Verb Class Discovery from Rich Syntactic Data

Lin Sun[1], Anna Korhonen[1] and Yuval Krymolowski[2]

[1] Computer Laboratory, University of Cambridge
15 JJ Thomson Avenue, Cambridge CB3 0FD, UK
alk23@cam.ac.uk, ls418@cam.ac.uk
[2] Department of Computer Science, University of Haifa
31905, Haifa, Israel
yuvalkry@gmail.com

Abstract. Previous research has shown that syntactic features are the most informative features in automatic verb classification. We investigate their optimal characteristics by comparing a range of feature sets extracted from data where the proportion of verbal arguments and adjuncts is controlled. The data are obtained from different versions of VALEX [1] – a large SCF lexicon for English which was acquired automatically from several corpora and the Web. We evaluate the feature sets thoroughly using four supervised classifiers and one unsupervised method. The best performing feature set includes rich syntactic information about both arguments and adjuncts of verbs. When combined with our best performing classifier (a novel Gaussian classifier), it yields the promising accuracy of 64.2% in classifying 204 verbs to 17 Levin (1993) classes. We discuss the impact of our results on the state-or-art and propose avenues for future work.

1 Introduction

Recent research shows that it is possible, using current natural language processing (NLP) and machine learning technology, to automatically induce lexical classes from corpus data with promising accuracy [2–5]. This research is interesting, since lexical classifications, when tailored to the application and domain in question, can provide an effective means to deal with a number of important NLP tasks (e.g. parsing, word sense disambiguation, semantic role labeling), as well as enhance performance in applications, (e.g. information extraction, question-answering, machine translation) [6–10].

Lexical classes are useful for NLP because they capture generalizations over a range of (cross-)linguistic properties. Being defined in terms of similar meaning components and (morpho-)syntactic behaviour of words [11, 12] they generally incorporate a wider range of properties than e.g. classes defined solely on semantic grounds [13]. For example, verbs which share the meaning component of 'manner of motion' (such as *travel*, *run*, *walk*), behave similarly also in terms of subcategorization (*I traveled/ran/walked to London*) and usually have zero-related nominals (*a run*, *a walk*).

NLP systems can benefit from lexical classes in many ways. For example, such classes can be used i) to define a mapping from surface realization of arguments to predicate-argument structure, ii) as a means to abstract away from individual words when required, or (iii) to build a lexical organization which predicts much of the syntax and semantics of a new word by associating it with an appropriate class.

While lexical classes have proved useful for various (multilingual) tasks, their large-scale exploitation in real-world or domain-sensitive tasks has not been possible because existing manually built classifications are incomprehensive. They are expensive to extend and do not incorporate important statistical information about the likelihood of different classes for words. Automatic classification can help since it is cost-effective and gathers statistical information as a side-effect of the acquisition process.

Most work on lexical classification has focussed on verbs, which are typically the main predicates in sentences. Syntactic features have proved the most informative for verb classification. The best results in automatic classification have been obtained using either (i) deep syntactic features (e.g. subcategorization frames (SCFs)) extracted using parsers and subcategorisation acquisition systems [14, 3, 4] or (ii) shallow ones (e.g. NPs/PPs preceding/following verbs) extracted using taggers and chunkers [2, 5]. (i) capture the arguments of verbs and correspond closely with the features used for manual classification [12]. The fact that promising results have also been reported using (ii) has led to suggestions that adjuncts can also be useful for the task, e.g. [5].

To gain a better understanding of the optimal characteristics of syntactic features in verb classification, we compare a range of feature sets extracted from data where the proportion of verbal arguments and adjuncts is controlled. The data are obtained from different versions of VALEX [1] – a large SCF lexicon for English which was acquired automatically from several corpora and the Web. We evaluate the feature sets thoroughly using four supervised classifiers and one unsupervised method. The best performing feature set is the one which includes the richest syntactic information about both arguments and adjuncts of verbs. When combined with the best performing classifier (our novel Gaussian classifier), it yields the promising accuracy of 64.2% in classifying 204 verbs to 17 Levin (1993) classes. We discuss the impact of our results on the state-or-art and propose avenues for future work.

We introduce our target classification in section 2 and syntactic features in section 3. The classification and clustering techniques are presented in section 4. Details of the experimental evaluation are supplied in section 5. Section 6 provides discussion and concludes with directions for future work.

