Assessing the Contribution of Shallow and Deep Knowledge Sources for Word Sense Disambiguation

Lucia Specia (l.specia@wlv.ac.uk)
Research Institute for Information and Language Processing, University of Wolverhampton, Stafford Street, Wolverhampton, WV1 1SB, UK

Mark Stevenson (m.stevenson@dcs.shef.ac.uk)
Department of Computer Science, University of Sheffield, Regent Court, 211 Portobello, Sheffield, S1 4DP, UK

Maria das Graças Volpe Nunes (gracan@icmc.usp.br)
Universidade de São Paulo, Caixa Postal 668, São Carlos, 13560-970, Brazil

September 24, 2009

Abstract. Corpus-based techniques have proved to be very beneficial in the development of efficient and accurate approaches to word sense disambiguation (WSD) despite the fact that they generally represent relatively shallow knowledge. It has always been thought, however, that WSD could also benefit from deeper knowledge sources. We describe a novel approach to WSD using inductive logic programming to learn theories from first-order logic representations that allows corpus-based evidence to be combined with any kind of background knowledge. This approach has been shown to be effective over several disambiguation tasks using a combination of deep and shallow knowledge sources. Is it important to understand the contribution of the various knowledge sources used in such a system. This paper investigates the contribution of nine knowledge sources to the performance of the disambiguation models produced for the SemEval-2007 English lexical sample task. The outcome of this analysis will assist future work on WSD in concentrating on the most useful knowledge sources.

Keywords: Word Sense Disambiguation, Knowledge Sources, Inductive Logic Programming

1. Introduction

Sense ambiguity has been recognised as one of the most important obstacles to successful language understanding since the early 1950's and many techniques have been proposed to solve the problem. Early approaches that relied on hand-coded linguistic knowledge, for example (Wilks, 1978; Small and Rieger, 1982), were difficult to scale beyond toy systems. Recent work has focused on the use of information derived from lexical resources and corpus-based techniques. These approaches have proved to be successful, particularly when used in combination with supervised machine learning (Mihalcea et al., 2004; Agirre et al., 2007). Current approaches rely on limited knowledge representation

and modeling techniques: almost all systems use attribute-value vectors to represent disambiguation instances and tend to use a small set of standard machine learning algorithms. This paradigm is suitable for exploiting information extracted from corpora like bags-of-words and collocations (which we refer to as *shallow knowledge sources*) but is less appropriate for making use of more complex forms of information such as selectional restrictions (*deep knowledge sources*). Specia et al. (2007a) presented a novel approach to WSD that combines corpus-based evidence with deeper knowledge. The approach uses Inductive Logic Programming (Muggleton, 1991) to induce theories based on examples and any type of background knowledge. Inductive Logic Programming produces disambiguation rules using a first-order model that allows deep and shallow knowledge sources to be represented. Using this learning technique and a range of (shallow and deep) knowledge sources it is possible to perform accurate WSD. An additional advantage is that the models generated are interesting for knowledge acquisition since they can convey potentially new knowledge in a format that can be easily interpreted by humans.

The approach was originally developed to disambiguate verbs in English-Portuguese Machine Translation (*translation disambiguation*) and has also been applied to a monolingual setting: the disambiguation of verbs and nouns from the Senseval-3 and SemEval-2007 lexical sample tasks (Specia et al., 2007a; Specia et al., 2007b). Promising results were reported in all cases. Nevertheless, the relative contribution of the various knowledge sources used by this WSD approach has not yet been explored. Such an analysis has potential to improve the performance of the system and is an important factor in the understanding of the WSD problem. For example, it may identify knowledge sources that have a negative impact on overall accuracy or are redundant (in the sense that another knowledge source provides the same information) and should be removed. In addition, some knowledge sources are more expensive to compute than others and may affect system efficiency without significantly improving results, while others may not be available for all languages. The analysis will also provide more general insight into which knowledge sources are most useful for WSD.

The remainder of this paper is organised as follows. Section 2 discusses previous studies comparing the usefulness of different knowledge sources for WSD. Section 3 discusses our approach to WSD and outlines how it differs from previous work. Section 4 presents the results of our experiments on the investigation of the relevance of several knowledge sources using the SemEval-2007 English lexical sample task data.

2. Related work

This section discusses the previous studies that have analysed the contribution of various knowledge sources for WSD. A more detailed description can be found in (Agirre and Stevenson, 2006). In this discussion we focus on systematic comparisons on the same data sets and algorithms, rather than on the comparison of results from independent approaches.

It is important to distinguish *knowledge sources* from *features*. Knowledge sources (KSs) are high-level abstract linguistic and semantic phenomena relevant to resolving ambiguity, for example, the domain of each word sense. On the other hand, features are ways of encoding the KSs used by actual systems. For instance, the domain of a word sense can be represented by the words co-occurring often with the word sense (bag-of-words). It is also important to differentiate the analysis of KSs from the process of *feature selection*. The latter aims to select the most relevant features either to remove the least useful, and thus improve efficiency, or to improve accuracy, for example (Mihalcea, 2002; Decadt et al., 2004). Daelemans et al. (2003) show that feature selection, together with parameter optimisation, plays an important role in the use of machine learning algorithms for NLP applications, including WSD. However, feature selection does not say much about the types of KSs that are most useful for the WSD problem in general. The selected features tend to be very specific (for example, a particular word occurring in the context) and hence vary considerably between datasets. In addition, these studies have usually been limited to the analysis of shallow features.

Ng and Lee (1996) describe an early analysis into the relative contribution of various KSs in a corpus-based approach to WSD. The KSs used by their system were collocations, topical word associations (bag-of-words), syntactic relations (verb-object), part-of-speech (POS) and morphological information. Their system used an example-based machine learning algorithm and was evaluated on a corpus containing instances of the word *interest*. KSs were compared by using each on its own. They found that local collocations were the most useful source in their system. Lee and Ng (2002) describe more comprehensive experiments using the same set of KSs. Four machine learning algorithms were compared and an evaluation was carried out against the Senseval-2 data set (Edmonds et al., 2002). The authors found little difference in the performance of each individual KS and that the combination of KSs usually performs better than any KSs individually.

