bayes classifier (in the context of WSD) is that it looks at the words around an ambiguous word within a context window. Each content word contributes potentially useful information about which sense of the ambiguous word is likely to be used with it. The classifier does no feature selection but, instead, it combines evidence coming from all features. The mentioned classifier [11] is an instance of a particular kind of Bayes classifier, the Naïve Bayes classifier.

2.1. The probability model of the corpus and the Bayes classifier

In order to formalize the described model, we shall present the probability structure of the corpus $\mathcal{C}$. The following *notations* will be used: w is the word to be disambiguated (target word); $s_1, ..., s_K$ are possible senses for w; $c_1, ..., c_I$ are contexts of w in a corpus $\mathcal{C}$; $v_1, ..., v_J$ are words used as contextual features for the disambiguation of w.

Let us note that the contextual features could be some attributes (morphological, syntactical, etc.), or they could be actual ”neighboring” content words of the target word. The contextual features occur in a fixed position near w, in a window of fixed length, centered or not on w. In what follows, a window of size n will denote taking into consideration n content words to the left and n content words to the right of the target word, whenever possible. The total number of words taken into consideration for disambiguation will therefore be $2n + 1$. When not enough features are available, the entire sentence in which the target word occurs will represent the window of context.

The probability structure of the corpus is based on one main assumption: *the contexts* $\{c_i, i\}$ *in the corpus* $\mathcal{C}$ *are independent*. Hence, the likelihood of $\mathcal{C}$ is given by the product

$$P(\mathcal{C}) = \prod_{i=1}^{I} P(c_i)$$

Let us note that this is a quite natural assumption, as the contexts are not connected, they occur at significant lags in $\mathcal{C}$.

On considering the possible senses of each context, one gets

$$P(\mathcal{C}) = \prod_{i=1}^{I} \sum_{k=1}^{K} P(s_k) \cdot P(c_i \mid s_k)$$

A model with independent features (usually known as the Naïve Bayes model) assumes that the contextual features are conditionally independent. That is,

$$P\left(c_i \mid s_k\right) = \prod_{v_j \ in \ c_i} P\left(v_j \mid s_k\right) = \prod_{j=1}^{J} \left(P\left(v_j \mid s_k\right)\right)^{|v_j \ in \ c_i|},$$

where by $|v_j \ in \ c_i|$ we denote the number of occurrences of feature v_j in context c_i. Then, the likelihood of the corpus $\mathcal{C}$ is

$$P\left(\mathcal{C}\right) = \prod_{i=1}^{I} \sum_{k=1}^{K} P\left(s_k\right) \prod_{j=1}^{J} \left(P\left(v_j \mid s_k\right)\right)^{|v_j \ in \ c_i|}$$

The parameters of the probability model with independent features are

$$\left\{P\left(s_k\right), k = 1, ..., K \ \ and \ \ P\left(v_j \mid s_k\right), j = 1, ..., J, k = 1, ..., K\right\}$$

Notation:

- $P\left(s_k\right) = \alpha_k$, $k = 1, ..., K$, $\alpha_k \geq 0$ for all k, $\sum_{k=1}^{K} \alpha_k = 1$
- $P\left(v_j \mid s_k\right) = \theta_{kj}$, $k = 1, ..., K$, $j = 1, ..., J$, $\theta_{kj} \geq 0$ for all k and j, $\sum_{j=1}^{J} \theta_{kj} = 1$ for all $k = 1, ..., K$

With this notation, the likelihood of the corpus $\mathcal{C}$ can be written as

$$P\left(\mathcal{C}\right) = \prod_{i=1}^{I} \sum_{k=1}^{K} \alpha_k \prod_{j=1}^{J} \left(\theta_{kj}\right)^{|v_j \ in \ c_i|}$$

The well known Bayes classifier involves the a posteriori probabilities of the senses, calculated by the Bayes formula for a specified context c,

$$P\left(s_k \mid c\right) = \frac{P\left(s_k\right) \cdot P\left(c \mid s_k\right)}{\sum\limits_{k=1}^{K} P\left(s_k\right) \cdot P\left(c \mid s_k\right)} = \frac{P\left(s_k\right) \cdot P\left(c \mid s_k\right)}{P\left(c\right)},$$

with the denominator independent of senses.

The Bayes classifier chooses the sense s' for which the a posteriori probability is maximal (sometimes called the Maximum A Posteriori classifier)

$$s' = \underset{k=1,...,K}{\arg\max} P\left(s_k \mid c\right)$$

Taking into account the previous Bayes formula, one can define the Bayes classifier by the equivalent formula

$$s' = \underset{k=1,...,K}{\arg\max}\left(\log P\left(s_k\right) + \log P\left(c \mid s_k\right)\right)$$

Of course, when implementing a Bayes classifier, one has to estimate the parameters first.

2.2. Parameter estimation

Parameter estimation is performed by the Maximum Likelihood Method, for the available corpus $\mathcal{C}$. That is, one has to solve the optimization problem

$$\max\left(\log P\left(\mathcal{C}\right) \mid \left\{P\left(s_k\right), k=1,...,K \ \ and \ \ P\left(v_j \mid s_k\right), j=1,...,J, k=1,...,K\right\}\right)$$

For the Naïve Bayes model, the problem can be written as

$$\max\left(\sum_{i=1}^{I}\log\left(\sum_{k=1}^{K}\alpha_k\prod_{j=1}^{J}\left(\theta_{kj}\right)^{|v_j\ in\ c_i|}\right)\right) \tag{1}$$

with the constraints

$$\sum_{k=1}^{K}\alpha_k = 1$$
$$\sum_{j=1}^{J}\theta_{kj} = 1 \ \ for \ \ all \ \ k=1,...,K$$

For unsupervised disambiguation, where no annotated training corpus is available, the maximum likelihood estimates of the parameters are constructed by means of the Expectation - Maximization (EM) algorithm.

The optimization problem (1) can be solved only by iterative methods. The Expectation - Maximization algorithm (Dempster, Laird and Rubin [6]) is a very successful iterative method, very well fitted for models with missing data.

Each iteration of the algorithm involves two steps:

- estimation of the missing data by the conditional expectation method (E - step)
- estimation of the parameters by maximization of the likelihood function for complete data (M - step)

The E - step calculates the conditional expectations given the current parameter values, and the M - step produces new, more precise parameter values. The two steps alternate until the parameter estimates in iteration $r+1$ and r differ by less than a threshold ε.

The EM algorithm is guaranteed to increase the likelihood $\log P\left(\mathcal{C}\right)$ in each step. Therefore, two stopping criteria for the algorithm could be considered: (1) Stop when the likelihood $\log P\left(\mathcal{C}\right)$ is no longer increasing significantly; (2) Stop when parameter estimates in two consecutive iterations no longer differ significantly.

Further on, we present the EM algorithm for solving the optimization problem (1).