2 Test Verbs and Classes

We adopt as a target classification Levin's (1993) well-known taxonomy where verbs taking similar diathesis alternations are assumed to share meaning components and are organized into a semantically coherent class. For instance, the class of "*Break* Verbs" (class 45.1) is partially characterized by its participation in the following alternations:

1. **Causative/inchoative alternation**:
   Tony broke the window ↔ *The window broke*
2. **Middle alternation**:
   Tony broke the window ↔ *The window broke easily*
3. **Instrument subject alternation**:
   Tony broke the window with the hammer ↔ *The hammer broke the window*

Alternations are expressed as pairs of SCFs. Additional properties related to syntax, morphology and extended meanings of member verbs are specified with some classes.

	LEVIN CLASS	EXAMPLE VERBS
9.1	PUT	*bury, place, install, mount, put, deposit, position, set*
10.1	REMOVE	*remove, abolish, eject, extract, deduct, eradicate, sever, evict*
11.1	SEND	*ship, post, send, mail, transmit, transfer, deliver, slip*
13.5.1	GET	*win, gain, earn, buy, get, book, reserve, fetch*
18.1	HIT	*beat, slap, bang, knock, pound, batter, hammer, lash*
22.2	AMALGAMATE	*contrast, match, overlap, unite, unify, unite, contrast, affiliate*
29.2	CHARACTERIZE	*envisage, portray, regard, treat, enlist, define, depict, diagnose*
30.3	PEER	*listen, stare, look, glance, gaze, peer, peek, squint*
31.1	AMUSE	*delight, scare, shock, confuse, upset, overwhelm, scare, disappoint*
36.1	CORRESPOND	*cooperate, collide, concur, mate, flirt, interact, dissent, mate*
37.3	MANNER OF SPEAKING	*shout, yell, moan, mutter, murmur, snarl, moan, wail*
37.7	SAY	*say, reply, mention, state, report, respond, announce, recount*
40.2	NONVERBAL EXPRESSION	*smile, laugh, grin, sigh, gas, chuckle, frown, giggle*
43.1	LIGHT EMISSION	*shine, flash, flare, glow, blaze, flicker, gleam, sparkle*
45.4	CHANGE OF STATE	*soften, weaken, melt, narrow, deepen, dampen, melt, multiply*
47.3	MODES OF BEING WITH MOTION	*quake, falter, sway, swirl, teeter, flutter, wobble, waft*
51.3.2	RUN	*swim, fly, walk, slide, run, travel, stroll, glide*

Table 1. Test classes and example verbs

The extended version of Levin's classification currently incorporated in VerbNet [15][3] provides a classification of 5,257 verb senses into 274 classes. We selected 17 of Levin's original classes and 12 member verbs per class (table 1) for experimentation. The small test set enabled us to evaluate our results thoroughly. The classes were selected in random, subject to the constraint that they (i) included both syntactically and semantically similar and different classes (to vary the difficulty of the classification task), and (ii) had enough member verbs whose predominant sense belongs to the class in question (we verified this according to the method described in [1]). As VALEX was designed for a maximum coverage most test verbs had 1000-9000 occurrences in the lexicon.

3 Syntactic Features

We employed as features distributions of SCFs specific to given verbs. We extracted them from the recent large VALEX [1] lexicon which provides SCF frequency information for 6,397 English verbs. VALEX was acquired automatically from five large corpora and the Web (up to 10,000 occurrences per verb) using the subcategorization acquisition system of Briscoe and Carroll [16]. The system incorporates RASP, a domain-independent robust statistical parser [17], and a SCF classifier which identifies 163 verbal SCFs. The basic SCFs abstract over lexically-governed particles and prepositions and predicate selectional preferences. Three versions of VALEX were employed[4]:

Valex 1: A noisy unfiltered version of VALEX which includes all the SCFs found in data.

Valex 2: A sub-lexicon created by selecting from Valex 1 only SCFs whose relative frequency is higher than a SCF-specific threshold.

Valex 3: A sub-lexicon created by selecting from Valex 1 SCFs which are also listed in the manually built ANLT [18] and the COMLEX syntax [19] dictionaries, and those whose relative frequency is higher than a SCF-specific threshold.

[3] See http://verbs.colorado.edu/verb-index/index.php for details.

[4] See [1] for the full description of these versions of VALEX and the details of their evaluation.

