Automatic Verb Classification Using a General Feature Space

by

Eric Joanis

A thesis submitted in conformity with the requirements
for the degree of Master of Science
Graduate Department of Computer Science
University of Toronto

Abstract

Automatic Verb Classification Using a General Feature Space

Eric Joanis
Master of Science
Graduate Department of Computer Science
University of Toronto
2002

We develop a general feature space that can be used for the semantic classification of English verbs. We design a technique to extract these features from a large corpus of English, while trying to maintain portability to other languages—the only language-specific tools we use to extract our core features are a part-of-speech tagger and a partial parser. We show that our general feature space reduces the chance error rate by 40% or more in ten experiments involving from two to thirteen verb classes. We also show that it usually performs as well as features that are selected using specific linguistic expertise, and that it is therefore unnecessary to manually do linguistic analysis for each class distinction of interest. Finally, we consider the use of an automatic feature selection technique, stepwise feature selection, and show that it does not work well with our feature space.

Acknowledgements

Suzanne Stevenson, my advisor, has provided me with continuous support and encouragement, through the times when I thought it wasn't worth going on, until the final completion of this thesis. I am grateful for ideas shared, endless useful (and sometimes chatty) discussions, and guidance through this research. Without her help, I can say that this thesis could not have been. Thanks!

Sabine Schulte im Walde, as a senior student doing related work, provided me with some coaching and ideas, both when she visited Toronto and at ACL-02. She warned me not to expect stepwise feature selection to work well when I still naively believed that it "of course" would (even though Suzanne was never really convinced). Thanks also to Diana McCarthy for engaging in really interesting discussions about our respective work at ACL-02.

Thanks to Gerald Penn, for reading my thesis and providing helpful feedback; Graeme Hirst, for supporting me during my initial years in Toronto and for significantly improving my ability to write English; Sam Roweis, for advice on machine learning; Tristan Miller, for his precise copy editing; and François Pitt, for being a LaTeX god and a good friend for all these years.

Of course, I owe great thanks to my partner, ZuZu Gadallah, who provided invaluable support throughout the process of writing my thesis, despite my mental incoherence in the last months. Through many heated debates on proper statistics, she also helped me shape up the analysis of my results.

Thanks also to Vivian Tsang and the CL group, Martha Hendriks, Jim Clarke, Alan Skelley, and Samir Girdhar for their support and assistance.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Statement of Purpose

In any language, verbs can be grouped together into semantic classes which share common elements of meaning. It is well-known in linguistics that verbs in such classes will also share syntactic characteristics, since the semantics of a verb at least partially determines its syntactic behaviour (Pinker, 1989; Levin, 1993). Using this syntactic–semantic link as a central assumption, Levin (1993) proposed a semantic classification of English verbs, bringing together decades of linguistic research. In computational linguistics, semantic classifications of verbs are useful in the construction of lexicons for a number of applications, such as machine translation (*e.g.*, Dorr, 1997), natural language generation (*e.g.*, Stede, 1999), and information retrieval (*e.g.*, Klavans and Kan, 1998).

Recently, a number of researchers have worked on using the syntactic–semantic link to automate, at least in part, the process of verb classification (see §1.2). Most of these studies have used some type of features to quantify or describe the syntactic characteristics of verbs, using the assumption that information about the syntactic use of a particular verb can provide information about its semantics. Supervised or unsupervised classification techniques are then used to recover semantic information from these features.

In particular, Merlo and Stevenson (2001) have used, as features, five linguistically motivated statistical indicators for the membership of verbs in three classes they considered. Building on the syntactic–semantic link, they hypothesize that statistics regarding the use of particular verbs can be automatically gathered from a large corpus and used as features in a machine-learning system to automatically perform the semantic classification.

The experiments Merlo and Stevenson (2001) conducted gave promising results, but linguistic expertise was required to develop features that were good indicators for the particular classes they considered. The need for linguistic expertise limits the applicability of the method,

since similar expertise would be required for each class distinction to be considered. Instead, we propose an analysis of possible verb class distinctions, which generalizes Merlo and Stevenson's features to a large space of features that could potentially cover any semantic classes. Accompanied by a technique to automatically select the relevant features for a given classification task, or a machine-learning system capable of ignoring irrelevant features, this general feature space could be used for the automatic learning of any semantic verb class distinction in English. If our approach is successful, it will no longer be necessary to analyze the differences between individual classes to devise relevant features for a specific class distinction. Furthermore, if the approach used to build this feature space was transferable across languages, general feature spaces could be built for other languages as well.

In this study, we experiment with a new approach for building a general feature space for the semantic classification of English verbs, and test it with a number of verb classification tasks of varying complexity. We do not address the transferability of the feature space directly. However, we view alternation behaviour as an integral part of the definition of verb classes in any language, since the semantic classification of verbs is largely based on the syntactic expression of their arguments. While our feature space would certainly need to be adapted and expanded to deal with other languages, we expect the sufficiently useful extraction of argument slots, as we perform it, to play a key role in feature extraction for any language. We therefore try to use only those tools that can reasonably be expected to exist or be readily developable for other languages. For example, we use a partial, rather than full, parser, which produces noisier results, but is more readily available for languages other than English.

1.2 Previous Work on Automatic Verb Classification

Levin (1993) (hereafter referred to simply as Levin) manually classifies English verbs into semantic categories according to their assignment of semantic arguments to syntactic positions, as well as the diathesis alternations they undergo. Diathesis alternations involve switching semantic arguments between different syntactic positions. For example, the thing being broken can occur in the direct object slot of the verb *break*, as in *The boy broke the window*, or in the subject slot, as in *The window broke* (causative/inchoative alternation, Levin §1.1.2.1). Since Levin published her classification, several approaches have been taken towards automating verb classification, and towards automatically identifying participation in a diathesis alternation.

Dorr and Jones (1996) investigate the theoretical potential of using syntactic frames to classify verbs. (A syntactic frame is a list of the syntactic slots in which verb arguments are found, *e.g.*, NP-V-NP-PP$_{for}$ or NP-V-NP-NP). They define the syntactic signature of a Levin verb class as two lists of syntactic frames—those allowed and those not allowed—according to

Levin's examples. Note that these are not statistical features, but simple lists, unlike most other work we review here. They show that 187 (97.9%) of the 191 Levin classes have unique signatures, which suggests that the syntactic–semantic link can be exploited computationally. They then argue that if one performs word-sense disambiguation (WSD) on a verb, the signature of its sense in a given Levin class will be the same as the signature of its class, and therefore that 97.9% accuracy can also be obtained when classifying disambiguated verbs into the Levin classes. For "novel verbs" (verbs and verb senses not covered in Levin), they propose to use the syntactic codes from Longman's Dictionary of Contemporary English (LDOCE) (Procter, 1978) to acquire syntactic signatures. They then assign the novel verb to the closest matching Levin class, based on the syntactic signature of the classes. They evaluated this method with 134 verb senses, and found that 82 (61%) were accurately classified—an impressive result given that the chance baseline accuracy is below 1/191 (since they also consider new classes not covered in Levin) (Dorr, 1997).

However, the syntactic signatures used by Dorr and Jones (1996) are either extracted directly from Levin's examples, or from LDOCE, and therefore they are essentially noise-free, a situation that cannot be reproduced with any corpus-processing tools. Therefore we cannot compare their results with classification experiments based on statistical features extracted from corpora, such as the ones we present later in this section. Nevertheless, by exhibiting a computational use of the syntactic–semantic link, Dorr and Jones (1996) show that it is worthwhile to try to acquire syntactic information automatically for the purpose of semantic verb classification.

Oishi and Matsumoto (1997) have produced a semantic classification of verbs in Japanese by combining what they call a thematic classification based on case frames with an aspectual classification based on adverbs. They take advantage of the case markings in Japanese, so that their case frames—lists of the case markings that the arguments of a verb can take—are equivalent to syntactic frames. They obtained the list of possible case frames and adverbs for each verb from a Japanese dictionary of dependency relations. They then used the acquired case frames and adverb lists to build very fine-grained thematic and aspectual classifications consisting of 173 and 328 classes, respectively. Manually combining these two classifications, they built a network of 38 semantic classes for Japanese verbs. They argue that their thematic and aspectual classifications made building the semantic classification easy, but they still had to build it manually. In our work, we attempt to do most of the classification work automatically, and we attempt to eliminate the need for external sources of knowledge, such as the dictionary they use.

Schulte im Walde (1998, 2000) has investigated the clustering of English verbs using distributional statistics over syntactic frames acquired using a statistical parser. She tested her approach with 153 verbs in 80 Levin classes, and showed that the approach had good poten-

tial. In more recent work, she applied the same technique to 57 German verbs in fourteen classes, and had reasonable success in clustering them (Schulte im Walde and Brew, 2002). It is difficult to interpret the results of these studies, however, since it is unclear how one can objectively quantify the quality of a set of automatically built clusters. Schulte im Walde and Brew (2002) propose an evaluation method, but the numbers are not meaningful without the detailed explanation presented in the paper, so we do not present them here.

An important part of the work in producing a verb classification like Levin's is identifying whether a given verb can participate in a particular diathesis alternation. For that purpose, McCarthy (2000, 2001) uses selectional preferences encoded over a cut in the WordNet 1.5 noun hierarchy.[1] Consider a diathesis alternation where a role can be switched between two syntactic slots. McCarthy was able to determine whether a verb can participate in the diathesis alternation with an accuracy of around 70% (baseline: 50%) by measuring the similarity of the selectional preference the verb has for those two slots. This technique could only be applied to a few languages other than English, however, since it relies on the availability of an ontology such as WordNet. McCarthy also experimented with a simple lemma overlap measure, and achieved an accuracy around 60%. We use a similar lemma overlap measure in this study (see §2.8).

Some researchers have used corpus-based statistics to acquire information about verb tokens, rather than verb types.[2] Many verbs have different senses which belong to different Levin classes. Lapata and Brew (1999) used joint frequency statistics over verbs, verb classes, and syntactic frames to infer the most likely Levin class of a verb token, given the syntactic frame it occurs in. They experimented with benefactive and dative verbs. Considering only the frames NP-V-NP-NP, NP-V-NP-PP$_{to}$, and NP-V-NP-PP$_{for}$, they were able to correctly identify the Levin class of unambiguous verb tokens 91.8% of the time (baseline: 65.7%), and of ambiguous ones 83.9% of the time (baseline: 61.3%).

Also working at the verb token level, Siegel (1999) used fourteen carefully selected linguistic indicators to automatically determine the aspect of a verb token. Using his method, he could distinguish states from events with an accuracy of 93.9% (baseline: 83.8%). He could also distinguish between culminated and non-culminated events with an accuracy of 74.0% (baseline: 63.3%).

The goal of Lapata and Brew's (1999) work above was to determine the semantic class of a

[1] A cut in a tree is set of nodes that induces a partition of its leaves. Thus, each element of a cut groups together all the leaves it dominates, and each leaf in the tree is dominated by one of the cut elements (Li and Abe, 1998). A WordNet cut is defined similarly, but since the noun hierarchy is a directed acyclic graph, each leaf may be dominated by more than one cut element.

