Combining EM Training and the MDL Principle for an Automatic Verb Classification incorporating Selectional Preferences

Sabine Schulte im Walde, Christian Hying, Christian Scheible, Helmut Schmid
Institute for Natural Language Processing
University of Stuttgart, Germany
{schulte,hyingcn,scheibcn,schmid}@ims.uni-stuttgart.de

Abstract

This paper presents an innovative, complex approach to semantic verb classification that relies on selectional preferences as verb properties. The probabilistic verb class model underlying the semantic classes is trained by a combination of the EM algorithm and the MDL principle, providing soft clusters with two dimensions (verb senses and subcategorisation frames with selectional preferences) as a result. A language-model-based evaluation shows that after 10 training iterations the verb class model results are above the baseline results.

1 Introduction

In recent years, the computational linguistics community has developed an impressive number of semantic verb classifications, i.e., classifications that generalise over verbs according to their semantic properties. Intuitive examples of such classifications are the MOTION WITH A VEHICLE class, including verbs such as *drive, fly, row*, etc., or the BREAK A SOLID SURFACE WITH AN INSTRUMENT class, including verbs such as *break, crush, fracture, smash*, etc. Semantic verb classifications are of great interest to computational linguistics, specifically regarding the pervasive problem of data sparseness in the processing of natural language. Up to now, such classifications have been used in applications such as word sense disambiguation (Dorr and Jones, 1996; Kohomban and Lee, 2005), machine translation (Prescher et al., 2000; Koehn and Hoang, 2007), document classification (Klavans and Kan, 1998), and in statistical lexical acquisition in general (Rooth et al., 1999; Merlo and Stevenson, 2001; Korhonen, 2002; Schulte im Walde, 2006).

Given that the creation of semantic verb classifications is not an end task in itself, but depends on the application scenario of the classification, we find various approaches to an automatic induction of semantic verb classifications. For example, Siegel and McKeown (2000) used several machine learning algorithms to perform an automatic aspectual classification of English verbs into event and stative verbs. Merlo and Stevenson (2001) presented an automatic classification of three types of English intransitive verbs, based on argument structure and heuristics to thematic relations. Pereira et al. (1993) and Rooth et al. (1999) relied on the Expectation-Maximisation algorithm to induce soft clusters of verbs, based on the verbs' direct object nouns. Similarly, Korhonen et al. (2003) relied on the Information Bottleneck (Tishby et al., 1999) and subcategorisation frame types to induce soft verb clusters.

This paper presents an innovative, complex approach to semantic verb classes that relies on selectional preferences as verb properties. The underlying linguistic assumption for this verb class model is that verbs which agree on their selectional preferences belong to a common semantic class. The model is implemented as a soft-clustering approach, in order to capture the polysemy of the verbs. The training procedure uses the Expectation-Maximisation (EM) algorithm (Baum, 1972) to iteratively improve the probabilistic parameters of the model, and applies the Minimum Description Length (MDL) principle (Rissanen, 1978) to induce WordNet-based selectional preferences for arguments within subcategorisation frames. Our model is potentially useful for lexical induction (e.g., verb senses, subcategorisation and selectional preferences, collocations, and verb alternations),

and for NLP applications in sparse data situations. In this paper, we provide an evaluation based on a language model.

The remainder of the paper is organised as follows. Section 2 introduces our probabilistic verb class model, the EM training, and how we incorporate the MDL principle. Section 3 describes the clustering experiments, including the experimental setup, the evaluation, and the results. Section 4 reports on related work, before we close with a summary and outlook in Section 5.

2 Verb Class Model

2.1 Probabilistic Model

This paper suggests a probabilistic model of verb classes that groups verbs into clusters with similar subcategorisation frames and selectional preferences. Verbs may be assigned to several clusters (soft clustering) which allows the model to describe the subcategorisation properties of several verb readings separately. The number of clusters is defined in advance, but the assignment of the verbs to the clusters is learnt during training. It is assumed that all verb readings belonging to one cluster have similar subcategorisation and selectional properties. The selectional preferences are expressed in terms of semantic concepts from WordNet, rather than a set of individual words. Finally, the model assumes that the different arguments are mutually independent for all subcategorisation frames of a cluster. From the last assumption, it follows that any statistical dependency between the arguments of a verb has to be explained by multiple readings.