The available data, called *incomplete data*, are given by the corpus $\mathcal{C}$. The *missing data* are the senses of the ambiguous words, hence they must be modeled by some random variables

$$h_{ik} = \begin{cases} 1, & context\ \ c_i\ \ generates\ \ sense\ \ s_k \\ 0, & otherwise \end{cases} , i=1,...,I; k=1,...,K$$

The *complete data* consist of incomplete and missing data, and the corresponding likelihood of the corpus $\mathcal{C}$ becomes

$$P_{complete}\left(\mathcal{C}\right)=\prod_{i=1}^{I}\prod_{k=1}^{K}\left(\alpha_k\prod_{j=1}^{J}\left(\theta_{kj}\right)^{|v_j\ in\ c_i|}\right)^{h_{ik}}$$

Hence, the log-likelihood for complete data is

$$\log P_{complete}\left(\mathcal{C}\right)=\sum_{i=1}^{I}\sum_{k=1}^{K}h_{ik}\left(\log\alpha_k+\sum_{j=1}^{J}|v_j\ in\ c_i|\cdot\log\theta_{kj}\right)$$

Each M - step of the algorithm solves the maximization problem

$$\max\left(\sum_{i=1}^{I}\sum_{k=1}^{K}h_{ik}\left(\log\alpha_k+\sum_{j=1}^{J}|v_j\ in\ c_i|\cdot\log\theta_{kj}\right)\right) \tag{2}$$

with the constraints

$$\sum_{k=1}^{K}\alpha_k=1$$
$$\sum_{j=1}^{J}\theta_{kj}=1\ \ for\ \ all\ \ k=1,...,K$$

For simplicity, we denote the vector of parameters by

$$\psi=\left(\alpha_1,...,\alpha_K,\theta_{11},...,\theta_{KJ}\right)$$

and notice that the number of independent components (parameters) is $(K-1)+(KJ-K)=KJ-1$.

The EM algorithm starts with a random initialization of the parameters, denoted by

$$\psi^{(0)}=\left(\alpha_1^{(0)},...,\alpha_K^{(0)},\theta_{11}^{(0)},...,\theta_{KJ}^{(0)}\right)$$

The *iteration* $(r+1)$ consists in the following two steps:

The E - step computes the missing data, based on the model parameters estimated at iteration r, as follows:

$$h_{ik}^{(r)}=P_{\psi^{(r)}}\left(h_{ik}=1\mid\mathcal{C}\right),$$

$$h_{ik}^{(r)}=\frac{\alpha_k^{(r)}\cdot\prod\limits_{j=1}^{J}\left(\theta_{kj}^{(r)}\right)^{|v_j\ in\ c_i|}}{\sum\limits_{k=1}^{K}\alpha_k^{(r)}\cdot\prod\limits_{j=1}^{J}\left(\theta_{kj}^{(r)}\right)^{|v_j\ in\ c_i|}},i=1,...I;k=1,...,K$$

The M - step solves the maximization problem (2) and computes $\alpha_k^{(r+1)}$ and $\theta_{kj}^{(r+1)}$as follows:

$$\alpha_k^{(r+1)}=\frac{1}{I}\sum_{i=1}^{I}h_{ik}^{(r)},\ k=1,...,K$$

$$\theta_{kj}^{(r+1)} = \frac{\sum\limits_{i=1}^{I} |v_j \; in \; c_i| \cdot h_{ik}^{(r)}}{\sum\limits_{j=1}^{J} \sum\limits_{i=1}^{I} |v_j \; in \; c_i| \cdot h_{ik}^{(r)}}, \; k = 1, ..., K; j = 1, ..., J$$

The stopping criterion for the algorithm is ”Stop when parameter estimates in two consecutive iterations no longer differ significantly”. That is, stop when

$$\left\| \psi^{(r+1)} - \psi^{(r)} \right\| < \varepsilon,$$

namely

$$\sum_{k=1}^{K} \left(\alpha_k^{(r+1)} - \alpha_k^{(r)} \right)^2 + \sum_{k=1}^{K} \sum_{j=1}^{J} \left(\theta_{kj}^{(r+1)} - \theta_{kj}^{(r)} \right) < \varepsilon$$

It is well known that the EM iterations $\left(\psi^{(r)} \right)_r$ converge to the Maximum Likelihood Estimate $\widehat{\psi} = \left(\widehat{\alpha}_1, ..., \widehat{\alpha}_K, \widehat{\theta}_{11}, ..., \widehat{\theta}_{KJ} \right)$.

Once the parameters of the model have been estimated, we can disambiguate contexts of w by computing the probability of each of the senses based on features v_j occurring in the context c. Making the Naïve Bayes assumption and using the Bayes decision rule, we can decide s' if

$$s' = \arg\max_{k=1,...,K} \left(\log \widehat{\alpha}_k + \sum_{j=1}^{J} |v_j \; in \; c| \cdot \log \widehat{\theta}_{kj} \right)$$

Our choice of recommending usage of the EM algorithm for parameter estimation in unsupervised WSD with an underlying Naïve Bayes model is also based on previously existing discussions and reported results. The EM algorithm has equally been used for parameter estimation in [15], to the results of which we shall be comparing our own disambiguation results.

3. WordNet

WordNet [27, 28, 30] is a large electronic and interactive lexical database for English. It has been developed during the last 20 years[5] at Princeton University by a group headed by George Miller, a psycholinguist who was inspired by experiments in Artificial Intelligence that tried to understand human semantic memory (e.g., [5][6]). The novelty of Miller's approach was the attempt to represent the entire bulk of the lexicalized concepts of a language in a network-like structure based on hierarchical relations.

As its authors note, WordNet (WN[7]) is a lexical knowledge base which was created as a machine-readable dictionary based on psycholinguistic principles. It is a lexical database that currently contains

[5] In 1986 George Miller has the initiative of creating WordNet and designs its structure, which was meant to serve testing current theories concerning human semantic memory. Verbs are added to the network the following year (1987) and its first version (1.0) is released in 1991. Already in 2006 approximately 8000 download operations were registered on a daily basis and similar, more or less developed, semantic networks of type WordNet existed for some 40 other languages.

[6] The Collins and Quillian model proposed a hierarchical structure of concepts, where more specific concepts inherit information from their superordinate, more general concepts. That is why only knowledge particular to more specific concepts needs to be stored with such concepts.

[7] For a comprehensive description of WN see also [10].

(ver. 3.0) approximately 155,287 English nouns, verbs, adjectives and adverbs organized by semantic relations into over 117,000 meanings, where a meaning is represented by a set of synonyms (a *synset*) that can be used (in an appropriate context) to express that meaning. These numbers are approximate since WN continues to grow. The building block of WN is the synset. A synset lexically expresses a *concept*. A word's membership in multiple synsets reflects that word's polysemy. Different relations link the WN synsets. An entry in WN consists of a synset, a definitional gloss, and (sometimes) one or more phrases illustrating usage. The major relations used to organize words and entries are synonymy and antonymy, hyponymy, troponymy and hypernymy, meronymy and holonymy[8].

WordNet contains very little syntax, because it was conceived as a semantic database only.

WN was primarily viewed as a lexical database. However, due to its structure, it can be equally considered a semantic network and a knowledge base. It has been recognized as a valuable resource in the human language technology and knowledge processing communities. In WSD, WN represents the most popular sense inventory, with its synsets being used as labels for sense disambiguation.

WSD can be performed at many levels of granularity. The various existing sense inventories have different such levels of granularity. WN is very fine-grained, a property from which information retrieval, for instance, can benefit.

As the American WN continues to grow, new features are added to it. Version 2.1, for instance, is the first to incorporate the distinctions between classes and instances reported in [31] which lead to a semi-ontology of WN nouns. And which facilitate the disambiguation of proper names.

According to specific semantic relations, in WN, noun and verb synsets are organized as hierarchies, while adjective and adverb synsets are part of a completely different structure - the *cluster*.

While nouns have offered the best disambiguation results so far (no matter what type of WSD technique has been applied), the verb has proven to be, as expected, the most difficult to disambiguate part of speech. That is why, in what follows, we shall exemplify the discussed theory with regard to adjectives (organized into WN adjective clusters). The results of syntactic-based feature selection for adjective sense disambiguation will constantly be compared to those obtained in [15] when performing WN-based semantic feature selection with respect to the same part of speech.