[2] A token is an individual occurrence of a word. A type is a category, of which a token is an instance. Thus, *verb type* refers to the verb in general, whereas *verb token* refers to its use in a particular sentence.

Class/Transitivity		Example	Subject	Object
Unergative	intr.	The horse raced past the barn.	Agent	
	trans.	The jockey raced the horse past the barn.	Causative Ag.	Agent
Unaccusative	intr.	The butter melted in the pan.	Theme	
	trans.	The cook melted the butter in the pan.	Causative Ag.	Theme
Object-Drop	intr.	The boy played.	Agent	
	trans.	The boy played soccer.	Agent	Theme

Table 1.1: Thematic role assignment for optionally transitive verb classes (Merlo and Stevenson, 2001)

verb token using syntactic frame information. In a different kind of task, Gildea and Jurafsky (2002) instead classified the elements of the syntactic frame itself—*i.e.*, the arguments of the verb—labelling each according to its semantic role with respect to the verb in a given sentence. Using probability estimates derived from a training corpus of sentences manually annotated with semantic role labels, they achieved about 80% accuracy on argument labelling in unseen test sentences.

Unfortunately the need for manual annotation of the semantic roles for training purposes greatly limits the current applicability of the Gildea and Jurafsky (2002) method to a fairly small number of verbs, and only in English. (For example, in their work, 927 different English verbs occurred in the annotated database.) However, it is clear that progress in such methods goes hand-in-hand with research on semantic verb classification (at the type or token level), as the latter is, in large part, an attempt to indirectly determine the possible semantic role labels for the arguments of a verb, by determining its class.

Returning to the task of semantic verb classification at the verb type level, we note that Merlo and Stevenson (2001) used five statistical features describing the usage of verbs in a corpus of English to classify verbs into three optionally transitive classes. The class distinction they considered is particularly interesting because it involves three classes of verbs that occur in the same syntactic frames, but assign different thematic roles to each syntactic slot (see Table 1.1).

The features used by Merlo and Stevenson (2001) (see Table 1.2) were carefully designed to correlate with the three optionally transitive classes. Classification tests gave 69.8% accuracy, a significant improvement over the 33.9% (20/59) baseline (obtained by classifying all verbs in one class). The work in this thesis builds on theirs, expanding their features to a general feature space.

Feature	Description
TRANS	The proportion of use of a verb in transitive frames (*vs.* intransitive).
PASS	The proportion of use of a verb in passive clauses (*vs.* active).
VBN	The relative frequency of the past participle among past tense uses, *i.e.*, VBN/(VBN + VBD).
CAUS	A measure of overlap between the lemmas occurring in the subject and direct object positions, intended to quantify the causativity of a verb.
ANIM	An estimate of the animacy of the subject position based on the use of personal pronouns.

Table 1.2: Features used by Merlo and Stevenson (2001)

1.3 Outline of the Study

Expanding the approach of Merlo and Stevenson (2001), we define a general feature space for English verb classification in Chapter 2. Our features are statistical indicators of, for example, general verb usage, the use of specific syntactic slots, and the participation in various alternations defined in Part I of Levin. In developing the feature space, we included indicators that are potentially useful for verb classification, but that are also easy to extract from a corpus.

In Chapter 3, we show how we estimate the value of our features by counting specific verb usage frequencies over a large corpus of English, the British National Corpus. We use a partial parser to identify the syntactic constructions required to estimate the value of our features. For each verb, we obtain a vector of features which are statistical indicators for the use of the verb in English and, if our approach is successful, membership in semantic verb classes.

Since we are using a large feature space (our core feature space has 234 features), which will contain features of varying usefulness for any particular classification task, we explore some approaches to automatic feature selection in Chapter 4. The machine-learning system we use, C5.0, performs feature selection internally. Additionally, we experiment with stepwise feature selection, a method which attempts to select the features with which the machine-learning system achieves the highest estimated accuracy on our training data.

In Chapter 5, we present ten verb classification tasks involving between two and thirteen classes, with which we experimentally verify the usefulness of our feature space and our feature selection techniques. Our experimental results (Chapter 6) show that classifiers built using our core feature space reduce the error rate by 40% or more from the chance baseline for these ten tasks, *e.g.*, achieving as high as 84% and 72% accuracy on two- and three-way classifications, respectively. They also generally perform as well as classifiers built using linguistically-motivated

subsets of our core feature space. However, our results show that stepwise feature selection, as we implemented it, generally hurts predictive accuracy on unseen data.

In Chapter 7, we review the contributions of this study and consider a number of directions for future work.

Chapter 2

A General Feature Space

In this chapter, we define our feature space, a first attempt at a general feature space for the classification of English verbs. Our features expand on the five features defined by Merlo and Stevenson (2001): we started with the five features they used and attempted to generalize each of them to a class of potentially useful features. We also used the alternations described in Part I of Levin for additional features, with the goal of developing a feature space that can be used for the automatic distinction of any verb classes, or at least any Levin verb classes.

The feature space is summarized in Table 2.1 and described in detail in the rest of this chapter. Five of the feature categories are motivated by the five features of Merlo and Stevenson (2001), as indicated in the table. The first three categories of features are also expected to capture general verb usage information about the verb classes, potentially providing some aspectual information as well. The remaining six categories of features are all motivated by alternations described in Part I of Levin, as well as the corresponding syntactic frames and the specific assignment of arguments to syntactic slots in those frames. It is important to note that the features are motivated by possible alternations in English, as described in Part I of Levin, and not by the behaviour of specific verb classes.

The first eight categories of features form our core features, and are the the main focus of our work. The last category, syntactic collocation features, are a preliminary attempt at encoding syntactic collocations, or a simplified form of selectional preference, within our feature space. They are experimental and should not be considered definitive. Furthermore, they make use of WordNet, a resource that we cannot expect to have access to for most languages other than English, and therefore we cannot consider them as a part of our core feature space.

All the features are proportions of use of a verb in a particular way, with respect to the overall frequency of the verb as the head verb in a clause in the corpus, unless otherwise specified. In the following text, the "target verb" refers to the verb for which these proportions are being calculated.

Feature Category	Unique Count	Actual Count	Motivation
Part of speech of the verb	6	6	M&S: VBN; general use and aspect
Frequency of auxiliaries and modals	11	11	general use and aspect
Combined feature: passive	1	2	M&S: PASS; general use and aspect
Relative frequency of derived forms	3	3	Levin: §5.4, §7.1,2
Semantically empty constituents	4	6	Levin: §6.1, §8.4
Frequency of various syntactic slots	78	84	M&S: TRANS; correlated with syntactic frames
Animacy of NPs in various slots	76	80	M&S: ANIM; Levin: many alternations
Measure of slot overlap	40	42	M&S: CAUS; Levin: §1–4
Core features: total	219	234	
Syntactic collocation features	1140	1260	Selectional preference
All features: total	1359	1494	

Table 2.1: Feature categories with counts and motivation. Unique counts reflect the number of features we define in this section. Actual counts reflect the number of features given to the machine-learning system. They are higher than unique counts because some features are calculated in two different ways (see §3.7). The motivation indicates from which source we developed each category of features, including Merlo and Stevenson (2001) (M&S), Levin alternations, and general and aspectual information.

Verb Form	Penn tag	CLAWS tag
Base form: infinitive	VB	VVI
Base form: imperative Base form: subjunctive		VVB
Non-third person singular present	VBP	
Third person singular present	VBZ	VVZ
Past participle	VBN	VVN
Past tense	VBD	VVD
Gerund or present participle	VBG	VVG

Table 2.2: The Penn and CLAWS tags for verbs

2.1 Part of Speech of the Verb

The first set of features encodes the proportion of occurrence of the six part-of-speech (POS) tags that verbs can take in the Penn tagset: VB, VBP, VBZ, VBG, VBD, and VBN. In this study, we mostly work with the British National Corpus (BNC), but we still use the Penn tagset because our corpus processing tools require it (see §3). The mapping of verb tags from the CLAWS tagset (used in the BNC) to the Penn tagset is straightforward, though not exact, as shown in Table 2.2 (BNC, 2000; Santorini, 1995). When we work with the BNC, we map VVI to VB and VVB to VBP, since this mapping preserves the most information, even though it is incorrect for imperatives and subjunctives.[1]

2.2 Auxiliaries and Modals

The second set of features measures the proportion of verb groups with the target verb as head that include each auxiliary or modal: *to have*, *to be*, *will*, *can*, *could*, *might*, *must*, *would*, *should*, and *may*. An additional feature gives the proportion of verb groups that include any of the eight modals.

[1]We could map VVB to VB instead, but this would mask the potentially useful distinction between infinitives and the non-third-person-singular present tense. We also suspect it would introduce more errors, since we expect the non-third-person-singular present tense to occur more frequently than imperatives or subjunctives in a balanced corpus of English, but we have not verified this.

2.3 Passive Voice

The passive voice is a feature category on its own because for English, it is really a combined feature. The passive feature measures the proportion of use of the target verb where a) the POS tag is VBN, and b) the auxiliary *to be* occurs in the verb group, and both of those conditions hold simultaneously. Following Merlo and Stevenson (2001), we use the total count of VBN and VBD as the denominator in one instance of this feature. We use the overall frequency of the target verb in a verb group as denominator in a second instance of this feature. In principle, we need only the latter, but we keep them both and let the classification system decide which one is more useful.

This feature is the only one we keep from the original study by Merlo and Stevenson (2001). As a combined feature, it does not really belong with any of our feature categories, but we feel it is an important feature in general. In future work, more combined features might be added to the feature space, and then the passive voice feature will belong to that class.

2.4 Derived Forms

The fourth set of features measures the relative frequency of derived forms with respect to the overall frequency of the verb itself. Specifically:

$$\text{rel-freq}(\textit{zero-nominal}) = \frac{\text{freq}(\textit{zero-nominal})}{\text{freq}(\textit{zero-nominal}) + \text{freq}(\textit{verb})}$$

$$\text{rel-freq}(\textit{vbn-adjectival}) = \frac{\text{freq}(\textit{vbn-adjectival})}{\text{freq}(\textit{vbn-adjectival}) + \text{freq}(\textit{verb})}$$

$$\text{rel-freq}(\textit{vbg-adjectival}) = \frac{\text{freq}(\textit{vbg-adjectival})}{\text{freq}(\textit{vbg-adjectival}) + \text{freq}(\textit{verb})}$$

The quantity freq(*zero-nominal*) is the frequency with which the base form of the target verb, possibly with the plural marker *-s* or *-es* added, occurs with a noun POS tag (NN or NNS) in the corpus. The quantities freq(*vbn-adjectival*) and freq(*vbg-adjectival*) are the frequencies with which the past and present participles, respectively, are deemed to be used as adjectives. A participle is deemed to be used as an adjective if it is tagged as an adjective (POS tag JJ) or if it is tagged as a participle (POS tag VBN or VBG, respectively) but is not part of a verb group. Finally, freq(*verb*) is the overall frequency of the target verb being used as a verb.

