

Automatic Lexical Classification

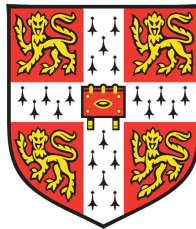
Lin Sun, Girton College

Candidate for the degree of

M.Phil. in Computer Speech, Text, and Internet Technology

Under the supervision of Dr Anna Korhonen

July 9, 2007



Declaration of Originality

I, Lin Sun of Girton college , being a candidate for M.Phil in Computer Speech, Text and Internet Technology, hereby declare that this dissertation and the work described in it are my own work, unaided except as may be specified below, and that the dissertation does not contain material that has already been used to any substantial extent for a comparable purpose.

Signed

Date

The length of this document is estimated to be 15,000.

Acknowledgement

I would like to thank my supervisor Dr. Anna Korhonen for her tremendous help on the project, and for her previous works on lexical acquisition and verb classification. I consider it as a privilege to work under her supervision. Moreover, I would like to thank Dr. Yuval Krymolowski for his support on the machine learning and other parts of my work. I would also like to thank Wenxiao Yuan for the discussions on Gaussian Model.

Abstract

Verbs that have similar meaning and syntactic behaviour can be grouped into lexical classes (Levin, 1993). These classes are useful for many NLP tasks. In this study, we investigate the use of supervised machine learning methods, including the K nearest neighbours, maximum entropy (Berger et al., 1996), Support Vector Machines (Vapnik, 1995) and the Gaussian model for verb classification, using as syntactic features SCF distributions in a large automatically acquired subcategorization lexicon recently built using Briscoe and Carroll's (1997) system. We experiment with SCF feature sets previously employed by Korhonen et al. (2003, 2006) and also with a new extended feature set. With the best performing method and feature set combination, we obtain the classification accuracy of 64.2% when classifying 204 verbs into 17 classes. This is clearly a better result than obtained by clustering the same data and, although our test set is not identical to that recently used by Joanis et al. (2007), our good result challenges their conclusion that features extracted by parsing and subcategorization acquisition are not helpful for verb classification. We discuss reasons for this, and outline directions for future work.

Contents

1	Introduction and Motivation	5
1.1	Statement of Purpose	5
1.2	Outline of Study	6
2	Background	8
2.1	Subcategorization Frames and Diathesis Alternations	8
2.2	Verb Classes	9
2.3	Previous Work	10
2.4	Automatic Acquisition of Subcategorization Frames	11
3	Automatic Verb Classification	13
3.1	Feature Sets	13
3.1.1	Feature Set 1: Basic Features	13
3.1.2	Feature Set 2: Refined Features	13
3.1.3	Feature Set 3: New Features	15
3.2	Data Preprocessing	15
3.3	Classification Methods	15
3.3.1	K Nearest Neighbours	17
3.3.2	Maximum Entropy Model	17
3.3.3	Support Vector Machines	19
3.3.4	Gaussian Model	20
4	Experiments	22
4.1	Data	22
4.2	Methodology	22
4.3	Parameter Estimation	24
4.4	Experimental Evaluation	25
4.4.1	Feature Set 1: Basic Features	26
4.4.2	Feature Set 2: Refined Features	26
4.4.3	Feature Set 3: New features	30
4.5	Summary of Quantitative Evaluation	35
4.6	Qualitative Analysis	35
4.7	Related Work	39
4.7.1	Recent work of Joanis, Stevenson and James (2007)	39
4.7.2	Comparison with Clustering	39

5	Conclusion and Future Work	41
5.1	Conclusion	41
5.2	Future Work	41
5.2.1	Dealing with Polysemous Verbs	41
5.2.2	Improving the Input Data	42
5.2.3	Feature selection	42
5.2.4	Combining Approaches	42

List of Figures

3.1	Sample features from the 3 feature sets for the verb <i>put</i>	16
4.1	Class level F-Score for Feature set 1	27
4.2	Class level F-Score for Feature set 2	29
4.3	Class specific F-Score results with 13 new features added from Feature set 3	33
4.4	Sensitivity of 3 machine learning methods to the sample size . . .	36
4.5	Verb frequency and accuracy using maximum entropy method with Feature Set 1	36
4.6	Error matrix for Gaussian	37
4.7	Error matrix for Maximum entropy	37

List of Tables

3.1	15 SCFs refined by their PPs, ranked by their frequency	14
3.2	Verb tense tags	14
4.1	Experiment data	23
4.2	Leave one out cross validation test for KNN to select the distance metric, k=4	25
4.3	'Leave one out' cross validation result for Feature set 1	26
4.4	Re-sampling results for Feature set 1	27
4.5	Leave one out cross validation result on Feature set 2 (refined feature set)	28
4.6	Re-sampling results for Feature set 2 (refined feature set)	29
4.7	Average contribution of Feature set 2 to all the methods for each class as compared with Feature set 1	30
4.8	Leave one out cross validation results with 3 new PP features added from Feature set 3	31
4.9	Re-sampling results with 3 new features added from Feature set 3	32
4.10	Leave one out cross validation result on 8 new features added from Feature set 3	32
4.11	Re-sampling results with 8 new features added from Feature set 3	32
4.12	Leave one out cross validation results on 13 new features added from Feature set 3	33
4.13	Re-sampling results with 13 new features added from Feature set 3)	33
4.14	Contribution of Feature set 2 as compared with Feature set 1 and contribution of Feature set 3 as compared with Feature set 2 for the Gaussian model	34
4.15	Leave one out cross validation results with the full Feature set 3, verb tense features included	34
4.16	Evaluation ($uPUR$) of the verb clusters	40

Chapter 1

Introduction and Motivation

1.1 Statement of Purpose

Verbs that share the common element of meaning and syntactic behaviour can be grouped into lexical classes (e.g. Pinker, 1989; Jackendoff, 1990; Levin, 1993). These classes generalize over the linguistic properties of verbs without defining the idiosyncratic details for each verb. For example, the verbs *murder*, *slay* and *massacre* all belong to the class of MURDER verbs.

Although there is rarely a perfect correspondence between the semantics and syntax of verbs, lexical classes can be helpful for a number of natural language tasks. For example, they can help to reduce the redundancy in the lexicon, since verbs in a class share some properties. They can also help and address the problem of data sparseness which affects many NLP tasks. A verb class can predict and refine the property of its member verbs, when not enough empirical evidence is available. The verb classifications have been used to help tasks, such as language machine translation (Dorr, 1997), document classification (Klavans and Kan, 1998), and word sense disambiguation (Dorr and Jones, 1996).

The largest manually built verb classification available for English is that of Levin's (1993) classification. It mainly deals with verbs which take noun and prepositional phrase complements, although the latest version incorporated in VerbNet (<http://verbs.colorado.edu/verb-index/index.php>) provides additional classes for verbs taking sentential complements. However, the classification does not provide exhaustive lists of member verbs with the classes, and does not include information about the likelihood of different classes for words. This information is difficult to obtain by hand, especially because it varies from one domain to another.

Recently, automatic verb classification has received considerable amount of interest (Merlo and Stevenson, 2001; Joanis and Stevenson, 2003; im Walde, 2006; Korhonen et.al. 2003 and 2006). The automatic approach is cheap and gathers statistical information during the acquisition process. Like manual classification, automatic classification is based on the assumption that the shared semantic behaviour verbs can be largely inferred by their shared syntactic behaviour. The methods proposed so far use machine learning techniques to clas-

sify syntactic features extracted from corpus data using part-of-speech tagging or statistical parsing techniques.

The syntactically richest features used in this work so far have been comprehensive sets of automatically extracted subcategorization frame (SCF) frequency distributions. So far these features have been mainly classified using unsupervised (clustering) methods. In English, this work has been conducted by Korhonen et al. (2003, 2006) who have clustered SCF distributions extracted from corpus data using the system of Briscoe and Carroll (1997). Korhonen et al. (2003) used the Nearest Neighbour and Information Bottleneck methods (Tishby et al., 1999) to cluster polysemous verbs in general corpora, while Korhonen et al. (2006b) used (in addition) Probabilistic Latent Semantic Analysis (Hofmann, 2001) and a modified version of the Information Bottleneck (Dimitrov and Miller, 2001) to cluster verbs in biomedical texts. These experiments have shown promising results. Due to its easy portability, clustering is ideal for classification in domains where the set of verb classes is unknown. However, supervised or semi-supervised techniques may provide optimal performance when the basic set of classes (and some member verbs) are already known, but one wishes to extend the classification.

The only published work which involves using supervised methods to classify SCF distribution data for English so far is the recent work by Joanis, Stevenson and James (2007). They perform an experiment with 835 verbs (from 14 classes) where they use Support Vector Machines (Vapnik, 1995) to classify a) shallow syntactic features obtained using a chunker and b) SCF distributions obtained using Briscoe and Carroll's (1997) system. They report 58% accuracy with a) and only 38% with b). On the basis of this result, they conclude that using syntactically rich subcategorization features is not helpful for English verb classification. Their poor result is surprising since better results than 38% accuracy have been obtained when clustering SCF distributions extracted using Briscoe and Carroll's system. Their conclusion is surprising as well, since verb classes are primarily based on alternating verb subcategorization patterns.

1.2 Outline of Study

In this study, we investigate the use of four supervised machine learning methods, including the K nearest neighbours, maximum entropy (Berger et al., 1996), Support Vector Machines (Vapnik, 1995) and the Gaussian model for verb classification, using as syntactic features SCF distributions in a large automatically acquired subcategorization lexicon built using Briscoe and Carroll's system. We experiment with SCF feature sets employed by Korhonen et al. (2003, 2006) and also with a new extended feature set. With the best performing method and feature set combination, we obtain the classification accuracy of 64.2% when classifying 204 verbs into 17 classes. This is clearly better result than obtained when clustering the same data and, although our test set is not identical to that of Joanis et al. (2007), the good result challenges their recent work. We discuss reasons for this, and outline directions for future work.

The report is structured as follows:

The chapter 2 describes the background and previous work on verb classification. It introduces the concepts of subcategorization, diathesis alternation and lexical verb class. We review previous work on verb classification and finally

explain the automatic subcategorization acquisition process.

In chapter 3, we introduce our approach to automatic verb classification. We start by describing our 3 feature sets (basic, refined and new features) and four classification methods: the K nearest neighbours, Maximum entropy, Support Vector Machines and the Gaussian model.

In chapter 4, we report verb classification experiments. We describe the data, experimental methodologies and evaluation measures used. We report the quantitative and qualitative analysis of the results for each method and feature set combination using 'leave one out' cross validation and re-sampling.

In chapter 5, we summarise our contributions and discuss several directions for future work.

Chapter 2

Background

2.1 Subcategorization Frames and Diathesis Alternations

Different subcategories of verbs impose different constraints on the number and the type of arguments they take. The syntactic variation can be captured by frames called subcategorization frames (SCFs). Similar verbs take similar SCFs and participate in similar alternating pairs of SCFs (*diathesis alternations*). For example, the verb *eat* can take a simple NP frame which consists of one subject and one direct object (Example 1).

Example 1 *I ate an apple.*

On the other hand, the verb *put* cannot take this frame (Example 2). It requires at least three syntactic arguments: subject, direct object and an indirect object, as shown in the SCF NP-PP in Example 3.

Example 2 *I put the apple**

Example 3 *I put the apple on the table.*

The semantics of the verb is said to determine its syntactic behaviour. To prove the interdependency, Hale and Keyser (1987) presented the below example showing that the verb semantics is a key to the verb syntactic behaviour. Suppose we don't know the meaning of the archaic English word 'gally', as in *The sailor galled the whales*. Some people might suggest that the word means 'see', as in *The sailor saw the whales*, while others might suggest that it means 'frighten', as in *The sailor frightened the whales*. At this point, we don't know which meaning is correct. To solve this problem, Hale and Keyser looked at the middle transitivity alternation, where the subject of the intransitive usage of the verb can be placed as the object the the transitive usage. For example, for the verb 'read', we can say 'this book reads well'; or we can move the subject 'book' to the object position, as in 'I read this book'. The people who believe 'gally' means 'see' would not allow for the sentence 'Whales gally easily', because middle transivity alternation is not allowed for 'see' (**Whales see easily*). The people who believe the meaning is 'frighten' would allow for the middle transivity alternation, as in *Whales frighten easily*. In fact, the verbs

with middle transitivity alternation would normally cause a change of state, e.g. *frighten, open, split, crush*.