The statistical model is characterised by the following equation which defines the probability of a verb v with a subcategorisation frame f and arguments $a_1, ..., a_{n_f}$:

$$p(v, f, a_1, ..., a_{n_f}) = \sum_c p(c)\ p(v|c)\ p(f|c)\ * \prod_{i=1}^{n_f} \sum_{r \in R} p(r|c, f, i)\ p(a_i|r)$$

The model describes a stochastic process which generates a verb-argument tuple like ⟨*speak, subj-pp.to, professor, audience*⟩ by

1. selecting some cluster c, e.g. c_3 (which might correspond to a set of *communication* verbs), with probability $p(c_3)$,

2. selecting a verb v, here the verb *speak*, from cluster c_3 with probability $p(\text{speak}|c_3)$,

3. selecting a subcategorisation frame f, here *subj-pp.to*, with probability $p(\text{subj-pp.to}|c_3)$; note that the frame probability only depends on the cluster, and not on the verb,

4. selecting a WordNet concept r for each argument slot, e.g. *person* for the first slot with probability $p(\text{person}|c_3, \text{subj-pp.to}, 1)$ and *social group* for the second slot with probability $p(\text{social group}|c_3, \text{subj-pp.to}, 2)$,

5. selecting a word a_i to instantiate each concept as argument i; in our example, we might choose *professor* for *person* with probability $p(\text{professor}|\text{person})$ and *audience* for *social group* with probability $p(\text{audience}|\text{social group})$.

The model contains two *hidden variables*, namely the clusters c and the selectional preferences r. In order to obtain the overall probability of a given verb-argument tuple, we have to sum over all possible values of these hidden variables.

The assumption that the arguments are independent of the verb given the cluster is essential for obtaining a clustering algorithm because it forces the EM algorithm to make the verbs within a cluster as similar as possible.[1] The assumption that the different arguments of a verb are mutually independent is important to reduce the parameter set to a tractable size

The fact that verbs select for concepts rather than individual words also reduces the number of parameters and helps to avoid sparse data problems. The application of the MDL principle guarantees that no important information is lost.

The probabilities $p(r|c, f, i)$ and $p(a|r)$ mentioned above are not represented as atomic entities. Instead, we follow an approach by Abney

[1] The EM algorithm adjusts the model parameters in such a way that the probability assigned to the training tuples is maximised. Given the model constraints, the data probability can only be maximised by making the verbs within a cluster as similar to each other as possible, regarding the required arguments.

and Light (1999) and turn WordNet into a Hidden Markov model (HMM). We create a new pseudo-concept for each WordNet noun and add it as a hyponym to each synset containing this word. In addition, we assign a probability to each hypernymy–hyponymy transition, such that the probabilities of the hyponymy links of a synset sum up to 1. The pseudo-concept nodes emit the respective word with a probability of 1, whereas the regular concept nodes are non-emitting nodes. The probability of a path in this (a priori) WordNet HMM is the product of the probabilities of the transitions within the path. The probability $p(a|r)$ is then defined as the sum of the probabilities of all paths from the concept r to the word a. Similarly, we create a partial WordNet HMM for each argument slot $\langle c, f, i\rangle$ which encodes the selectional preferences. It contains only the WordNet concepts that the slot selects for, according to the MDL principle (cf. Section 2.3), and the dominating concepts. The probability $p(r|c, f, i)$ is the total probability of all paths from the top-most WordNet concept *entity* to the terminal node r.

2.2 EM Training

The model is trained on verb-argument tuples of the form described above, i.e., consisting of a verb and a subcategorisation frame, plus the nominal[2] heads of the arguments. The tuples may be extracted from parsed data, or from a treebank. Because of the hidden variables, the model is trained iteratively with the Expectation-Maximisation algorithm (Baum, 1972). The parameters are randomly initialised and then re-estimated with the Inside-Outside algorithm (Lari and Young, 1990) which is an instance of the EM algorithm for training Probabilistic Context-Free Grammars (PCFGs).

The PCFG training algorithm is applicable here because we can define a PCFG for each of our models which generates the same verb-argument tuples with the same probability. The PCFG is defined as follows:

(1) The start symbol is TOP.

(2) For each cluster c, we add a rule TOP $\rightarrow$ V_c A_c whose probability is $p(c)$.

(3) For each verb v in cluster c, we add a rule $\mathrm{V}_c \rightarrow$ v with probability $p(v|c)$.