Let us finally note that WN represents *words* and *concepts* as an interrelated system which, according to G. A. Miller [29], is consistent with evidence of the way speakers organize their mental lexicons. And which incorporate**s** knowledge into the lexicon, bringing it closer to the mental one, that contains both word and world knowledge [19]. This should be of the essence for artificial intelligence applications such as WSD and is in contrast to linguistic theories that attempt to model human grammar, or linguistic competence. Unlike such linguistic theories (as the one used in Section 4), the structure of WordNet is motivated by theories of human knowledge organization [10, p. 2].

3.1. Adjectives in WordNet

Antonymy is the dominant relation among adjectives in WN, with most adjectives being organized into "direct" antonym pairs (such as *wet-dry* or *fast-slow*), which seems to be in accordance with their organization in the human mind, as results from various word association tests.

[8] Synonymy and antonymy are the major lexical (word-word) relations encoded in WordNet. As noted in [9], "Another kind of lexical relation, dubbed ≪morphosemantic≫ is the only one that links words from all four parts of speech. It connects words that are both morphologically and semantically related...All other relations in WordNet are conceptual-semantic relations and connect not just words (synset members) but entire synsets. For each part of speech, different relations were identified."

WordNet divides adjectives into two major classes - descriptive and relational - while chromatic color adjectives are regarded as a special case. Relational adjectives are related by derivation to nouns (as in *electrical engineer*, where the adjective *electrical* is related to the noun *electricity*). Descriptive adjectives represent the largest category and this is mainly why, in what follows, we have chosen such adjectives in order to exemplify the discussed disambiguation method.

As WordNet's authors note [33], a *descriptive adjective* typically ascribes to a noun a value of an attribute[9]. WordNet contains pointers between descriptive adjectives and the nouns by which appropriate attributes are lexicalized[10]. In fact, the same authors [33] regard adjectives as "words whose sole function is to modify nouns". As underlined by Miller and Johnson-Laird [32], "the nominal information must be given priority; the adjectival information is then evaluated within the range allowed by the nominal information".

A point made by the authors of WordNet is that all descriptive adjectives have antonyms. Descriptive adjectives that do not have direct antonyms are regarded as having indirect antonyms via their semantic similarity to other adjectives that do have direct antonyms[11]. Gross, Fischer, and Miller [13] proposed to organize adjectives in clusters of synsets associated by semantic similarity[12] to a central adjective that relates the cluster to a contrasting cluster at the opposite pole of the same attribute.

The polysemy of adjectives has been a topic of debate among various authors over time. Justeson and Katz [18] consider that different senses of polysemous adjectives occur with specific nouns, or specific senses of polysemous nouns, while Murphy and Andrew [34] regard adjectives as being monosemous but having different extensions. WordNet takes the same position as Justeson and Katz [18] who also point out that the different (direct or indirect) antonyms can be useful in the disambiguation of polysemous adjectives. A point confirmed by our own test results (for which see Section 5). The importance of antonymy, which can not be captured by a strictly syntactical approach, has long become obvious. Specifically, in WordNet, antonymy is the organizing principle for descriptive adjectives, two of which we are trying to disambiguate in the following section.

3.2. WordNet-based feature selection

When implementing the mathematical model described in Section 2, discussion among specialists focuses almost entirely on the issue of feature selection. Unlike previous approaches that rely mostly on distributional and local type features, the approach to WSD of Hristea and Popescu [15] feeds semantic knowledge to the Naïve Bayes model. It is based on a set of features formed by the actual words occurring near the target word (within the context window) and reduces the size of this feature set by performing knowledge-based feature selection that relies entirely on WordNet. The WN semantic network provides the words considered relevant for the set of senses taken into consideration corresponding to the target word [15].

[9] For example, *heavy* and *light* are values for the attribute *weight*.

[10] See the *attribute relation* in WordNet that was used in our experiments. This relation is established between noun and adjective synset pairs in which the adjective is a value of the noun.

[11] For example, *wet* has *dry* as direct antonym. The adjective *watery* is considered *similar* to *wet* in WN and therefore has *dry* as indirect antonym.

[12] WN encodes the *similarity relation* between two synsets designating a synset which is the satellite of another which, in turn, is the cluster head. This relation only holds for adjective synsets contained in adjective clusters and has been used by us in our experiments.

First of all, words occurring in the same WN synsets as the target word (WN synonyms) have been chosen, corresponding to all senses of the target. Additionally, the words occurring in synsets related (through explicit relations provided in WordNet) to those containing the target word have also been considered as part of the vocabulary used for disambiguation. Synsets and relations were restricted to those associated with the part of speech of the target word. The content words of the glosses of all types of synsets participating in the disambiguation process, using the corresponding example strings as well, have been equally taken into consideration. The latter choice has been made since previous studies [3], performed for knowledge-based disambiguation, have come to the conclusion that the *example relation* - which simply returns the example string associated with the input synset - seems to provide useful information in the case of all parts of speech. A conclusion which is not surprising, as the examples contain words related syntagmatically to the target.

Corresponding to adjectives, this disambiguation method has taken into account [15] the *similarity* relation, which is typical of this part of speech. The *also-see* relation and the *attribute* relation have also been taken into consideration since these relations are viewed as most informative and have been found [3] to rank highest among the useful relations for adjectives. The *pertaining-to* relation has also been considered, whenever possible. Finally, the *antonymy* relation has represented a source of "negative information" that has proven itself useful in the disambiguation process. This was to be expected considering that antonymy is the organizing principle for descriptive adjectives in WN. This is also in accordance with previous findings of studies performed for knowledge-based disambiguation [2] that consider the antonymy relation a source of negative information allowing a disambiguation algorithm "to identify the sense of a word based on the absence of its antonymous sense in the window of context". The disambiguation vocabulary providing the best test results [15] was one in the formation of which all mentioned types of synsets have taken part (together with their glosses and associated example strings). Disambiguation results were computed [15] with and without antonym synsets participating in the disambiguation process. Results were always in favor of antonym participation, as expected (tests conducted for the descriptive adjectives *common* and *public*).

As a result of using only those words indicated as being relevant by WordNet, a much smaller vocabulary was obtained, and therefore a much smaller number of features were taking part in the disambiguation process. In the case of this method [14, 15, 16] each word (feature) contributes to the final score being assigned to a sense with a weight given by $P(v_j \mid s_k)$. This weight (probability) is not a priori established, but is learned by means of the EM algorithm.

By performing such feature selection, the disambiguation process is being placed at the border between unsupervised and knowledge-based techniques, with the Naïve Bayes model "reacting well"[13] when being fed semantic knowledge.

4. A dependency-based semantic space for word sense disambiguation

With the benefits of placing the disambiguation process at the border between unsupervised and knowledge-based techniques having become so obvious, our next concern is to augment the role of linguistic knowledge in informing the construction of the semantic space for WSD. Specifically, in the case of unsupervised WSD with an underlying Naïve Bayes model, our intention is to construct a semantic space that takes into account *syntactic relations*. The Naïve Bayes model will be fed syntactic knowledge based

[13] See the test results and comparisons presented in [15] and in Section 5.2 of the present paper.

on the consideration that “because syntax-based models capture more linguistic structure than word-based models, they should at least in theory provide more informative representations of word meaning” [36]. The choice of the syntactic formalism to be used is not an easy one. Our main concern was that of having the semantic relationships between concepts and the words that lexicalize them mirrored in some way, considering that semantic knowledge of this type had already proven useful in the disambiguation process [14, 15, 16]. Once again following the line of reasoning of Padó and Lapata [36], “an ideal syntactic formalism should abstract over surface word order, mirror semantic relationships as closely as possible, and incorporate word-based information is addition to syntactic analysis ... These requirements point towards dependency grammar, which can be considered as an intermediate layer between surface syntax and semantics”.