From this example, we can see that a verb's behaviour has a very strong relationship with its semantics.

2.2 Verb Classes

On the basis of the interdependency between the alternation behaviour of verbs and their meanings, Levin (1993) manually created classification for English verbs. The verbs within the same class share some meaning component and a similar syntactic frames.

Firstly, Levin defined 78 possible diathesis alternations. Then, each verb was associated with the possible set of alternations. Finally, the verbs with the same or similar alternation behaviour were assigned to the same class. Levin classified 3104 verbs into 49 broad verb classes, some of which divide further into subclasses, making the total number of classes 192. For example, the class of 'COOK' verbs includes verbs such as *cook, bake, boil, roast, heat . . .* which share the following syntactic behaviour and alternations (Levin, 1993):

1. Causative/Inchoative Alternation
 - (a) Jennifer baked the potatoes.
 - (b) The potatoes baked.
2. Middle Alternation:
 - (a) Jennifer baked Idaho potatoes.
 - (b) Idaho potatoes bake beautifully.
3. Instrument subject Alternation
 - (a) Jennifer baked the potatoes in the oven.
 - (b) This oven bakes potatoes well.
4. Resultative Phrase
 - (a) Jennifer boiled the pot dry.
 - (b) Jennifer baked the potatoes to a crisp.

Besides defining the alternations, Levin also defines other alternations that the verbs cannot take. The resulting verb classes are impressive in capturing the interdependency between the syntax and and the verb meaning.

VerbNet (<http://verbs.colorado.edu/verb-index/index.php>) is the most extensive on-line verb lexicon currently available for English. It provides detailed syntactic-semantic descriptions of Levin classes organized into a refined taxonomy. It has recently been extended with additional classes for verbs not covered comprehensively by Levin. It provides now 274 fine-grained classes which cover a wider number of verb senses and member verbs (5257). The resource has been used to aid a number of NLP applications such as automatic verb acquisition (Swift, 2005), semantic role labelling (Swier and Stevenson, 2004), robust semantic parsing (Shi and Mihalcea, 2005), word sense disambiguation (Dang,

2004), building conceptual graphs (Hensman and Dunnion, 2004), and creating a unified lexical resource for knowledge extraction (Croch and King, 2005), among others. However, the classification is still not comprehensive, it includes no statistical information about the likelihood of different classes for individual verbs, and is not helpful for processing texts in specific domains where verb senses (and thus classes) may only partially overlap with the ones in general language.

2.3 Previous Work

Since Levin released her classification, there have been attempts to automatically classify verbs into lexical classes.¹

Merlo and Stevenson (2001) presented an automatic classification of English intransitive verbs into three syntactical classes. The three classes are: unergative, unaccusative and object-drop verbs. The classification is based on argument structure and thematic relations. The features used are transitivity, causativity, animacy, passive voice and verb tense. Each verb is associated with a vector of 5 features. The vectors are the input to a decision tree classification algorithm. 60 verbs are classified into three classes with accuracy of 69.8%. The baseline accuracy is 33.3%.

In further research, Stevenson and Joanis (2003), classified 841 verbs into 14 Levin’s classes. They extended their feature set e.g. with features based on simple syntactic slots (identified by a chunker) and compared supervised, semi-supervised and unsupervised techniques. They used a decision tree learner (C5.0 <http://www.rulequest.com>) and a set of 220 features. The supervised decision tree classifier gave the best result. They also found that a semi-supervised approach where a classifier trained with five seed verbs from each verb class outperformed both manual selection of features and the unsupervised approach of Dash, Liu, and Yao (1997) which used an entropy measure to organise data into multi-dimensional space.

In the most recent research, Joanis et. al. (2007) performed experiment with almost identical data and very similar feature sets using support Vector Machines. They report 58% accuracy using this approach, and 38% accuracy when they replace their own feature sets with basic SCF distributions obtained using Briscoe and Carroll’s system (1997) which distinguishes 163 SCFs for verbs.

Schulte im Walde (2003) classified 168 mostly monosemous German verbs to 43 classes on the basis of cues related to SCF frequency information. A set of 38 SCFs was employed, some of which were parameterised for prepositions and enriched with information about selectional preferences. The frequency information was extracted automatically from data parsed using a statistical parser. She obtained 0.182 adjusted Rand Index when she used K-means (a standard clustering technique) for classification

Korhonen et al. (2003) conducted a similar experiment with English verbs. They reported similar results (60% modified cluster purity) in their more challenging experiment which involved assigning 110 difficult, highly polysemous English verbs into 34 Levin-style classes. Like Schulte-im-Walde, they clustered

¹Note that although purely semantic classification of verbs is a related and partly overlapping NLP task, we focus here only on previous work on lexical classification.

cues related to SCF frequency information, but employed more robust technology capable of also dealing with sense variation. Cues were extracted from corpus data using the SCF acquisition system of Briscoe and Carroll (1997). They parameterized two of the SCFs for prepositions, and applied a clustering technique (the Information Bottleneck) and the Nearest Neighbours method to assign verbs into classes corresponding to (any of) their senses in the corpus data.

Korhonen et al. (2006b) extended this system with additional clustering methods (Probabilistic Latent Semantic Analysis and a modified version of the Information Bottleneck) and applied it to the biomedical domain. 192 medium to high frequency verbs were selected from biomedical journals for the experiment. The resulting classification was highly accurate (they obtained 77 F-measure) and domain specific.

The approaches most related to ours are those of Schulte im Walde (2003) and Korhonen et al. (2003, 2006). These approaches make use of detailed syntactic (subcategorization frequency) features extracted automatically from corpora. So far this related work has used unsupervised (clustering) methods. Only Joanis et al. (2007) have attempted classifying SCF distributions using SVMs, but the results have been poor.

2.4 Automatic Acquisition of Subcategorization Frames

Our approach relies essentially on automatically extracted SCF frequency information. We make use of the Valex (Korhonen et al., 2006a) lexicon which was acquired automatically from five corpora and Web using Briscoe and Carroll (1997)'s system. It includes SCF and frequency information for 6,397 English verbs. There are 163 types of SCF in the lexicon, a superset of those in the (ANLT) (Boguraev and Briscoe, 1987) and the COMLEX syntax dictionaries (Grishman et al., 1994).

The system of Briscoe and Carroll is one of the most comprehensive subcategorization acquisition systems available for English. The version employed for building Valex is capable of categorizing 163 verbal SCFs and returning relative frequencies for each SCF found for a verb. It makes use of the first release of RASP (Robust Accurate Statistical Parsing) toolkit (Briscoe and Carroll, 2002).

As explained in detail in Briscoe and Carroll (1997), raw corpus data are first tokenized, tagged, lemmatised, and parsed with RASP using a tag-sequence grammar written in a feature-based unification formalism. This yields (complete) intermediate parses. Briscoe and Carrolls system invokes RASP in a mode which outputs intermediate phrase structure analyses. Verb subcategorization patterns (local syntactic frames including the syntactic categories and head lemmas of constituents) are then extracted from parsed sentences, from subanalyses which begin or end at the boundaries of specified predicates. The patterns are classified into SCFs using a comprehensive classifier which is capable of categorizing the 163 verbal SCFs. They abstract over lexically-governed particles and prepositions and specific predicate selectional preferences. Lexical entries are constructed for each verb and SCF combination, and the basic lexicon is built.

As no lexicon or semantic information is exploited during parsing, the basic lexicon is noisy. A filtering component may therefore be applied which can be used to remove noisy SCFs from the lexicon. In our work, we experimented with three versions of Valex:

Valex 1: The basic unfiltered lexicon

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

Valex 4: A sub-lexicon created by selecting from the basic lexicon all the SCFs which are also listed in the ANLT and/or COMLEX dictionaries plus the ones whose relative frequency is higher than a SCF-specific threshold.

The accuracy of these lexicons is 21.9, 58.6 and 83.7 according to F-measure, respectively. See Korhonen et al. (2006a) for the details of the evaluation.

Chapter 3

Automatic Verb Classification

We first extract feature sets from corpus data and then classify them using different methods. These steps are described in the following two sections, respectively.

3.1 Feature Sets

3.1.1 Feature Set 1: Basic Features

In the previous section we mentioned that the SCF frequency information was extracted from three versions of the automatically built Valex lexicon. Recall that in the Valex lexicon, each verb is associated with some of the 163 SCF types and that each SCF is associated with the relative frequency with the verb in question. The 163 SCFs and their relative frequencies form our **basic feature set**. For the detailed description of the 163 SCFs, see Briscoe (2000).

3.1.2 Feature Set 2: Refined Features

Some of the SCFs can be further refined by the prepositional phrases involved. For example, consider the SCF 49 (NP-PP):

Example 4 *she put the flowers on the table.*

Example 5 *she removed the flowers from the table.*

As the examples 4 and 5 show, "put" and "remove" have same SCF NP-PP. However, *put* prefers a PP headed by *on*, while *remove* prefers a PP headed by *from*. Therefore, to tell the difference between "put" and "remove" (which belong to different verb classes, those of PUTTING and REMOVING verbs) it helps to know the prepositions occurring in their SCFs. The lexical entries in the Valex lexicon include the preposition data, so it is easy to parameterise the SCFs for them. This feature set, where we enrich two high frequency SCFs (49 which is NP-PP and 87 which is PP) with the preposition types, is called the **refined feature set**.

SCF	Name of SCF	Example sentence
49	NP-PP	she_PPHS1 added_VVD the_AT flowers_NN2 to_II the_AT bouquet_NN1
87	PP	they_PPHS2 apologized_VVD to_II him_PPHO1
31	NP-FOR-NP	she_PPHS1 bought_VVD a_AT1 book_NN1 for_IF him_PPHO1
56	NP-TO-NP	he_PPHS1 gave_VVD a_AT1 big_JJ kiss_NN1 to_II his_AT mother_NN1
147	(SUBCAT OC_AP, SUBTYPE EQUI, PREP as)	he_PPHS1 condemned_VVD him_PPHO1 as_CSA stupid_JJ
14	S-SUBJ-TO-NP-OBJ	that_CST she_PPHS1 left_VVD matters_VVZ to_II them_PPHO2
162	*AS-VPPRT	he_PPHS1 accepted_VVD him_PPHO1 as_II/CSA associated_VVN
122	(SUBCAT NP_PP_PP, PFORM)	he_PPHS1 turned_VVD it_PPHO1 from_II a_AT disaster_NN1 into_II a_AT victory_NN1
43	NP-P-NP-ING	he_PPHS1 attributed_VVD his_AT failure_NN1 to_II noone_NP1 buying_VVG his_AT books_NN2
75	PART-ING-SC	he_PPHS1 ruled_VVD out_II paying_VVG her_AT debts_NN2
67	P-NP-TO-INF-OC	he_PPHS1 beckoned_VVD to_II him_PPHO1 to_TO come_VV0
85	POSSING-PP	she_PPHS1 attributed_VVD his_PP\$ drinking_VVG too_RG much_DA1 to_II his_AT anxiety_NN1
156	*NP-HOW-S	he_PPHS1 asked_VVD him_PPHO1 how_RGQ he_PPHS1 came_VVD
69	P-POSSING	they_PPHS2 argued_VVD about_II his_PP\$ coming_VVG

Table 3.1: 15 SCFs refined by their PPs, ranked by their frequency

Tag	Description
VV0	Verb, base form
VVD	Verb, past tense
VVG	Gerund or present participle
VVN	Verb, past participle
VVZ	Verb, 3rd person singular present

Table 3.2: Verb tense tags

3.1.3 Feature Set 3: New Features

The feature sets 1 and 2 have been previously used by Korhonen et al. (2003 and 2006). In addition, we experimented with new a new feature set consisting of a number of new features:

1. In addition to SCFs 49 and 87, we parameterized additional SCFs involving PPs for prepositions, those which were the 5, 10 and 15 most frequent ones in the Valex lexicon, respectively. We examined different frequency ranges because we were concerned about the effects of sparse data (as some of the SCFs involving PPs are low in frequency). The list of all the SCFs parameterized by PPs is shown in table 3.1, starting from the highest ranked SCF 49.
2. We used the tense of the verb, indicated by the simple part-of-speech tag (POS) of the verb (e.g. VVN, VV0), and its relative frequency with specific verbs as additional features. The list of all the POS tags indicating verb tense are shown in the table 3.1.2. This feature type is meaningful for verb classification, is readily available in the SCF lexical entries and has been previously used e.g. by Merlo and Stevenson (2001).