Stevenson and Wilks (2001) assigned senses from the LDOCE dictionary (Procter, 1978). Their WSD approach used a wide range of

information types (including some extracted from LDOCE itself): POS and surface form of the ambiguous term, a range of collocations, word associations (computed using dictionary definitions), selectional preferences and domain codes. The KSs were applied in different ways, some were used to reduce the search space of possible senses under consideration, while others are combined using a nearest neighbor learning algorithm to generate the final output. When tested on a version of the Semcor corpus (Miller et al., 1994), the KS based on domain codes was the most successful and selectional preferences the least. The combination of KSs performed better than any applied individually.

Agírre and Martinez (2001) compared a wide range of KS: frequency of senses (first sense in WordNet (Fellbaum, 1998)), topical word association (dictionary definition overlap), paradigmatic relations (conceptual density on the WordNet taxonomy), semantic classes, selectional preferences and, finally, decision lists with n-grams and small windows of POS and lemmas, argument-head relations, subcategorisation information and bag-of-words. The various KSs were applied using a range of algorithms and used to assign senses from WordNet. The approaches were evaluated against all occurrences of eight nouns in Semcor or all polysemous nouns in a set of four random files from Semcor. They reported wide variation between the various KSs in terms of both accuracy and proportion of instances to which they were applicable.

Yarowsky and Florian (2002) also experimented with a wide range of features: local context (n-grams, small windows with raw words, lemmas and POS tags), syntactic relations depending on the POS of the ambiguous word and bag-of-words. The approach was evaluated on the Senseval-2 data. They compared all KS, all KS apart from one (*leave-one-out*) and each KS applied individually. Verbs were found to be the most difficult to disambiguate and also the most affected by the removal of syntactic features. Nouns benefited more from information about their wide context and basic collocations.

The methodology used by Yarowsky and Florian (2002) is the most closely related to ours but they only considered relatively shallow KSs. Agírre and Martinez (2001) compared a wider set of KS but only considered each in isolation. Other approaches were limited in a number of ways, such as restricting their studies to shallow KS, applying the KS in different ways or only evaluating their systems against a small set of words (normally all nouns). This paper presents a more comprehensive evaluation on a large set of nouns and verbs, taking into account both deep and shallow KSs, which are all applied using the same learning algorithm (Inductive Logic Programming).

3. A hybrid approach to WSD

A wide variety of approaches to WSD have been proposed in the literature and can be grouped into three main types. Knowledge-based approaches make use of linguistic knowledge, either coded manually or extracted from lexical resources, for example (Agirre and Rigau, 1996). Corpus-based approaches make use of shallow knowledge automatically acquired from text and learning algorithms to induce disambiguation models, for example (Yarowsky, 1995). Hybrid approaches mix characteristics from the two other approaches to automatically acquire disambiguation models by combining shallow knowledge extracted from text with linguistic knowledge, for example (Stevenson and Wilks, 2001).

Hybrid approaches can combine advantages from both strategies with the potential to produce accurate and comprehensive systems, particularly when deep knowledge is explored. A vast amount of linguistic knowledge is available from resources and tools such as WordNet, dictionaries and parsers. However, the use of this information has been hampered by the limitations of the standard modeling techniques: using deep sources of domain knowledge is beyond the capabilities of such techniques, which are generally based on attribute-value vector representations.

In attribute-value vectors each attribute has a type (its name) and a single value for each example. Therefore, attribute-value vectors have the same expressiveness as propositional formalisms: they only allow the representation of atomic propositions and constants. With first-order logic, a more expressive formalism which is employed by Inductive Logic Programming, it is possible to represent both variables and n-ary predicates. This allows relational knowledge to be expressed naturally.

In the hybrid approaches that have been explored to date, deep knowledge is either pre-processed into an attribute-value vector representation to accommodate the use of machine learning algorithms, or used in previous steps to filter out possible senses, for example (Stevenson and Wilks, 2001). This may cause information to be lost. For example, one of the KSs used in our approach is the set of all possible selectional restrictions of a verb for a given sense, expressed in terms of the semantic features required by its arguments. The verb *ask* in the sense of "inquire, seek an answer" requires a human subject and an abstract object or a human subject, a human direct object and an abstract indirect object. Other senses of the verb may have different or sometimes the same restrictions. It is not possible to directly represent this information by means of attribute-value vectors: the number of alternative restrictions varies from sense to sense. Moreover, it is difficult to represent the fact that arguments with different combinations of fea-

tures can satisfy the selectional restrictions of the same sense. One way to try to express such information with attribute-value representations is to create one attribute for each possible sense of the verb and have a true/false value assigned to it depending on whether the arguments of the verb satisfies any restrictions referring to that sense. However, this means that information is partially lost: it is not possible to retrieve, for example, which are the actual semantic features of the arguments. As a consequence, the models produced reflect only the shallow knowledge that is provided to the learning algorithm.

Another limitation of attribute-value vectors is the need for a unique representation for all the examples: one attribute is created for every feature and the same structure is used to characterise all the examples. This usually results in a very sparse representation of the data, given that values for certain features will not be available for many examples. Data sparseness increases as more knowledge is exploited and this can cause problems for machine learning algorithms.

A final disadvantage of attribute-value vectors is that equivalent features may have to be bounded to distinct identifiers. An example of this occurs when the syntactic relations between words in a sentence are represented by attributes for each possible relation. Sentences in which there is more than one instantiation for a particular grammatical role cannot be easily represented. For example, in the sentence "*The company was forced to sell stocks and shares.*" the direct object of *sell* is a conjunction and, since each feature requires a unique identifier, two are required:

$$\text{obj}_1\text{-verb}_1 = \text{stocks} \qquad \text{obj}_2\text{-verb}_1 = \text{shares}$$

These attributes would be treated as two independent pieces of knowledge by the learning algorithm. First-order formalisms, on the other hand, allow a generic predicate to be created for every possible syntactic role relating two or more elements. For example *has_object(verb, object)*, which could then have two instantiations: *has_object(sell, stocks)* and *has_object(sell, shares)*.