Despite the various existing linguistic theories, which lead to different ways of viewing sentence structure and therefore syntactic analysis, most linguists today agree that at the heart of sentence structure are *the relations among words*. These relations refer either to grammatical functions (subject, complement etc.) or to the links which bind words into larger units like phrases or even sentences. The dependency grammar[14] approach to syntactic analysis takes into consideration the latter, viewing each word as *depending* on another word that links it to the rest of the sentence. Unlike generative grammars therefore, dependency grammars (DG) are not based on the notion of constituent but on the direct relations existing among words.

The relation between the *dependent* word and the word on which it depends (the *head*) is at the basis of DG. The syntactical analysis of a sentence signifies, from the point of view of DG, the description of all *dependency relations* (between the head and the dependent) which occur among all words of the sentence. The dependency relations may or may not lack directionality (from head to dependent) in the relation between words, according to which variant or alternative dependency-based grammatical theory[15] is used. A variety of dependency relations may exist among the words of a sentence if no restrictions are specified. The role of dependency grammars is mainly that of specifying the restrictions which the dependency relations should meet so that the structure they define is linguistically correct. The *dependency structure* will specify, in the case of each word, what other word it depends on. The dependencies indicated by the dependency structure of a sentence map straightforwardly onto a directed graph representation, in which the words of the represented sentence are nodes in the graph and grammatical relations are edge labels. It is various combinations of such dependencies[16] that will form the context over which we shall be constructing our semantic space for WSD. Just as Padó and Lapata [36], we adopt the working hypothesis that syntactic structure in general and argument structure in particular are a close reflection of lexical meaning [22]. When using dependency relations we model meaning by quantifying the degree to which words occur in similar syntactic environments.

4.1. Dependency-based feature selection

Of the several existing dependencies parsers, we have used the Stanford Parser [20] in order to automatically extract typed dependency parses of English sentences.

While the classical dependency-based linguistic theory does not allow the arches denoting the dependency relations to intersect (thus leading to an oriented graph which has no cycles), the dependency

[14] *Dependency grammars* were introduced by Lucien Tesnière [44].

[15] See also *Link grammar* [42, 43] and *Word grammar* [17].

[16] For which see Section 5.1.

analysis performed by the Stanford parser can be either projective (disallowing crossing dependencies) or non-projective (permitting crossing dependencies). When using this specific syntactic parser we have performed a dependency syntactical analysis of non-projective type, in order to maximize the number of dependencies between content words. Although this may increase the number of features (words included into the disambiguation vocabulary) and therefore of parameters which must be estimated by the EM algorithm, it is our belief it could give a better indication of the ambiguous word's sense in context. The number of resulting features should then be decreased by taking into account only dependency relations of specific types (see Section 5.1).

Tests concerning the construction of the semantic space for WSD by feeding the Naïve Bayes model syntactic knowledge (provided by dependency relations) have so far concentrated on adjectives. However, we believe that the discussion which is to follow holds for all parts of speech (POS) and should only be subject to certain adaptations, depending on the particularities of each syntactic POS category. Our intention is to study the effectiveness of syntactic features (determined by dependency relations) as compared to semantic ones, more precisely to the ones provided by WordNet, which has been created in the spirit of understanding human semantic memory.

In order to inform the construction of the semantic space for WSD with syntactic knowledge of the mentioned type, we have conducted a two-stage experiment. At the first stage of our study, we have made no qualitative distinction between the different relations, by not taking into account the type of the involved dependencies[17]. This approach was inspired by existing syntax-based semantic space models, where the construction of the space is either based on all relations [12, 23] or on a fixed subset [21], but always with no qualitative distinction between the different relations being made. At the second stage of our experiment we have taken into account the *dependency type*, thus informing the construction of the semantic space in a more linguistically rich manner. We have therefore eliminated certain paths (of the associated dependency graph) from the semantic space, on the basis of linguistic knowledge, by making use only of specific dependency relations, which are considered more informative than others relatively to the studied part of speech.

At both stages, our experiment takes place in the same type of setup. Namely, in defining the syntactic context of the target word, we have first taken into consideration *direct relationships*[18] between the target and other words (denoted by dependency relations where the target is either the head or a dependent and which correspond to paths of length 1 anchored[19] at the target in the associated dependency graph). We subsequently consider *indirect relationships*[20] between the target and other words by taking into account paths of length 2 in the same associated dependency graph. At the present stage we have limited our study to *second order dependencies*. However, the length (order) of these dependencies (paths in the associated graph) represent a parameter that can vary and which we consider analogous to the classical "window size" parameter. This parameter should have relatively small values, since it is a known fact that linguistically interesting paths are of limited length. By taking into account second order dependencies we additionally represent indirect semantic relations which could prove to be important.

[17] We have only eliminated the potentially unuseful relations - for WSD - provided by the Stanford parser, such as: determiner, predeterminer, numeric determiner, punctuation relations, etc.

[18] In what follows, such dependencies will be called *first order dependencies*.

[19] A path anchored at the target word w is a path in the dependency graph starting at w. If the dependency relations have directionality, leading to an associated oriented graph, a path anchored at w is either a path starting at w or arriving at w.

[20] In what follows, such dependencies will be called *second order dependencies*.

When using dependency-based syntactic features we shall form the disambiguation vocabulary by taking into account all words that participate in the considered dependencies.

In our experiments we have considered both dependencies having directionality and dependencies lacking it. Contrary to other studies [36], which consider that "directed paths would limit the context too severely", we have taken into account both undirected and directed paths, with the latter providing the best test results (see Section 5.2). The Naïve Bayes model seems to react well to the directionality of dependency relations.

5. Empirical evaluation

In order to compare our disambiguation results to those of other previous studies [15, 16] that have made use of the same Naïve Bayes model, trained with the EM algorithm, but have performed semantic WordNet-based feature selection, we hereby try to disambiguate the same target adjectives using the same corpora (see Section 5.2). Specifically, we shall be reporting disambiguation results in the case of adjectives *common* and *public* (see Section 5.2).

5.1. Design of the experiments

With respect to adjectives we have used as test data the [4] data containing twelve words taken from the ACL/DCI Wall Street Journal corpus and tagged with senses from the Longman Dictionary of Contemporary English. We have chosen this data set for our tests concerning adjectives since it has equally been used in the case of the Hristea and Popescu [15] approach to WSD (WordNet-based feature selection), to which we shall be comparing the results of our own disambiguation method (dependency-based feature selection). As already mentioned, test results will be reported in the case of two adjectives, *common* and *public*. The senses of *common* that have been taken into consideration and their frequency distribution are shown in Table 1, while Table 2 provides the same type of information corresponding to the adjective *public*. In these tables *total count* represents the number of occurrences in the corpus of each word, with each of the adjectives being limited to the 3 most frequent senses, while *count* gives the percentage of occurrence corresponding to each of these senses.

In order for our experiments to be conducted, the data set was preprocessed in the usual required way for WSD: the stop words were eliminated and Porter stemmer was applied to the remaining words.