The figure 3.1 shows some of the features for the verb *put*. The first box shows some basic features and the second shows refined ones (SCFs 49 and 87 parameterized by PPs), and new ones (the tense features).

3.2 Data Preprocessing

We constructed a feature vector for each verb in our data. In the basic feature set, each verb had 163 features. Each feature corresponds to a SCF class, and its value is the relative frequency of the SCF with the verb in question. Some of the feature values are zero, because most verbs only take a subset of the possible 163 SCFs. For the two other feature sets, the number of dimensions vary depending on the feature set used. However, for all the feature sets, all the verbs have the same vector template, hence the high number of dimensions.

Vector template: [*Class label Basic features*(SCF1 SCF2 ... SCF163)*Refined features Tense features*]

The feature vectors constitute the input to machine learning. This data is used to train the automatic classifiers to determine the class labels, given the feature vector of a unknown verb (not in the training data).

3.3 Classification Methods

Machine learning enables the computer to 'learn' by using various algorithms. There are mainly two types of algorithms: supervised and unsupervised. As mentioned above, a number of unsupervised learning (clustering) techniques for verb classification have been previously investigated by Korhonen et al. (2003, 2006). In this study, we focus on supervised techniques, namely on the 'classification' tasks.

Put: <Unrefined>		
SCF	SCF NAME	Freq
24	NP	0.639
76	NP PRT	0.144
49	NP_PP	0.134
87	PP	0.066
77	NP_PP PRT+	0.011
.....		



Put: <Refined by tense and SCF 49 and 87 are refined by PP>		
SCF	SCF Name	Freq
24	NP	0.639
76	NP PRT	0.144
77	NP_PP PRT+	0.011
.....		
----- <i>Refined feature</i> -----		
49_NP_at		0.0092
49_NP_on		0.0377
49_NP_in		0.0409
49_NP_into		0.0214
87_in		0.0136
87_into		0.0115
87_by		0.0062
----- <i>Tense feature</i> -----		
VV0		0.486
VVD		0.198
VVG		0.106
VVN		0.147
VVZ		0.062

Figure 3.1: Sample features from the 3 feature sets for the verb *put*

We experiment with two types of classifiers: model-based classifiers and discriminative classifiers.

A **model-based classifier** builds a model independently for each class from samples during the training stage. During the test stage, the classification is performed according to the similarity between the test sample and the model. Maximum entropy model and the Gaussian classifier are both model-based classifiers.

A **discriminative classifier** tries to find an optimal boundary between classes during the training stage. Test samples are classified according to the class boundary. The K Nearest Neighbours and Support Vector Machines are discriminative classifiers.

3.3.1 K Nearest Neighbours

The k nearest neighbour method (KNN) is a classification method based on the K closest training sample in the feature space. For each verb feature vector in the test data, we measure its distance to each verb in the training data. Then, the verb class label is the most frequent class label in the top k closest training samples.

Three distance metrics are tried: the Euclidean distance, Kullback-Leibler (KL) divergence, and Jensen-Shannon (JS) divergence (Lin, 1991). The euclidean distance is the distance between two points (P,Q).

$$D(P, Q) = \sqrt{\sum_{i=1}^n (p_i - q_i)^2} \quad (3.1)$$

The KL and JS divergence are entropy based. They measure the similarity between two SCF distributions. KL divergence between SCF distribution P and Q measures the expected number of bits necessary for identifying an event based on Q when the true distribution is P.

$$D(P||Q) = \sum_i p_i \log\left(\frac{p_i}{q_i}\right) \quad (3.2)$$

The JS divergence is similar to KL divergence. It relies on the assumption that if P and Q are similar, then they are close to their average.

$$JS(P, Q) = \frac{1}{2} \left[D\left(P \parallel \frac{P+Q}{2}\right) + D\left(Q \parallel \frac{P+Q}{2}\right) \right] \quad (3.3)$$

The value of k is the only free parameter. A larger k value reduces the effect of noise, but it also makes the boundary between the classes less clear. The k value is determined by performing cross validation on the training data. The detail of estimating the k value is discussed in section 4.3.

3.3.2 Maximum Entropy Model

The idea behind maximum entropy is simple: **given a collection facts, we want to choose a model consistent with all the facts, and avoid assumption on anything that is unknown.**

The fact we know in this experiment is the SCF distribution of the verb. A verb can have many SCFs out of the 163 possible and each SCF is associated

with the relative frequency of usage with the verb in question. We define the verb class as y and its SCF as x . To express the fact that a verb $\mathbf{v1}$ is in the verb class **10.1** and takes the SCF **49** with relative frequency **0.6**, we have:

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

This is called the feature function or feature. The expected value of a feature f with respect to the empirical distribution (training data) is

$$\tilde{\mathcal{E}}(f) \equiv \sum_{x,y} \tilde{p}(x, y) f(x, y) \quad (3.4)$$

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

$$\mathcal{E}(f) \equiv \sum_{x,y} \tilde{p}(x) p(y|x) f(x, y) \quad (3.5)$$

$\tilde{p}(x)$ is the empirical distribution of x in the training data. If the training data is representative, we constrain the expected value $\mathcal{E}(f)$ to be the same as the expected value of training data.

$$\mathcal{E}(f) = \tilde{\mathcal{E}}(f) \quad (3.6)$$

From the equations 3.4 and 3.5, the equation 3.6 can be rewritten as

$$\sum_{x,y} \tilde{p}(x, y) f(x, y) = \sum_{x,y} \tilde{p}(x) p(y|x) f(x, y) \quad (3.7)$$

We require our probability model $p(y|x)$ to satisfy the equation 3.7 for every feature. We eliminate the probability models that do not agree with the expected feature value of the training sample. The constrained search space is defined as \mathcal{C} .

$$\mathcal{C} \equiv \{p \in \mathcal{P} | \mathcal{E}(f_i) = \tilde{\mathcal{E}}(f_i) \text{ for } i \in \{1, 2, \dots, n\}\} \quad (3.8)$$

So far, we already found all the models \mathcal{C} that are consistent with a known fact. Next, we need to find the most uniform model, which is also the model with the least assumption. Entropy is a measure of uniformity. For all $p \in \mathcal{C}$, we need to find the distribution p^* that has the maximum entropy $H(Y|X)$.

$$H(Y|X) \equiv - \sum_{x,y} \tilde{p}(x) p(y|x) \log p(y|x) \quad (3.9)$$

This problem is called the **primal problem**. It cannot be solved directly. Pietra et al. (1997) shows that the Lagrange multiplier can be used for this constrained optimization problem (see the paper for details). A **dual function** $\Psi(\lambda)$ is defined as in equation 3.10. The **dual problem** is to find the parameter λ^* that will maximize the dual function. We can also rewrite the primal problem according to the dual problem as in equation 3.11. Based on the Kuhn-Tucker theorem, if λ^* is the solution to the dual problem, then p_{λ^*} is the solution to the primal problem.

$$\Psi(\lambda) = - \sum_x \tilde{p}(x) \log Z_\lambda(X) + \sum_i \lambda_i \tilde{\mathcal{E}}(f_i) \quad (3.10)$$

$$p_\lambda(y|x) = \frac{1}{Z_\lambda(x)} \exp\left(\sum_i \lambda_i f_i(x, y)\right) \quad (3.11)$$

The solution to the dual problem can be found by Generalised Iterative Scaling (Berger, 1997) or Limited-Memory Variable Metric (Malouf, 2002). The number of iterations for these two algorithms is set to be 30 if not stated.¹

We used Zhang (2004)'s maximum entropy toolkit to implement the model. The toolkit supports both generalised iterative scaling and limited-memory variable metric, and more importantly, it supports real-valued feature.

3.3.3 Support Vector Machines

The Support Vector Machines (SVM) (Vapnik, 1995) try to find a maximal margin hyperplane to separate two group of vectors. The fundamental theory is that *if a set of vectors are mapped to a sufficiently high dimension, they will always be linearly separable*. The maximal margin is just to optimize the choice of dimensions to avoid over-fitting. The parameters of the hyperplane are found by solving a quadratic programming optimization problem.

Suppose we want to find a decision function f to separate the verbs $x_1 \dots x_n$ into two classes $(-1, +1)$ $f(x_i) = y_i, \forall i$. A hyperplane that correctly classifies the data must satisfy the following equation:

$$y_i[(w \cdot x_i) + b] \geq 1, \forall i \quad (3.12)$$

In practise, a separating hyperplane often does not exist. So we add a 'slack variable' ξ to allow the violation of equation 3.12

$$\xi_i > 0, \forall i \quad (3.13)$$

$$y_i[(w \cdot x_i) + b] \geq 1 - \xi, \forall i \quad (3.14)$$

In order to minimize the guaranteed risk bound, we need to minimize the equation

$$\iota(w, \xi) = \frac{1}{2}(w \cdot w) + \gamma \sum_{i=1}^n \xi_i \quad (3.15)$$

subject to equations 3.13 and 3.14. By the Kuhn Tucker theorem (Kuhn, 1982) of optimization theory, the solution is shown in equation 3.16 where a_i is the Lagrange multipliers.

$$w = \sum_{i=1}^n y_i a_i x_i \quad (3.16)$$

Only the training sample x_i that precisely satisfies equation 3.14 would have a non-zero coefficient a_i . These x_i are called **support vectors**. All other training samples are irrelevant. The coefficients a_i are found by solving the following quadratic programming problem, maximize:

$$W(a) = \sum_{i=1}^n a_i - \frac{1}{2} \sum_{i,j=1}^n a_i a_j y_i y_j (x_i \cdot x_j) \quad (3.17)$$

¹In the experiment, the Limited-Memory Variable Metric always outperform Generalized Iterative Scaling

subject to

$$0 \leq a_i \leq \gamma, i = 1, \dots, n, \quad (3.18)$$

and

$$\sum_{i=1}^n a_i y_i = 0 \quad (3.19)$$

Due to the linearity of the dot product $x \cdot x$, we need to replace $x \cdot x$ in equation 3.17 with a kernel function to solve the non-linear problem. The kernel is used to map the input space to a new space to solve the non-linear problem. The kernels experimented with in our experiment are the "Polynomial" and "Radial basis function". Polynomial:

$$K(x_i, x_j) = (\gamma x_i^t x_j + r)^d, \gamma > 0 \quad (3.20)$$

Radial basis function(RBF):

$$K(x_i, x_j) = \exp(-\gamma \|x_i - x_j\|^2), \gamma > 0 \quad (3.21)$$

Gamma γ , coefficient r , and degree d are the kernel parameters. The cost C is the penalty for the error term. Gamma and cost are found by performing cross-validation on the training data. All the other parameters are set to be default: The degree d is set to be 3. The coefficient r is set to be 0. The tolerance of termination criterion is set to be 0.001.

We use Chang and Lin (2001) 's LIBSVM library to implement the SVMs. In LIBSVM, the binary classification is extended to multi-class classification. One against one approach is used, where for each verb, a set of binary classifiers is built to compare each pair of classes. The resulting class label is the class that is voted most times. In the case of a tie, the resulting class is selected arbitrarily. For more information, please consult Hsu and Lin (2001).

3.3.4 Gaussian Model

Modelling data using the Gaussian or Gaussian mixture model is common in many applications, such as speech recognition. The central limit theorem is a justification of the popularity of the Gaussian model.