For our 204 test verbs, these three lexicons include 44, 5, and 5 SCFs per verb on average. According to the evaluation reported in [1], the SCF accuracy of Valex 1, 2, and 3 is 21.9, 58.6 and 83.7 according to F-measure, respectively, when evaluated on a set of 183 test verbs. Although standard text processing and parser errors result in some noise, the main source of error in SCF acquisition is the difficulty of argument-adjunct distinction. Therefore the higher the SCF accuracy of a lexicon, the higher the number of genuine arguments in the SCFs (e.g. *I sang* ***a song*** correctly analysed as SCF NP), as opposed to adjuncts (e.g. *I sang* ***in the party*** incorrectly analysed as SCF PP), and vice versa. Thus by controlling the SCF accuracy we can control the proportion of arguments and adjuncts in input data and examine the effects on classification.

A lexical entry for each verb and SCF combination provides e.g. the frequency of the entry in corpora, the POS tags of verb tokens, the argument heads in argument positions, and the prepositions in PP slots. We experimented with five feature sets:

Feature set 1: SCFs and their frequencies

Feature set 2: Feature set 1 with two high frequency PP frames parameterized for prepositions: the simple PP (e.g. *they apologized to him*) and NP-PP (e.g. *he removed the shoes from the bag*) frames refined according to the prepositions provided in the VALEX SCF entries (e.g. PP_*at*, PP_*on*, PP_*in*).

Feature sets 3-5: Feature set 2 with additional 3, 8 and 13 high frequency PP frames parameterized for prepositions, respectively.

Although prepositions are an important part of the syntactic description of Levin style classes and therefore feature set 5 should be the most informative one (and feature set 1 the least informative one), we controlled the number of PP frames parameterized for prepositions in order to examine the effects of sparse data in automatic classification.

4 Classification

4.1 Preparing the Data

A feature vector was constructed for each verb. For example, Valex 1 includes 107, 287 and 305 SCF types for feature sets 1, 2, and 3, respectively. Each feature corresponds to a SCF type. Its value is the relative frequency of the SCF with the verb in question. Some of the feature values are zero because most verbs take only a subset of the possible SCFs.

4.2 Machine Learning Methods

We experimented with five methods for classification: four supervised ones (K nearest neighbours, support vector machines, maximum entropy, Gaussian) and one unsupervised one (cost-based pairwise clustering). To our knowledge, two of these methods (the Gaussian and clustering methods) have not been previously used for verb classification. The free parameters of classifiers were optimised for each feature set by (i) defining the value range (as explained in below sections), and (ii) searching for the optimal value on the training data using 10 fold cross validation (section 5.2).

K Nearest Neighbours K Nearest Neighbours (KNN) is a memory-based classification method based on the distances between verbs in the feature space. For each verb in the test data, we measure its distance from each verb in the training data. We then assign it the label which is the most frequent among the top K closest training verbs. We use the entropy-based Jensen-Shannon (JS) divergence as the distance measure:

$$JS(P,Q) = \frac{1}{2}\left[D(P\|\frac{P+Q}{2}) + D(Q\|\frac{P+Q}{2})\right]$$

The range of the parameter K is 2-20.

Support Vector Machines The Support Vector Machines (SVM) [20] try to find a maximal margin hyperplane to separate between two groups of verb feature vectors. In practice, a linear separator is unlikely to exist in the original feature space. SVM uses a kernel function to map the original feature vectors to a higher dimension space. The 'maximal margin' optimizes our choice of dimensionality to avoid over-fitting. We used Chang and Lin's LIBSVM library [21] to implement the SVM. Following [22], we use the radial basis function as the kernel function:

$$K(x_i, x_j) = \exp\left(-\gamma||x_i - x_j||^2\right), \gamma > 0$$

γ and the cost of the error term C (the penalty for margin errors) are optimized. The search ranges of [22] are used:

$$C = 2^{-5}, 2^{-3}, \ldots, 2^{15}, 2^{17}\,; \gamma = 2^{-17}, 2^{-15}, \ldots, 2^{1}, 2^{3}$$

Maximum Entropy Maximum entropy (ME) constructs a probabilistic model that maximizes entropy on test data subject to a set of feature constraints. If verb x is in class 10.1 and takes the SCF 49 (NP-PP) with the relative frequency of 0.6 in feature function f, we have:

$$f(x,y) = 0.6 \text{ if } y = 10.1 \text{ and } x = 49$$

The expected value of a feature f with respect to the empirical distribution (training data) is:

$$\tilde{\mathcal{E}}(f) \equiv \sum_{x,y} \tilde{p}(x,y) f(x,y)$$

The expected value of the feature f (on test data) with respect to the model $p(y|x)$ is:

$$\mathcal{E}(f) \equiv \sum_{x,y} \tilde{p}(x) p(y|x) f(x,y)$$

$\tilde{p}(x)$ is the empirical distribution of x in the training data. We constrain $\mathcal{E}(f)$ to be the same as $\tilde{\mathcal{E}}(f)$:

$$\mathcal{E}(f) = \tilde{\mathcal{E}}(f)$$

The model must maximize the entropy $H(Y|X)$:

$$H(Y|X) \equiv -\sum_{x,y} \tilde{p}(x) p(y|x) \log p(y|x)$$

The constraint-optimization problem is solved by the Lagrange multiplier [23]. We used Zhang's [24] maximum entropy toolkit for implementation. The number of iterations i (5-50) of the parameter estimation algorithm is optimised.