Inductive Logic Programming provides a general-purpose framework for dealing with such problems: it makes explicit provisions for the inclusion of background knowledge of any form and the representation language is powerful enough to capture contextual relationships. In what follows we provide an introduction to Inductive Logic Programming and then outline the KSs used in our experiments.

3.1. Inductive Logic Programming

Inductive Logic Programming (ILP) (Muggleton, 1991) employs techniques from Machine Learning and Logic Programming. From Machine

Learning come methods and tools for inducing hypotheses from examples and synthesising new knowledge from experience. From Logic Programming comes the representation formalism, which is based on first-order logic, with its well defined semantic orientation and techniques. These are combined to build first-order theories from examples and background knowledge, also represented by first-order clauses. ILP allows the efficient representation of substantial knowledge about the problem and produces disambiguation models that can make use of this knowledge. The general approach underlying ILP can be outlined as follows.

Given:

- A finite set of examples $E = E^+ \cup E^-$ where:
  - *Positive Examples*: $E^+ = \{e_1, e_2, \ldots\}$ is a non-empty set of definite clauses[1], usually expressed as ground facts (i.e. without variables).
  - *Negative Examples*: $E^- = \{\overline{f_1}, \overline{f_2} \ldots\}$ is a set of Horn clauses[2] (this may be empty).
- Background knowledge B consisting of a finite set of extensional (ground) or intentional (with variables) clauses $= \{C_1, C_2, \ldots\}$

The goal is: to induce a hypothesis (or theory) H, with relation to E and B, which covers all the E^+, without covering the E^-, that is, a theory that is a generalisation of the positive examples. These restrictions are very strict, in practice, a theory is acceptable if the following conditions are met (Muggleton, 1994)[3]:

- *Prior Satisfiability:* B and E^- are satisfiable[4], that is, $B \wedge E^- \not\models \square$
- *Posterior Satisfiability:* B and H and E^- are satisfiable, that is, $B \wedge H \wedge E^- \not\models \square$
- *Prior Necessity:* The background knowledge complements the examples, that is, $B \not\models E^+$
- *Posterior Sufficiency:* B and H logically imply all examples, that is, $B \wedge H \models E^+$

[1] Definite clauses are first-order clauses containing one positive literal.

[2] Horn clauses are first-order clauses that can contain at most one positive literal. A Horn clause with exactly one positive literal is a definite clause.

[3] Where $\wedge$ represents *logical and*, $\models$ *logically proves* and $\square$ *falsity*.

[4] A clause is satisfiable if there exists at least one model for it, i.e., there exists one interpretation (a set of ground facts) that assigns a true value for such clause.

The examples E, background knowledge B and induced hypothesis H are logical programs. The induction process to find H can be seen as a search problem in a space of hypotheses. In general, H also needs to satisfy constraints specified by the user to restrict or bias the search space. These constraints can be defined on the structure or semantics of the clauses, specify a stop criterion or optimise the search process. For example, most ILP systems require the specification of a predicate p defining the target relation to be learned, i.e., which will appear in the head of the clauses in H, and a number of predicates $q_1, \ldots, q_n$ defining which knowledge sources can appear in the body of these clauses. It is also possible to specify in which way these predicates can be used, for example, the number of instantiations in a clause, whether intentional definitions are permitted and the variables used for input and output.

Different approaches can be used to structure the search space. These are usually grouped in *generalisation approaches*, which start the search from the examples (most specific hypotheses) and generalise them by means of generalisation operators, and *specialisation approaches*, which start the search from the descriptions of the most general concepts and specialise such concepts by using specialisation operators. In general, ILP systems use both generalisation and specialisation operators in different steps of the search process.

Structuring the search space consists of sorting the hypotheses according to some strategy. In general, sorting strategies are based in the *θ-subsumption* relation (Muggleton and Raedt, 1994). A clause c_1 *θ-subsumes* a clause c_2 if and only if there exists a substitution θ such that $c_1 \subseteq c_2$, that is, c_1 is a generalisation of c_2 and c_2 is a specialisation of c_1 under *θ-subsumption*. A substitution $\theta = \{V_1/t_1, \ldots, V_n/t_n\}$ consists in assigning terms t_i to variables V_i.

We use the Aleph ILP system (Srinivasan, 1999), which provides a complete inference engine and can be customised in various ways. The default inference engine induces a theory iteratively, in a bottom-up and batch (non-incremental) mode, until it finds clauses that explain all the examples. We say that a clause *covers* an example if the example satisfies all the conditions in the body of the clause and has the same head as the clause.

Aleph uses the following steps:

1. A positive example (seed) is selected to be generalised, using the order of the examples in the training data.

2. The bottom clause, a more specific clause covering that example, is built using inverse entailment (Muggleton, 1995). The bottom clause generally represents all knowledge about the example within any constraints provided to the learning algorithm, for example

which (and how) knowledge sources can be used as part of the body and the maximum number of literals in a clause. The bottom clause usually contains many literals and covers only the seed example; it is the most specific clause that covers that example. This step is often referred to as *saturation*.

3. A search is carried out for a clause that is more general than the bottom clause. The goal is to find a consistent generalisation that is more compact in the lattice of clauses that subsume the bottom clause within the constraints provided (structure, semantics, etc). This generalisation must cover both the saturated example and other positive examples. This can be done, for example, by removing literals from the bottom clause or replacing terms in literals by variables. The search for the best clause is performed using pre-defined search (e.g. best-first) and evaluation strategies (e.g. number of positive examples covered). This step is usually called *reduction*.

4. The best clause is added to the theory and the examples covered by that clause are removed from the training set. This process stops if there are no more examples in the training set, otherwise returns to step 1.