The verbs with feature vector $x_i \dots x_n$ in a verb class y are assumed to be independent from each other, and have an arbitrary probability distribution. The central limit theory states that: if we take the average of all the verbs in the class:

$$\bar{x} = \frac{1}{n} \sum_{i=1}^n x_i \quad (3.22)$$

then, irrespective of how the original samples were distributed, the distribution of the mean (\bar{x}) is always **normally distributed** as $n \Rightarrow \text{Limit}$.

As we will see in the next chapter, we have a relatively small number of verbs in a class, which does not completely satisfy the central limit theorem. However, the Gaussian is a good model when the statistical distributions of the training and test data for a class are very similar to each other.

The covariance matrix of the features is assumed to be diagonal. This means that no correlation is assumed between features. Suppose x is feature vector

of a verb, the feature vector has d dimensions, and y is the verb class. The likelihood is:

$$p(x|y) = \prod_{a=1}^d \frac{1}{\sqrt{2\pi\sigma_a^2}} \exp\left(-\frac{(x_i - \mu_a)^2}{2\sigma_a^2}\right) \quad (3.23)$$

We need to find the mean and the variance in order to calculate the likelihood. Because all the verbs have a feature vector with same number of dimensions, the mean (μ_i) and variance (σ^2) for each dimension i for verbs in a class can be calculated separately as shown below:

If there are N verbs with feature vectors $x_1 \dots x_n$ in a verb class the mean of a dimension i is:

$$\mu_i = \frac{\sum_{i=1}^n x_i}{N} \quad (3.24)$$

The variance of the dimension i is:

$$\sigma_i^2 = \frac{\sum_{i=1}^n (x_i - \mu_i)^2}{N} \quad (3.25)$$

With μ_i and σ_i^2 , by using the equation 3.23, we can construct a Gaussian model for each verb class. To predict an unknown verb, we use the feature vector of the unknown verb (x_i) and calculate the likelihood using equation 3.23 for each verb class. Then we chose the verb class with highest probability.

The data sparseness is a serious problem with the Gaussian model. For any feature vector in a verb class, there is a possibility that some dimension may never appear. This can be interpreted in that some SCFs never show up with a verb class. In fact, this happens quite frequently. With our data on average, each verb class had 30 to 40 'blank dimensions'. For these dimensions, the variance μ_i is zero. We assigned these zero-variance cases a small value ν . This is the same idea as behind the "vFloor" in the Hidden Markov Model Toolkit (HTK) (Young et al., 2000).

The value of ν is largely affected by system performance due to the high number of zero-variance dimensions. The relationship between ν and the system accuracy is found to be a convex. A simple hill climbing algorithm was implemented to find the global optimum. The optimal ν was found by performing 'leave one out' cross validation on the training data as explained in detail in the next chapter.

Chapter 4

Experiments

4.1 Data

We selected our test verbs and classes from VerbNet (Kipper et al., 2000). 17 fine-grained classes were selected, not more than 2 from each broad class (recall that Levin’s classification is a 2-3 level taxonomy where some broad classes divide into subclasses or further). The classes were selected subject to the constraint that they had a sufficient number of member verbs whose predominant sense belongs to the class in question. We verified the relevant predominant sense by looking at the predominant verb sense classification used for constructing the Valex lexicon.

For each class, we selected 12 member verbs. This was the average maximum number we could find given the predominant sense and fine-grained class requirement. We also required that each verb should have at least 100 occurrences in our input data, i.e. in the Valex subcategorization lexicon. However, most of the selected verbs had 1000-9000 occurrences in Valex. The chosen classes and member verbs are listed in table 4.1.

Our set of 17 classes represents various Levin classes. Some classes are very different in terms of semantics and/or syntax, such as LIGHT OMISSION (e.g. sparkle) vs. SAY (e.g. say), verbs, while others are similar, e.g. SAY verbs vs. MANNER OF SPEAKING (e.g. whisper) verbs:

*Ellen said / whispered / *sparkled a few words.*
*Ellen said / whispered / *sparkled a few words to Helen.*
*Susan said / whispered / *sparkled that the party would be tonight.*
*Susan said / whispered / *sparkled, 'Leave the room.'*
*Susan said / whispered / *sparkled to Rachel that the party would be tonight.*
*Susan said / whispered / *sparkled to Rachel how to avoid the crowd.*

Thus the difficulty of the classification task varies from one class to another.

4.2 Methodology

For the experiments, we split the data into two parts: the training data, and the test data. Each classification method was trained on the training data, and

Class label	Description	Verbs
10.1	REMOVE	remove, abolish, discharge, extract, deduct, eradicate, sever, evict, eject, subtract, retract, delete
11.1	SEND	ship, post, send, transport, transmit, transfer, deliver, slip, dispatch, forward, mail, shunt
13.5.1	GET	win, gain, earn, buy, get, book, reserve, fetch, attain, pick, charter, conserve
18.1	HIT	beat, slap, bang, knock, pound, batter, hammer, lash, bash, smack, whack, bump
22.2	AMALGAMATE	integrate, match, coincide, incorporate, compare, unite, contrast, affiliate, unify, overlap, correlate, alternate
29.2	CHARACTERIZE	envisage, portray, regard, treat, identify, define, depict, diagnose, enlist, reinstate, picture, class
30.3	PEER	listen, stare, look, glance, gaze, peer, peek, squint, snoop, gape, leer, peep
31.1	AMUSE	stimulate, threaten, shock, confuse, upset, overwhelm, scare, disappoint, delight, exhaust, intimidate, frighten
36.1	CORRESPOND	cooperate, collide, concur, compromise, flirt, interact, dissent, mate, banter, haggle, commiserate, elope
37.3	MANNER SPEAKING	shout, yell, whisper, mutter, murmur, snarl, moan, wail, mumble, grunt, whimper, stutter
37.7	SAY	say, reply, mention, state, report, respond, announce, recount, utter, exclaim, retort, confide
40.2	NONVERBAL EXPRESSION	smile, laugh, grin, sigh, gasp, chuckle, frown, giggle, sneer, yawn, sob, snore
43.1	LIGHT EMISSION	shine, flash, flare, glow, blaze, flicker, gleam, sparkle, glisten, twinkle, glitter, flame
45.4	OTHER CHANGE OF STATE	soften, weaken, strengthen, narrow, deepen, dampen, melt, multiply, sharpen, shorten, broaden, freeze
47.3	MODES OF BEING WITH MOTION	quake, falter, sway, swirl, teeter, flutter, wobble, waft, quiver, wiggle, vibrate, oscillate
51.3.2	RUN	swim, fly, walk, slide, run, travel, stroll, glide, jog, march, trot, gallop
9.1	PUT	bury, place, install, mount, put, deposit, position, set, situate, immerse, insert, stash

Table 4.1: Experiment data

tested on the test data. We used two methods for partitioning the training and test data.

The first one is *leave one out cross validation*. With N verbs, the system runs N times. For each run, one verb is held out as the test data, and the remaining $N-1$ verbs are used as training data. The overall accuracy is the average of the accuracy of N rounds. This method allows us to examine the classification performance at both verb and verb class levels.

The second method is *re-sampling*. There are 12 verbs in each class. For each class, 3 verbs are selected randomly as test data, and the rest of the verbs are used as training data. In total 25% of the verbs are used as test data and 75% as training data. The sampling process is repeated for 30 times. The final result is the average result over the sampling processes. The re-sampling method creates a larger test set than the leave one out cross validation method. The downside is that it does not measure the performance at the verb level.

Two evaluation measures are used: accuracy, and F-score. Accuracy is the percentage of correct classifications out of all the classifications. At the verb level, there are no false positives or true negatives because a verb can only be classified correctly (true positive) or incorrectly (false negative).

$$Accuracy = \frac{truePositives}{truePositives + falseNegatives} \quad (4.1)$$

When we evaluate the performance at the class level, we have the false positives and true negatives as well. We calculate precision as follows:

$$Precision = \frac{truePositives}{turePositives + falsePositives} \quad (4.2)$$

Recall is the class level accuracy.

$$Recall = \frac{truePositives}{truePositives + falseNegatives} \quad (4.3)$$

F-score is the balance over recall and precision.

$$F - score = \frac{2 \times Precision \times Recall}{Precision + Recall} \quad (4.4)$$

Given there are 17 classes in the data, the accuracy of randomly assigning a verb into one of the 17 classes is $1/17 \approx 5.8\%$. Therefore, our random baseline accuracy is 5.8%.

4.3 Parameter Estimation

The classification methods were introduced in section 3. The following list summarises the settings and free parameters for each method:

KNN We tested three distance metrics using our basic feature set. The result is shown in table 4.2. The JS divergence outperforms the KL divergence and euclidean distance. Therefore, the JS divergence was chosen as the default distance metric. The value k is the only free parameter for KNN.

Feature Set	Distance metric	Accuracy(%)	F-Score(%)
Basic	Euclidian distance	37.7	37.3
Basic	KL divergence	40.2	40.9
Basic	JS divergence	47.0	47.0

Table 4.2: Leave one out cross validation test for KNN to select the distance metric, k=4

Maximum entropy We used the built-in L-BFGS as the default parameter estimation method, as suggested by Zhang (2004). The only free parameter for maximum entropy is the number of iterations i in the parameter estimation algorithm. When this value is too small, the parameters have a poor fit with the training data. When the value is too large, we have the over-fitting problem.

SVM We used the standard C-SVC Cost-Based Support Vector Classification model, and the radial basis function kernel as shown in equation 3.21. This setting is suggested by the author of the toolkit (Hsh et al., 2003). There are two free parameters: the cost C for C-SVC, and a scaling parameter for the kernel γ .

Gaussian In the Gaussian model, we assumed that the features are uncorrelated, so the covariance is diagonal. The only free parameter is the value of zero-variance ν .

KNN, maximum entropy model and Gaussian have one free parameter each. We use the same parameter estimation technique for them. We first define a value range for the free parameter. For KNN, the value k is set to be in the range 2-20. For maximum entropy, the iterations i are set to be in the range 5-50. The original default value of i is 30. For the Gaussian model, the value of zero variance ν is set between 10^{-8} and 10^{-7} . Then, we perform 'leave one out' cross validation for the Gaussian model and 10-fold cross validation for the other methods on the training data to find the best parameter value within the range.

SVM has two free parameters. The recommended range of values for these two parameters according to Hsh et al. (2003) are:

$$C = 2^{-5}, 2^{-3}, \dots, 2^{15}, 2^{17}$$

$$\gamma = 2^{-17}, 2^{-15}, \dots, 2^1, 2^3$$

We tested all the possible combinations of the two values on the training data using 10-fold cross validation. The combination with the highest accuracy was selected. This approach (called 'grid search') was suggested by the author of the toolkit (Hsh et al., 2003).

For each classification method and feature set, the parameter estimation is repeated according to the cross validation and re-sampling processes explained above.

4.4 Experimental Evaluation

In the following sections we report the classification results for the three feature sets introduced earlier in section 3.1.

Classification method	Lexicon	Accuracy(%)	F-Score (%)
KNN	Valex 1	44.1	44.0
KNN	Valex 2	34.3	32.4
KNN	Valex 4	39.2	39.0
Maximum entropy	Valex 1	47.5	47.6
Maximum entropy	Valex 2	37.2	35.5
Maximum entropy	Valex 4	44.1	43.7
SVM	Valex 1	47.1	47.8
SVM	Valex 2	40.7	41.3
SVM	Valex 4	47.5	47.8
Gaussian	Valex 1	49.5	46.2
Gaussian	Valex 2	34.3	28.4
Gaussian	Valex 4	41.2	36.3

Table 4.3: 'Leave one out' cross validation result for Feature set 1

4.4.1 Feature Set 1: Basic Features

The average performance of each classifier (with each version of Valex) according to leave one out cross-validation is shown in figure 4.4.1. The accuracy ranges from 34.3% to 49.5%, and the F-Score ranges from 28.4% to 47.8%.

SVM yields the best performance (47.8% in F-score), although maximum entropy and the Gaussian model have very similar performance (47.6% and 46.2%, respectively). All the other methods except SVM have the best performance with the raw version of the Valex lexicon, indicating that the large noisy lexicon is more useful for classification than the smaller more accurate lexicons.

Figure 4.4.1 shows the performance figures for individual classes on the Valex 1 (raw) lexicon.