Gaussian The Gaussian model assumes that the SCF frequencies of verbs in a class are normally distributed with averages and standard deviations characteristic of the class. We assume the dimensions of a feature vector to be independent of each other. The covariance matrix of the statistical distribution of features is diagonal. Suppose x is a feature vector of a verb, which has d dimensions, and y is the mean vector of the verb class which has N verbs with feature vectors $x_1 \ldots x_n$. The likelihood is:

$$p(x|y) = \Pi_{a=1}^{d} \frac{1}{\sqrt{2\pi\sigma_a^2}} \exp\left(-\frac{(x_i-\mu_a)^2}{2\sigma_a^2}\right)$$

The mean and variance of a dimension a are:

$$\mu_a = \frac{\sum_{i=1}^{n} x_i}{N} \, , \sigma_a^2 = \frac{\sum_{i=1}^{n} (x_i-\mu_i)^2}{N}$$

To predict the class membership of a test verb, we calculate its likelihood for each class and choose the class with the highest likelihood.

Pairwise Clustering We adopt a cost-based framework for pairwise clustering (PC) [25] where a cost criterion guides the search for a suitable clustering configuration. This criterion is realized through a cost function $H(S, M)$ where

(i) $S = \{\mathrm{sim}(a, b)\}$, $a, b \in A$: a collection of pairwise similarity values, each of which pertains to a pair of data elements $a, b \in A$.

(ii) $M = (A_1, \ldots, A_k)$: a candidate clustering configuration, specifying assignments of all elements into the disjoint clusters (that is $\cup A_j = A$ and $A_j \cap A_{j'} = \phi$ for every $1 \leq j < j' \leq k$).

The cost function is defined as follows:

$$H = -\sum n_j \cdot \mathrm{Avgsim}_j \, ,$$
$$\mathrm{Avgsim}_j = \frac{1}{n_j \cdot (n_j - 1)} \sum_{\{a,b \in A_j\}} \mathrm{sim}(a, b)$$

where n_j is the size of the j^{th} cluster and Avgsim_j is the average similarity between cluster members. The similarity is measured by the JS divergence.

The number of clusters, k, is specified as an input parameter. We varied the value of k from 10 to 35. The best performance was obtained for k values close to the number of gold standard classes ($n = 17$). We report the results for $k = 17$.

5 Experiments

5.1 Methodology for Supervised Methods

We split the data into training and test sets using two methods. The first is 'leave one out' *cross-validation* where one verb is held out as test data, and the remaining N-1 verbs are used as training data. The overall accuracy is the average accuracy of N rounds. The second method is *re-sampling*. For each class, 3 verbs are selected randomly as test data and 9 are used as training data. The process is repeated 30 times and the average result is recorded.

5.2 Measures

Supervised methods are evaluated using accuracy – the percentage of correct classifications out of all the classifications:

$$Accuracy = \frac{truePositives}{\text{Number of verbs}}$$

When evaluating the performance at class level, precision and recall are calculated as follows:

$$Precision = \frac{truePositives}{truePositives + falsePositives}$$
$$Recall = \frac{truePositives}{truePositives + falseNegatives}$$

F-measure is the balance over recall and precision. We report the average F-measure over the 17 classes. Given there are 17 classes in the data, the accuracy of randomly assigning a verb into one of the 17 classes is $1/17 \approx 5.8\%$.

Clustering is evaluated using *unique purity* (uPUR) which evaluates the output as if it were the output of a classifier. For each cluster, the *dominant class* is the class with most cluster members. If a class is dominant in several clusters we choose the cluster with the highest number of class members. This is analogous to the output of a classifier where only one output class can correspond to an actual class. uPUR is the total number of these verbs divided by the number of verbs N. The experiments were run 50 times on each input to get the distribution of performance due to the randomness in the initial clustering.