2.5 Semantically Empty Constituents

The fifth set of features measures the relative frequency of constructions involving semantically empty constituents. The first such construction we consider is the impersonal *it*, which can

occur in the subject position (*e.g.*, *it occurs to me that ...*; *it rains*) and in the direct object position (*e.g.*, *I like it that ...*; *they've got it made*). We cannot distinguish the impersonal *it* from the regular personal pronoun *it*, but we still expect that the occurrence of *it* in the subject or direct object positions should be higher for verbs which allow the impersonal *it*. Therefore, the relative frequency of *it* in those two positions, with respect to the overall frequency of the target verb, should be informative features, and these are the two we retain.[2]

The second construction we consider is the insertion of *there*, as in *There runs a cat in the alley* or *There developed a problem* (Levin, §6.1). Conveniently, *there* occurs in the subject position of any non-copulative verb only when *there*-insertion has occurred.[3] Thus, we can accurately estimate the relative frequency of *there*-insertion by the proportion of uses of the verb for which *there* is the subject with respect to the overall frequency of the target verb. We also measure this proportion with respect to transitive frames only, which indicates what proportion of transitive frames are really intransitive frames which have undergone *there*-insertion.

2.6 Syntactic Slots

The sixth set of features measures the relative frequency of occurrence of the verb with particular syntactic slots, including subject, direct object, indirect object, adverbs (occurring after the verb group), and prepositional phrases. Since transitive and intransitive frames sometimes have different arguments assigned to the subject slot, we add the relative frequency of the subject in a transitive frame and in an intransitive frame, *i.e.*, what proportion of transitive (or intransitive, respectively) clauses using the target verb have a subject (including expletive subjects). Finally, the middle alternation (Levin §1.1.1) often involves a transitive verb being used intransitively with some modifier, such as an adverb. We therefore add the relative frequency of adverbs in intransitive frames.

For prepositional phrases, we use individual counts for the 51 prepositions which occur at least 10 000 times in the BNC, as well as nineteen groups of related prepositions. Note that many prepositions are counted both individually and as part of a group. To these we add an additional feature: the proportion of uses of the verb where any prepositional phrase occurs at all.

[2]Here we do not distinguish between transitive and intransitive frames. However, the impersonal *it* subject frequently occurs in intransitive frames, so it would have been useful to consider the relative frequency of *it* used as the subject in intransitive frames separately. We thought about this after completing the evaluation on unseen test data, when it was too late to change the feature space for this study, so this feature should be added in future work.

[3]In some grammatical analyses, *there* is not considered to be in the subject position, but in others it is considered an expletive subject. Our corpus-processing techniques treat it as a subject, but we make no linguistic claim about what the correct analysis should be.

The 51 prepositions considered individually are *about, above, according to, across, after, against, along, among, around, as, as well as, at, away from, because of, before, behind, between, beyond, by, despite, during, for, from, in, in terms of, including, into, like, near, of, off, on, out of, outside, over, per, rather than, round, since, such as, through, throughout, to, towards, under, until, up to, upon, with, within,* and *without*. Note that this list includes prepositions which introduce an adjunct, as well as prepositions which introduce an argument.

The nineteen preposition groups we consider were created with the help of Quirk *et al.* (1985, §9), by putting together prepositions that are very close in meaning or in expected usage. Table 2.3 shows the nineteen groups, and gives prepositions which, although considered only individually, are treated as groups to allow for alternate spellings which occur in the corpus.

2.7 Animacy

The seventh set of features expands on the animacy feature used by Merlo and Stevenson (2001). In their work, they estimate the proportion of animate nouns occurring in the subject position by counting the proportions of subjects which are nominative personal pronouns except *it*—one of *I, you, he, she, we,* and *they*. They justified including *they* in this list by confirming that most instances of *they* in the WSJ (the corpus they used) refer to animate entities.

In this study, we consider pronouns in all positions, so we must include objective and reflexive personal pronouns. We assume that a pronoun refers to an animate being if it is any personal pronoun except *it* and *itself*, *i.e.*, one of *I, me, myself, you, yourself, he, him, himself, she, her, herself, we, us, ourselves, yourselves, they, them,* and *themselves*. We also consider the interrogative/relative pronouns *who* and *whom* and the reflexive indefinite pronoun *oneself*. Finally, we consider any proper noun phrase that we can identify as naming a person to be animate (see §3.7). Our animacy feature for a given slot and target verb is the total frequency of the above pronouns and proper NPs that occur in the slot with the target verb, divided by the frequency of occurrence of the slot with the target verb. In future work, we might also consider words which are hyponyms of *person* or *causal agent* in WordNet, but for now, the feature is intentionally kept simple to see how helpful it can be as-is, and to maintain its transferability to other languages.

For most personal pronouns, it is reasonable to assume that they refer to animate beings most of the time. However, it is not clear that this assumption holds for *they, them,* and *themselves*. Aone and McKee (1996), who define a similar animacy feature in their verb classification work, exclude *they*, along with *it*, from their animacy estimate for that reason. In order to determine how valid that assumption is, we have randomly extracted 100 occurrences each of *they, them* and *themselves* from the BNC, and manually classified them as animate

Group Name	Preposition List
%above	*above, on top of*
%add	*besides, as well as, in addition to, in conjunction with, plus, including*
%behind	*behind, in front of*
%between	*between, among, amongst, amid, amidst, in between*
%cause	*because of, on account of, for fear of*
%despite	*despite, in spite of, notwithstanding*
%dest	*on to, onto, into*
%during	*during, throughout*
%except	*except, except for, apart from, other than, but, but for, excluding, save, aside from, save for, excepting, bar, barring*
%inside	*outside, inside, outside of*
%instead	*rather than, instead of*
%like	*like, unlike*
%near	*near, nearer, nearest, beside, near to, nearer to, nearest to, close to, opposite, adjacent to, next to*
%path	*through, toward, towards, round, around, across, along, past, up, down, beyond*
%regard	*as to, concerning, as for, regarding, with regard to, in view of, with respect to, re, as regards, in regard to, pertaining to*
%reltime	*after, before, until, since, till, prior to*
%source	*out of, out, away from, off, off of*
%spatial	the union of the groups %under, %near, %above, %over, %behind, and %inside
%under	*under, below, beneath, underneath*
Prepositions for which alternate spellings are accepted	
%about	*about, 'bout*
%for	*for, fer*
%of	*of, o'*
%over	*over, o'er*
%with	*with, wi'*

Table 2.3: Preposition groups

Pronoun	Animate Proportion in Sample
they	63%
them	54%
themselves	82%

Table 2.4: Animacy of the personal pronouns *they, them* and *themselves*, each estimated using a random sample of 100 occurrences from the corpus.

or inanimate. When animacy could not be established from the sentence alone, we looked at the surrounding text, so that we could establish the animacy of every example extracted. The results show that the animacy assumption holds for *themselves*, but for *they* and *them*, only slightly more than half the examples are animate, so including them in our animacy estimate adds almost as much noise as signal (see Table 2.4). We include them regardless, since leaving them out would also result in lost information. In future work, we might consider leaving them out or counting only a proportion of their occurrence counts based on estimates of how often they refer to animate beings.

We calculate the animacy estimate for all the slots listed in §2.6 except adverb—*i.e.*, for subject, subject in a transitive frame, subject in an intransitive frame, direct object, indirect object, and for object of preposition slots as described above (*i.e.*, for individual prepositions and for preposition groups).

We should note here that the idea of calculating the animacy estimates for the transitive and intransitive subjects separately is not strictly ours. Aone and McKee (1996) used a subject animacy estimate similar to ours, but they did not distinguish between transitive and intransitive frames. They had inconclusive results in their work for verbs that can alternatively express a process or state, or a caused process, because the expected animacy of the subject for those verbs depends on the transitivity of the verb. For that reason, they noted that they would calculate the animacy of the subject separately for transitive and intransitive frames in future work.

2.8 Slot Overlap

The eighth set of features measures overlap between various syntactic slots. Many alternations involve the expression of the same semantic argument in two different syntactic slots. It is expected that if a verb (or a class of verbs) can undergo a particular alternation, the sets of lemmas (nouns stripped of any plural markers) found in two slots that express the same semantic argument should overlap to some degree. Consequently, a measure of the overlap of

lemma usage between two slots can be used as an indicator for the participation of a verb in an alternation which involves those two slots (McCarthy, 2000, 2001; Merlo and Stevenson, 2001).

Merlo and Stevenson (2001) measured the lemma overlap between the subject and the direct object slots. This overlap was expected to be higher for verbs which undergo causative alternations (Levin §1.1.2), since they allow the expression of the agent role either as an intransitive subject, or as a transitive object when a causative role is introduced in the subject slot—*e.g.*, *The horse races* can alternate to *John races the horse*.

For our feature space, we consider the overlap between each pair of slots that corresponds to a specific alternation in Part I of Levin. For example, Levin's §2.3.5 states that in the intransitive *clear* alternation, the object of *from* can alternate with the intransitive subject. Using Levin's examples, *Clouds cleared from the sky* alternates with *The sky cleared*. We therefore consider the overlap between the intransitive subject and the object of the preposition *from*. In a number of cases, it was more appropriate to consider a group of prepositions instead of a single preposition. For example, Levin's §1.4.1 states that in the locative preposition drop alternation, directional information can be expressed as a direct object or as the object of a directional preposition, which corresponds to our preposition group %path. Again using Levin's examples, *Martha climbed up the mountain* can alternate with *Martha climbed the mountain*. To cover this alternation, we consider the overlap between the direct object and the object of the prepositions in the %path group. See Table 2.5 for a list of all the slot overlap features we use, with the corresponding alternations.

We use the method of Merlo and Stevenson (2001) to calculate slot overlaps. Let S_X be the set of lemmas occurring in slot X for a given verb, MS_X be the multiset of lemmas occurring in slot X, and w_X be the number of times lemma w occurs in MS_X. Then we define the overlap between slots A and B, $over(A, B)$, as follows:[4]

$$\text{over}(A,B) = \frac{\sum_{w \in S_A \cap S_B} \max(w_A, w_B)}{|MS_A| + |MS_B|}$$

For example, if $\{a, a, a, b, b, c\}$ is the multiset of lemmas occurring in the subject slot for a verb, and $\{a, b, b, b, b, d, d, e\}$ is the multiset of lemmas occurring in the direct object slot for the same verb, then over(*subject*, *object*) $= (3+4)/(6+8) = 0.5$. We have briefly experimented with different definitions of the overlap, replacing max with min or the geometric mean, which might have more intuitively quantified the overlap between two slots. This made no significant difference to the classification performance, and therefore we retained the original formula used

[4]When either multiset is empty, we defined the overlap to be 0, but we should have used *undefined*, as is allowed by many machine-learning systems, including the one we use. We thought about this after completing the evaluation on unseen test data, when it was too late to use it for this study, but *undefined* should be used where appropriate in future work.