3.2. Knowledge sources

Our system uses a range of nine deep and shallow KS to disambiguate verbs and nouns. These are illustrated using the following example sentence (snt_1) *"If there is such a thing as reincarnation, I would not mind coming back as a squirrel."* in which the verb "coming" is the word being disambiguated. For this example the correct sense in OntoNotes (Hovy et al., 2006) is "1" = *move, travel, arrive.*[5]

KS_1. Topical word associations The sense with the highest count of **overlapping words** (excluding stop words) in its definition in the LDOCE dictionary (Procter, 1978) and in the sentence containing the target word. The mapping between senses in LDOCE and OntoNotes was performed using WordNet senses as intermediate (a mapping between WordNet and LDOCE senses has been previously compiled and OntoNotes provides a mapping into WordNet senses). These are represented by *has_overlapping(snt, sense)*. In the following example, sense 1 has the highest overlapping count in sentence snt_1:

[5] See Section 4.1 for a discussion of OntoNote's treatment of phrasal verbs such as "come back".

has_overlapping(snt$_1$, 1).

KS$_2$. Topical word associations Represented using a **bag-of-words** consisting of five words to the right and left of the target word (excluding stop words). These are represented using definitions of the form *has_bag(snt, word)*:

has_bag(snt$_1$, mind).

has_bag(snt$_1$, not).

KS$_3$. Collocations A range of collocations including the ambiguous word as defined by Stevenson and Wilks (2001): first preposition to the left and right, first and second words to the left and right, first noun, adjective and verb to the left and right. These are represented using definitions of the form *has_collocation(snt, type, collocation)*:

has_collocation(snt$_1$, 1st_prep_right, back).

has_collocation(snt$_1$, 1st_noun_left, mind).

KS$_4$. Syntactic relations Minipar (Lin, 1993) is used to identify syntactic relations in which the ambiguous word participates. For verbs constituents in the subject and object roles are identified. For nouns the verb it is governed by is identified, together with any noun or verb it modifies. These are represented by *has_rel(snt, type, word)*:

has_rel(snt$_1$, subject, i).

has_rel(snt$_1$, object, nil).

KS$_5$. POS tags Mxpost (Ratnaparkhi, 1996) is used to find the part of speech tags of the five words to the left and right of the ambiguous word. These are represented by *has_pos(snt, word_position, pos)*:

has_pos(snt$_1$, 1st_word_left, nn).

has_pos(snt$_1$, 1st_word_right, rb).

KS$_6$. Topical word associations Represented by **frequent bigrams** consisting of pairs of adjacent words in a sentence (other than the ambiguous word) which occur more than ten times in the training corpus. These are represented using definitions of the form *has_bigram(snt, word$_1$, word$_2$)*:

has_bigram(snt$_1$, back, as).

has_bigram(snt$_1$, such, a).

KS$_7$. Content word collocations The five content words to the left and right of the ambiguous word, identified using POS tags. These are represented by *has_narrow(snt, word_position, word)*:

has_narrow(snt$_1$, 1st_word_left, mind).

has_narrow(snt$_1$, 1st_word_right, back).

KS$_8$. Phrasal verbs For each ambiguous verb a list of its phrasal forms was extracted from various dictionaries (LDOCE, WordNet etc.) and simple heuristics were used to check whether these phrasal forms occurred in the sentences containing the ambiguous verb. The heuristics are based on pattern matching that allow for words between the verb and its particle for separable phrasal verbs. It is important to note that the occurrence of a verb followed by a particle in the sentence does not always indicate a phrasal expression, since particles can also be used as prepositions (e.g. in *come in red*, 'come in' is not a phrasal verb). Additionally, phrasal verbs themselves are not necessarily unambiguous ('come in' has five senses as a verb in WordNet). The potential occurrence of phrasal verbs in a sentence is represented by definitions of the form *has_expression(snt, verbal_expression)*:

has_expression(snt$_1$, 'come back').

KS$_9$. Selectional restrictions The selectional restrictions for each sense of an ambiguous verb, defined in terms of the semantic features required by its arguments (nouns), are extracted from LDOCE (Procter, 1978). In LDOCE each sense of a noun is labelled with codes from a set of 35 semantic categories. For example, the second sense of *reincarnation* is labelled with the category `abstract`, while the first sense of *squirrel* is labelled `animal`. Each verb sense also lists the semantic features required by its arguments (subject and object). For example, the sense "*to move towards the speaker or a particular place*" of the verb *come* requires a subject with the feature `animal` or `human` and no object (since this sense is an intransitive usage). LDOCE senses are mapped into OntoNotes senses via WordNet using the same mapping employed for KS$_1$.

Two mechanisms are used to increase the coverage of this knowledge source. Firstly, a hierarchy of feature types (Bruce and Guthrie,

1992) is used to account for restrictions established by the verb that are more generic than the features describing its arguments in the sentence, for example a noun labelled with the feature `human` would satisfy the restriction `animate`. Additionally, if the restrictions for a particular sense of a verb are not satisfied then synonyms and hypernyms taken from WordNet can be used instead. For example, if the verb sense requires an `abstract` subject, but the subject in the sentence does not have this feature or cannot be found in LDOCE, we look for a synonym in WordNet that contains such feature, like *rebirth* for *reincarnation*.

Selectional restrictions are represented by definitions of the form *satisfy_restriction(snt, rest_subject, rest_object)*, for example:

```
satisfy_restriction(snt1, [human], nil).
satisfy_restriction(snt1, [animal], nil).
```

These examples indicate that in sentence snt_1 the ambiguous verb, *come*, imposes two sets of selectional restrictions: one where the subject satisfies the restriction `human` and there is no object, another where the subject satisfies `animal` and also without an object. These restrictions may refer to the same or different senses but this information is not relevant.

Note that KS_8 and KS_9 can only be used when the ambiguous word is a verb. The KSs vary from superficial (like topical word associations in the form of bag-of-words) to deep (like selectional restrictions).