5.3 Results from Quantitative Evaluation

Table 2 shows the average performance of each classifier and feature set according to 'leave one out' cross-validation when the features are extracted from Valex 1. Each classifier performs considerably better than the random baseline. The simple KNN method produces the lowest accuracy (44.1-54.9). SVM and ME perform better (47.1-57.8 and 47.5-59.8 accuracy, respectively), with GS yielding the best accuracy (49.5-64.2). The clustering method performs similarly with KNN, yielding uPUR of 39.6-51.6.

The performance of all methods improves sharply when moving from feature set 1 to the refined feature set 2: both accuracy and F-measure improve c. 10%. When moving further to feature set 3 (which includes a higher number of low frequency PP features) KNN worsens clearly (c. 5% in accuracy and F-measure) while the other methods perform similarly. With sparser feature sets 4-5, KNN, PC SVM and ME show similar performance than with feature set 3 (the changes in SVM and ME are not statistically significant). GS is the only method which performs the best with the most refined feature set 5 (64.2 accuracy and 62.5 F-measure). ME produces the same results as with feature set 4: 59.8 accuracy and 59.9 F-measure. The differences in accuracy and F-measure of GS and ME are significant at the 3.4σ and 2σ level, respectively.

The resampling results in table 3 reveal that some classifiers perform worse than others when less training data is available[5]. KNN produces considerably lower results with resampling, particularly with the sparse feature set 5: 20.3 F-measure vs. 48.5 with

[5] Recall that the amount of training data is smaller with resampling evaluation, see section 5.2.

	Feature set 1		Feature set 2		Feature set 3		Feature set 4		Feature set 5	
	ACC	F	ACC	F	ACC	F	ACC	F	ACC	F
RAND	5.8		5.8		5.8		5.8		5.8	
KNN	44.1	44.0	54.9	53.9	49.5	48.2	49.5	48.3	49.5	48.5
ME	47.5	47.6	59.3	59.9	59.3	60.0	59.8	59.9	59.8	59.9
SVM	47.1	47.8	57.8	57.9	57.8	58.2	57.3	57.5	57.3	57.4
GS	49.5	46.2	59.3	57.1	59.3	56.5	59.8	56.6	64.2	62.5
	uPUR		uPUR		uPUR		uPUR		uPUR	
PC	$39.6\% \pm 1.5$		$51.4\% \pm 2.5$		$51.5\% \pm 2.6$		$51.6\% \pm 2.5$		$51.2\% \pm 3.7$	

Table 2. 'Leave one out' cross-validation results for supervised methods; PC results with $\mathcal{K} = 17$ clusters. The standard deviation of ACC and F is $\sigma = 0.9$. Feature sets extracted from Valex 1.

	Feature set 1		Feature set 2		Feature set 3		Feature set 4		Feature set 5	
	ACC	F	ACC	F	ACC	F	ACC	F	ACC	F
RAND	5.8		5.8		5.8		5.8		5.8	
KNN	37.3	36.5	42.7	42.6	27.1	28.2	23.6	24.0	20.5	20.3
ME	47.1	47.0	58.1	58.1	60.1	59.8	59.8	59.6	57.5	57.8
SVM	47.3	47.7	56.8	57.1	54.4	54.6	56.8	56.9	55.6	55.5
GS	39.0	40.0	49.9	51.4	50.0	51.7	52.4	53.2	54.0	55.5
σ	6.0	6.1	6.5	6.8	6.1	6.2	5.7	5.8	5.8	5.9
	uPUR		uPUR		uPUR		uPUR		uPUR	
PC	$39.6\% \pm 1.5$		$51.4\% \pm 2.5$		$51.5\% \pm 2.6$		$51.6\% \pm 2.5$		$51.2\% \pm 3.7$	

Table 3. Re-sampling results for supervised methods; PC results with $\mathcal{K} = 17$ clusters. The σ line presents the average standard deviation for each measure. Feature sets extracted from Valex 1.

cross-validation. Also other methods perform worse. Although a considerable improvement can be seen in GS with the use of more sophisticated feature sets, ME produces the best results with resampling.