Slots		Relevant Alternations
subj	obj	1.1.1, 1.1.2, 4.1, 4.2
subj	dobj	1.1.1, 1.1.2, 4.1, 4.2
subjintr	obj	1.1.1, 1.1.2
subjintr2	dobj	1.1.1, 1.1.2
subjtrans	from	1.1.3, 3.8, 3.10
subjintr	%with	1.2.6.2, 2.3.4
subjtrans	%with	1.2.6.2, 2.13.4, 3.3, 3.4, 3.5
obj	at	1.3
obj	on	1.3, 2.3.1, 2.7, 2.10, 2.12
obj	against	1.3, 2.8
obj	%dest	1.4.1, 2.3.1
obj	around	1.4.1, 2.3.1
obj	%over	1.4.1, 2.3.1
obj	into	1.4.1, 2.3.1, 2.4.1
obj	in	1.4.1?, 2.3.1?, 2.11, 2.12, 2.13.3
obj	%path	1.4.1, 2.3.1?
obj	%spatial	1.4.1?, 2.3.1
obj	%with	1.4.2, 2.3.1, 2.6, 2.7, 2.8, 2.9
iobj	to	2.1
iobj	%for	2.2
obj	from	2.3.2, 2.3.3
obj	%of	2.3.2, 2.3.3
obj	off	2.3.3
subjintr	in	2.3.4, 2.13.5
subjintr	%spatial	2.3.4?
subjintr	from	2.3.5, 2.4.2
subjtrans	%of	2.3.5
obj	out_of	2.4.1
subjintr	into	2.4.2
obj	to	2.6
obj	through	2.9
obj	%for	2.10, 2.11, 2.13.1, 2.13.3
obj	as	2.14
subjtrans	in	3.1, 3.2, 3.6
subjtrans	on	3.6
subjtrans	%spatial	3.6
subjtrans	into	3.7
subjtrans	out_of	3.8
subjtrans	%for	3.9
subjtrans	obj	4.1, 4.2
subj	pp-any	
obj	pp-any	

Table 2.5: Slot overlap features we consider, with the lists of alternations for which they are relevant. The alternations are all given as section numbers in Levin (1993). Alternations for which the relevance of an overlap measure is uncertain are marked with "?". Slots of the form "%group" refer to preposition groups (see Table 2.3).

by Merlo and Stevenson (2001).

McCarthy (2000, 2001) proposed a lemma overlap formula which uses min and divides by the cardinality of the smallest multiset instead of the sum of the two. This formula has the advantage of yielding more intuitive values between 0 (no overlap) and 1 (complete overlap), whereas ours would give a value nearer 0.5 for complete overlap.

As another indicator of participation in an alternation, McCarthy (2000, 2001) proposed to use measures of similarity between the selectional preference of a verb for two slots involved in the alternation. She used a tree-cut model within the WordNet hierarchy to represent selectional preference, and experimented with measures like the Kullback–Leibler distance and the α-norm of Lee (1999). In her experiments, she found that she could identify whether a verb participates in causative alternations (Levin §1.1.2) with 73% accuracy (baseline: 50%) using a selectional preference similarity measure, but with only 63% accuracy using her lemma overlap measure. However, this approach uses WordNet, a resource which we cannot expect to have for all languages. Since we want to keep our study as transferable as possible, we prefer to use a simple lemma overlap measure.

In this study, we modify the lemma overlap approach used by Merlo and Stevenson (2001) to treat names of people in a special way. As mentioned in the previous section, we can identify some proper noun phrases as being the name of a person (details in §3.7). When we measure the slot overlap, we take advantage of this by treating all proper names identifying a person as if they were the same lemma. This captures the intuition that, in terms of semantic slot selection, names of people are interchangeable. The occurrence of names of people in two syntactic slots is an indicator for a possible diathesis alternation, just as the occurrence of the same lemma in both slots does.

2.9 Syntactic Collocations

The ninth and final set of features is a simplified representation of selectional preference, which we will call syntactic collocations. We use a simple and preliminary definition, which we have not tried to refine. Although we would like to fine-tune the representation in future work, it falls outside the scope of this study to do so.

We measure syntactic collocations for the slots listed in §2.6, except that we do not use all 51 individual prepositions. Since each slot considered adds a large number of features to our feature space, we have chosen to restrict the preposition list to only the most frequently used ones. Therefore we consider the nineteen preposition groups, but only thirteen individual prepositions—namely, those that occur at least 50 000 times in the corpus and are not part of any group: *about, against, as, at, by, for, from, in, of, on, over, to,* and *with.*

Schulte im Walde (1998, 2000) encodes the selectional preferences of a verb for a particular syntactic slot using a cut in the WordNet noun hierarchy. Simplifying the approach of Resnik (1997, 1998), she summarizes the head nouns occurring in a particular slot over the eleven root synsets of the WordNet 1.5 noun hierarchy, as well as the thirteen children of `entity%1`,[5] since the latter was considered too generic to be meaningful.

In her work on automatically determining whether a verb participates in a given alternation (mentioned in §2.8), McCarthy (2000, 2001) uses a similar definition of selectional preferences. She investigated using different tree-cut models to encode selectional preference. The first tree-cut model she considered was based on Li and Abe (1995), who proposed to automatically select the optimal model according to the minimum description length (MDL) principle. The second model she considered was the root cut, where only the eleven root nodes of the WordNet 1.5 noun hierarchy were used. In her experiments, McCarthy found that the choice of cut did not make a significant difference for her task. We therefore conclude that it is not worth investigating the more complex MDL-based tree-cut models for our work.

To encode our syntactic collocation features, we use the simple cut proposed by Schulte im Walde (1998, 2000), and we use WordNet 1.7, the latest version available at the time we designed this feature space.[6]

We summarize a set of nouns using a WordNet cut as follows: we find each noun in the WordNet 1.7 hierarchy, follow the hypernym links until we get to a synset in the cut, and add the occurrence count for the noun to that synset. In this process, however, two issues come up: ambiguity and multiple hypernyms. When a noun has several senses (*i.e.*, it appears in several synsets) we divide the occurrence count evenly among all the senses, following Schulte im Walde (1998, 2000) and McCarthy (2000, 2001). Sometimes a synset has more than ancestor in our cut, since the WordNet noun hierarchy is a directed acyclic graph, and not a tree or a forest. For example, `person%1` has hypernyms `organism%1` and `causal_agent%1`, which are both in our cut since they are hyponyms of `entity%1`. Thus, all hyponyms of `person%1` have at least two ancestors in our cut. In such cases, we follow Schulte im Walde (1998, 2000) and further divide the fractional count evenly between each hypernym of any synset with more than one

[5] We refer to WordNet synsets using the first word in the synset together with its sense number. For example, the synset composed of *organism* (sense 1), *being* (sense 2) and *living thing* (sense 1) is referred to as `organism%1`.

[6] WordNet 1.7.1 has since been released, and some clean-up was performed in the hierarchy, including in the direct hyponyms of `entity%1`. This new hierarchy seems preferable to the one we used, but the update was released too late for use in this study. Another change we could have done to the cut is to remove the hyponyms of `entity%1` which have very small subtrees. Eleven hyponyms of `entity%1` have fewer than ten synsets each in their respective subtrees, and one has 29. These subtrees would have been best summarized with `entity%1` rather than treated separately. However, this should not have any adverse effects on our experiments, because the important semantic categories are still in the cut we use. The categories that should have been removed will add a very small amount of noise, since their values will be null most of the time. Their presence does not affect the values of the relevant semantic categories.

hypernym. In contrast, McCarthy (2000, 2001) assigns the full fractional count to each element of the cut that can be reached from the synset. Her approach has the disadvantage that the total weight assigned for one occurrence of a word may exceed 1. Finally we divide the total count accumulated by each cut element by the number of noun occurrences in the slot, so that we get proportions for each cut element. This gives us 30 numerical features approximating the selectional preference of a verb for each slot. Since we calculate these features for 42 slots, we get a total of 1260 features in this category.

Chapter 3

Feature Extraction

In this chapter, we describe how we estimate the features defined in the previous chapter using a large corpus of English. We describe the corpus we use, each of the processing steps required, and how we calculate the features.

3.1 The British National Corpus

In this study, we use the British National Corpus (BNC) to estimate all features. The BNC is a corpus of about 100 million words, tagged with the CLAWS POS tagset. It consists of small text samples ranging over a wide spectrum of domains, including 10% from spoken sources. It covers British English only, including imaginative texts from 1960 and informative texts from 1975. Refer to (BNC, 2000) for more information about the corpus.

Unlike the Wall Street Journal corpus (WSJ), which was used in previous related studies (Merlo and Stevenson, 2001), the BNC is a general corpus. In the WSJ, we can expect the financial sense of a polysemous verb to dominate, whereas in the BNC, we do not expect any strong domain-specific bias. We have therefore decided to use the BNC instead of the WSJ, so that we can obtain estimates of the features we measure for each verb that are more representative of the general use of English.

3.2 Data Processing Overview

Since our features involve argument slots of the verb, we use an existing tool to extract some of them: the partial (or chunking) parser of Abney (1991, 1997). This program, called SCOL, allows us to extract subjects and direct objects with reasonable confidence. It also allows us to identify potential prepositional phrases associated with the verb, and (with some post-processing) indirect objects, although with lower confidence.

We perform the following data processing on the corpus:

1. Conversion of the BNC POS tags to the Penn tagset (needed for SCOL).
2. Extraction of the relevant sentences from the corpus.
3. Partial parsing with `cass`, a component of SCOL.
4. Tuple identification with `tuples`, a second component of SCOL.
5. Indirect object identification using TGrep2.

3.3 Corpus Preprocessing

Since the BNC is already POS tagged, we need only convert its tagset (CLAWS) to the PennTreebank tagset, as required by SCOL. SCOL provides a tag conversion utility for the BNC, but it maps the CLAWS tag VVB to the Penn tag VB, which is not the mapping we use (see §2.1), so we modified it to map VVB to VBP instead.

A further issue with POS tags in the BNC is that when a tag is uncertain, the BNC uses an "ambiguous" tag indicating the two most likely tags, with the single most likely tag listed first. For example, the ambiguous tag VVB–NN1 means that the word is most likely a verb in the present tense, but could be a singular noun. For simplicity, we always replace ambiguous tags by the most likely tag and ignore the ambiguity. This conversion is also done by the utility provided with SCOL.

We then extract all the sentences which contain one of our target verbs used as a verb. We do so simply by using `grep` to search for lines identified in the corpus as sentences that contain any of the inflected forms of the verb tagged with any of the possible verb POS tags.

The inflected forms of the verb are determined with the help of the list of irregular verb forms provided with WordNet 1.7 for the purpose of lemmatization. We retain each form mentioned in that list, and we add regular *-ing*, *-s* and *-ed* forms if no corresponding irregular forms were found. In the few cases where this method fails to provide the correct results, we use hard-coded lists. (Of all the verbs listed by Levin, only nineteen require such hard-coded lists.)