(4) For each subcategorisation frame f of cluster c with length n, we add a rule $\mathrm{A}_c \rightarrow$ f $\mathrm{R}_{c,f,1,entity}$ ... $\mathrm{R}_{c,f,n,entity}$ with probability $p(f|c)$.

(5) For each transition from a node r to a node r' in the selectional preference model for slot i of the subcategorisation frame f of cluster c, we add a rule $\mathrm{R}_{c,f,i,r} \rightarrow \mathrm{R}_{c,f,i,r'}$ whose probability is the transition probability from r to r' in the respective WordNet-HMM.

(6) For each terminal node r in the selectional preference model, we add a rule $\mathrm{R}_{c,f,i,r} \rightarrow \mathrm{R}_r$ whose probability is 1. With this rule, we "jump" from the selectional restriction model to the corresponding node in the a priori model.

(7) For each transition from a node r to a node r' in the a priori model, we add a rule $\mathrm{R}_r \rightarrow \mathrm{R}_{r'}$ whose probability is the transition probability from r to r' in the a priori WordNet-HMM.

(8) For each word node a in the a priori model, we add a rule $\mathrm{R}_a \rightarrow$ a whose probability is 1.

Based on the above definitions, a partial "parse" for $\langle$*speak subj-pp.to professor audience*$\rangle$, referring to cluster 3 and one possible WordNet path, is shown in Figure 1. The connections within R_3 ($R_{3,\ldots,entity}$–$R_{3,\ldots,person/group}$) and within R ($R_{person/group}$–$R_{professor/audience}$) refer to sequential applications of rule types (5) and (7), respectively.

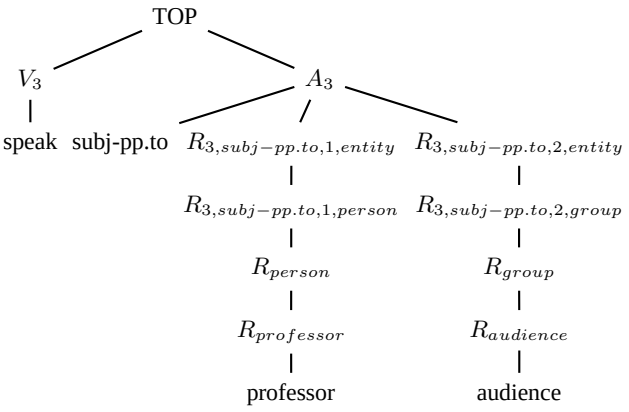

Figure 1: Example parse tree.

The EM training algorithm maximises the likelihood of the training data.

[2] Arguments with lexical heads other than nouns (e.g., subcategorised clauses) are not included in the selectional preference induction.

2.3 MDL Principle

A model with a large number of fine-grained concepts as selectional preferences assigns a higher likelihood to the data than a model with a small number of general concepts, because in general a larger number of parameters is better in describing training data. Consequently, the EM algorithm a priori prefers fine-grained concepts but – due to sparse data problems – tends to overfit the training data. In order to find selectional preferences with an appropriate granularity, we apply the Minimum Description Length principle, an approach from Information Theory. According to the MDL principle, the model with minimal *description length* should be chosen. The description length itself is the sum of the *model length* and the *data length*, with the model length defined as the number of bits needed to encode the model and its parameters, and the data length defined as the number of bits required to encode the training data with the given model. According to coding theory, an optimal encoding uses $-log_2 p$ bits, on average, to encode data whose probability is p. Usually, the model length increases and the data length decreases as more parameters are added to a model. The MDL principle finds a compromise between the size of the model and the accuracy of the data description.

Our selectional preference model relies on Li and Abe (1998), applying the MDL principle to determine selectional preferences of verbs and their arguments, by means of a concept hierarchy ordered by hypernym/hyponym relations. Given a set of nouns within a specific argument slot as a sample, the approach finds the cut[3] in a concept hierarchy which minimises the sum of encoding both the model and the data. The *model length (ML)* is defined as

$$ML = \frac{k}{2} \ * \ log_2 \ |S|,$$

with k the number of concepts in the partial hierarchy between the top concept and the concepts in the cut, and $|S|$ the sample size, i.e., the total frequency of the data set. The *data length (DL)* is defined as

$$DL = -\sum_{n \in S} \ log_2 \ p(n).$$

[3]A *cut* is defined as a set of concepts in the concept hierarchy that defines a partition of the "leaf" concepts (the lowest concepts in the hierarchy), viewing each concept in the cut as representing the set of all leaf concepts it dominates.