When performing syntactic (dependency-based) feature selection, at the first stage of the testing process, we have taken into account both directed and undirected dependency relations. No information concerning the types of the considered dependencies was used. At this stage of our study, we have designed a set of eight experiments (with the first two referring to undirected dependencies and the following six referring to directed ones).

The first performed experiment considers all undirected first order dependencies anchored at the target word. All words participating in these dependencies (with the exception of the target) will be included in the so-called disambiguation vocabulary. This experiment is referred to in Table 3 of Section 5.2 (corresponding to adjective *common*) and Table 4 of Section 5.2 (corresponding to adjective *public*) as *Undirected first order dependencies*.

The second performed experiment also refers to undirected dependencies. It takes into account all first order and second order dependencies which are anchored at the target word. All words participating

Table 1. Distribution of senses of *common*

Sense	Count
As in the phrase “common stock”	84%
Belonging to or shared by 2 or more	8%
Happening often; usual	8%
Total count	1,060

Table 2. Distribution of senses of *public*

Sense	Count
Concerning people in general	68%
Concerning the government and people	19%
Not secret or private	13%
Total count	715

in these dependencies are part of the disambiguation vocabulary. This experiment is referred to in Table 3 of Section 5.2 (corresponding to adjective *common*) and Table 4 of Section 5.2 (corresponding to adjective *public*) as *Undirected first and second order dependencies*.

The following six experiments were all designed with reference to dependency relations that have directionality[21]. Both dependencies that view the target word as head[22] and dependencies that view it as dependent[23] have been considered. Within this group of experiments, the first two refer to directed first order dependencies and the following four to directed first and second order dependencies, respectively (see Tables 3 and 4 of Section 5.2).

The third performed experiment which is presented in Tables 3 and 4 of Section 5.2 views the target word as *head*. It takes into account all head-driven dependencies of first order anchored at the target word and collects all corresponding dependents, which form the considered disambiguation vocabulary. This experiment must be looked up, in Table 3 of Section 5.2 (corresponding to adjective *common*) and Table 4 of Section 5.2 (corresponding to adjective *public*) under *Directed first order dependencies*. It is referred to as *Head-driven dependencies*.

The fourth performed experiment views the target word as *dependent*. It takes into account all dependent-driven dependencies of first order anchored at the target word and collects all corresponding heads, which form the considered disambiguation vocabulary. This experiment must also be looked up, in Table 3 of Section 5.2 (corresponding to adjective *common*) and Table 4 of Section 5.2 (corresponding to adjective *public*) under *Directed first order dependencies*. It is referred to as *Dependent-driven dependencies*.

The fifth performed experiment views the target word as *head*. It takes into account all first order head-driven dependencies anchored at the target word and collects all corresponding dependents. Furthermore, it takes into consideration all first order head-driven dependencies anchored at the previously

[21] The considered directionality is from head to dependent.

[22] In what follows, these dependencies will be called *head-driven dependencies*.

[23] In what follows, these dependencies will be called *dependent-driven dependencies*.

obtained dependents and collects the corresponding dependents of these dependents[24]. All such collected words are included in the disambiguation vocabulary. This experiment must be looked up, in Table 3 of Section 5.2 (corresponding to adjective *common*) and Table 4 of Section 5.2 (corresponding to adjective *public*) under *Directed first and second order dependencies* and *Head-driven dependencies*, respectively. It is referred to as *Two head-driven dependencies*.

The sixth performed experiment also views the target word as *head*. It takes into account all first order head-driven dependencies anchored at the target word and collects all corresponding dependents. Furthermore, it takes into consideration all first order dependent-driven dependencies anchored at the previously obtained dependents and collects the corresponding heads of these dependents[25]. All such collected words are included in the disambiguation vocabulary. This experiment must be looked up, in Table 3 of Section 5.2 (corresponding to adjective *common*) and Table 4 of Section 5.2 (corresponding to adjective *public*) under *Directed first and second order dependencies* and *Head-driven dependencies*, respectively. It is referred to as *Head-driven dependencies and dependent-driven dependencies*.

The seventh performed experiment views the target word as *dependent*. It takes into account all first order dependent-driven dependencies anchored at the target word and collects all corresponding heads. Furthermore, it takes into consideration all first order dependent-driven dependencies anchored at the previously obtained heads and collects the corresponding heads of these heads[26]. All such collected words are included in the disambiguation vocabulary. This experiment must be looked up, in Table 3 of Section 5.2 (corresponding to adjective *common*) and Table 4 of Section 5.2 (corresponding to adjective *public*) under *Directed first and second order dependencies* and *Dependent-driven dependencies*, respectively. It is referred to as *Two dependent-driven dependencies*.

The eighth and last performed experiment also views the target word as *dependent*. It takes into account all first order dependent-driven dependencies anchored at the target word and collects all corresponding heads. Furthermore, it takes into consideration all first order head-driven dependencies anchored at the previously obtained heads and collects the resulted dependents of these heads[27]. All such collected words are included in the disambiguation vocabulary. This experiment must be looked up, in Table 3 of Section 5.2 (corresponding to adjective *common*) and Table 4 of Section 5.2 (corresponding to adjective *public*) under *Directed first and second order dependencies* and *Dependent-driven dependencies*, respectively. It is referred to as *Dependent-driven dependencies and head-driven dependencies*.

Contrary to more general comments made in other studies [36], as far as WSD with an underlying Naïve Bayes model is concerned, test results will show (see Section 5.2) that considering directionality of the dependency relations is essential when forming the disambiguation vocabulary.

During the second stage of our testing process we have been taking into account the *type* of the dependency relations and we have chosen only typed dependencies that have been considered relevant for the study of adjectives. Such typed dependencies were the providers of the words to be included in the disambiguation vocabulary.

[24] The case *Two head-driven dependencies* can be summarized as follows: let us denote the target word by A; collect all words of type B and C such that B is a dependent of A and C is a dependent of B.

[25] The case *Head-driven dependencies and dependent-driven dependencies* can be summarized as follows: let us denote the target word by A; collect all words of type B and C such that B is a dependent of A and B is a dependent of C.

[26] The case *Two dependent-driven dependencies* can be summarized as follows: let us denote the target word by A; collect all words of type B and C such that A is a dependent of B and B is a dependent of C.

[27] The case *Dependent-driven dependencies and head-driven dependencies* can be summarized as follows: let us denote the target word by A; collect all words of type B and C such that A is a dependent of B and C is a dependent of B .

Of the various dependency relations provided by the Stanford parser, we have chosen a restricted set of relations (that we view as minimal) in order to conduct our experiments: *adjectival modifier*, *nominal subject*, *noun compound modifier* and *preposition collapsed*. The latter, which is not typical for adjectives, is a common Stanford collapsed dependency relation. In the chosen non-projective analysis of the Stanford parser, the dependencies that involve prepositions, conjunctions or multi-word constructions are collapsed in order to get direct dependencies between content words [26]. In our case, this relation is applicable for adjectival constructs where the adjective can be accompanied/intensified by an adverb particle such as “more common” or “very difficult”.

Our choice of dependency relations for this type of WSD task was guided mainly by the comments of Miller and Johnson-Laird [32] who clearly state that the nominal information should be given priority, while the adjectival information can and should be evaluated strictly within the range allowed by the nominal one. We have therefore constantly looked for, or looked at, the noun that the target adjective modifies.

This principle has guided us when choosing the *nominal subject* relation, for instance. This relation refers to the predicative form of the adjective linked via a copula verb to the noun that the adjective modifies.