The class 47.3 has the worst performance with all the 4 methods. This is not surprising since the class (MODES OF BEING IN MOTION) includes intransitive verbs (e.g. *wobble*) with few distinguishing SCFs. The classes 11.1, 29.2 and 40.2 had the best performance (more than 70% in F-score). Interestingly, the Gaussian model fails with classes 47.3, 43.2 and 36.1. In return, it produces very good performance with the remaining of the classes. Precision is therefore low with the Gaussian model, but recall is better than for the other classification methods (49.5%).

Re-sampling results for the basic feature set are shown in table 4.4. These results are consistent with the 'leave one out' results, except for the Gaussian model. The performance of the Gaussian model has declined approximately 4%. A possible reason for this is the smaller training data with re-sampling. In comparison with the 'leave one out' method, the re-sampling method has 25% less training data, and 51 times more test data. The resampling method picks up randomly 3 verbs for each round, so the performance may vary. So a 1-2% difference in accuracy and F-score is not necessarily meaningful.

4.4.2 Feature Set 2: Refined Features

As mentioned in section 3.1.2, the refined feature set 2 includes the basic features from the feature set 1 for all the other SCFs, except for 49 and 87 which are parameterized according to the preposition in the PP of the SCF.

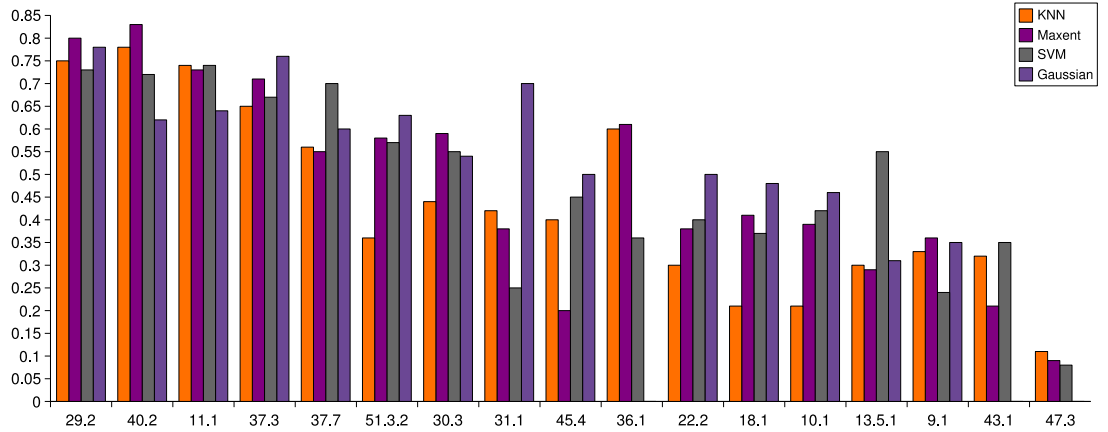


Figure 4.1: Class level F-Score for Feature set 1

Classification method	Lexicon	Accuracy(%)	F-Score (%)
KNN	Valex 1	37.3	36.5
KNN	Valex 2	29.8	29.2
KNN	Valex 4	34.5	34.1
Maximum entropy	Valex 1	47.1	47.0
Maximum entropy	Valex 2	42.2	41.5
Maximum entropy	Valex 4	49.2	48.9
SVM	Valex 1	47.3	47.7
SVM	Valex 2	41.1	40.5
SVM	Valex 4	46.4	46.4
Gaussian	Valex 1	39.0	40.0
Gaussian	Valex 2	36.9	34.4
Gaussian	Valex 4	43.6	42.2

Table 4.4: Re-sampling results for Feature set 1

Classification method	Lexicon	Accuracy(%)	F-Score (%)
KNN	Valex 1	54.9	53.9
KNN	Valex 2	46.1	44.1
KNN	Valex 4	50.9	49.2
Maximum entropy	Valex 1	59.3	59.9
Maximum entropy	Valex 2	55.3	56.8
Maximum entropy	Valex 4	59.3	59.8
SVM	Valex 1	57.8	57.9
SVM	Valex 2	55.4	54.9
SVM	Valex 4	54.4	54.1
Gaussian	Valex 1	59.3	57.1
Gaussian	Valex 2	55.3	51.8
Gaussian	Valex 4	55.3	52.5

Table 4.5: Leave one out cross validation result on Feature set 2 (refined feature set)

The 'leave one out' results are shown in table 4.5. They show that the refined feature set improves the performance of each classification method by around 10% and that each classification method produces again its best performance with the raw version of the Valex lexicon. The maximum entropy method has 12% better F-score here than with the basic feature set. Its performance is the best among the 4 methods. KNN is again the method with the lowest performance.

Table 4.7 shows the difference in performance for each class when the refined feature set is used as opposed to the basic one. 13 classes out of the 17 have better F-score. The classes 10.1, 43.1, 31.1 and 30.3 have more than 20% better performance. This is not surprising as these classes share the SCFs 49 and 87, but differ in the type of prepositions they prefer (e.g. the class 10.1, REMOVE verbs, has a clear preference for the preposition *from* as in *she removed the bag from the car*, while the class 31.1, AMUSE verbs, has a preference for the preposition *with* as in *she amused her with her stories*). The performance of four verb classes declined by 3.5% on average, but overall 11% increase in F-score was gained with the refined feature set.

Figure 4.2 shows the F-score results for each verb class on sub-lexicon Valex 1. When compared with figure 4.4.1, we can see clearly that the performance is significantly better with each method for most of the verb classes. Class 47.3 is still the class with the worst performance, but the Gaussian model has now non-zero F-score for classes 43.1 and 36.1, although the results are still lower than with maximum entropy and SVM. Despite the boost in performance, precision is lower for the Gaussian model than for maximum entropy and SVM, but the Gaussian model produces again the highest accuracy (59.3%).

The re-sampling results are shown in table 4.6. These results are consistent with the 'leave-one-out' cross validation results, except for the Gaussian model. As with the basic feature set, the Gaussian model yields a considerably lower performance here than with the 'leave-one-out' cross validation.

Classification method	Lexicon	Accuracy(%)	F-Score (%)
KNN	Valex 1	42.7	42.6
KNN	Valex 2	36.6	35.3
KNN	Valex 4	41.6	40.1
Maximum entropy	Valex 1	58.1	58.1
Maximum entropy	Valex 2	56.8	56.3
Maximum entropy	Valex 4	58.8	58.6
SVM	Valex 1	56.7	57.1
SVM	Valex 2	53.7	50.3
SVM	Valex 4	53.9	54.3
Gaussian	Valex 1	49.9	51.4
Gaussian	Valex 2	44.8	45.2
Gaussian	Valex 4	54.3	54.9

Table 4.6: Re-sampling results for Feature set 2 (refined feature set)

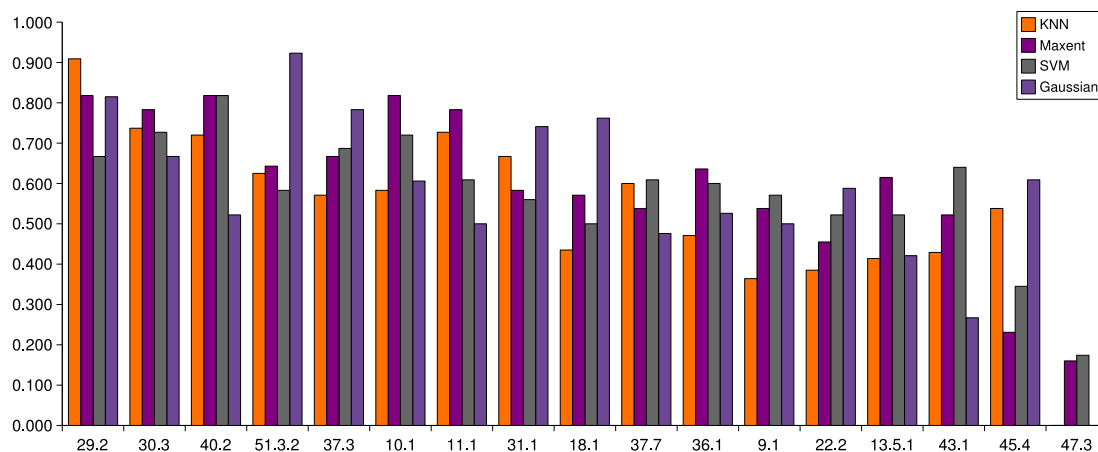


Figure 4.2: Class level F-Score for Feature set 2

Class label	Average F-score change
10.1	+31.1
43.1	+24.4
31.1	+20.1
30.3	+20.0
18.1	+19.9
9.1	+17.3
36.1	+16.5
51.3.2	+13.0
13.5.1	+13.2
22.2	+9.3
45.4	+4.3
29.2	+3.7
47.3	+1.5
40.2	-1.8
37.3	-2.1
37.7	-4.5
11.1	-5.5

Table 4.7: Average contribution of Feature set 2 to all the methods for each class as compared with Feature set 1

4.4.3 Feature Set 3: New features

Additional PP Features

Since the refined features improved the performance considerably, we evaluated whether we would see a similar or better boost in performance with the new feature set described in section 3.1.2. Recall that the new features had additional 13 SCFs involving PPs parameterized for prepositions (instead of just the SCFs 49 and 87 in the refined feature set). The additional SCFs are lower in frequency and classification may thus suffer from sparse data problems.

The 13 new PP features (i.e. refined SCFs) were added in three steps on the top of Feature set 2. First, the 3 most frequent new features were added, then 5 further features that are less frequent were added, and finally 5 least frequent features were added. This approach was adopted so that we could examine the possible sparse data effect.

Tables 4.8, 4.10, and 4.12 show the 'leave one out' cross validation results for the new feature set. The performance of KNN method is worse with the new feature set and keeps on decreasing with the increasing number of features. The performance of maximum entropy and SVM does not change significantly over different numbers of new features. In fact, the performance is very similar to that with feature set 2. However, as shown in table 4.12, the performance of the Gaussian model is significantly better when all the 13 new features are used as opposed to using subsets of them. The F-Score is 62.5%, which is 2.5% higher than with the maximum entropy model, and 5% higher than with SVM. The best results with each method are once again obtained with the raw version of the Valex. The raw version of Valex is likely to include a higher number of the new features as many are absent in the filtered versions of Valex.

Table 4.1.4 shows the contribution of the Feature Sets 2 as compared with

Classification method	Lexicon	Accuracy(%)	F-Score (%)
KNN	Valex 1	49.5	48.2
KNN	Valex 2	45.6	43.2
KNN	Valex 4	51.0	49.2
Maximum entropy	Valex 1	59.3	60.0
Maximum entropy	Valex 2	55.8	56.7
Maximum entropy	Valex 4	59.3	60.0
SVM	Valex 1	57.8	58.2
SVM	Valex 2	55.3	54.9
SVM	Valex 4	54.4	54.1
Gaussian	Valex 1	59.3	56.5
Gaussian	Valex 2	50.9	48.8
Gaussian	Valex 4	56.3	52.8

Table 4.8: Leave one out cross validation results with 3 new PP features added from Feature set 3

the Feature set 1, and the contribution of Feature set 3 as compared with the Feature set 2 for the Gaussian model. Most classes have a consistent performance increase with additional PP features. With Feature set 2, 13 out of 17 classes have improvement. With Feature set 3, the performance improves further for 9 out of the 17 classes. Most classes have net increase on performance. Exceptions are classes 11.1, 37.3, 37.7 and 45.4. The class 11.1 has the best performance with Feature set 1, indicating that although members in the class take prepositions, considering the basic SCFs instead is more helpful. The same applies to classes 37.3 and 45.4 which have improved performance with Feature set 2, but decrease with Feature set 3. For class 37.7, the performance declines with Feature set 2 but improves with Feature set 3. We discuss various sources of error in the qualitative analysis section 4.6.

Figure 4.3 shows the performance for each class. We can see that the Gaussian model produces consistently better performance for most of the classes. 16 out of the 17 classes have F-score better than 55%. The only exception is still the problematic class 47.3, which has the F-score zero.