The probability of a noun $p(n)$ is determined by dividing the total probability of the concept class the noun belongs to, $p(concept)$, by the size of that class, $|concept|$, i.e., the number of nouns that are dominated by that concept:

$$p(n) = \frac{p(concept)}{|concept|}.$$

The higher the concept within the hierarchy, the more nouns receive an equal probability, and the greater is the data length.

The probability of the concept class in turn is determined by dividing the frequency of the concept class $f(concept)$ by the sample size:

$$p(concept) = \frac{f(concept)}{|S|},$$

where $f(concept)$ is calculated by upward propagation of the frequencies of the nominal lexemes from the data sample through the hierarchy. For example, if the nouns *coffee, tea, milk* appeared with frequencies $25, 50, 3$, respectively, within a specific argument slot, then their hypernym concept *beverage* would be assigned a frequency of 78, and these 78 would be propagated further upwards to the next hypernyms, etc. As a result, each concept class is assigned a fraction of the frequency of the whole data set (and the top concept receives the total frequency of the data set). For calculating $p(concept)$ (and the overall data length), though, only the concept classes within the cut through the hierarchy are relevant.

Our model uses WordNet 3.0 as the concept hierarchy, and comprises one (complete) a priori WordNet model for the lexical head probabilities $p(a|r)$ and one (partial) model for each selectional probability distribution $p(r|c, f, i)$, cf. Section 2.1.

2.4 Combining EM and MDL

The training procedure that combines the EM training with the MDL principle can be summarised as follows.

1. The probabilities of a verb class model with c classes and a pre-defined set of verbs and frames are initialised randomly. The selectional preference models start out with the most general WordNet concept only, i.e., the partial WordNet hierarchies underlying the probabilities $p(r|c, f, i)$ initially only contain the concept r for *entity*.

2. The model is trained for a pre-defined number of iterations. In each iteration, not only the model probabilities are re-estimated and maximised (as done by EM), but also the cuts through the concept hierarchies that represent the various selectional preference models are re-assessed. In each iteration, the following steps are performed.

(a) The partial WordNet hierarchies that represent the selectional preference models are expanded to include the hyponyms of the respective leaf concepts of the partial hierarchies. I.e., in the first iteration, all models are expanded towards the hyponyms of *entity*, and in subsequent iterations each selectional preference model is expanded to include the hyponyms of the leaf nodes in the partial hierarchies resulting from the previous iteration. This expansion step allows the selection models to become more and more detailed, as the training proceeds and the verb clusters (and their selectional restrictions) become increasingly specific.

(b) The training tuples are processed: For each tuple, a PCFG parse forest as indicated by Figure 1 is done, and the Inside-Outside algorithm is applied to estimate the frequencies of the "parse tree rules", given the current model probabilities.

(c) The MDL principle is applied to each selectional preference model: Starting from the respective leaf concepts in the partial hierarchies, MDL is calculated to compare each set of hyponym concepts that share a hypernym with the respective hypernym concept. If the MDL is lower for the set of hyponyms than the hypernym, the hyponyms are left in the partial hierarchy. Otherwise the expansion of the hypernym towards the hyponyms is undone and we continue recursively upwards the hierarchy, calculating MDL to compare the former hypernym and its cohyponyms with the next upper hypernym, etc. The recursion allows the training algorithm to remove nodes which were added in earlier iterations and are no longer relevant. It stops if the MDL is lower for the hyponyms than for the hypernym.
This step results in selectional preference models that minimally contain the top concept *entity*, and maximally contain the partial WordNet hierarchy between *entity* and the concept classes that have been expanded within this iteration.

(d) The probabilities of the verb class model are maximised based on the frequency estimates obtained in step (b).

3 Experiments

The model is generally applicable to all languages for which WordNet exists, and for which the WordNet functions provided by Princeton University are available. For the purposes of this paper, we choose English as a case study.

3.1 Experimental Setup

The input data for training the verb class models were derived from Viterbi parses of the whole British National Corpus, using the lexicalised PCFG for English by Carroll and Rooth (1998). We took only active clauses into account, and disregarded auxiliary and modal verbs as well as particle verbs, leaving a total of 4,852,371 Viterbi parses. Those input tuples were then divided into 90% training data and 10% test data, providing 4,367,130 training tuples (over 2,769,804 types), and 485,241 test tuples (over 368,103 types).