In the case in which the head role of the target adjective is not imposed, that is in the case of undirected dependency relations, we have also taken into consideration the *adjectival modifier* relation, a very frequent dependency relation for adjectives that connects them as dependents directly to the noun they modify. This is probably the most informative of the considered relations and corresponds to the *attribute* semantic relation which we have used when performing WordNet-based feature selection.

Our experiments have once again used both first order and second order dependencies. In order to obtain second order dependencies, the first order dependency relations were composed only with the modifier-type relations for nouns, as the first order relations (usually) return the modified noun of the target adjective (in our case by means of the previously specified relations - *nominal subject* and *preposition collapsed*). The modifier-type relations we have considered are *adjectival modifier* (returning the modifying adjective) and *noun compound modifier* (for the modifying noun).

The presented test results (see Section 5.2) correspond to experiments performed with this restricted set of chosen dependency relations, which is meant to ensure a minimal number of features for WSD, as well as a restricted number of parameters to be estimated by the EM algorithm. However, experiments of the same type could be conducted with an enlarged set of such relations, a choice which should be made according mainly to linguistic criteria and by the linguistic community.

During this second stage of our study, which takes into account the type of the involved dependencies, we have designed a set of four experiments. Both directed and undirected dependency relations have again been considered.

The first performed experiment uses undirected first order dependencies. The considered relations are *adjectival modifier, preposition collapsed* and *nominal subject*, respectively. The disambiguation vocabulary is formed, as before, with all words participating in these dependency relations (with the exception of the target adjective). This experiment is referred to in Table 5 of Section 5.2 (corresponding to adjective *common*) and in Table 6 of Section 5.2 (corresponding to adjective *public*) as *Undirected first order dependencies*.

The second performed experiment also refers to undirected dependencies, this time of second order as well. The disambiguation vocabulary is formed with all words provided by the undirected first order dependencies of the previous experiment, to which all words indicated by the considered undirected second order dependencies are added. When forming the second order dependencies the following modifier-type dependency relations are used: *adjectival modifier* and *noun compound modifier*. This experiment is referred to in Table 5 of Section 5.2 (corresponding to adjective *common*) and in Table 6 of Section 5.2 (corresponding to adjective *public*) as *Undirected first and second order dependencies*.

The next two experiments were both designed with reference to dependency relations that have directionality. Within both experiments the target adjective is viewed as *head*[28] only. Therefore all considered dependencies are head-driven ones.

The third performed experiment takes into account first order head-driven dependencies and forms the disambiguation vocabulary with all words they provide. The dependency relations that were used are *preposition collapsed* and *nominal subject*. All these dependency relations take the target adjective as the head word and, as consequence, the resulted disambiguation vocabulary is made of all the dependents of the target. This experiment is referred to in Table 5 of Section 5.2 (corresponding to adjective *common*) and in Table 6 of Section 5.2 (corresponding to adjective *public*) as *Head-driven first order dependencies*.

The fourth and last performed experiment refers to first and second order head-driven dependencies. The disambiguation vocabulary is formed with all words provided by the first order head-driven dependencies of the previous experiment, to which all words indicated by the considered second order head-driven dependencies, representing dependents of the target's dependents, are added. The experiment therefore considers the head role of both the target and of its dependents. When forming the second order dependencies, the following modifier-type dependency relations are used: *adjectival modifier* and *noun compound modifier*. This experiment is referred to in Table 5 of Section 5.2 (corresponding to adjective *common*) and in Table 6 of Section 5.2 (corresponding to adjective *public*) as *Head-driven first and second order dependencies*.

Test results will show (see Section 5.2) that taking into account the *type* of the dependency relations when forming the disambiguation vocabulary is of the essence.

5.2. Test results

Performance is evaluated in terms of accuracy. In the case of unsupervised disambiguation defining accuracy is not as straightforward as in the supervised case. Our objective is to divide the I given instances of the ambiguous word into a specified number K of sense groups, which are in no way connected to the sense tags existing in the corpus. In our experiments, sense tags are used only in the evaluation of the sense groups found by the unsupervised learning procedure. These sense groups must be mapped to sense tags in order to evaluate system performance. As in previous studies [15, 16] we have used the mapping that results in the highest classification accuracy[29].

Test results are presented in Tables 3, 4, 5 and 6. Each result represents the average accuracy and standard deviation obtained by the learning procedure over 20 random trials while using a threshold ε having the value 10^{-9}.

[28] This is the approach suggested by the first series of performed experiments, which had disregarded the dependency type. Test results have shown (see Section 5.2) that directionality of the relations counts and that the best disambiguation results are obtained when the target word plays the role of head.

[29] In order to conduct our experiments we have chosen a number of sense groups equal to the number of sense tags existing in the corpus. Therefore a number of $K!$ possible mappings (with K denoting the number of senses of the target word) should be taken into account. For a fixed mapping, its accuracy is given by the number of correct labellings (identical to the corresponding corpus sense tags) divided by the total number of instances. From the $K!$ possible mappings, the one with maximum accuracy has been chosen.

Table 3. First stage of the experiments corresponding to adjective *common*

Name of experiment			No of features	Instances having only null features	Accuracy
Undirected first order dependencies			279	.02	.586±.09
Undirected first and second order dependencies			553	.05	.532±.09
Directed first order dependencies	*Head-driven dependencies*		135	.00	.616±.07
	Dependent-driven dependencies		178	.02	.547±.05
Directed first and second order dependencies	*Head-driven dependencies*	*Two head-driven dependencies*	224	.01	.643±.09
		Head-driven and dependent-driven dependencies	154	.00	.614±.10
	Dependent-driven dependencies	*Two dependent-driven dependencies*	281	.03	.517±.07
		Dependent-driven and head-driven dependencies	336	.03	.540±.10

Apart from accuracy, the following type of information is also included in Tables 3, 4, 5 and 6: number of features resulting in each experiment and percentage of instances having only null features (i.e. containing no relevant information).

As previously mentioned, within the present approach to disambiguation, the value of a feature is given by the number of occurrences of the corresponding word in the given context window. Since the process of feature selection is based on the restriction of the disambiguation vocabulary, it is possible for certain instances not to contain any of the relevant words forming this vocabulary. Such instances will have null values corresponding to all features. These instances do not contribute to the learning process. However, they have been taken into account in the evaluation stage of our experiments. Corresponding to these instances, the algorithm assigns the sense s_k for which the value $P(s_k)$ (estimated by the EM algorithm) is maximal.

As far as the Stanford parser [20] is concerned, we generate the output in dependency relation format [25] and we preprocess the data in the usual way: edges that do not connect open-class words are filtered out, words are lemmatized. The first of the mentioned operations could lead to some instances having only null features. However, corpus coverage will be much greater here than in the case of WordNet-based feature selection, where an independent knowledge source (WN) is used. Which makes the obtained results even more valuable.

Test results are presented in Tables 3 and 5 (corresponding to adjective *common*) and in Tables 4 and 6 (corresponding to adjective *public*).