The re-sampling results are shown in table 4.13. As with the other feature sets, the Gaussian and KNN models still produce lower F-score than they do with the 'leave one out' cross validation method. The maximum entropy model and SVM produce similar results as with the 'leave one out' cross validation method.

Verb Tense Features

We finally added the verb tense features (see section 3.1.3) to each classification method with the best configuration. Recall that the KNN, maximum entropy and SVM methods had their best performance with feature set 2. The Gaussian model had its best performance with the new feature set 3.

We can see from table 4.15 that the performance is not improved for any of the methods by adding the verb tense feature. The performance of the maximum entropy and SVM models does not change, while the performance of the KNN and Gaussian models is decreased. Also Joanis et al. (2007) noted

Classification method	Lexicon	Accuracy(%)	F-Score (%)
KNN	Valex 1	27.1	28.2
KNN	Valex 2	19.1	19.6
KNN	Valex 4	25.6	25.9
Maximum entropy	Valex 1	60.1	59.8
Maximum entropy	Valex 2	56.4	56.3
Maximum entropy	Valex 4	59.3	59.4
SVM	Valex 1	54.4	54.6
SVM	Valex 2	54.7	54.7
SVM	Valex 4	54.7	54.6
Gaussian	Valex 1	50.0	51.7
Gaussian	Valex 2	48.1	48.2
Gaussian	Valex 4	53.5	52.8

Table 4.9: Re-sampling results with 3 new features added from Feature set 3

Classification method	Lexicon	Accuracy(%)	F-Score (%)
KNN	Valex 1	49.5	48.3
KNN	Valex 2	46.1	44.1
KNN	Valex 4	51.0	49.2
Maximum entropy	Valex 1	59.8	59.9
Maximum entropy	Valex 2	53.9	54.2
Maximum entropy	Valex 4	59.3	59.7
SVM	Valex 1	57.3	57.5
SVM	Valex 2	55.3	54.9
SVM	Valex 4	54.4	54.0
Gaussian	Valex 1	59.8	56.6
Gaussian	Valex 2	53.4	49.6
Gaussian	Valex 4	55.8	53.4

Table 4.10: Leave one out cross validation result on 8 new features added from Feature set 3

Classification method	Lexicon	Accuracy(%)	F-Score (%)
KNN	Valex 1	23.6	24.0
KNN	Valex 2	17.6	17.6
KNN	Valex 4	21.0	20.7
Maximum entropy	Valex 1	59.8	59.6
Maximum entropy	Valex 2	56.6	56.5
Maximum entropy	Valex 4	59.8	59.7
SVM	Valex 1	56.8	56.9
SVM	Valex 2	51.5	52.1
SVM	Valex 4	53.6	54.0
Gaussian	Valex 1	52.4	53.2
Gaussian	Valex 2	47.6	48.3
Gaussian	Valex 4	56.1	56.6

Table 4.11: Re-sampling results with 8 new features added from Feature set 3

Classification method	Lexicon	Accuracy(%)	F-Score (%)
KNN	Valex 1	49.5	48.5
KNN	Valex 2	46.6	44.4
KNN	Valex 4	47.1	45.7
Maximum entropy	Valex 1	59.8	59.9
Maximum entropy	Valex 2	53.9	54.4
Maximum entropy	Valex 4	59.3	59.7
SVM	Valex 1	57.3	57.4
SVM	Valex 2	55.8	55.5
SVM	Valex 4	53.9	53.7
Gaussian	Valex 1	64.2	62.5
Gaussian	Valex 2	53.9	50.1
Gaussian	Valex 4	56.3	53.5

Table 4.12: Leave one out cross validation results on 13 new features added from Feature set 3

Classification method	Lexicon	Accuracy(%)	F-Score (%)
KNN	Valex 1	20.5	20.3
KNN	Valex 2	17.0	16.6
KNN	Valex 4	22.6	22.1
Maximum entropy	Valex 1	57.5	57.8
Maximum entropy	Valex 2	52.6	52.8
Maximum entropy	Valex 4	59.8	59.7
SVM	Valex 1	55.6	55.5
SVM	Valex 2	50.7	50.7
SVM	Valex 4	55.6	55.6
Gaussian	Valex 1	54.0	55.5
Gaussian	Valex 2	45.8	46.4
Gaussian	Valex 4	56.1	56.6

Table 4.13: Re-sampling results with 13 new features added from Feature set 3)

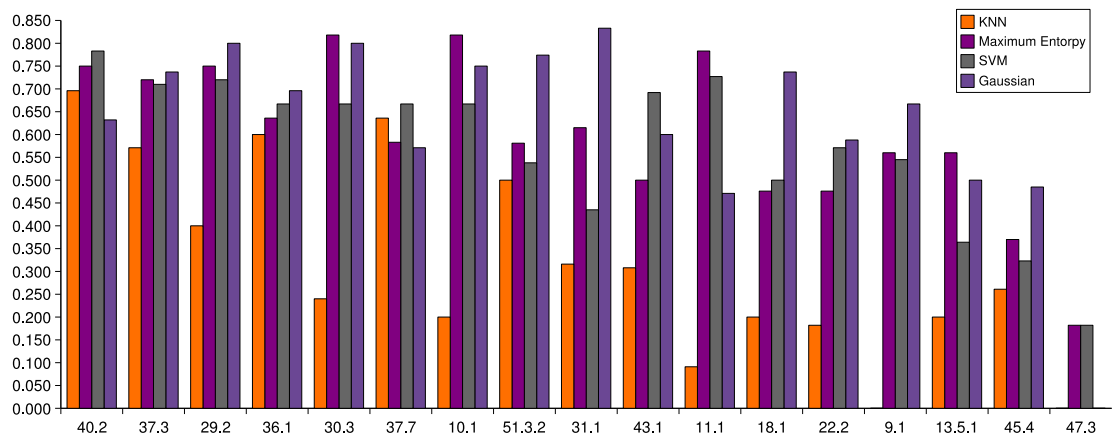


Figure 4.3: Class specific F-Score results with 13 new features added from Feature set 3

Class label	Feature set 2 against feature set 1 (%)	Class label	Feature set 3 against feature set 2(%)
36.1	+52.6	43.1	+33.3
51.3.2	+29.3	36.1	+17.0
18.1	+28.2	9.1	+16.7
43.1	+26.7	10.1	+14.4
9.1	+15.0	30.3	+13.3
10.1	+14.6	40.2	+11.1
30.3	+12.7	37.7	+9.5
13.5.1	+11.1	31.1	+9.2
45.4	+ 10.9	13.5.1	+ 7.9
22.2	+8.8	47.3	0
31.1	+4.1	22.2	0
29.2	+3.5	29.2	-1.5
37.3	+0.2	18.1	-2.5
47.3	0	11.1	-2.9
40.2	-9.8	37.3	-4.6
37.7	-12.4	45.4	-12.4
11.1	-14.0	51.3.2	-14.9

Table 4.14: Contribution of Feature set 2 as compared with Feature set 1 and contribution of Feature set 3 as compared with Feature set 2 for the Gaussian model

Classification method	Lexicon	Accuracy(%)	F-Score (%)
KNN	Valex 1	38.4	38.8
Maximum entropy	Valex 1	59.8	59.9
SVM	Valex 1	58.8	59.5
Gaussian	Valex 1	62.7	62.0

Table 4.15: Leave one out cross validation results with the full Feature set 3, verb tense features included

that the contribution of verb tense feature was very small in their classification experiment.

4.5 Summary of Quantitative Evaluation

To sum up, the system achieved the best performance at F-Score = 62.5% and accuracy = 64.2% using Gaussian model with the new feature set 3. The accuracy is 59% higher than the baseline accuracy. The Gaussian model proved very sensitive to the amount of training data. In contrast, the maximum entropy model and SVM are more robust, as they can produce more stable results when less training data is available. Maximum entropy model is the second best classification algorithm. It outperforms Gaussian and SVM in most of the experiments conducted. However, it fails to take advantage of the new features. SVM performs very similarly with the maximum entropy model. It always has a significantly better performance than the KNN method. The KNN method is the simplest method, and it produces acceptable result with feature sets 1 and 3, but the performance declined significantly with the new feature set 3.

In the re-sampling evaluation, the Gaussian model produces considerably lower performance than in the 'leave one out' cross validation evaluation. This seems to indicate that the Gaussian model is more sensitive to the amount of training data than the Maximum entropy model and SVM. As shown in figure 4.4, the performance of all the three methods declines with larger sample size. The Maximum entropy model is the method which produces the most stable performance over different sample sizes. Only less than 2% change can be observed from sample size 1 to 4. The SVM changes around 3% from sample size 1 to 4. The performance of the Gaussian model drops dramatically with larger sample size. The performance drops by more than 8% from sample size 1 to 4.

4.6 Qualitative Analysis

We did some qualitative analysis to trace the origin of various error types. We first investigated whether verb frequency in the Valex lexicon plays any role in the results. Previous research has shown that verb clustering performance increases with the size of the SCF lexicon (Korhonen et. al., 2003). From the 204 verbs in our data, 54 had less than 1000 occurrences in the Valex lexicon. Most of these are found in classes 30.3 (6), 36.1 (4), 40.2 (5), 43.1 (7) and 47.3 (7). While it is true that class 47.3 was the worst performing class, there appears to be little correlation between verb frequency and class performance, as most other classes have average or very good performance, see e.g. class 40.2. Figure 4.5 which shows the correlation between verb frequency and classification performance indicates similarly that verb frequency played little part in the results. As the Valex lexicon was extracted from several corpora, it provides sufficient data for verb classification.

We then examined the classification errors for each class. Figures 4.6 and 4.7 show the error matrix for the Gaussian and maximum entropy models with Feature set 3. In each confusion matrix a column indicates a correct class and a row a mistakenly predicted class. For example, in figure 4.7 the number 2 in

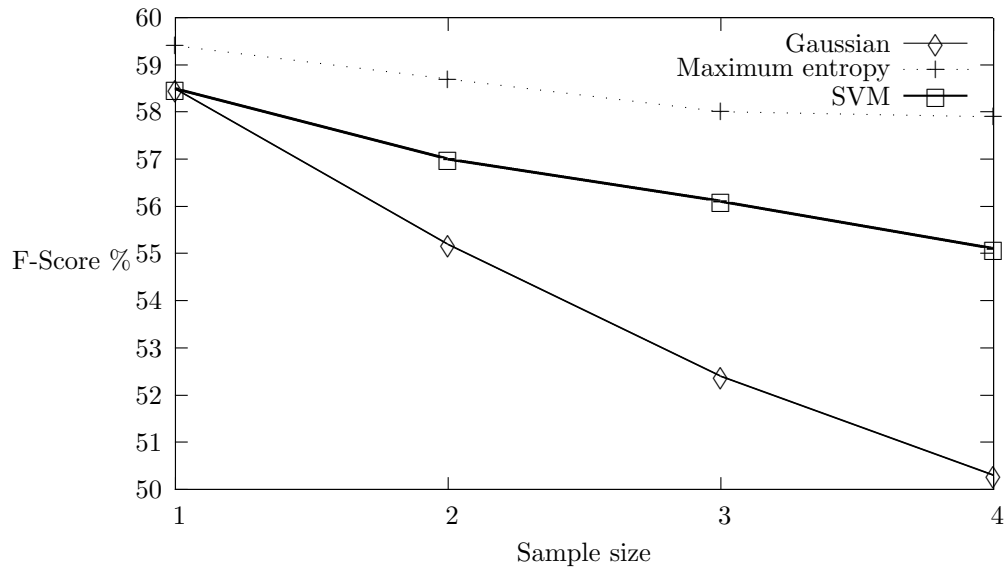


Figure 4.4: Sensitivity of 3 machine learning methods to the sample size

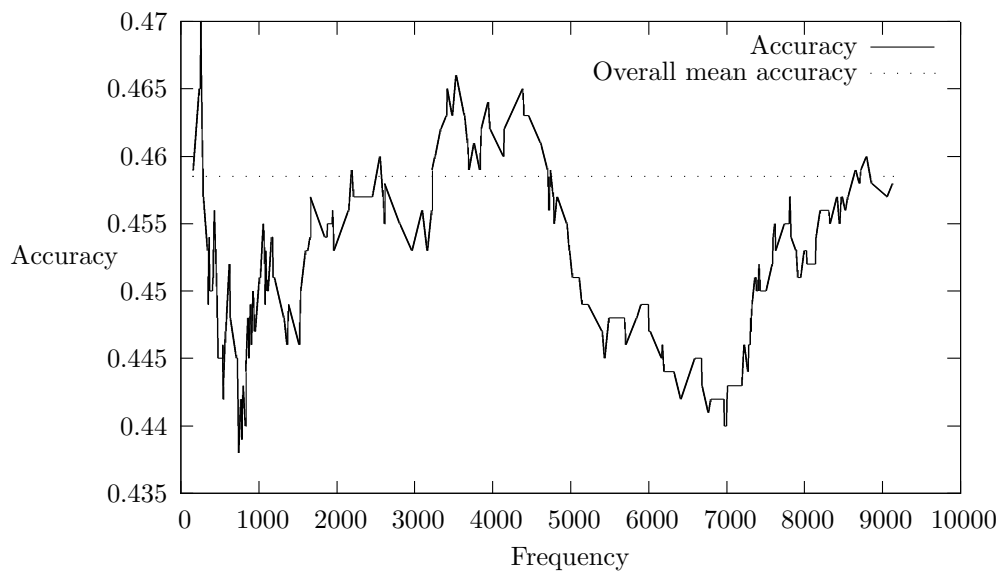


Figure 4.5: Verb frequency and accuracy using maximum entropy method with Feature Set 1