As we wanted to train and assess our verb class model under various conditions, we used different fractions of the training data in different training regimes. Because of time and memory constraints, we only used training tuples that appeared at least twice. (For the sake of comparison, we also trained one model on all tuples.) Furthermore, we disregarded tuples with personal pronoun arguments; they are not represented in WordNet, and even if they are added (e.g. to general concepts such as *person, entity*) they have a rather destructive effect. We considered two subsets of the subcategorisation frames with 10 and 20 elements, which were chosen according to their overall frequency in the training data; for example, the 10 most frequent frame types were *subj:obj, subj, subj:ap, subj:to, subj:obj:obj2, subj:obj:pp-in, subj:adv, subj:pp-in, subj:vbase, subj:that*.[4] When relying on theses 10/20 subcategorisation frames, plus including the above restrictions, we were left with 39,773/158,134 and 42,826/166,303 training tuple types/tokens, respectively. The overall number of training tuples

[4] A frame lists its arguments, separated by ':'. Most arguments within the frame types should be self-explanatory. `ap` is an adjectival phrase.

was therefore much smaller than the generally available data. The corresponding numbers including tuples with a frequency of one were 478,717/597,078 and 577,755/701,232.

The number of clusters in the experiments was either 20 or 50, and we used up to 50 iterations over the training tuples. The model probabilities were output after each 5th iteration. The output comprises all model probabilities introduced in Section 2.1. The following sections describe the evaluation of the experiments, and the results.

3.2 Evaluation

One of the goals in the development of the presented verb class model was to obtain an accurate statistical model of verb-argument tuples, i.e. a model which precisely predicts the tuple probabilities. In order to evaluate the performance of the model in this respect, we conducted an evaluation experiment, in which we computed the probability which the verb class model assigns to our test tuples and compared it to the corresponding probability assigned by a baseline model. The model with the higher probability is judged the better model.

We expected that the verb class model would perform better than the baseline model on tuples where one or more of the arguments were not observed with the respective verb, because either the argument itself or a semantically similar argument (according to the selectional preferences) was observed with verbs belonging to the same cluster. We also expected that the verb class model assigns a lower probability than the baseline model to test tuples which frequently occurred in the training data, since the verb class model fails to describe precisely the idiosyncratic properties of verbs which are not shared by the other verbs of its cluster.

The Baseline Model The baseline model decomposes the probability of a verb-argument tuple into a product of conditional probabilities:[5]

$$p(v, f, a_1^{n_f}) = p(v)\ p(f|v) \prod_{i=1}^{n_f} p(a_i|a_1^{i-1}, \langle v, f\rangle, f_i)$$

[5] f_i is the label of the i[th] slot. The verb and the subcategorisation frame are enclosed in angle brackets because they are treated as a unit during smoothing.

The probability of our example tuple ⟨*speak, subj-pp.to, professor, audience*⟩ in the baseline model is then $p(\text{speak})\ p(\text{subj-pp.to}|\text{speak})\ p(\text{professor}|\langle\text{speak, subj-pp.to}\rangle, \text{subj})\ \ p(\text{audience}|\ \text{professor}, \langle\text{speak, subj-pp.to}\rangle, \text{pp.to})$.

The model contains no hidden variables. Thus the parameters can be directly estimated from the training data with relative frequencies. The parameter estimates are smoothed with modified Kneser-Ney smoothing (Chen and Goodman, 1998), such that the probability of each tuple is positive.

Smoothing of the Verb Class Model Although the verb class model has a built-in smoothing capacity, it needs additional smoothing for two reasons: Firstly, some of the nouns in the test data did not occur in the training data. The verb class model assigns a zero probability to such nouns. Hence we smoothed the concept instantiation probabilities $p(\text{noun}|\text{concept})$ with Witten-Bell smoothing (Chen and Goodman, 1998). Secondly, we smoothed the probabilities of the concepts in the selectional preference models where zero probabilities may occur.

The smoothing ensures that the verb class model assigns a positive probability to each verb-argument tuple with a known verb, a known subcategorisation frame, and arguments which are in WordNet. Other tuples were excluded from the evaluation because the verb class model cannot deal with them.