At the first stage of the testing process, when no information concerning the dependency type was used (see the designed experiments in Section 5.1), the best obtained accuracy was .643±.09 in the case of adjective *common* and .607±.02 in the case of adjective *public*, respectively. These disambiguation results are more modest than the ones obtained in [15] when performing WordNet-based feature selection

Table 4. First stage of the experiments corresponding to adjective *public*

Name of experiment			No of features	Instances having only null features	Accuracy
Undirected first order dependencies			340	.03	.424±.04
Undirected first and second order dependencies			698	.07	.436±.04
Directed first order dependencies	*Head-driven dependencies*		40	.00	.597±.03
	Dependent-driven dependencies		312	.02	.435±.02
Directed first and second order dependencies	*Head-driven dependencies*	*Two head-driven dependencies*	53	.01	.607±.02
		Head-driven and dependent-driven dependencies	51	.01	.569±.03
	Dependent-driven dependencies	*Two dependent-driven*	527	.04	.434±.04
		Dependent-driven dependencies and head-driven dependencies	515	.05	.426±.03

(as described in Section 3.2). The mentioned study reports an accuracy of .775±.02 (obtained with 83 features and 19.2% instances having only null features) in the case of adjective *common* and an accuracy of .559±.03 (obtained with 74 features and 43.3% instances having only null features) in the case of adjective *public*, respectively.

When analyzing Table 3 and Table 4 of the present study, several conclusions can be drawn.

The best result is never obtained in the presence of undirected dependencies. When using undirected dependencies, taking into account second order dependencies as well does not increase accuracy in the case of adjective *common* and leads to only a slight increase in accuracy in the case of adjective *public*.

The highest accuracy is attained within the same testing setup, both corresponding to adjective *common* and corresponding to adjective *public* - namely in the case *Two head-driven dependencies* - which takes into consideration the head role of both the target and of its dependents[30]. Disambiguation accuracy obtained in this case is higher than the one attained by head-driven directed first order dependencies. A result which suggests that, when ignoring the dependency type, one should move to second order dependencies for increasing accuracy.

High accuracy is never attained in the case of dependent-driven dependencies. Considering solely the head role, corresponding to both first and second order dependencies, proves to be of the essence, a principle which has guided our second series of experiments. During which we have tried to decrease the number of features, and therefore of parameters that the EM algorithm must estimate, by taking into account the *type* of the involved dependency relations (see test results in Tables 5 and 6).

[30] Let us note that accuracy is always higher in the case *Two head-driven dependencies* than in the case *Head-driven dependencies and dependent-driven dependencies*, which shows that, in the case of directed first and second order dependencies, it is essential to consider the head role not only of the target word but also of its dependents.

Table 5. Second stage of the experiments corresponding to adjective *common*

Name of experiment	No of features	Instances having only null features	Accuracy
Undirected first order dependencies	225	.02	.605±.09
Undirected first and second order dependencies	416	.04	.568±.09
Head-driven first order dependencies	73	.00	.775±.07
Head-driven first and second order dependencies	112	.07	.753±.08

When comparing results of Table 3 with those of Table 5 and results of Table 4 with those of Table 6 respectively, it becomes obvious that disambiguation accuracy improves as a result of taking into consideration the type of the existing dependencies.

Table 6. Second stage of the experiments corresponding to adjective *public*

Name of experiment	No of features	Instances having only null features	Accuracy
Undirected first order dependencies	294	.02	.428±.03
Undirected first and second order dependencies	443	.06	.423±.03
Head-driven first order dependencies	11	.00	.669±.01
Head-driven first and second order dependencies	18	.00	.662±.01

When analyzing Tables 5 and 6, one notices that the best obtained accuracy was .775±.07 in the case of adjective *common* and .669±.01 in the case of adjective *public*, respectively. Both results were obtained with full corpus coverage, ensured by a small number of features (73 words corresponding to adjective *common* and 11 words corresponding to adjective *public*). Both accuracy values are superior to the ones obtained in our previous experiments, which did not take into account the dependency type, and are attained with a much smaller number of features. Maximum disambiguation accuracy corresponding to adjective *common* is practically the same as the one obtained when performing WordNet-based feature selection, the latter being attained with a more or less similar number of features but with less corpus coverage. In the case of adjective *public* accuracy increases significantly when performing dependency-based feature selection (as compared to WN-based feature selection), while the number of features used in disambiguation decreases significantly in spite of the ensured full corpus coverage. Both discussed accuracies were obtained within the same testing setup, namely in the case of head-driven first order dependencies, which take into account the head role of the target. In fact, this role is so significant that moving to second order dependencies becomes unnecessary in this case[31].

[31] Disambiguation results are close but slightly inferior in the case of head-driven first and second order dependencies (see Tables 5 and 6).

Accuracies obtained when considering only undirected dependencies are always much lower, even when taking into account the dependency type. In both studied cases the head role of the target is of the essence.

6. Conclusions

The present study has concentrated on the issue of feature selection for unsupervised WSD performed with an underlying Naïve Bayes model. It has extended the disambiguation method presented in [15] by introducing dependency-based feature selection and has tested the efficiency of such syntactic features in the case of adjectives. Performing this type of knowledge-based feature selection has placed the disambiguation process at the border between unsupervised and knowledge-based techniques, while reinforcing the benefits of combining the unsupervised approach to the WSD problem with usage of a knowledge source for feature selection. The discussed method has once again proven that a basic, simple knowledge-lean disambiguation algorithm, hereby represented by the Naïve Bayes model, can perform quite well when provided knowledge in an appropriate way.

Specifically, the Naïve Bayes model needs to be fed knowledge in order to perform well as clustering technique for unsupervised WSD. This knowledge can be fed in various ways and can be of various natures. The present paper has examined syntactic knowledge provided by dependency relations as compared to semantic knowledge offered by the semantic network WordNet.

Our conclusion is that the Naïve Bayes model reacts well in the presence of syntactic knowledge of this type. The fact that 11 words (features) only, for instance, are sufficient in order to attain a higher disambiguation accuracy[32] than the one obtained by WordNet-based feature selection, while ensuring full corpus coverage, determines us to recommend syntactic (dependency-based) feature selection as a reliable alternative to the semantic one.

The challenge is, however, great for dependency-based feature selection. Especially when keeping in mind that the structure of WordNet is motivated by theories of human knowledge organization and that it is consistent with evidence of the way speakers organize their mental lexicons. We have attempted to replace such knowledge with information coming from a linguistic theory that models linguistic competence. The essence of the considered dependencies, namely the head role of a word (in our case the target), seems to represent crucial information for the Naïve Bayes model when acting as clustering technique for unsupervised WSD. Together with the dependency type, which should always be taken into account. Since it expresses in what way the head links the dependent to the sentence in which they both occur.

Whether or not disambiguation accuracy can be improved by taking into consideration dependencies of various other types or by performing a projective-type analysis could represent a topic of discussion (primarily for the linguistic community). Whether or not this type of syntactic information can replace the mentioned semantic one should probably be subject to further investigation. And should also involve other parts of speech. We hope to have initiated an open discussion concerning the *type of knowledge* that is best suited for the Naïve Bayes model when performing the task of unsupervised WSD.

[32] In the case of adjective *public*; see Table 6 of Section 5.2.

Acknowledgements

The work of author Florentina Hristea has been supported by the National University Research Council of Romania (the "Ideas" research programme, PN II - IDEI), Contract No. 659/2009.
The authors express their deepest gratitude to Professor Ted Pedersen for having provided the data set necessary for performing the presented tests and comparisons with respect to adjectives.

References

[1] Agirre, E., Edmonds, P. G., (Eds.): *Word Sense Disambiguation: Algorithms and Applications*, Springer, 2006.

[2] Banerjee, S., Pedersen, T.: An Adapted Lesk Algorithm for Word Sense Disambiguation Using WordNet, in: *Proceedings of the Third International Conference on Computational Linguistics and Intelligent Text Processing*, CICLing '02, 2002, ISBN 3-540-43219-1, 136–145.