In addition to this background knowledge, the system learns from a set of examples. In the case of the lexical sample tasks, in which a single word per sentence is to be disambiguated, these are represented using predicates containing the sentence identifier and the sense of the ambiguous word in that sentence, e.g. *sense(snt, sense)*:

```
sense(snt1, 1).
sense(snt2, 3).
```

3.3. Creating WSD Models

The ILP system Aleph is provided with the examples and background knowledge together with definitions of the predicate that can form the conditional part of the rule. This information is used in a number of iterations involving steps 1-4 (as described in Section 3.1). Assuming that sentence snt_1 (Section 3.2) is the first training example, the first iteration of the learning process could proceed as follows:

1. The first positive example (seed) is selected to be generalised:

```
sense(snt1, 1).
```

2. The bottom clause with all possible instantiaitons of KSs covering the seed example is built (saturation):

```
sense(A, 1) :-  has_overlapping(A, 1),
                has_bag(A, not),
                has_bag(A, mind),
                has_bag(A, back), ...
                has_collocation(A, 1st_prep_right, back),
                has_collocation(A, 1st_noun_left, mind),
                ...
                has_rel(A, subject, i),
                has_rel(A, object, nil),
                has_pos(A, 1st_word_left, nn),
                has_pos(A, 1st_word_right, rb), ...
                has_bigram(A, back, as),
                has_bigram(A, such, a), ...
                has_narrow(A, 1st_word_left, mind),
                has_narrow(A, 1st_word_right, back),
                ...
                has_expression(A, 'come back'),
                satisfy_restriction(A, [human], nil),
                satisfy_restriction(A, [animal], nil),
                ...
```

3. Generalisations of the bottom clause are searched (reduction):

```
sense(A, 1) :-  satisfy_restriction(A, B, nil).
sense(A, 1) :-  satisfy_restriction(A, [animate], nil),
                has_narrow(A, 1st_word_right, back).
...
```

4. The best clause found is added to the theory and all examples covered by it are removed from the training set. Returns to step 1.

```
sense(A, 1) :-  satisfy_restriction(A, [animate], nil),
                has_narrow(A, 1st_word_right, back).
```

After a number of iterations to cover all training examples, the result is a set of symbolic rules. Figure 1 shows an example rule induced for the verb "come". This rule states that the sense of the verb in a sentence `A` will be "1" (*move, travel, arrive*) if the subject of the verb has the feature `animate` and there is no object, or if the verb has a subject `B` that occurs in a position, `C`, as either a proper noun (`nnp`) or a personal pronoun (`prp`). Note that a rule such as this contains complex combinations of KSs that would be difficult to learn from standard attribute-value vectors.

```
sense(A, 1) :-  satisfy_restriction(A, [animate], nil);
                (has_rel(A, subj, B),
                (has_collocation(A, C, B),
                (has_pos(A, C, nnp); has_pos(A, C, prp))).
```

Figure 1. Example rule learned for "come"

Aleph generates an ordered list of rules that are applied in sequence. The first rule to match an instance is used to identify its sense.

3.4. Performance

This approach to WSD has been evaluated on a variety of mono- and multilingual scenarios. It was originally developed to identify the correct translation of verbs in an English-Portuguese Machine Translation system. Specia et al. (2007a) describe a corpus containing examples of ten frequently occurring English verbs which are difficult to translate to Portuguese. This approach correctly disambiguated 74% of examples. Results using Aleph as the learning algorithm were significantly better than using a simplified representation of the same KS within attribute-value vectors and three widely used learning algorithms (decision lists, Naive Bayes and Support Vector Machine). In the official SemEval-2007 evaluation, the system came in fifth out of 15 participants with a performance three points below the best reported system (Specia et al., 2007b). These results are very promising, considering that many of the KSs available were designed for verb disambiguation while SemEval also includes ambiguous nouns.

4. Experiments

Data from SemEval-2007 English lexical sample task was used to assess the performance of the KSs used in our approach. This contains examples of 65 verbs and 35 nouns taken from the WSJ Penn Treebank II and Brown corpora. There is an average of 222 examples for training and 49 for test per target word, although there is a large variation: the lexical item with the fewest has 19 training examples and 2 for testing while the item with the most has 2,536 examples for training and 541 for testing. The examples were annotated with senses from OntoNotes, which are formed from groupings of WordNet senses and are therefore more coarse-grained. There are an average of 3.6 possible senses for each ambiguous word. Further details about the task and dataset can be found in (Pradhan et al., 2007).

We produced three sets of models for each target word by varying the types of KSs made available to the inference engine:

1. All KSs (KS_1 to KS_9 for verbs and KS_1 to KS_7 for nouns), resulting in a single model for each target word.

2. Each KSs individually, resulting in nine models for each verb and seven for each noun (KS_8 and KS_9 are not available for nouns).

3. All KSs apart from one (*leave-one-out*), also resulting in nine models for each verb and seven for each noun.

We optimise a small set of relevant parameters in Aleph using 3-fold cross-validation on the training data for each of the possible combinations of KSs. The models were evaluated by testing each on the corresponding set of test cases by applying the rules in a decision-list like approach, i.e., retaining the order in which they were produced and backing off to the most frequent sense in the training set to classify cases that were not covered by any of the rules. Each models is evaluated in terms of average accuracy (correctly classified examples divided by number of examples) as computed by the *scorer* program provided by the SemEval organisers.

Results of the experiments are shown in Table I for nouns, verbs and all words together. Results from a baseline system which classifies all instances of a term with the most frequent sense are shown in the first row. The next row shows the accuracy of the best WSD system participating in SemEval-2007. The third row shows the accuracy of the ILP models created using all available KS.