3.3 Results

The evaluation results of our classification experiments are presented in Table 1, for 20 and 50 clusters, with 10 and 20 subcategorisation frame types. The table cells provide the log_e of the probabilities per tuple token. The probabilities increase with the number of iterations, flattening out after approx. 25 iterations, as illustrated by Figure 2. Both for 10 and 20 frames, the results are better for 50 than for 20 clusters, with small differences between 10 and 20 frames. The results vary between -11.850 and -10.620 (for 5-50 iterations), in comparison to baseline values of -11.546 and -11.770 for 10 and 20 frames, respectively. The results thus show that our verb class model results are above the baseline results after 10 iterations; this means that our statistical model then assigns higher probabilities to the test tuples than the baseline model.

No. of Clusters	Iteration									
	5	10	15	20	25	30	35	40	45	50
	10 frames									
20	-11.770	-11.408	-10.978	-10.900	-10.853	-10.841	-10.831	-10.823	-10.817	-10.812
50	-11.850	-11.452	-11.061	-10.904	-10.730	-10.690	-10.668	-10.628	-10.625	-10.620
	20 frames									
20	-11.769	-11.430	-11.186	-10.971	-10.921	-10.899	-10.886	-10.875	-10.873	-10.869
50	-11.841	-11.472	-11.018	-10.850	-10.737	-10.728	-10.706	-10.680	-10.662	-10.648

Table 1: Clustering results – BNC tuples.

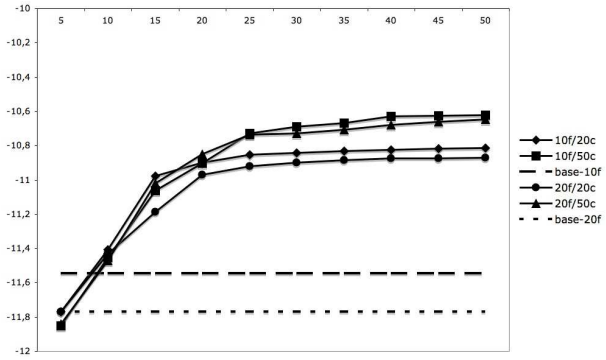

Figure 2: Illustration of clustering results.

Including input tuples with a frequency of one in the training data with 10 subcategorisation frames (as mentioned in Section 3.1) decreases the log_e per tuple to between -13.151 and -12.498 (for 5-50 iterations), with similar training behaviour as in Figure 2, and in comparsion to a baseline of -17.988. The differences in the result indicate that the models including the hapax legomena are worse than the models that excluded the sparse events; at the same time, the differences between baseline and clustering model are larger.

In order to get an intuition about the qualitative results of the clusterings, we select two example clusters that illustrate that the idea of the verb class model has been realised within the clusters. According to our own intuition, the clusters are overall semantically impressive, beyond the examples. Future work will assess by semantics-based evaluations of the clusters (such as pseudo-word disambiguation, or a comparison against existing verb classifications), whether this intuition is justified, whether it transfers to the majority of verbs within the cluster analyses, and whether the clusters capture polysemic verbs appropriately.

The two examples are taken from the 10 frame/50 cluster verb class model, with probabilities of 0.05 and 0.04. The ten most probable verbs in the first cluster are *show, suggest, indicate, reveal, find, imply, conclude, demonstrate, state, mean*, with the two most probable frame types *subj* and *subj:that*, i.e., the intransitive frame, and a frame that subcategorises a *that* clause. As selectional preferences within the intransitive frame (and quite similarly in the *subj:that* frame), the most probable concept classes[6] are *study, report, survey, name, research, result, evidence*. The underlined nouns represent specific concept classes, because they are leaf nodes in the selectional preference hierarchy, thus referring to very specific selectional preferences, which are potentially useful for collocation induction. The ten most probable verbs in the second cluster are *arise, remain, exist, continue, need, occur, change, improve, begin, become*, with the intransitive frame being most probable. The most probable concept classes are *problem, condition, question, natural phenomenon, situation*. The two examples illustrate that the verbs within a cluster are semantically related, and that they share obvious subcategorisation frames with intuitively plausible selectional preferences.

4 Related Work

Our model is an extension of and thus most closely related to the latent semantic clustering (LSC) model (Rooth et al., 1999) for verb-argument pairs $\langle v, a \rangle$ which defines their probability as follows:

$$p(v,a) = \sum_c p(c)\ p(v|c)\ p(a|c)$$

In comparison to our model, the LSC model only considers a single argument (such as direct objects),

[6] For readability, we only list one noun per WordNet concept.

or a fixed number of arguments from one particular subcategorisation frame, whereas our model defines a probability distribution over all subcategorisation frames. Furthermore, our model specifies selectional preferences in terms of general WordNet concepts rather than sets of individual words.