These results were obtained using features extracted from Valex 1. Table 4 shows 'leave one out' cross-validation results for all the three versions of Valex with feature set 5. Each method performs the best with the very noisy Valex 1 which includes a lot of adjunct data in addition to argument data. Most methods perform the second best with Valex 3 which includes highly accurate argument data supplemented with high frequency adjunct data. In these results with feature set 5 we can see the biggest difference in GS which yields 62.5 F-measure with Valex 1, 53.5 with Valex 3 and 50.1 with Valex 2. For other methods big differences can be detected between the three lexicons when using less ambitious feature sets 1-3.[6]

	Valex 1			Valex 2			Valex 3		
	ACC	P	F	ACC	P	F	ACC	P	F
RAND	5.8			5.8			5.8		
KNN	49.5	47.5	48.5	46.6	42.4	44.4	47.1	44.3	45.7
ME	59.8	59.7	59.9	53.9	54.9	54.4	59.3	60.1	59.7
SVM	57.3	57.5	57.4	55.8	55.2	55.5	53.9	53.5	53.7
GS	64.2	60.8	62.5	53.9	46.8	50.1	56.3	51.0	53.5
	uPUR			uPUR			uPUR		
PC	$51.2\% \pm 3.7$			$44.9\% \pm 1.5$			$48.3\% \pm 1.0$		

Table 4. 'Leave one out' cross-validation results with the three versions of Valex (feature set 5). The standard deviation of ACC, P, and F is $\sigma = 0.9$.

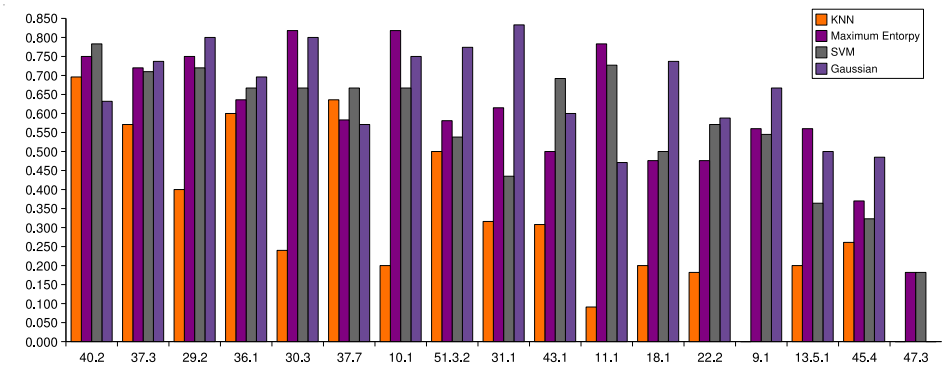

Fig. 1. Class level F-score for feature set 5 (cross-validation)

5.4 Qualitative Evaluation

Figure 1 shows the F-measure for 17 individual classes when supervised methods are used with feature set 5 extracted from Valex 1. Levin classes 40.2, 29.2, and 37.3 (see table 1) which take fewer prepositions with higher frequency have the best average performance (65% or more) among all the methods, and classes 47.3, 45.4 and 18.1 the worst (40% or less). GS outperforms other methods with 11 of the 17 classes.

Figure 2 shows the error matrix for ME with feature set 5. Examination of the worst performing class 47.3 (MODES OF BEING INVOLVING MOTION verbs) illustrates well the various error types. For ME 10 of the 12 verbs in this class are classified incorrectly:

- **3** (*flutter, falter, teeter*) in class 43.1 (LIGHT EMISSION verbs): Verbs in 47.3 and 43.1 describe intrinsic properties of their subjects (e.g. *a jewel sparkles*, *a flag flutters*). Their similar alternations and PP SCFs make it difficult to separate them on syntactic grounds.
- **2** (*swirl, waft*) in class 51.3.2 (RUN verbs): 47.3 and 51.3.2 share the meaning component of motion. Their members take similar alternations and SCFs, which causes the confusion.
- **2** (*quiver, vibrate*) in class 36.1 (CORRESPOND verbs): 47.3 and 36.1 are semantically very different, but their members take similar intransitive and PP SCFs with high frequency.

[6] These figures are not show in table 4 due to space restrictions.

	10.1	11.1	13.5.1	18.1	22.2	29.2	30.3	31.1	36.1	37.3	37.7	40.2	43.1	45.4	47.3	51.3.2	9.1
10.1	-	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11.1	0	-	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0
13.5.1	1	1	-	2	0	1	0	0	0	0	0	0	0	1	0	0	0
18.1	0	0	1	-	0	0	1	0	0	0	0	0	0	1	0	0	1
22.2	0	1	0	0	-	0	0	0	1	0	0	0	0	0	0	0	2
29.2	0	0	1	0	0	-	0	0	0	0	0	0	0	0	0	0	1
30.3	0	0	0	0	0	0	-	0	0	1	0	1	0	0	0	0	0
31.1	1	0	0	2	0	0	0	-	0	0	0	0	0	2	0	0	1
36.1	0	0	0	0	2	0	0	0	-	0	0	0	0	0	2	0	0
37.3	0	0	0	0	0	0	1	0	0	-	2	2	0	0	0	1	0
37.7	0	0	0	0	0	0	1	0	0	1	-	0	0	1	2	0	0
40.2	0	0	0	0	0	0	0	0	0	1	0	-	0	0	0	0	0
43.1	0	0	0	1	0	0	0	0	1	0	0	0	-	0	3	1	0
45.4	1	0	0	0	1	1	0	3	1	0	1	0	0	-	1	0	0
47.3	0	0	0	0	1	0	0	0	2	0	0	0	4	1	-	0	0
51.3.2	0	1	1	1	0	0	0	1	0	0	1	0	2	0	2	-	0
9.1	0	0	1	1	3	1	0	0	0	0	1	0	0	0	0	0	-