Results in Table I can be analysed from three main perspectives:

Table I. Accuracies of different combinations of KSs

System	**All Words**	**Verbs**	**Nouns**
Baseline (most frequent sense)	0.780	0.762	0.809
Top performing system in SemEval	0.887	-	-
All KSs	0.851	0.817	0.882
Single Knowledge Source			
Overlap of definitions (KS_1)	0.778	0.754	0.800
Bag-of-words (KS_2)	0.813	0.776	0.845
Collocations (KS_3)	0.846	0.812	0.877
Subject-object relations (KS_4)	0.796	0.771	0.819
POS (KS_5)	0.819	0.789	0.845
Bigrams (KS_6)	0.802	0.764	0.835
Content words (KS_7)	0.810	0.774	0.842
Phrasal verbs (KS_8)	-	0.776	-
Selectional restrictions (KS_9)	-	0.760	-
Leave-One-Out			
All KSs - overlap of definitions (KS_1)	0.843	0.809	0.874
All KSs - bag-of-words (KS_2)	0.830	0.801	0.856
All KSs - collocations (KS_3)	0.824	0.794	0.852
All KSs - subject-object relations (KS_4)	0.830	0.799	0.857
All KSs - POS (KS_5)	0.830	0.797	0.860
All KSs - bigrams (KS_6)	0.835	0.799	0.867
All KSs - content words (KS_7)	0.830	0.802	0.855
All KSs - phrasal verbs (KS_8)	-	0.795	-
All KSs - selectional restrictions (KS_9)	-	0.809	-

1. Comparison of each individual KS's performance with the most frequent sense baseline.

2. Comparison of each individual KS's performance with the performance of the models generated using all KS.

3. Comparison of models produced when one KS is removed (leave-one-out) with those generated using all KS.

When used individually, the majority of KSs perform better than the baseline, particularly collocations (KS_3), which show an improvement of 0.066 over all words. These results are consistent with previous studies that have shown that collocations and other forms local context are very useful for WSD, particularly for nouns, for example (Yarowsky and Florian, 2002). Some KSs do not perform as well. Word overlap (KS_1) actually performs worse than the baseline. Similarly, the performance of selectional restrictions (KS_9) is below the baseline for verbs. We believe the poor performance of selectional restrictions may be due to the noise added by the multiple mappings between senses (from OntoNotes to WordNet and then from WordNet to LDOCE) that were necessary to extract this KS. Another reason may be that the coarse grained sense distinctions in OntoNotes prevent Aleph from taking full advantage of this complex KS.

Item 2 refers to the comparison of accuracy of each individual KS with respect to the combination of all KSs. When used alone, the performance of each KS is significantly worse (paired t-test, $p < 0.05$) than the combination of all KSs. KS_1 (word overlap) was the worst performing. KS_9 (selectional restrictions) also performed badly for verbs and KS_4 (subject-object relations) for nouns. KS_3 (collocations) achieved accuracy within 0.005 of the combination of all KSs, again highlighting the usefulness of this KS for WSD.

To gain further insight into the contribution of the deep KSs we generated models for verbs using the best KS (KS_3) and two deep KSs (KS_8 and KS_9). This led to a slight, although not statistically significant, improvement in accuracy from 0.812 (using KS_3 alone) to 0.813. This result demonstrates that the deep KS can improve WSD accuracy when combined with a shallow KS, albeit marginally, and suggests that they encode different information which can be exploited by the ILP approach.

It is important to note that items 1 and 2 refer to the behaviour of KSs when applied individually, but it has been shown that performance improves when KS are combined (see Section 2). We believe this is particularly relevant to our approach in which there is a strong interaction amongst different KSs. Table II shows the reduction in performance when using the leave-one-out strategy for each KS. A first observation is that removing a KS always reduces significantly the performance (paired t-test, $p < 0.05$), both for nouns and verbs. The leave-one-out strategy results in an accuracy of around 0.83 for the majority of KS, approximately 0.02 lower than when all are used. KS_1 is an exception since performance drops by a lower amount (0.008). Interestingly, although the performance of this KS is worse than the baseline when it is used alone, removing it still reduces accuracy. Similarly, for verbs

Table II. Drop in accuracy of each leave-one-out combination compared to the accuracy of all KSs together

	All Words	**Verbs**	**Nouns**
All KSs - overlap of definitions (KS_1)	-0.008	-0.008	-0.008
All KSs - bag-of-words (KS_2)	-0.021	-0.016	-0.026
All KSs - collocations (KS_3)	-0.027	-0.023	-0.030
All KSs - subject-object relations (KS_4)	-0.021	-0.018	-0.025
All KSs - POS (KS_5)	-0.021	-0.020	-0.022
All KSs - bigrams (KS_6)	-0.016	-0.018	-0.015
All KSs - content words (KS_7)	-0.021	-0.015	-0.027
All KSs - phrasal verbs (KS_8)	-	-0.022	-
All KSs - selectional restrictions (KS_9)	-	-0.008	-

removing the two KSs whose individual performance is lower than the baseline (KS_1 and KS_9) leads to some reduction in performance compared to using all KSs. On the other hand, removing phrasal verbs (KS_8) results in a considerable drop in accuracy, the second largest for verbs after collocations (KS_3). For nouns, the KSs resulting in larger drops in accuracy when removed are both sources of collocations (KS_3 and KS_7), followed by bag-of-words (KS_2) and syntactic relations (KS_4).

The analysis presented so far describes the effects of removing selected KSs but does not tell us how often individual KS are used by the system, if at all. To investigate whether the KSs appear in the models produced for each target word we counted the number of times each KS appeared in the rules, either as the only one in a rule or in combination with others. The fact that the KSs appear in rules does not necessarily mean it is useful to classify new cases. Therefore, we also determined the KSs which are actually used to classify the test cases. The proportion of rules in which a given KS appears and the proportion of times a rule including a given KS is applied is shown in Table III. For each KS figures are shown over all words as well as individually for verbs and nouns.

The figures in Table III show that the use of a KS is directly proportional to the number of times it appears in the rules. The fact that a KS appears very few times in the rules may indicate that it is not particularly discriminative (KS_1 and KS_7) or may be simply due to the sparsity of the data (KS_8 and KS_9). These very sparse KSs appear and

Table III. Proportions of times KSs appear in the rules and are used to classify test cases

KS	Rules including the KS (%)			Rules using the KS (%)		
	All words	Verbs	Nouns	All words	Verbs	Nouns
$\mathbf{KS}_1$	3.97	5.68	2.63	5.09	3.49	6.53
$\mathbf{KS}_2$	23.70	18.89	27.45	14.88	8.07	20.98
$\mathbf{KS}_3$	50.92	52.67	49.55	52.90	55.10	50.92
$\mathbf{KS}_4$	5.45	11.41	0.81	7.03	14.05	0.74
$\mathbf{KS}_5$	69.99	66.65	72.59	55.99	49.21	61.98
$\mathbf{KS}_6$	13.48	3.88	20.95	13.05	2.71	22.24
$\mathbf{KS}_7$	0.98	0.87	1.06	0.37	0.09	0.63
$\mathbf{KS}_8$	-	2.02	-	-	3.01	-
$\mathbf{KS}_9$	-	5.02	-	-	12.04	-

are used mostly individually, that is, not in combination with other KSs. All the remaining KSs appear and are used in combination with other KSs in above 80% of the cases.