In a similar vein, our model is both similar and distinct in comparison to the soft clustering approaches by Pereira et al. (1993) and Korhonen et al. (2003). Pereira et al. (1993) suggested deterministic annealing to cluster verb-argument pairs into classes of verbs and nouns. On the one hand, their model is asymmetric, thus not giving the same interpretation power to verbs and arguments; on the other hand, the model provides a more fine-grained clustering for nouns, in the form of an additional hierarchical structure of the noun clusters. Korhonen et al. (2003) used verb-frame pairs (instead of verb-argument pairs) to cluster verbs relying on the Information Bottleneck (Tishby et al., 1999). They had a focus on the interpretation of verbal polysemy as represented by the soft clusters. The main difference of our model in comparison to the above two models is, again, that we incorporate selectional preferences (rather than individual words, or subcategorisation frames).

In addition to the above soft-clustering models, various approaches towards semantic verb classification have relied on hard-clustering models, thus simplifying the notion of verbal polysemy. Two large-scale approaches of this kind are Schulte im Walde (2006), who used k-Means on verb subcategorisation frames and verbal arguments to cluster verbs semantically, and Joanis et al. (2008), who applied Support Vector Machines to a variety of verb features, including subcategorisation slots, tense, voice, and an approximation to animacy. To the best of our knowledge, Schulte im Walde (2006) is the only hard-clustering approach that previously incorporated selectional preferences as verb features. However, her model was not soft-clustering, and she only used a simple approach to represent selectional preferences by WordNet's top-level concepts, instead of making use of the whole hierarchy and more sophisticated methods, as in the current paper.

Last but not least, there are other models of selectional preferences than the MDL model we used in our paper. Most such models also rely on the WordNet hierarchy (Resnik, 1997; Abney and Light, 1999; Ciaramita and Johnson, 2000; Clark and Weir, 2002). Brockmann and Lapata (2003) compared some of the models against human judgements on the acceptability of sentences, and demonstrated that the models were significantly correlated with human ratings, and that no model performed best; rather, the different methods are suited for different argument relations.

5 Summary and Outlook

This paper presented an innovative, complex approach to semantic verb classes that relies on selectional preferences as verb properties. The probabilistic verb class model underlying the semantic classes was trained by a combination of the EM algorithm and the MDL principle, providing soft clusters with two dimensions (verb senses and subcategorisation frames with selectional preferences) as a result. A language model-based evaluation showed that after 10 training iterations the verb class model results are above the baseline results.

We plan to improve the verb class model with respect to (i) a concept-wise (instead of a cut-wise) implementation of the MDL principle, to operate on concepts instead of combinations of concepts; and (ii) variations of the concept hierarchy, using e.g. the sense-clustered WordNets from the Stanford WordNet Project (Snow et al., 2007), or a WordNet version improved by concepts from DOLCE (Gangemi et al., 2003), to check on the influence of conceptual details on the clustering results. Furthermore, we aim to use the verb class model in NLP tasks, (i) as resource for lexical induction of verb senses, verb alternations, and collocations, and (ii) as a lexical resource for the statistical disambiguation of parse trees.

References

Steven Abney and Marc Light. 1999. Hiding a Semantic Class Hierarchy in a Markow Model. In *Proceedings of the ACL Workshop on Unsupervised Learning in Natural Language Processing*, pages 1–8, College Park, MD.

Leonard E. Baum. 1972. An Inequality and Associated Maximization Technique in Statistical Estimation for Probabilistic Functions of Markov Processes. *Inequalities*, III:1–8.

Carsten Brockmann and Mirella Lapata. 2003. Evaluating and Combining Approaches to Selectional Preference Acquisition. In *Proceedings of the 10th Conference of the European Chapter of the Association for Computational Linguistics*, pages 27–34, Budapest, Hungary.

Glenn Carroll and Mats Rooth. 1998. Valence Induction with a Head-Lexicalized PCFG. In *Proceedings of the 3rd Conference on Empirical Methods in Natural Language Processing*, Granada, Spain.

Stanley Chen and Joshua Goodman. 1998. An Empirical Study of Smoothing Techniques for Language Modeling. Technical Report TR-10-98, Center for Research in Computing Technology, Harvard University.