3.4 Partial Parsing with SCOL

SCOL is a partial parser, or chunker, provided by Steven Abney for public use (Abney, 1991, 1997). It takes as input English text tagged using the Penn tagset, performs shallow syntactic analysis on it, and provides the results in two output formats, each supplying different information about the analysis. The first format, the output of the `cass` program, is in the form of

Slot	Description
`subj`	Subject (active frames only)
`obj`	Active frames: the first object after the verb; Passive frames: the surface subject
`obj2`	Passive frames only: surface object
prep	Object of the preposition *prep* (for any preposition)
`pp-comp`	Object of the preposition *before, after, until,* or *since*
`cdqlx`	Object of the preposition *about, only, above, under,* or *over*
`rb, rx`	Adverb and adverb phrase

Table 3.1: Slots identified by `tuples` that are of interest to us

partial parse trees which can be analyzed using tree searching or manipulation techniques. The second format, the output of the `tuples` program, is in the form of slot/value pairs, with one tuple being produced for each clause. Each tuple includes the head verb of the clause, along with the head word found in any syntactic slot identified as belonging with the verb. We are interested primarily in the slots which correspond to arguments of the verbs (see Table 3.1). Other slots cover, for example, adjective phrases, date phrases, and incomplete chunks (groups of words that could not be labelled as phrases, usually because the sentence is too complex to be analyzed correctly), which we simply ignore. Since SCOL is a partial parser, it does not attempt to identify embedded clauses except the simplest ones, namely, those that occur between the subject and the verb. Furthermore, `tuples` treats all clauses identically, embedded or not.

A few slots require further explanations:

`pp-comp:` Because the four prepositions *before, after, until,* and *since* allow particular constructs that other prepositions do not, SCOL treats them as a special group and labels them all `p-comp`. As a result, `tuples` uses `pp-comp` as the slot name for objects of those prepositions, instead of the prepositions themselves.

`cdqlx:` This label is intended to capture pre-numeral chunks, which include *about, only,* and *over*, and should not be associated with the verb, but rather with numeral expressions. However, `tuples` also uses this slot name for objects of those prepositions, and thus we also find `cdqlx` associated with the verb.

`obj, obj2:` The label `obj` is always intended to capture the underlying direct object. Therefore, this label is given to the surface object in an active frame and to the surface subject in a passive frame.

SCOL will recognize a noun phrase following the verb as a surface object, even in passive

frames. Since the label `obj` is already assigned to the surface subject, the surface object in a passive frame is given the label `obj2`. However, when a surface object is present in a passive frame, as in *John was given a book*, the surface object is the direct object, and the surface subject is the indirect object. In this case, the label `obj` assigned incorrectly by SCOL.

Furthermore, SCOL does not try to identify double object frames—it recognizes only one object after the verb. When an indirect object is present in an active frame, as in *I gave John a book*, the indirect object is incorrectly labelled with `obj` and the direct object is ignored. To identify double object frames, and indirect objects, we apply some simple heuristics to SCOL's partial parse trees (see §3.5).

3.4.1 Processing of the SCOL Output

We use the sample sentences in Table 3.2 to illustrate the output produced by SCOL. Note that these sentences were created to illustrate how SCOL works, and not to be clear or good English.

The output of `tuples`

The `tuples` output for this set of sample sentences is presented in Table 3.3. Each line, or tuple, corresponds to a clause in the input text. Each tuple starts with the index of the first token of the clause, followed by the head of the clause.[1] The head of the clause is then followed by an unordered list of all the slots SCOL finds accompanying the verb phrase. For each slot, the name of the slot is followed by the head of the phrase filling the slot, as identified by SCOL. The head can be a leaf node in the partial parse tree (*i.e.*, a word) or an internal node, in which case the type of the internal node is given, prefixed by "`%`". For example, `%person` fills several

[1] Normally the head of the clause is the head verb, but if the verb is copular and the predicate complement is an adjective, this adjective is reported as the head of the clause. For example, *tall* is reported as the head of *John is tall*.

John is tall.
I gave the man a cat.
A chair was built by John for his mother.
Andrew, the new chair, was elected by acclamation and also accepted after some thought over important issues the position of chair of the subcommittee on equity as the committee recessed.

Table 3.2: Sample sentences used to illustrate our corpus processing.

```
0   tall  :subj %person
4   gave  :obj man  :subj I
11  built  :by %person  :for mother  :obj chair
21  elected  :by acclamation  :obj %person
32  accepted  :pp-comp thought  :cdqlx issues  :obj position  :on equity
 :as committee
52  recessed
```

Table 3.3: Output of `tuples` on the sample sentences.

slots in the sample output above, since `person` is the internal node used to represent proper noun phrases which SCOL's named-entity recognizer classifies as a person.

Since SCOL is a partial parser, the identification of clauses and clause boundaries is only approximate. In a complex sentence, SCOL may fail to identify the correct arguments of a verb phrase. Some constituents might be attached to the wrong phrase, or not be attached at all, in which case `tuples` would ignore them. Furthermore, dependent clauses are not identified as such, but treated as independent clauses.

We consider all these errors as noise in our data, and do not attempt to correct any of them, except for the lack of indirect object identification (see §3.5).

The output of `cass`

The output of `tuples` is ideal for counting occurrences of particular slots, but it does not provide all the information we need for extracting our features. In particular, we need the POS tag of the target verb, the whole verb phrase, and the POS tag of the head of some of the slots. This information is all available to the `tuples` program, but no means are provided by the software to include it in the output. However, we can extract it by matching the `tuples` output with the `cass` output, which is presented in the form of a partial parse tree. Table 3.4 shows the output of `cass` on the sample sentences above.

Merging the two output formats

Using the `cass` output to extract the additional information we need, we augment each tuple by adding the POS tag to the verb and each slot filler, as well as the whole verb group so that we can extract auxiliaries and modals. We also rename the slots `cdqlx` and `pp-comp` to the actual preposition appearing in the text, so that we can later deal with these prepositions individually. We call the result the intermediate tuples, since we augment them further in the next section. The intermediate tuples for the sample sentences are shown in Table 3.5.

```
   <s>
00 [c
00 h=[c0
00   s=[nx
00     h=[person
00         [fn John]]]
01   h=[vx
01       [bez is]
02     h=[jj tall]]]]
03 [per .]
   </s>
   <s>
04 [c
04 h=[c0
04   s=[nx
04     h=[prp I]]
05   h=[vx
05     h=[vbd gave]]]
06 o=[nx
06     [dt the]
07   h=[nn man]]]
08 [nx
08   [dt-a a]
09 h=[nn cat]]
10 [per .]
   </s>
   <s>
11 [c
11 h=[c0
11   s=[nx
11       [dt-a A]
12     h=[nn chair]]
13   h=[vx
13     f=[bedz was]
14     h=[vbn built]]]
15   [pp
15   f=[by by]
16   h=[nx
16     h=[person
16         [fn John]]]]
17   [pp
17   f=[in for]
18   h=[nx
18       [prps his]
19     h=[nn mother]]]]
20 [per .]
   </s>
   <s>
21 [c
21 h=[c0
21   s=[nx
21     h=[person
21         [fn Andrew]]]
22     [cma ,]
23     [nx
23       [dt the]
24       [jj new]
25     h=[nn chair]]
26     [cma ,]
27   h=[vx
27     f=[bedz was]
28     h=[vbn elected]]]
29   [pp
29   f=[by by]
30   h=[nx
30     h=[nn acclamation]]]]
31 [cc and]
32 [vnp
32 h=[vnx
32     [rb also]
33   h=[vbn accepted]]
34   [pp-comp
34     [p-comp after]
35   h=[nx
35       [dtp-q some]
36     h=[nn thought]]]
37   [pp
37   f=[cdqlx
37       [over over]]
38   h=[nx
38       [jj important]
39     h=[nns issues]]]
40 o=[ng
40   h=[nx
40       [dt the]
41     h=[nn position]]
42     [of of]
43   k=[nx
43     h=[nn chair]]
44     [of of]
45     [nx
45       [dt the]
46     h=[nn subcommittee]]]
47   [pp
47   f=[in on]
48   h=[nx
48     h=[nn equity]]]
49   [pp
49   f=[as as]
50   h=[nx
50       [dt the]
51     h=[nn committee]]]]
52 [vnp
52 h=[vnx
52   h=[vbn recessed]]]
53 [per .]
   </s>
```

Table 3.4: Output of `cass` on the sample sentences, reformatted in three columns. Role legend: `h`: head, `s`: subject, `o`: object, `f`: function word, `k`: part.

```
0   tall/jj  :subj %person/person  [vx [bez is] [jj tall]]
4   gave/vbd  :obj man/nn  :subj I/prp  [vx [vbd gave]]
11  built/vbn  :by %person/person  :for mother/nn  :obj chair/nn  [vx [be
dz was] [vbn built]]
21  elected/vbn  :by acclamation/nn  :obj %person/person  [vx [bedz was]
[vbn elected]]
32  accepted/vbn  :after thought/nn  :over issues/nns  :obj position/nn
:on equity/nn  :as committee/nn  [vnx [rb also] [vbn accepted]]
52  recessed/vbn  [vnx [vbn recessed]]
```

Table 3.5: Intermediate tuples, with information from `tuples` and `cass` merged

3.5 Indirect Object Identification with TGrep2

Before we describe our final step in augmenting tuples, we need to discuss direct and indirect object identification in more detail.

We use SCOL to extract the subject, the direct object, and the object of prepositions, but SCOL does not attempt to identify indirect objects and it has poor recall when identifying direct objects.[2] Since we need to identify both indirect and direct objects for our work, we attempted to go beyond what SCOL provides.

By inspection, we observed that when a double object frame is processed by `cass`, the first object is attached to the verb phrase and labelled as an object, but the second object is left unattached and immediately follows the core of the clause in the tree structure. (The core of the clause is the subtree containing the verb group and, when they are identified as belonging with the verb, the subject, the first object and any adverbs.) We therefore make the following assumptions:

Assumption 1. When a clause core contains an object and is followed immediately by an unattached noun phrase, the unattached noun phrase is the second object.

Assumption 2. When a clause core does not contain an object, but is followed immediately by an unattached noun phrase, the unattached noun phrase is an object.

[2] To get an idea of SCOL's accuracy, we evaluated how well SCOL identifies direct objects. We found precision in the high eighties but recall was only around one half. To estimate precision, we extracted 100 random clauses where SCOL identified a direct object, and manually found 87 of them to be correct (95% confidence interval: [78.8%–92.9%], calculated using the binomial exact method of Newcombe, 1998 and Uitenbroek, 1997a). To estimate recall, we extracted 100 random clauses and manually identified those with a direct object. This gave us 61 clauses. SCOL correctly identified the direct object of 30 (49%) of those (95% confidence interval: [36.1%–62.3%]). This high precision and low recall reflects the *easy-first parsing* philosophy embedded in partial parsing—attachments that can be made easily or with high confidence are made first, whereas complex or ambiguous structures are sometimes left as unlinked chunks (Abney, 1991).

We know that these assumptions will often be incorrect, but we hope that they will provide more signal than noise, and that the noise will be sufficiently randomly distributed that the statistics we gather for indirect objects, based on these assumptions, will still be helpful.