[3] Banerjee, S., Pedersen, T.: Extended Gloss Overlaps as a Measure of Semantic Relatedness, in: *Proceedings of the 8th International Joint Conference on Artificial Intelligence*, 2003, 805–810.

[4] Bruce, R., Wiebe, J., Pedersen, T.: The Measure of a Model, in: *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 1996, 101–112.

[5] Collins, A. M., Quillian, M. R.: Retrieval time from semantic memory, *Journal of Verbal Learning and Verbal Behavior*, **8**, 1969, 240–247.

[6] Dempster, A., Laird, N., Rubin, D.: Maximum likelihood from incomplete data via the EM algorithm, *Journal of the Royal Statistical Society B*, **39**(1), 1977, 1–38.

[7] Domingos, P., Pazzani, M.: On the optimality of the simple Bayesian classifier under zero-one loss, *Machine Learning*, **29**, 1997, 103–130.

[8] Eberhardt, F., Danks, D.: Confirmation in the Cognitive Sciences: The Problematic Case of Bayesian Models, *Minds and Machines*, **21**, 2011, 389–410.

[9] Fellbaum, C.: WordNet, in: *Theory and Applications of Ontology: Computer Applications (R. Poli, M. Healy and A. Kameas Eds.)*, Dordrecht, London: Springer, 2010, 231–243.

[10] Fellbaum, C., (Eds.): *WordNet: an Electronic Lexical Database*, The MIT Press, Cambridge, MA, 1998.

[11] Gale, W., Church, K., Yarowsky, D.: A method for disambiguating word senses in a large corpus, *Computers and the Humanities*, **26**(5–6), 1992, 415–439, ISSN 0010-4817.

[12] Grefenstette, G.: *Explorations in Automatic Thesaurus Discovery*, Dordrecht: Kluwer Academic Publishers, 1994.

[13] Gross, D., Fischer, U., Miller, G. A.: The organization of adjectival meanings., *Journal of Memory and Language*, **28**, 1989, 92–106.

[14] Hristea, F.: Recent Advances Concerning the Usage of the Naïve Bayes Model in Unsupervised Word Sense Disambiguation., *International Review on Computers and Software*, **4**(1), 2009, 58–67.

[15] Hristea, F., Popescu, M.: Adjective Sense Disambiguation at the Border Between Unsupervised and Knowledge-Based Techniques, *Fundamenta Informaticae*, **91**(3–4), 2009, 547–562, ISSN 0169-2968.

[16] Hristea, F., Popescu, M., Dumitrescu, M.: Performing word sense disambiguation at the border between unsupervised and knowledge-based techniques, *Artificial Intelligence Review*, **30**(1–4), December 2008, 67–86, ISSN 0269-2821.

[17] Hudson, R.: *Word Grammar*, Oxford: Blackwell, 1984.

[18] Justeson, J. S., Katz, M. K.: *Principled disambiguation: Discriminating adjective sense with modified nouns*, Unpublished manuscript, IBM Thomas J. Watson Research Center, NY, 1993.

[19] Kay, M.: The concrete lexicon and the abstract dictionary, in: *Proceedings of the Fifth Annual Conference of the UW Centre for the New Oxford English Dictionary*, 1989, 35–41.

[20] Klein, D., Manning, C. D.: Accurate unlexicalized parsing, in: *Proceedings of the 41st Meeting of the Association for Computational Linguistics (ACL 2003)*, 2003, 423–430.

[21] Lee, L.: Measures of distributional similarity, in: *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*, 1999, 25–32.

[22] Levin, B.: *English Verb Classes and Alternations: A Preliminary Investigation*, Chicago, IL: University of Chicago Press, 1993.

[23] Lin, D.: Automatic retrieval and clustering of similar words, in: *Proceedings of the Joint Annual Meeting of the Association for Computational Linguistics and International Conference on Computational Linguistics*, 1998, 768–774.

[24] Manning, C., Schütze, H.: *Foundations of Statistical Natural Language Processing*, Cambridge, MA: The MIT Press, 2003.

[25] de Marneffe, M.-C., MacCartney, B., Manning, C. D.: Generating Typed Dependency Parses from Phrase Structure Parses, in: *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC-06)*, 2006, 449–454.

[26] de Marneffe, M.-C., Manning, C. D.: *Stanford typed dependencies manual*, Technical report, Stanford University, 2008.

[27] Miller, G. A.: Nouns in WordNet: a lexical inheritance system, *International Journal of Lexicography*, **3**(4), 1990, 245–264.

[28] Miller, G. A.: WordNet: a lexical database for English, *Communications of the ACM*, **38**(11), November 1995, 39–41.

[29] Miller, G. A.: Nouns in WordNet, in: *WordNet: An Electronic Lexical Database (C. Fellbaum Ed.)*, Cambridge, MA: The MIT Press, 1998, 23–46.

[30] Miller, G. A., Beckwith, R., Fellbaum, C., Gross, D., Miller, K.: WordNet: An on-line lexical database, *International Journal of Lexicography*, **3**, 1990, 235–244.

[31] Miller, G. A., Hristea, F.: WordNet Nouns: Classes and Instances, *Computational Linguistics*, **32**(1), 2006, 1–3.

[32] Miller, G. A., Johnson-Laird, P. N.: *Language and perception*, Cambridge, MA: Harvard University Press, 1976.

[33] Miller, K.: Modifiers in WordNet, in: *WordNet: An Electronic Lexical Database (C. Fellbaum Ed.)*, Cambridge, MA: The MIT Press, 1998, 47–67.

[34] Murphy, G. L., Andrew, J. M.: The conceptual basis of antonymy and synonymy in adjectives, *Journal of Memory and Language*, **32**, 1993, 301–319.

[35] Năstase, V.: Unsupervised All-words Word Sense Disambiguation with Grammatical Dependencies, in: *Proceedings of the Third International Joint Conference on Natural Language Processing*, 2008, 757–762.

[36] Padó, S., Lapata, M.: Dependency-Based Construction of Semantic Space Models, *Computational Linguistics*, **33**(2), 2007, 161–199.

[37] Pedersen, T.: Unsupervised Corpus-Based Methods for WSD, in: *Word Sense Disambiguation: Algorithms and Applications*, Springer, 2006, 133–166.

[38] Pedersen, T., Bruce, R.: Distinguishing Word Senses in Untagged Text, in: *Proceedings of the Second Conference on Empirical Methods in Natural Language Processing*, 1997, 197–207.

[39] Pedersen, T., Bruce, R.: Knowledge Lean Word-Sense Disambiguation, in: *Proceedings of the 15th National Conference on Artificial Intelligence*, AAAI Press, 1998, 800–805.

[40] Ponzetto, S. P., Navigli, R.: Knowledge-rich Word Sense Disambiguation rivaling supervised systems, in: *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, 2010, 1522–1531.

[41] Schütze, H.: Automatic word sense discrimination, *Computational Linguistics*, **24**(1), 1998, 97–123.

[42] Sleator, D. D. K., Temperley, D.: *Parsing English with a Link Grammar*, Technical report CMU-CS-91-196, Carnegie Mellon University, Pittsburgh, PA, 1991.

[43] Sleator, D. D. K., Temperley, D.: Parsing English with a Link Grammar, in: *Proceedings of the Third International Workshop on Parsing Technologies (IWPT93)*, 1993, 277–292.

[44] Tesnière, L.: *Eléments de syntaxe structurale*, Paris: Klincksieck, 1959.