Massimiliano Ciaramita and Mark Johnson. 2000. Explaining away Ambiguity: Learning Verb Selectional Preference with Bayesian Networks. In *Proceedings of the 18th International Conference on Computational Linguistics*, pages 187–193, Saarbrücken, Germany.

Stephen Clark and David Weir. 2002. Class-Based Probability Estimation using a Semantic Hierarchy. *Computational Linguistics*, 28(2):187–206.

Bonnie J. Dorr and Doug Jones. 1996. Role of Word Sense Disambiguation in Lexical Acquisition: Predicting Semantics from Syntactic Cues. In *Proceedings of the 16th International Conference on Computational Linguistics*, pages 322–327, Copenhagen, Denmark.

Aldo Gangemi, Nicola Guarino, Claudio Masolo, and Alessandro Oltramari. 2003. Sweetening WordNet with DOLCE. *AI Magazine*, 24(3):13–24.

Eric Joanis, Suzanne Stevenson, and David James. 2008? A General Feature Space for Automatic Verb Classification. *Natural Language Engineering*. To appear.

Judith L. Klavans and Min-Yen Kan. 1998. The Role of Verbs in Document Analysis. In *Proceedings of the 17th International Conference on Computational Linguistics and the 36th Annual Meeting of the Association for Computational Linguistics*, pages 680–686, Montreal, Canada.

Philipp Koehn and Hieu Hoang. 2007. Factored Translation Models. In *Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 868–876, Prague, Czech Republic.

Upali S. Kohomban and Wee Sun Lee. 2005. Learning Semantic Classes for Word Sense Disambiguation. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 34–41, Ann Arbor, MI.

Anna Korhonen, Yuval Krymolowski, and Zvika Marx. 2003. Clustering Polysemic Subcategorization Frame Distributions Semantically. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 64–71, Sapporo, Japan.

Anna Korhonen. 2002. *Subcategorization Acquisition*. Ph.D. thesis, University of Cambridge, Computer Laboratory. Technical Report UCAM-CL-TR-530.

Karim Lari and Steve J. Young. 1990. The Estimation of Stochastic Context-Free Grammars using the Inside-Outside Algorithm. *Computer Speech and Language*, 4:35–56.

Hang Li and Naoki Abe. 1998. Generalizing Case Frames Using a Thesaurus and the MDL Principle. *Computational Linguistics*, 24(2):217–244.

Paola Merlo and Suzanne Stevenson. 2001. Automatic Verb Classification Based on Statistical Distributions of Argument Structure. *Computational Linguistics*, 27(3):373–408.

Fernando Pereira, Naftali Tishby, and Lillian Lee. 1993. Distributional Clustering of English Words. In *Proceedings of the 31st Annual Meeting of the Association for Computational Linguistics*, pages 183–190, Columbus, OH.

Detlef Prescher, Stefan Riezler, and Mats Rooth. 2000. Using a Probabilistic Class-Based Lexicon for Lexical Ambiguity Resolution. In *Proceedings of the 18th International Conference on Computational Linguistics*.

Philip Resnik. 1997. Selectional Preference and Sense Disambiguation. In *Proceedings of the ACL SIGLEX Workshop on Tagging Text with Lexical Semantics: Why, What, and How?*, Washington, DC.

Jorma Rissanen. 1978. Modeling by Shortest Data Description. *Automatica*, 14:465–471.

Mats Rooth, Stefan Riezler, Detlef Prescher, Glenn Carroll, and Franz Beil. 1999. Inducing a Semantically Annotated Lexicon via EM-Based Clustering. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*, Maryland, MD.

Sabine Schulte im Walde. 2006. Experiments on the Automatic Induction of German Semantic Verb Classes. *Computational Linguistics*, 32(2):159–194.

Eric V. Siegel and Kathleen R. McKeown. 2000. Learning Methods to Combine Linguistic Indicators: Improving Aspectual Classification and Revealing Linguistic Insights. *Computational Linguistics*, 26(4):595–628.

Rion Snow, Sushant Prakash, Daniel Jurafsky, and Andrew Y. Ng. 2007. Learning to Merge Word Senses. In *Proceedings of the joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, Prague, Czech Republic.

Naftali Tishby, Fernando Pereira, and William Bialek. 1999. The Information Bottleneck Method. In *Proceedings of the 37th Annual Conference on Communication, Control, and Computing*, Monticello, IL.