Is is interesting to note that deep KSs like phrasal verbs (KS_8) contribute to improving performance, even with very few instantiations in the training and test data. KSs that can be extracted from all sentences, bag-of-words (KS_2), collocations (KS_3) and POS tags (KS_5), appear and are used in many more rules than other KSs. Table III also shows that, of those KSs that can be applied to both nouns and verbs, some are more useful for verbs (subject-object relations, collocations), while others are more useful for nouns (bag-of-words, POS tags, bigrams, overlap of definitions). In general, shallow sources are more useful for disambiguation of nouns. Verbs, on the other hand, benefit from more elaborate knowledge sources.

4.1. Discussion

Our analysis showed that KSs using collocations are highly discriminative, both individually and in combination with other KSs. This is particularly true for the disambiguation of nouns. Collocations are a shallow KS that require only information from a POS tagger to be extracted. However, as we showed in Figure 1, the way they are represented and manipulated by the inference engine in our approach makes it possible for this KS to interact with others in a complex way

that would not be possible in attribute-value based approaches. For example, sometimes it is not necessary to know which word is in a certain position in the sentence relative to the ambiguous word (that is, the collocation itself) but only that this word is in a given syntactic relation with the words (such as its subject) or has a certain POS tag (as shown in Figure 1). This flexibility allows collocations to play a different role in our approach to the way they are usually applied in WSD.

Subject-object syntactic relations, which can be considered as an intermediate KS, performed relatively poorly in isolation. This is not surprising considering the limited coverage of this KS and the fact that some of the relations generated by a parser may be incorrect. However, they prove to be very helpful for both disambiguation of nouns and verbs when combined with other KS. This suggests that syntactic relations are a useful KS but without enough coverage to be used alone.

Different results were obtained from the two deepest KS in our experiments: selectional restrictions and phrasal verbs. Performance of the first was lower than the most frequent sense baseline when used in isolation and leaving it out only reduced the system's performance by a small amount compared to other KSs. It is likely that the disappointing performance in this experiment is due to the fact that only coarse-grained distinctions are considered in the dataset, while the selectional restrictions are defined in terms of the finer-grained distinctions used by LDOCE. It is difficult to discriminate between coarse-grained sense distinctions for verbs since a single sense may allow more than one syntactic frame (e.g. both transitive and intransitive usage). In addition, the verb's arguments may be shared by other senses, making them difficult to distinguish. Consequently, features that are more superficial than selectional restrictions may perform better. The mapping between LDOCE and OntoNotes senses used by this KS may also have been a problem. LDOCE senses are mapped to OntoNotes via WordNet but this mapping is not comprehensive: there are senses in LDOCE that are not mapped onto WordNet and, consequently, there is no mapping to OntoNotes. Likewise, there are OntoNotes senses with no WordNet mapping. Therefore, the coverage of this KS in this dataset is limited. Specia et al. (2007a) found selectional restrictions to be very discriminative for the disambiguation of verbs in experiments using Senseval-3 data and a translation task in which the sense distinctions were very fine-grained (WordNet senses and translations of the verbs).

The other deep KS, phrasal verbs, performed well despite the fact that OntoNotes defines only a few senses which are specific to occurrences of the verb in phrasal expressions. For example, the verb "come"

has 11 possible senses in OntoNotes. In this resource, sense "1" (*move, travel, arrive*) also covers several phrasal verbs that would have been assigned different senses in other repositories like WordNet: *come away, come back, come by, come down, come forward, come in, come near, come on, come out, come through, come together, come up, come up to*. Some of these phrasal verbs are also represented in other senses in OntoNotes. For example, "come out" is also in senses "2", "3", "6", "9", and, more importantly, sense "10", which is defined simply as "idioms". This last sense is common to many verbs in OntoNotes: most occurrences of the verb in phrasal expressions are simply grouped and tagged with the sense label "idioms" despite the fact they do not share the same meaning. We believe the performance of this KS could be further improved with access to lexical resources with more appropriate information about phrasal verbs.

5. Conclusion

We investigated the use of ILP as a mechanism for incorporating a mixture of shallow and deep knowledge sources into the construction of WSD models. Evaluation was carried out using data from the SemEval-2007 lexical sample task. Using a combination of nine KSs consistently outperformed the most frequent sense baseline. In this paper the approach was investigated further by analysing the performance of each KS individually and in combination with other KSs.

The combination of all KSs always performs better than each KS individually and better than any of the leave-one-out combinations (all KSs apart from one). We also showed that most of the KSs outperform the baseline when used individually and, even those which do not, improve performance when combined with others. We believe this is due to the relational nature of the representation formalism and modeling technique, which allows complex interactions among different KSs.

Our experiments also show that the most relevant KSs for disambiguation differ for nouns and verbs, as others have shown (e.g. (Stevenson and Wilks, 2001)). However, these are broad grammatical categories and further work is required to understand the importance of various KSs for individual tokens.

The approach described in this paper demonstrates how deep and shallow KS for WSD can be combined using ILP. We found that the inclusion of deep KSs improved the accuracy of our WSD system although this improvement was only marginal in comparison to using only collocations, a shallow KS. However, the use of deep KS have received less attention than shallow ones within machine learning approaches to

WSD and it is possible that refinements to them could yield further improvements.