We evaluated the precision of our two assumptions as follows. We randomly selected 100 frames which were identified as double-object frames using Assumption 1. We manually inspected these and found that only 22 of them were indeed double-object frames, with the two objects correctly identified (95% confidence interval: [14.3%–31.4%], calculated using the binomial exact method of Newcombe, 1998 and Uitenbroek, 1997a). We also randomly selected 100 frames which were not identified as having an object by SCOL, but which were identified as one-object frames using Assumption 2. We manually inspected these and found that 55 of them were indeed one-object frames, with the direct object correctly identified (95% confidence interval: [45.7%–65.9%]).

This evaluation gives us estimates of precision, although it does not tell us anything about accuracy or recall. It would be useful to assess the recall achieved by these assumptions. However, given the low frequency of indirect objects in the corpus, obtaining enough random samples to make this assessment would require the manual annotation of an unreasonably large number of sentences.

Given the low precision of Assumption 1, we might consider discarding it. However, at this point it is the only method that is simple and readily available to identify indirect objects. We therefore accept that features involving indirect objects will be noisy ones, and hope that the machine-learning system can still extract useful information from them. If the noise is sufficiently evenly distributed, these features can be useful, at least in theory. We review this question in §6.5.

Assumption 2 does not affect the identification of indirect objects, but only direct objects. When we designed this method of identifying indirect objects, we noticed that SCOL misses a number of direct objects that would be easy to identify. With the machinery in place to identify indirect objects, it seemed most sensible to improve the direct object identification where it was easy to do so, hence Assumption 2. However, Assumption 2's precision is significantly lower than SCOL's, so it is unclear whether it is preferable to use it or not. For that reason, we keep two versions of features using the direct object, one using neither assumption above, the other using both, and let the machine-learning system pick the most useful. We also review this question in §6.5.

3.6 Final Augmented Tuples

We extract double object frames and attempt to increase direct object recall using the assumptions described in the previous section. To implement these assumptions, we process the partial parse trees output by `cass` using TGrep2 version 1.06, a publicly available tree-searching utility (Rohde, 2002).

Using these assumptions, we assign the slot labels `dobj` and `iobj`. In active frames, if one object is found, it is labelled `dobj`; if two objects are found, the first is labelled `iobj` and the second, `dobj`. In passive frames, if there is no object, the subject is labelled `dobj`; if there is one object, the subject is labelled `iobj` and the object, `dobj`. Occasionally, two objects are found in passive frames. Since such a frame is not possible in English, we assume this is an error; we ignore the second object and treat the frame as if only one object had been found.

We identify passives by the use of the past participle and the auxiliary *to be* in the same verb group. Note that in the case of passives, we make no attempt at recovering the underlying subject. We could look for the preposition *by*, but it is ambiguous whether it is an underlying subject, as in *The chair was built by John* or an actual object of the preposition *by*, as in *John was elected by acclamation*. We therefore prefer the more conservative solution of always treating the preposition *by* as distinct from the subject.

Finally, we add the `dobj` and `iobj` slots to our intermediate tuples to obtain augmented tuples. In the augmented tuples, we preserve the `obj` slot, even though it is superseded by `dobj`, and we preserve the `obj2` slot, even though it has little meaning. Occasionally, `tuples` fails to identify a passive frame as such (specifically, with long verb groups like *would have been picked*). In those cases, we change `obj` to `obj2` and `subj` to `obj`, the transformation `tuples` applies to passives it recognizes. The augmented tuples for the sample sentences are shown in Table 3.6.

3.7 Feature Calculations

Using the augmented tuples described above, it is fairly straightforward to calculate most of the features presented in Chapter 2 (summarized in Table 3.7). In most cases, the relevant frequency is divided by the overall frequency of the target verb, but there are exceptions, so we explain the calculation of each feature category individually here. For each target verb, we first extract all verb groups from the `cass` output which contain any form of the target verb with a verb POS tag.

The relative frequencies of the parts of speech of the verb (§2.1) are calculated by tabulating the occurrences of the inflected forms of the target verb with each POS tag in the extracted

```
0   tall/jj  :subj %person/person  [vx [bez is] [jj tall]]
4   gave/vbd  :obj man/nn  :subj I/prp  :dobj cat/nn  :iobj man/nn  [vx [
vbd gave]]
11  built/vbn  :by %person/person  :for mother/nn  :obj chair/nn  :dobj c
hair/nn  [vx [bedz was] [vbn built]]
21  elected/vbn  :by acclamation/nn  :obj %person/person  :dobj %person/p
erson  [vx [bedz was] [vbn elected]]
32  accepted/vbn  :after thought/nn  :over issues/nns  :obj position/nn
:on equity/nn  :as committee/nn  :dobj position/nn  [vnx [rb also] [vbn a
ccepted]]
52  recessed/vbn  [vnx [vbn recessed]]
```

Table 3.6: Augmented tuples, with merged information from `tuples`, `cass` and TGrep2, *i.e.*, the intermediate tuples with the slots `dobj` and `iobj` added.

verb groups, and dividing by the total number of occurrences of the target verb with a verb POS tag.

The relative frequencies of modals and auxiliaries (§2.2) are calculated by tabulating the occurrences *to be* and *to have* (and their inflected forms) and each of the eight modals in the extracted verb groups, and dividing by the total number of occurrences of the verb. The feature giving the proportion of verb groups which include any of the eight modals is also calculated using these extracted verb groups.

The relative frequency of the passive voice (§2.3) is calculated by counting the number of extracted verb groups where two conditions are met: the verb group contains the auxiliary *to be* in some form, and the head verb has the POS tag VBN. We divide in one case by the sum of the occurrence counts of VBN and VBD as calculated above, following Merlo and Stevenson (2001), and in the other by the total number of occurrences of the verb.

For the relative frequencies of the derived forms (§2.4), we count how often the verb occurs as an adjective or a noun in the original corpus. To avoid having to process the entire corpus multiple times, we first build a list of all tagged tokens appearing in the corpus with their frequency counts, and we extract from this list the counts we need for each verb. The application of the formulas given in §2.4 is then straightforward.

To produce the remaining sets of features, we first extract, out of our augmented tuples, all the tuples for which the verb of interest is the head verb. Then for each slot, we create a list of all the words that occur in that slot across all tuples, with frequency counts. Additionally, we create the list of words occurring in the subject slot in transitive clauses (*i.e.*, those whose tuple contains a direct object) and the list of words occurring in the subject slot in intransitive

Feature Category	Defined in Section
Part of speech of the verb	2.1
Frequency of auxiliaries and modals	2.2
Combined feature: passive	2.3
Relative frequency of derived forms	2.4
Semantically empty constituents	2.5
Frequency of various syntactic slots	2.6
Animacy of NPs in various slots	2.7
Measure of slot overlap	2.8
Syntactic collocation features	2.9

Table 3.7: Summary of the feature categories

clauses (*i.e.*, those whose tuple does not contains a direct object).

Note that we have two ways to identify the direct object, the first being those direct objects identified by `tuples`, the second adding to those the ones we find with TGrep2 as part of our indirect object identification process, and removing those that are identified as indirect objects. Consequently, we keep two distinct lists for each of direct object, subject in a transitive frame and subject in an intransitive frame. Each feature using these lists has two versions—we keep both and let the machine-learning system determine which one is more useful. (We review this question in §6.5.)

The relative frequencies of semantically empty components (§2.5) are calculated using the list of words occurring in the subject and direct object slots for the impersonal *it*, and the subject and subject in a transitive frame slots for *there*-insertion. We tally how often *it* or *there* occurs in the relevant slot, and divide by the total number of occurrences of the verb. In the case of the relative frequency of *there*-insertion with respect to transitive frames, we divide by the number of tuples which have a direct object instead.

The relative frequencies of syntactic slots (§2.6) are calculated by dividing the sum of the frequencies from the list of words occurring in the slot by the total number of occurrences of the verb. This includes the subject, subject in a transitive frame, subject in an intransitive frame, direct object, indirect object, and the object of individual prepositions.

The relative frequencies of preposition groups (§2.6) are calculated by adding the relative frequencies of each preposition in the group. This will be an overestimate when two prepositions from the same group occur in the same clause, but the resulting error is sufficiently small that we can ignore it.

The relative frequency of the occurrence of the verb with any preposition (§2.6) is calculated in two ways. In the first way, we simply add the relative frequencies of all prepositions. However, this incorrectly counts a clause with multiple prepositional phrases multiple times. This number is therefore not really a relative frequency, since it could be greater than 1. In the second way, we count the number of tuples in which any preposition occurs, and divide by the total number of occurrences of the verb. Since we are treating a large number of prepositions, these two numbers could be significantly different. We calculate and keep them both, as they convey potentially different information about the verb.

All the animacy features (§2.7) are calculated using the lists of words occurring in the relevant slots. For each slot or group of prepositions, we go through the list of words and tally those that are deemed to refer to an animate being, and divide by the sum of all word frequencies in the list. In §2.7, we list the pronouns we deem to refer to animate beings, and mention that we also count proper noun phrases that can be identified as the name of a person. We rely on SCOL's crude named entity recognizer to do this. If a proper noun phrase starts with one of the five titles *Mr., Ms., Mrs., Miss.,* or *Dr.*, or if it contains a known first name (from a list of 178 English first names), SCOL classifies it as a person's name, `tuples` labels it `%person`, and we count it as animate.

To calculate the slot overlap features (§2.8), we again use the list of words occurring in each slot. We lemmatize each word, treating `%person` as a special lemma. We then apply the slot overlap formula given in §2.8.

The syntactic collocation features (§2.9) are calculated using the same lists of words and following the procedure described in §2.9.

Chapter 4

Machine-Learning Approaches

The purpose of this study is to develop a general feature space for verb classification and test it on a number of sample classification tasks. Specifically, we focus on two questions. First, do our features work well, and second, can we identify which features are most useful or relevant?

We use the decision-tree-induction system C5.0 (`http://www.rulequest.com`), the successor of C4.5 (Quinlan, 1993), as our machine-learning software. We made this choice because it is the system that was used in studies upon which our work is built (Merlo and Stevenson, 2001) and because it performs well for our purposes. In previous work, different machine-learning paradigms were compared, and they performed similarly (Stevenson *et al.*, 1999). Without repeating this comparison, we consider additional approaches to solve the feature-selection problem.

The feature space we defined is a relatively large one, given the small size of our training data—in our experiments, we have over 200 features for 40 to 60 training samples. Our feature space includes many partially redundant features, and for any particular verb classification task, we can expect a number of the features to be irrelevant. C5.0 internally selects the features it identifies as most useful when building decision trees. In our experience, however, it does not always use the best features available when faced with numerous redundant and irrelevant ones (*e.g.*, in Chapter 6, we report a number of cases where a set of features has lower accuracy than a subset of the same features). We therefore assume that we cannot rely solely on the tree-induction mechanism to select the most relevant features for us. We consider methods to filter out irrelevant features and/or to select the most useful features for a particular classification task. These methods should improve the selection of relevant features and, by reducing the number of features used, make the interpretation of classifiers built by the machine-learning system, to the extent that such interpretation is meaningful, easier.