	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	3	1	0	0	0	0	0	0	0	2	0	0	1
11.1	0	-	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
13.5.1	0	5	-	0	1	1	0	2	0	0	0	0	1	1	0	0	1
18.1	0	0	0	-	0	0	0	0	0	0	0	0	0	0	0	0	0
22.2	0	0	0	0	-	0	0	0	0	0	0	0	0	0	0	0	0
29.2	0	0	0	0	0	-	0	0	0	0	1	0	0	0	0	0	2
30.3	0	0	0	0	0	0	-	0	0	0	0	0	0	0	0	0	0
31.1	0	0	0	0	0	0	0	-	0	0	0	0	0	1	0	0	1
36.1	0	0	0	0	2	0	0	0	-	0	1	0	0	0	0	0	0
37.3	0	0	0	0	0	0	0	0	0	-	0	0	0	0	0	0	0
37.7	0	1	1	0	1	0	0	0	1	2	-	0	0	0	2	0	0
40.2	0	0	0	1	0	0	4	0	2	3	0	-	2	0	2	0	0
43.1	0	0	0	0	0	0	0	0	0	0	0	0	-	0	2	0	0
45.4	0	1	2	4	0	0	0	0	1	0	2	0	1	-	1	0	1
47.3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-	0	0
51.3.2	0	1	0	0	0	0	0	0	0	0	0	0	1	0	5	-	0
9.1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-

Figure 4.6: Error matrix for Gaussian

	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	-

Figure 4.7: Error matrix for Maximum entropy

the cell in column 47.3 and row 51.3.2 indicates that class 51.3.2 was predicted wrongly for 2 verbs belonging to class 47.3.

Examination of the worst performing class 47.3 (MODES OF BEING INVOLVING MOTION verbs) illustrates well the various causes of error we found in the classification. For the maximum entropy model 10 of the 12 verbs in this class are classified incorrectly:

- 3 are assigned to class 43.1 (LIGHT EMISSION verbs): Verbs in classes 47.3 and 43.1 both describe intrinsic properties of their subjects (e.g. *a jewel sparkles* and *a flag flutters*). They take very similar SCFs and alternations, making it difficult to separate them on syntactic grounds. For example, verbs in both classes take intransitive and PP SCFs and participate in locative inversion (e.g. *a diamond sparkled on his finger - on his finger sparkled a diamond*). Distinguishing these two classes would ideally require classifying the subjects of the verbs semantically and looking at the differences in their semantic classes.
- 2 are assigned to class 51.3.2 (RUN verbs): These errors are caused by the fact that class 47.3 is closely related to 51.3.2 semantically. Verbs in both classes share the semantic component of motion and take similar alternations and SCFs. The main difference is that verbs in class 47.3 describe the existence of inanimate entities that involve types of motion typical to these entities, while most verbs in class 51.3.2 describe the manners in which animate entities can move. Again it would help to know the semantic type of the arguments (particularly subjects) of these verbs.
- 2 are assigned to class 36.1 (CORRESPOND verbs): these errors seem to be caused by purely syntactic similarities between the two classes, as semantically verbs in class 36.1 are very different to those in class 47.3. The confusion is likely to arise from the fact that verbs in both classes take simple intransitive and PP SCFs with high frequency, although they do not participate in similar alternations.
- 2 are assigned to class 37.7 (SAY verbs): These errors are surprising as verbs in 47.3 and 37.7 have little in common in terms of semantics or syntax. The misclassification seems to be caused by idiosyncratic properties of two individual verbs.
- 1 is assigned to class 45.4 (OTHER CHANGE OF STATE verbs): Verbs in classes 47.3 and 45.4 are semantically quite different, and the verbs in class 45.4 are richer in terms of subcategorization and alternation behaviour, but they do share similar SCFs (e.g. PP ones) which may explain the misclassification.

Interestingly, when looking at the errors the Gaussian method made with class 47.3, they are fairly similar to the ones we just described with the Maximum entropy. Both methods confuse 47.3 with classes 51.3.2, 37.7 and 43.1. The Gaussian method assigns as many as 5 verbs to class 51.3.2. While this is considered an error here, the classes are closely related, as mentioned above. Our gold standard does not capture semantic relatedness of different Levin classes, as neither Levin nor VerbNet does it. Error analysis of the type we

have just given could be used as a step towards identifying semantically related classes.

4.7 Related Work

4.7.1 Recent work of Joanis, Stevenson and James (2007)

As mentioned in the introduction, Joanis et al. (2007) have recently published an experiment where they used SVMs to classify 835 verbs into 14 classes. They compared the performance of their best shallow syntactic features (obtained using a chunker) against deeper ones (SCF distributions obtained using Briscoe and Carroll’s system). 58% accuracy is reported with the shallow features, and only 38% with the SCF features. They used basic SCFs without parameterizing them for prepositions. We obtained considerably better results than 38% (47.5%) when using the same classification method (SVMs) with a similar basic SCF feature set. The data sets are not comparable, but our task may be more difficult as we classified 204 verbs into 17 classes (i.e. we had more target classes and less training data).

Joanis et al. say that parameterizing SCFs for prepositions might help, but that they cannot do this because Briscoe and Carroll’s system does not include preposition information. This is a wrong claim as the system does include this information. The authors do not say which version of Briscoe and Carroll’s system they used in their experiments or whether they filtered some SCFs out or used a raw version of the lexicon. They conclude that using syntactically rich subcategorization features is not helpful for English verb classification and that this might be because including information about adjuncts (not only about arguments of verbs) can be useful for verb classification. In fact, the unfiltered (raw) version of Valex includes information about both SCFs and adjuncts, and this is the likely explanation for why in most of our experiments we obtained optimal results using the raw version of Valex.

4.7.2 Comparison with Clustering

One of the starting points of this work was to investigate the performance of supervised methods as compared with unsupervised ones. We asked Anna Korhonen and Yuval Krymolowski to run their clustering methods on same data and feature sets as the ones employed in our work¹. They used their current best method for the task - a cost-based framework for pairwise clustering introduced by Puzicha et al. (2000). In this method, a cost criterion guides the search for a suitable clustering configuration. This criterion is realized through a cost function $H(S, M)$ which takes the following parameters:

- (i) $S = \{\text{sim}(a, b)\}$, $a, b \in A$: a collection of pairwise similarity values, each of which pertains to a pair of data elements $a, b \in A$.
- (ii) $M = (A_1, \dots, A_k)$: 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 $1 \leq j < j' \leq k$).

¹The results reported in this section are provided by Anna Korhonen and Yuval Krymolowski.

Lexicon	FSet1	FSet2	Fset3 (3)	Fset3 (8)	Fset3 (13)
Valex 1	39.6% ± 1.5	51.4% ± 2.5	51.5% ± 2.6	51.6% ± 2.5	51.2% ± 3.7
Valex 2	30.3% ± 1.4	44.0% ± 1.6	45.1% ± 1.8	45.0% ± 1.5	44.9% ± 1.5
Valex 3	34.6% ± 2.4	48.9% ± 1.6	49.2% ± 1.0	49.1% ± 1.1	48.3% ± 1.0

Table 4.16: Evaluation ($uPUR$) of the verb clusters

The number of clusters, k , is pre-determined and specified as an input parameter to the clustering algorithm.

The main idea underlying the clustering criteria is the preference of configurations in which similarity of elements within each cluster is generally high and similarity of elements that are not in the same cluster is correspondingly low. The following cost function is used:

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

where n_j is the size of the j^{th} cluster and AvgSim_j is the average similarity between cluster members.

They clustered the data in the three Valex lexicons using the three feature sets. As a similarity function they used the JS divergence (eq. 3.3).

In order to compare the clustering results with classification results they used a variation of the purity measure (see Korhonen et al. (2003) for details) in which they allowed each class to be dominant only in a single cluster. They then summed up the verbs in dominant classes and reported the proportion of these verbs. This measure is called *unique purity - uPUR*. It is not identical but it is quite similar to the accuracy measure used in our evaluation. Korhonen and Krymolowski ran 50 experiments on every input in order to get the distribution of performance due to the randomness in the initial clustering.

The value of k in the clustering runs ranged from 10 to 35, the best performance was obtained for k values close to the number of classes ($n = 17$). They provided the results for $k = 17$. The results for nearby k values were within the range of the standard deviation from the reported results.

Table 4.16 presents the $uPUR$ evaluation of the clusters. Clusters based on the refined features (Fset2) are significantly better than those based on the basic features (Fset1), but differences between Fset2 and Fset3 are not statistically significant. The best performance was obtained with the raw lexicon. As we can see, the best results here are more than 10% lower than our best accuracy results, indicating that supervised classification can indeed be helpful for verb classification.

Chapter 5

Conclusion and Future Work

5.1 Conclusion

In this study, we investigated various supervised machine learning techniques for automatically classifying 204 verbs into 17 lexical classes (Levin, 1993). We experimented with four supervised machine learning techniques: K nearest neighbour, Maximum entropy model, Support Vector Machines and the Gaussian model. The results show that the supervised machine learning techniques can produce accurate result for automatic verb classification and that the accuracy is clearly better than that obtained using clustering. Model based classifiers (Gaussian, Maximum entropy model) and discriminative classifiers (SVM) do not differ considerably in performance, but the K nearest neighbours, our simplest classifier, performs clearly worse than the other methods. The Gaussian model gives the final best performance at 62.5% in F-Score. However, the maximum entropy model and SVM seem more robust than the Gaussian model over varying amount of training data.

Moreover, our evaluation confirms the previous findings by Korhonen et. al. (2003, 2007) and Schulte im Walde (2006) that SCFs are helpful and informative features for lexical classification. All the classifiers generate accuracy greater than 40% with the basic feature set 1, and the accuracy is improved greatly when SFCs are parameterized for prepositions. The new extended feature set has a positive effect for the Gaussian model and SVM, except when the tense features are added. The fact that the best performance is obtained with the noisy version of the Valex lexicon suggests that Joanis et al. (2007) are right in suggesting that using also adjuncts (not only SCFs) as features can be helpful.

5.2 Future Work

5.2.1 Dealing with Polysemous Verbs

In this study, we focused on verbs which have one predominanting sense in general language. However, predominanting verb senses can vary across corpora (Roland et al., 2000) and many important medium and high frequency verbs

have flat rather than zipfian sense distributions in general language (Korhonen and Preiss, 2003). In the future, we plan to extend our system so that we can such polysemous verbs to multiple classes corresponding to their different senses. This could be treated as a multi-label classification problem and approach e.g. via binary classification (Riguzzi, 2005).