Acknowledgments

We are grateful for the feedback provided by the anonymous reviewers of this paper. Mark Stevenson was supported by the UK Engineering and Physical Sciences Research Council (grants EP/E004350/1 and EP/D069548/1).

References

Agirre, E. and Martínez, D. Knowledge Sources for Word Sense Disambiguation. *Proceedings of the 4th International Conference on Text Speech and Dialogue (TSD)*, Plzen, 1–10, 2001.

Agirre, E., Marquez, L. and Wicentowski, R. *4th International Workshop on Semantic Evaluations (SemEval-07)*, 48–53, Prague, 2007.

Agirre, E. and Rigau, G. Word Sense Disambiguation using Conceptual Density. *Proceedings of the 15th Conference on Computational Linguistics (COLING-96)*, 16–22, Copenhagen, 1996.

Agirre, E. and Stevenson, M. Knowledge Sources for Word Sense Disambiguation. E. Agirre and P. Edmonds (eds), *Word Sense Disambiguation: Algorithms, Applications and Trends*, Springer, 2006.

Bruce, R. and Guthrie, L. Genus disambiguation: A study in weighted performance. *14th Conference on Computational Linguistics (COLING-92)*, Nantes, 1187–1191, 57–60, 1992.

Daelemans, W., Hoste, V., Meulder, F. and Naudts, B. Combined optimization of feature selection and algorithm parameter interaction in machine learning of language. *Proceedings of the 14th European Conference on Machine Learning (ECML-03)*, Croatia, 84–95, 2003.

Decadt, B., Hoste, V., Daelemans, W., and van den Bosch, A. GAMBL, Genetic Algorithm Optimization of Memory-Based WSD. *Senseval-3: 3rd International Workshop on the Evaluation of Systems for the Semantic Analysis of Text*, Barcelona, 108–112, 2004.

Edmonds, P., Mihalcea, R., Saint-Dizier, P. *Proceedings of the Workshop Word Sense Disambiguation: Recent Successes and Future Directions*, Philadelphia, 2002.

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

Hovy, E.H., Marcus, M., Palmer, M., Pradhan, S., Ramshaw, L., and Weischedel, R. OntoNotes: The 90% Solution. *Human Language Technology / North American Association of Computational Linguistics conference (HLT-NAACL 06)*, New York, 57–60, 2006.

Lee, Y.K. and Ng, H.T. An empirical evaluation of knowledge sources and learning algorithms for word sense disambiguation. *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 41–48, Philadelphia, 2002.

Lin, D. Principle based parsing without overgeneration. *Proceedings of the 31st Meeting of the Association for Computational Linguistics (ACL-93)*, 112–120, Columbus, 1993.

Mihalcea, R.F. Word sense disambiguation with pattern learning and automatic feature selection. *Natural Language Engineering*, 8(4):343–358, Cambridge University Press, 2002.

Mihalcea, R.F., Chklovski, T. and Kilgariff, A. The SENSEVAL-3 English Lexical Sample Task. *SENSEVAL-3: 3rd Int. Workshop on the Evaluation of Systems for Semantic Analysis of Text*, 25–28, 2004.

Miller, G.A., Chorodow, M., Landes, S., Leacock, C, Thomas, R.G. Using a Semantic Concordancer for Sense Identification. *ARPA Human Language Technology Workshop*, 240–243, Washington, 1994.

Muggleton, S. Inductive Logic Programming. *New Generation Computing*, 8(4):29-5-318, 1991.

Muggleton, S. Inductive Logic Programming: derivations, successes and shortcomings. *SIGART Bulletin*, 5(1):5–11, 1994.

Muggleton, S. Inverse Entailment and Progol. *New Generation Computing*, 13:245–286, 1995.

Muggleton, S. and Raedt, L. D. Inductive logic programming: Theory and methods. *Journal of Logic Programming*, 19(20):629–679, 1994.

Ng, H.T. and Lee, H.B. Integrating multiple knowledge sources to disambiguate word sense: An exemplar-based approach. *Proceedings of the 34th Meeting of the Association for Computational Linguistics (ACL-96)*, 40–47, Santa Cruz, 1996.

Pradhan, S., Loper, E., Dligach, D. and Palmer, M. SemEval-2007 Task-17: English Lexical Sample, SRL and All Words *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-07)*, 87–92, Prague, 2007.

Procter, P. (editor). *Longman Dictionary of Contemporary English*, Longman Group, Essex, 1978.

Ratnaparkhi, A. A Maximum Entropy Part-Of-Speech Tagger. *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 133–142, New Jersey, 1996.

Small, S. and Rieger, C. Parsing and Comprehending with Word Experts (a Theory and its Realisation) Lehnert, W. and Ringle, M. (eds), *Strategies for Natural Language Processing*, 1982.

Specia, L., Nunes, M.G.V., Stevenson, M. Learning Expressive Models for Word Sense Disambiguation. *45th Annual Meeting of the Association for Computational Linguistics (ACL-07)*. 41–148, Prague, 2007.

Specia, L., Nunes, M.G.V., Srinivasan, A., Ramakrishnan, G. USP-IBM-1 and USP-IBM-2: The ILP-based Systems for Lexical Sample WSD in SemEval-2007. *Proceedings of the 4th International Workshop on Semantic Evaluations (SemEval-07)*, 442–445, Prague, 2007.

Srinivasan, A. *The Aleph Manual*. Computing Laboratory, Oxford University, 1999.

Stevenson, M. and Wilks, Y. The Interaction of Knowledge Sources in Word Sense Disambiguation. *Computational Linguistics*, 27(3):321–349, 2001.

Wilks, Y. Making Preferences More Active. *Artificial Intelligence*, 11(3):197-223, 1978.

Yarowsky, D. Unsupervised Word-Sense Disambiguation Rivaling Supervised Methods. *Proceedings of the 33rd Meeting of the Association for Computational Linguistics (ACL-95)*, 189–196, Cambridge, 1995.

Yarowsky, D. and Florian, R. Evaluating sense disambiguation across diverse parameter spaces. *Natural Language Engineering*, 8(2):293–310, 2002.