Fig. 2. Error matrix for ME, a column indicates a correct class and a row a mistakenly predicted class.

- **2** (*quake, wiggle*) in class 37.7 (SAY verbs): 47.3 differs in semantics and syntax from 37.7. The confusion is due to idiosyncratic properties of individual verbs.
- **1** (*sway*) in class 45.4 (OTHER CHANGE OF STATE verbs): Classes 47.3 and 45.3 are semantically different. Their similar PP SCFs explains the misclassification.

Interestingly, the errors of GS with class 47.3 are very similar. Both methods confuse 47.3 with classes 51.3.2, 37.7 and 43.1, but GS assigns as many as 5 verbs to class 51.3.2. Most errors concern classes which are in fact semantically related. Unfortunately no existing gold standard comprehensively captures the semantic relatedness of Levin classes. This kind of error analysis could be used as a step towards creating one. Other errors concern semantically unrelated but syntactically similar classes – cases which we can try to address with careful feature engineering. Some errors relate to syntactic idiosyncracy. These show the true limits of lexical classification - the fact that the correspondence between the syntax and semantics of verbs is not always perfect.

6 Discussion and Conclusion

In our experiments, rich syntactic features incorporating extensive information about both arguments and adjuncts of verbs proved the most useful for verb classification. This result not only confirms the earlier observations that adjuncts can be useful for the task [4, 5], but demonstrates that adjuncts are actually very important for the task. SCFs containing arguments have been used as primary features in manual verb classification (Levin, 2003) for good theoretical reasons. However, adjuncts may be absent in manual work for practical reasons. It may be that they are best identified via general statistics about verb usages in corpora (rather than via specific syntactic tests). This kind of statistical information would be prohibitively difficult to capture manually.

Our best performing method (the new GS method) produced the best results with the challenging feature set 5 which is the most informative feature set but challenging to most methods due to its sparseness. The 64.2 accuracy and 62.5 F-measure produced by GS is especially promising considering that in these experiments focussing on the basic

characteristics of syntactic features we performed no sophisticated feature engineering or selection based on the properties of the target classification.

Due to differences in target languages and evaluation procedures, direct comparison of our results against previously published ones is difficult. The closest comparison point is the recent experiment reported in [5] which involved classifying 835 English verbs to 14 Levin classes using SVM. Features were specifically selected via analysis of alternations that are used to characterize Levin classes. Both (i) shallow syntactic features (syntactic slots obtained using a chunker) and (ii) deep ones (SCFs extracted using Briscoe and Carroll's system) were used. The accuracy was 58% with (i) and only 38% with (ii). This experiment is not directly comparable with ours as we classified a smaller number of verbs (204) to a higher number of Levin classes (17) (i.e. we had less training data) and did not select the optimal set of features using Levin's alternations. We nevertheless obtained better accuracy with our best performing method, and better accuracy (47%) with the same method (SVM) when the comparable feature set 1 was acquired using the very same subcategorization acquisition system. [5] does not reveal whether noise was filtered from the system output to obtain a set of SCFs containing mostly arguments. In the light of our experiments it seems likely that this was done, and that using much larger and noisier SCF data including a high proportion of adjuncts explains our better result.

Further experiments are required to determine the optimal set of features. The fact that we obtained the best results using noisy Valex 1 and the second best using highly accurate but filtered Valex 3 suggests that combining these two lexicons into a single lexicon may be the best approach. It would yield a large lexicon with good coverage in adjuncts and high accuracy in arguments. In the future, we also plan to improve the features by e.g. enriching them with additional syntactic information recoverable from the parser output and/or available in VALEX lexical entries (see section 3). Incorporating semantic information e.g. about selectional preferences [4] would also be interesting, although this has proved challenging in earlier works.