In this chapter, we describe the two approaches we explored for dealing with our large feature space. Section 4.1 describes our first, unsuccessful approach, namely to filter out features that

clearly should not be relevant. Specifically, we remove features that we expect not to be useful, based solely on the values they take, and without regard to the classification of the training data. Section 4.2 describes stepwise feature selection, our second and more successful approach. With this approach, we try to identify useful features based on their performance on the classification task. Finally, we survey some relevant literature in §4.3.

4.1 Feature Filters

Knowing that C5.0 is not always successful at selecting the best features when faced with large numbers of irrelevant and redundant features, as found in our feature space, we first applied filters to remove features that should clearly not be useful. The justification for this approach is that when faced with a large number of features, but restricted training data, it is preferable to first exploit any information available for removing features other than the classification of the training data. This will reduce the number of statistical tests, or other types of feature comparisons based on the training data, and thereby reduce the chances of overfitting, *i.e.*, the chances of finding seemingly good features that are not actually relevant.

This approach did not prove useful. The main problem is that it completely ignores the classification of the training data when making decisions about features to remove. We therefore discarded it in favour of stepwise feature selection (see §4.2), but we describe it here for completeness.

4.1.1 Highly Correlated Features

When two useful features are highly correlated (positively or negatively), we expect that either one of them will be sufficient to perform the classification. If both were kept, the second one would provide only redundant information and not only would it not help the classification, but it could make the decision trees built by the tree-induction system more complex and harder to interpret. Therefore, when we find two features having a correlation coefficient r higher than a fixed threshold in absolute value, we discard one of the two as follows. Some features are believed to be better than others, because of how they are extracted or what information they encode. If, of the two features, one is believed to be better, the other one is discarded; otherwise, one of the two is chosen at random and discarded.

We experimented with the threshold set at $r \geq 0.8$, $r \geq 0.9$, and with the correlated-feature filter disabled. The results were disappointing, as filtering features seldom improved predictive performance and frequently degraded it, as estimated using tenfold cross-validation repeated 50 times.

One of the problems lies in the use of the correlation coefficient. Some of our features have

null values for many verbs (though not enough to be filtered by the mostly-null-feature filter described below), and in some cases, two features might be zero for the same verbs but yet not be otherwise correlated. The weight of all these zero values can raise the correlation coefficient enough to filter out one of the features, even if they provide complementary information about the verbs when they are not null. For example, `%of` is the most important feature in the *Clear* Alternation task (described in Chapter 5). However, it is highly correlated with `over.obj.%of`, `%of_person` and `over.subjtrans.of`, since if `%of` is null, then the others must also be null. In our experiments, `%of` was sometimes removed because of this correlation, and we observed a dramatic increase in mean cross-validation error in those cases. We could have looked for a better measure of feature relatedness, but this approach did not seem useful or promising. Consequently, we did not investigate it further.

4.1.2 Mostly Null Features

Features for which 90% or more of the verbs had a null value were considered to be meaningless, and were removed from the feature set before passing it to C5.0. We compared performance with and without this filter, and we found again that it made no appreciable difference in classification performance, except in one case where the filter slightly hurt performance. We therefore discarded this filter as well. (Although as a speed optimization, we kept it with a 99% threshold.)

4.2 Stepwise Feature Selection

Our second approach to automatically selecting the most relevant features, stepwise feature selection (§4.2.3), was more successful. It is a greedy algorithm combining forward selection (§4.2.1) and backward elimination (§4.2.2) of possible features to include in the final classifier. This approach requires a method to evaluate the "goodness" of feature subsets—we use accuracy as estimated by repeated tenfold cross-validation on the training data (§4.2.4).

4.2.1 Forward Selection

Forward selection is a greedy feature selection approach which starts with the empty set, and at each step selects the feature which most improves estimated accuracy when added to the already-selected features. If the best accuracy improvement does not meet a fixed "add" criterion, the feature is not added and the algorithm stops (Olden and Jackson, 2000; Sokal and Rohlf, 1981; John *et al.*, 1994).

In the typical statistical approach to forward selection for regression, the add criterion is a test of statistical significance of the reduction in estimated error (Olden and Jackson, 2000; Sokal

and Rohlf, 1981). However, it is not clear that the significance test applies for classification tasks, especially since we use repeated cross-validation to estimate accuracy. How best to test for the statistical significance of a difference between cross-validation results is debated, as we discuss in §4.3.6. Furthermore, since we consider hundreds of different features at each forward selection step, the probability of finding statistical significance by chance when it does not really exist would be very high (Salzberg, 1997). Thus, even if we used a valid test of the significance of the difference between two cross-validation results, it would not be valid when used in forward selection.

Since the statistical approach to forward selection is very involved, and does not clearly apply to classification tasks, we use a simpler approach. We set the add criterion to a fixed minimum accuracy increment, 0.5%, and we ensure that the standard error of the mean accuracy is 0.5% or less (see §4.2.4). This may seem like an excessively relaxed add criterion, but Kohavi and John (1997) relax this even further by accepting any improvement at all, giving a penalty of only 0.1% per feature to the larger feature sets, in order to prefer smaller sets in the case of a tie. Our approach gives a bit more weight to parsimony, under the theory that smaller feature sets are better if they explain the data almost as well as larger ones. In future work, we might want to examine the effect of the add criterion more carefully.

4.2.2 Backward Elimination

Backward elimination is also a greedy feature selection approach, but this time we start from the full set of features. At each step we remove the feature such that, when it is removed, accuracy improves most (or degrades least)—we will call this feature the "worst feature". If the feature does not meet a fixed "remove" criterion, we leave it in and the algorithm stops (Olden and Jackson, 2000; Sokal and Rohlf, 1981; John *et al.*, 1994).

In the typical statistical approach to backward elimination for regression, the remove criterion is also a test of statistical significance, applied as follows: if removing the worst feature degrades performance in a statistically significant way, the feature is not removed and the algorithm stops (Olden and Jackson, 2000; Sokal and Rohlf, 1981). Just as with forward selection, slightly lower accuracies are tolerated for the sake of parsimony.

Note that for regression, such a remove criterion must be used, because removing a feature is guaranteed either to leave the estimated error untouched or to make it worse, but never to improve it. The reason is that the fit of a feature set is assessed against the training data only, as seems to be normal practice in regression tasks, at least during the feature selection phase (Olden and Jackson, 2000; Sokal and Rohlf, 1981). (Olden and Jackson (2000) warn against using the training data for final validation, but do not seem to hesitate using it for feature selection.)

When using backward elimination for classification tasks, we do not necessarily have to allow any performance degradation when removing a feature, since removing a feature can improve performance, unlike with regression. However, it remains a good idea, for the sake of parsimony—we prefer smaller feature sets to larger ones, if they perform similarly.

For our work, we again use a simple remove criterion, similar to our add criterion. The worst feature is removed if removing it increases the accuracy estimate at all, or reduces it by at most 0.25%. This is in keeping with the standard statistical approach, where stricter requirements are applied before adding a feature, but we are more lenient in allowing features to stay in the current set. But it is also consistent with the desire to prefer parsimonious feature sets, since a feature might be removed even if removing it slightly reduces the accuracy estimate.

4.2.3 Stepwise Feature Selection

Stepwise feature selection combines forward selection and backward elimination. If all our features were independent, a feature that was deemed very useful would normally remain so even when more features were added. But when various correlations exist between features, it is possible for a feature to be informative on its own, but to be useless or even harmful when some particular combination of other features is also available. In order to deal with this problem, forward selection and backward elimination are combined together in an algorithm called stepwise feature selection (Olden and Jackson, 2000; Sokal and Rohlf, 1981; John *et al.*, 1994). This method can start from any subset of features, be it the empty set, the set of all available features, or some other subset believed to be a useful starting point. The algorithm then alternates between forward selection and backward elimination, alternatively trying to add and remove features, and stopping when both directions miss their respective criterion and cannot change the feature set any further (Olden and Jackson, 2000; Sokal and Rohlf, 1981; John *et al.*, 1994).

Note that stepwise feature selection is related to hill-climbing, in that it is a greedy algorithm, except that in hill-climbing, we would not alternate between adding and removing features. Instead, we would select at each step the addition or removal operation which would yield the best feature subset if applied to the current feature subset. Stepwise feature selection has the advantage of allowing the add and remove criteria to be set independently.

As mentioned in the previous two sections, in our study the add criterion is an increase of at least 0.5% in the estimated accuracy, and the remove criterion is any increase or a decrease of at most 0.25% in the estimated accuracy.[1] The choice was somewhat arbitrary, but seemed

[1]Another approach would be, using the empty set as the initial set, to make the add criterion more relaxed than the remove criterion—*e.g.*, 0.1% for the add criterion, but 0.5% for the remove criterion. This unusual setting would create loops in the search, so a loop detection mechanism would be required. However, it would

appropriate given that we force the standard error of the mean accuracy estimate down to 0.5% (see below). For the initial set, we used the empty set.

4.2.4 Accuracy Estimation

In all the algorithms described above, we need a method to estimate the accuracy of a subset of features. The estimator we use is repeated tenfold cross-validation using the subset of features to evaluate. This method can be used with any machine-learning system, but since we use C5.0 for our study, we have to use C5.0 when running the cross-validation. To reduce the variance of the estimate, we repeat the cross-validation 20 times, and then add repeats by groups of five until the standard error of the mean accuracy is 0.5% or less.

Kohavi and John (1997) report that they also use repeated cross-validation, having found that using a single cross-validation run gave insufficiently predictable results because of the high variance in the estimate. They argue that the number of repeats must be set keeping in mind what they call the exploration versus exploitation problem, which they attribute to (Kaelbling, 1993). A higher number of repeats favours exploitation by reducing the confidence interval on the accuracy estimates. At the cost of more CPU time, the right feature subset is more likely to be picked. Fewer repeats favour exploration, since evaluating each feature subset takes less CPU time, and more states can be explored in the search space. At the cost of rougher accuracy estimates, good feature subsets further away in the search space can be found sooner. Now, with stepwise feature selection, we explore far fewer states that Kohavi and John (1997) do with their search heuristic, so we can afford to exploit the data more than they do.[2] We force all our standard errors to be below 0.5%, using 20 or more repeats, whereas they run only up to five cross-validation repeats, stopping earlier if the standard error is below 1%. For some of our larger experiments, we might consider allowing a higher standard error, because the largest one took several days to run, which is not very practical.

provide an interesting advantage when two correlated features carry useful information for the classification task. If the first was more useful than the second when considered alone, but less useful when other features were available, it would be added first, even if it was not the best one to keep in the end. This setting would give the second feature a better chance to replace the first one later in the search. In situations without such pairs of features, this approach yields results similar to the 0.5%/0.25% setting we use. The first feature to be added with an improvement smaller than 0.5% would be immediately removed by the next backward elimination step. This would create a loop and terminate the search, with the smaller set being retained. Experimenting with this approach, however, is left to future work.

[2] John *et al.* (1994) use forward selection and backward elimination. Kohavi and John (1997) differ somewhat from John *et al.* (1994), in that they use best-first search instead, and they stop only when the last five search node expansions produced no improvements. At the end, they simply retain the best feature subset that was encountered throughout the search, giving a 0.1%-per-feature penalty to each subset, basically to give the advantage to the smaller subset in the case of a tie.