5.2.2 Improving the Input Data

In this experiment, our SCF data was obtained from three versions of the Valex lexicon (section 2.4). According to Korhonen et al. (2006a), the Valex 4 is a lot more accurate than the raw lexicon (with F-Score at 83.7% which is 60% higher than with the raw lexicon). However, in the verb classification we get the best results using the raw lexicon. Because the noise in the raw lexicon contains valuable information about adjuncts, but on the other hand having accurate information about SCFs is important for the task as well, we could investigate improving the input data by merging Valex 4 with the raw lexicon.

5.2.3 Feature selection

In this study, we used one feature set for all the classification techniques. The feature selection is crucial to the performance and is an aspect of this work which requires further attention. For the Gaussian model, we could perform the dimensionality reduction. Liu (2001) shows that the dimensionality reduction schemes can help to find a feature space with low dimensions and rich discriminant features. The performance of the Gaussian model could be improved by simply filtering out the low frequency features (e.g. many of the PP features in our new feature set were low in frequency), but there are also more sophisticated techniques available that could be used, e.g. the Principle Component Analysis (Kambhatla and Leen, 1997) and Linear Discriminant analyses (Keyser and Ney, 2004). For the maximum entropy model, the new feature set did not improve performance. A possible reason is that we had also irrelevant features among the relevant ones. We could address the problem by using an iterative feature selection algorithm (e.g. as suggested by Berger et al. (1996)) to select the most relevant and discriminant features.

5.2.4 Combining Approaches

Combining Different Classification Methods

In the experiments, some classification techniques outperformed others with different feature sets, leading to different performances with different verb classes. This shows that different classification techniques have different strengths. We could combine the strengths of different classifiers using linear interpolation. If we got the results P_1 , P_2 and P_3 from three different classifiers, we could combine them together as:

$$P = w_1 \times P_1 + w_2 \times P_2 + w_3 \times P_3 \quad (5.1)$$

The weights w_1 , w_2 and w_3 are interpolation weights. The sum of them must equal to one. They can be estimated by deleted interpolation. We divide the training data into parts, and we go through each part. Each time, we leave one

part as leave out data, and we train the model on the other parts. We estimate the weights by maximising the likelihood of the models on the leave out data.

Combining Supervised and Unsupervised Approaches

The supervised techniques are trained on the labelled data, while the unsupervised techniques do not require any training data. Many studies (e.g. Ando and Zhang (2005)) have found that the supervised methods, when combined with unsupervised learning method and a large amount of unlabelled data, often produces improved performance. This approach falls between unsupervised and supervised learning, and is therefore called semi-supervised learning.

One of the standard approaches is to use expectation maximization (EM) algorithm. The unlabelled data are treated as unseen data. We can use the EM algorithm to maximise the likelihood of the model. The transductive learning is another semi-supervised learning technique. An implementation of this algorithm is the Transductive Support Vector Machine (TSVM). (Vapnik, 1998) The TSVM tries to find the boundary that maximises the margin for both the labelled and unlabelled data. We can supply TSVM with a large amount of unlabelled verb feature vectors. It would be interesting to implement TSVM and compare the result to our inductive SVM result.

Of course, when doing classification in a new unknown domain we may not have any labelled data available. Unsupervised methods offer then a good starting point, but supervised and semi-supervised methods can be used once the set of basic classes and some labelled data is already available.

Bibliography

- Rie Kubota Ando and Tong Zhang. A high-performance semi-supervised learning method for text chunking. In *ACL '05: Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 1–9, Morristown, NJ, USA, 2005. Association for Computational Linguistics. doi: <http://dx.doi.org/10.3115/1219840.1219841>.
- A. Berger. The improved iterative scaling algorithm: A gentle introduction, 1997. URL citeseer.ist.psu.edu/berger97improved.html.
- Adam L. Berger, Stephen Della Pietra, and Vincent J. Della Pietra. A maximum entropy approach to natural language processing. *Computational Linguistics*, 22(1):39–71, 1996.
- Bran Boguraev and Ted Briscoe. Large lexicons for natural language processing: utilising the grammar coding system of ldoce. *Comput. Linguist.*, 13(3-4): 203–218, 1987. ISSN 0891-2017.
- Ted Briscoe. *Dictionary and System Subcategorisation Code Mappings*, 2000. Included in the valex release, please check <http://www.cl.cam.ac.uk/alk23/subcat/lexicon.html>.
- Ted Briscoe and John Carroll. Automatic extraction of subcategorization from corpora. In *Proceedings of the fifth conference on Applied natural language processing*, pages 356–363, San Francisco, CA, USA, 1997. Morgan Kaufmann Publishers Inc.
- Chih-Chung Chang and Chih-Jen Lin. *LIBSVM: a library for support vector machines*, 2001. Software available at <http://www.csie.ntu.edu.tw/~cjlin/libsvm>.
- Dick Croch and Tracy Holloway King. Unifying lexical resources. In *In Proceedings of Interdisciplinary Workshop on the Identification and Representation of Verb Features and Verb Classes*, Saarbruecken, Germany, 2005.
- Hoa Trang Dang. *Investigations into the Role of Lexical Semantics in Word Sense Disambiguation*. PhD thesis, CIS, University of Pennsylvania, 2004.
- Alexander G. Dimitrov and John P. Miller. Neural coding and decoding: Communication channels and quantization. *Network: Computation in Neural Systems*, 12(4):441–472, 2001.
- B. Dorr and D. Jones. Role of word-sense disambiguation in lexical acquisition, 1996.

- Bonnie J. Dorr. Large-scale dictionary construction for foreignlanguage tutoring and interlingual machine translation. *Machine Translation*, 12(4):271–322, 1997. ISSN 0922-6567.
- Ralph Grishman, Catherine Macleod, and Adam Meyers. Complex syntax: building a computational lexicon. In *Proceedings of the 15th conference on Computational linguistics*, pages 268–272, Morristown, NJ, USA, 1994. Association for Computational Linguistics.
- Kenneth Hale and Jay Keyser. A view from the middle, lexicon project work paper 10, 1987.
- Svetlana Hensman and John Dunnion. Automatically building conceptual graphs using VerbNet and WordNet. In *In Proceedings of the 3rd International Symposium on Information and Communication Technologies (ISICT)*, pages 115–120, Las Vegas, NV, 2004.
- Thomas Hofmann. Unsupervised learning by probabilistic latent semantic analysis. *Mach. Learn.*, 42(1-2):177–196, 2001. ISSN 0885-6125.
- Chih-Wei Hsh, Chih-Chung Chang, and Chih-Jen Lin. A practical guide to support vector classification, 2003.
- C. Hsu and C. Lin. A comparison of methods for multi-class support vector machines, 2001.
- Sabine Schulte im Walde. Experiments on the automatic induction of german semantic verb classes. *Comput. Linguist.*, 32(2):159–194, 2006. ISSN 0891-2017.
- Ray Jackendoff. *Semantic Structures*. The M.I.T. Press, Cambridge, MA, 1990.
- E. Joanis and S. Stevenson. A general feature space for automatic verb classification, 2003.
- Eric Joanis, Suzanne Stevenson, and David James. A general feature space for automatic verb classification. *Natural Language Engineering*, Forthcoming, 2007. doi: 10.1017/S135132490600444X.
- Nandakishore Kambhatla and Todd K. Leen. Dimension reduction by local principal component analysis. *Neural Computation*, 9(7):1493–1516, 1997. doi: 10.1162/neco.1997.9.7.1493. URL <http://www.mitpressjournals.org/doi/abs/10.1162/neco.1997.9.7.1493>.
- Daniel Keysers and Hermann Ney. Linear discriminant analysis and discriminative log-linear modeling. *17th International Conference on Pattern Recognition (ICPR'04)*, 01:156–159, 2004. ISSN 1051-4651.
- Karin Kipper, Hoa Trang Dang, and Martha Palmer. Class-based construction of a verb lexicon. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on Innovative Applications of Artificial Intelligence*, pages 691–696. AAAI Press / The MIT Press, 2000. ISBN 0-262-51112-6.

- Judith Klavans and Min-Yen Kan. Role of verbs in document analysis. In *COLING-ACL*, pages 680–686, 1998.
- A. Korhonen and J. Preiss. Improving subcategorization acquisition using word sense disambiguation, 2003.
- Anna Korhonen, Yuval Krymolowski, and Zvika Marx. Clustering polysemic subcategorization frame distributions semantically. In *ACL '03: Proceedings of the 41st Annual Meeting on Association for Computational Linguistics*, pages 64–71, Morristown, NJ, USA, 2003. Association for Computational Linguistics.
- Anna Korhonen, Yuval Krymolowski, and Ted Briscoe. A large subcategorization lexicon for natural language processing applications. In *Proceedings of LREC*, 2006a.
- Anna Korhonen, Yuval Krymolowski, and Nigel Collier. Automatic classification of verbs in biomedical texts. In *ACL '06: Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the ACL*, pages 345–352, Morristown, NJ, USA, 2006b. Association for Computational Linguistics.
- Harold W. Kuhn. Nonlinear programming: a historical view. *SIGMAP Bull.*, pages 6–18, 1982. ISSN 0163-5786.
- Beth Levin. *English Verb Classes and Alternations: a preliminary investigation*. University of Chicago Press, Chicago and London, 1993.
- J. Lin. Divergence measures based on the shannon entropy. *IEEE Transactions on Information Theory*, 37(1):145 – 151, 1991.
- Xunying Liu. Linear projection schemes for automatic speech recognition. 2001.
- R. Malouf. A comparison of algorithms for maximum entropy parameter estimation, 2002.
- Paola Merlo and Suzanne Stevenson. Automatic verb classification based on statistical distributions of argument structure. *Comput. Linguist.*, 27(3):373–408, 2001. ISSN 0891-2017.
- Stephen Della Pietra, Vincent J. Della Pietra, and John D. Lafferty. Inducing features of random fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(4):380–393, 1997.
- Steven Pinker. *Learnability and Cognition: The acquisition of Argument Structure*. Cambridge, Mass.: MIT Press, 1989. URL <http://www.isrl.uiuc.edu/amag/langev/paper/pinker89book.html>.
- J. Puzicha, T. Hofmann, and J. M. Buhmann. A theory of proximity-based clustering: structure detection by optimization. *Pattern Recognition*, 33(4): 617–634, 2000.
- Fabrizio Riguzzi. A simple approach to a multi-label classification problem, 2005.

- Douglas Roland, Daniel Jurafsky, Lise Menn, Susanne Gahl, Elizabeth Elder, and Chris Riddoch. Verb subcategorization frequency differences between business-news and balanced corpora: the role of verb sense. In *Proceedings of the Association for Computational Linguistics (ACL-2000) Workshop on Comparing Corpora*, pages 28–34, Hong Kong, 2000.
- Lei Shi and Rada Mihalcea. 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.
- Suzanne Stevenson and Eric Joanis. Semi-supervised verb class discovery using noisy features. In *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003*, pages 71–78, Morristown, NJ, USA, 2003. Association for Computational Linguistics.
- Robert Swier and Suzanne Stevenson. Unsupervised semantic role labelling. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 95–102, Barcelona, Spain, August 2004.
- Mary Swift. Towards automatic verb acquisition from VerbNet for spoken dialog processing. In *In Proceedings of Interdisciplinary Workshop on the Identification and Representation of Verb Features and Verb Classes*, Saarbruecken, Germany, 2005.
- N. Tishby, F. Pereira, and W. Bialek. The information bottleneck method. In *Proceedings of the 37-th Annual Allerton Conference on Communication, Control and Computing*, pages 368–377, 1999.
- Vladimir N. Vapnik. *The nature of statistical learning theory*. Springer-Verlag New York, Inc., New York, NY, USA, 1995. ISBN 0-387-94559-8.
- Vladimir N. Vapnik. *Statistical Learning Theory*. Wiley-Interscience, 1998. ISBN 0471030031.
- Steve Young, Dan Kershaw, Julian Odell, Dave Ollason, Valtcho Valtchev, and Phil Woodland. *The HTK Book for HTK V3.0*. Cambridge University Press, Cambridge, UK, 2000.
- Le Zhang. *Maximum Entropy Modeling Toolkit for Python and C++*, December 2004.