We evaluated our features using both supervised and unsupervised methods. This was important since the two types of methods can serve different purposes: unsupervised methods are ideal for class discovery (e.g. in new domains) while supervised methods are useful for supplementing existing classifications with additional members. However, supervised techniques may perform optimally when combined with unsupervised techniques and a large unlabelled data [26]. In the future, we will therefore investigate a semi-supervised approach to verb classification.

Acknowledgement

This work was supported by the EPSRC project 'ACLEX' and the Royal Society, UK.

References

1. Korhonen, A., Krymolowski, Y., Briscoe, T.: A large subcategorization lexicon for natural language processing applications. In: Proceedings of LREC. (2006)
2. Merlo, P., Stevenson, S.: Automatic verb classification based on statistical distributions of argument structure. Computational Linguistics **27** (2001) 373–408

3. Korhonen, A., Krymolowski, Y., Collier, N.: Automatic classification of verbs in biomedical texts. In: Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual meeting of the ACL. (2006) 345–352
4. Schulte im Walde, S.: Experiments on the automatic induction of german semantic verb classes. Computational Linguistics **32** (2006) 159–194
5. Joanis, E., Stevenson, S., James, D.: A general feature space for automatic verb classification. Natural Language Engineering **Forthcoming** (2007)
6. Dorr, B.J.: Large-scale dictionary construction for foreign language tutoring and interlingual machine translation. Machine Translation **12** (1997) 271–322
7. Prescher, D., Riezler, S., Rooth, M.: Using a probabilistic class-based lexicon for lexical ambiguity resolution. In: 18th International Conference on Computational Linguistics, Saarbrücken, Germany (2000) 649–655
8. Swier, R., Stevenson, S.: Unsupervised semantic role labelling. In: Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing, Barcelona, Spain (2004) 95–102
9. Dang, H.T.: Investigations into the Role of Lexical Semantics in Word Sense Disambiguation. PhD thesis, CIS, University of Pennsylvania (2004)
10. Shi, L., Mihalcea, R.: Putting pieces together: Combining FrameNet, VerbNet and WordNet for robust semantic parsing. In: Proceedings of the Sixth International Conference on Intelligent Text Processing and Computational Linguistics, Mexico City, Mexico (2005)
11. Jackendoff, R.: Semantic Structures. MIT Press, Cambridge, Massachusetts (1990)
12. Levin, B.: English Verb Classes and Alternations. Chicago University Press, Chicago (1993)
13. Miller, G.A.: WordNet: An on-line lexical database. International Journal of Lexicography **3** (1990) 235–312
14. Schulte im Walde, S.: Clustering verbs semantically according to their alternation behaviour. In: Proceedings of COLING, Saarbrücken, Germany (2000) 747–753
15. Kipper, K., Dang, H.T., Palmer, M.: Class-based construction of a verb lexicon. In: AAAI/IAAI. (2000) 691–696
16. Briscoe, E.J., Carroll, J.: Automatic extraction of subcategorization from corpora. In: Proceedings of the 5^{th} ACL Conference on Applied Natural Language Processing, Washington DC (1997) 356–363
17. Briscoe, E.J., Carroll, J.: Robust accurate statistical annotation of general text. In: Proceedings of the 3^{rd} LREC, Las Palmas, Gran Canaria (2002) 1499–1504
18. Boguraev, B., Briscoe, T.: Large lexicons for natural language processing: utilising the grammar coding system of ldoce. Comput. Linguist. **13** (1987) 203–218
19. Grishman, R., Macleod, C., Meyers, A.: Comlex syntax: building a computational lexicon. In: Proceedings of the 15th conference on Computational linguistics, Morristown, NJ, USA, Association for Computational Linguistics (1994) 268–272
20. Vapnik, V.N.: The nature of statistical learning theory. Springer-Verlag New York, Inc., New York, NY, USA (1995)
21. Chang, C., Lin, J.: LIBSVM: a library for support vector machines. (2001)
22. Hsu, W., Chang, C., Lin, J.: A practical guide to support vector classification (2003)
23. Pietra, S.D., Pietra, J.D., Lafferty, J.D.: Inducing features of random fields. IEEE Transactions on Pattern Analysis and Machine Intelligence **19** (1997) 380–393
24. Zhang, L.: Maximum Entropy Modeling Toolkit for Python and C++. (2004)
25. Puzicha, J., Hofmann, T., Buhmann, J.M.: A theory of proximity-based clustering: structure detection by optimization. Pattern Recognition **33** (2000) 617–634
26. Ando, R.K., Zhang, T.: A high-performance semi-supervised learning method for text chunking. In: Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics. (2005) 1–9