4.3 Survey of Selected Literature

The problem of feature selection frequently comes up in studies that involve prediction tasks, including classification tasks (which predict the class an item belongs to) and regression tasks (which predict the value of a given quantity for an item). This problem is usually called model selection in the statistical literature (*e.g.*, Olden and Jackson, 2000; Sokal and Rohlf, 1981), but we follow the machine-learning literature and refer to it as feature selection, for consistency and to avoid confusion with other senses of "model".

In the present survey, we focus on work that is most relevant for our study. The reader interested in a broader survey of feature-selection literature is referred to Blum and Langley (1997). (Of all the methods reviewed there, the approach of John *et al.* (1994) is the most appropriate for the problem we are solving, and is discussed in §4.3.3.)

4.3.1 Feature Selection for Regression

Olden and Jackson (2000) survey the practices of feature selection for regression found in the ecological literature, and perform a study of the correctness of different selection methods. They clearly demonstrate that examining all subsets of features is inappropriate, even in cases where it is practical, as it tends to include many features which are not relevant, but which are correlated with the predicted variable by chance. In fact, examining all subsets of features results in the worst overfitting behaviour of all the feature selection approaches they survey.

Based on their experiments, they recommend two feature selection approaches. For small sample sizes (30 training examples in their study), they find that stepwise feature selection (and also its components forward selection and backward elimination used individually, see §4.2) performs best. For larger sample sizes (60 training examples in their study), they find that an approach called bootstrapping performs best for regression problems. Since it is not clear how to apply bootstrapping to classification tasks, we do not describe it here. With this sample size, however, they find that stepwise feature selection still performs well.

4.3.2 The Filter Model

John *et al.* (1994) study feature selection for classification tasks. One method they discuss is the feature-filter model, where determination of the relevant features is done before the machine-learning system is used. Each feature is individually evaluated for relevance, and those with the top relevance scores are used. Note that this model should not be confused with the feature filters we described in §4.1. There, features were discarded based on their hypothesized uselessness, without regard to the classification task, *i.e.*, without looking at the classification

of the training data. Here, features are selected based on some assessment of their individual relevance to the classification task.

As they mention in the paper, this model does not deal with redundant features very well. The first problem is that if two features are highly relevant, but also highly redundant, we would like to keep only one of them, unless having them both actually helps classification. But this cannot be determined without testing them together, so this model would keep them both. The second problem is that if a set of weakly relevant features are, together, highly informative, they might all get discarded anyway because they are not sufficiently relevant individually.

The use of any method which individually assesses features, such as using Mutual Information, or the Relief filter system (Kira and Rendell, 1992) will have the same problem. Since our feature set contains numerous redundant features, it is clear that the filter model is not appropriate for our classification tasks.

4.3.3 The Wrapper Model

John *et al.* (1994) also introduce what they call the wrapper model, which treats the machine-learning system as a black box, guiding feature selection by evaluating different feature subsets using cross-validation. Unlike with the filter model, feature values are not directly examined. With this model, various strategies can be used to determine which feature subsets to consider, from the brute-force all-subsets approach to more clever strategies which search for the best subset in a more focused way. Stepwise feature selection, our approach, is an example of a search strategy that can be used with the wrapper model.

The key to the wrapper model is that interactions between features are always taken into account, since feature sets are evaluated together using the machine-learning system itself, rather than individually. It is important to note here that the feature sets selected in this way are only appropriate for the machine-learning system used in selecting them, since the biases of the system itself are also embedded in the selection procedure. Different systems cannot always be expected to exploit features the same way, so the best feature set with one system might not be the best feature set with another.

Based on their study, John *et al.* (1994) find that the wrapper model, using forward selection in particular, performs well for classification tasks, just as Olden and Jackson (2000) found that it works well for regression tasks. John *et al.* (1994) show that forward selection enables C4.5 and ID3, the precursor to C4.5, to build smaller decision trees using fewer features than when they have access to all the features, but that generalization performance remained roughly the same. They conclude that the main advantage of this method lies in these smaller decision trees, which, to the extent that decision trees can be meaningfully analyzed, are easier to interpret than the large trees produced when all features are available to the tree-induction systems.

In later work, the wrapper model is generalized. The traditional statistical terminology of forward selection, backward elimination and stepwise feature selection is replaced by the more standard machine-learning terminology of a state space with operators to move through the space. The state space is the set of all possible feature subsets, and the goal is to search through this space for the best subset. Three kinds of operators are defined to move through the state space during the search. The two basic kinds of operators, adding a feature and removing a feature, correspond respectively to a step in forward selection and a step in backward elimination. The third kind, which they call compound operators, combines the best operators available at a given point into a single operator that adds and/or removes multiple features at once. Intuitively, if adding two features separately helps, it is likely (though not necessarily) true that adding them together will help even more, so it is worth exploring that hypothesis right away. At each step, they consider combining the best k operators so far into a compound operator c_k, starting with $k = 2$, and continue as long applying c_k to the current parent node being expanded improves the estimated performance. (Kohavi and John (1997) provide the most comprehensive report on this research; see also Kohavi and Sommerfield (1995), Kohavi and John (1998) and Kohavi (1995b, §4).)

The effect of compound operators is similar to that of the filter model, in that they take the features that were best individually, and consider them together. However, only the decision to consider a subset is based on the individual performance of the features. The decision to accept a subset is always based on the performance of the features when given together to the machine-learning system. Thus, interactions between features are taken into account, in contrast to the filter model. With this approach, if redundant features are accepted together but are not both required for optimal performance, we can expect that one of them will be removed by a backward elimination operator in a subsequent step. We can also expect that groups of weakly relevant features that are useful when used together will eventually be considered and retained.

Kohavi and John (1997) show that the use of compound operators, together with a search heuristic called best-first search, often improves generalization performance, although slightly larger feature subsets tend to be selected when compound operators are allowed than otherwise. Intuitively, compound operators allow the search to reach better states more rapidly, since features do not have to be removed or added one at a time, and more of the information available at each step of the search is exploited. compound operators increase the chances of overfitting, however, especially when only a small number of training examples are available.

In the present study, we restrict ourselves to adding or removing one feature at a time, as described in §4.2. However, we would like to explore the potential of compound operators for verb classification in future work. To avoid overfitting, we will have to find a more conservative

way to apply them, since we have a large number of features and a small number of training examples. One approach would be to give a more significant penalty to operators that add many features at once.

4.3.4 Overfitting and the Wrapper Model

When using the wrapper model, including when stepwise feature selection is used, the probability of overfitting increases as we explore more distinct subsets. The more feature subsets we test, the more likely we are to find one which is only good by chance, performing very well on the training data, but having poor predictive power (Kohavi and Sommerfield, 1995; Kohavi and John, 1997). For this reason, exploring all possible subsets is clearly inappropriate, as confirmed by Olden and Jackson (2000) and suggested by the work of Ng (1998).

Kohavi and Sommerfield (1995) study the issue of overfitting in stepwise feature selection. They found that overfitting was mostly an issue for small training sets, and therefore we will have to be concerned about overfitting in our work. They also showed that overfitting became increasingly worse as the search was allowed to expand more nodes. In the problematic cases they show, however, overfitting only became a serious problem when the search was allowed to continue expanding nodes even if a large number of previous node expansions had produced no improvements at all. Thus, we should not expect too much trouble with overfitting when performing stepwise feature selection, as long as we maintain a sufficiently conservative add criterion.

Finally, Kohavi and Sommerfield (1995) mention, as we know, that in order for accuracy estimates to be unbiased, they must be based on test data that was not used to perform the feature selection. Thus, the cross-validation results on the set of features selected cannot be considered meaningful.

Ng (1998) uses the wrapper model in two approaches to feature selection in cases where a large number of irrelevant features are present and only a small training sample is available, as in our case. His first approach is essentially that of Kohavi and John (1997), where the state space is searched using some heuristic and the feature subset selected in the end is the one that has the best estimated accuracy using cross-validation or a held-out set. Ng demonstrates that, because the same data is used repeatedly, the approach is limited by the severity of overfitting.

In his second approach, Ng proposes "ORDERED-FS", a theoretical algorithm that exhaustively examines feature subsets. The set of training samples S is subdivided into an active training set S' and a held-out set S''. For all possible subsets of features, performance on S' is measured (*e.g.*, using cross-validation). For each possible subset size (0–f if there are f features), the best-performing feature subset is retained (yielding $f + 1$ feature subsets). The retained feature sets are evaluated on S'' and the one with the smallest error is selected.

The key to ORDERED-FS is that, although all subsets are examined, only $f + 1$ subsets are tested on the hold-out set, so that the severity of overfitting is reduced. We describe this algorithm as theoretical, however, because it requires the exploration of all possible subsets, which rapidly becomes intractable with large numbers of features. Ng (1998) performs empirical experiments with an approximation of this algorithm, and finds that it almost always outperforms the standard wrapper approach. It is possible to adapt stepwise feature selection or the approach of Kohavi and John (1997) to make use of the underlying idea of ORDERED-FS. For example, stepwise feature selection would be run on the training data with a hold-out portion removed. The best feature subset identified at each step of the algorithm would be retained. At the end, we would evaluate the retained feature subsets against the held-out set and keep the best one. Note, however, that experimenting with this approach falls beyond the scope of this study and is proposed as an avenue of future work.

4.3.5 Gaussian Process Models

Another machine-learning approach we investigated is Gaussian process models (Neal, 1997), which use automatically determined relevance hyperparameters to reduce or increase the effect of individual features on classification, normally reflecting their actual relevance to the classification task. We tried this approach in a preliminary experiment, to determine if the relevance hyperparameters can help with the feature-selection problem. It performed similarly to C5.0 in terms of estimated accuracy, but it did not help with feature selection.

Even though Gaussian process models can deal with redundant features fairly well, they do not help in determining which features are the relevant ones, and which feature set might be the best. The key problem is that if two features are very helpful but redundant, their relevance hyperparameters will tend to be set in such a way that their combined relevance is high. However, one of them could have a high relevance hyperparameter and the other a low one, or both of them could be set somewhere in the middle. This approach is appropriate if the goal is to build a good classifier which uses all the available features. However, we cannot expect a feature's relevance hyperparameter to be informative of its actual relevance when there are potentially many redundant features in the feature space, such as is the case in this study. Since feature selection and the identification of relevant features is an important goal of this study, we decided it was not worth investigating the use of this approach any further, and we therefore have no results to report using it.

4.3.6 Comparing Classifiers

In this study, we need to compare a number of classifiers to each other. When a lot of data is available, one can randomly draw a sizeable test set out of the available data, and use this test set to perform classifier comparisons. However, when data is limited, as is our case since we are working with relatively small verb classes, comparing classifiers to determine whether we can say that one is better than the other in a statistically significant way is a difficult and debated problem.

Dietterich (1998) studied the behaviour of a number of descriptive statistical tests that can be used to compare classifiers using a small data sample. Of the four tests he studied, he found that a paired t test applied to the results of a tenfold cross-validation run was the most powerful, *i.e.*, it was most likely to detect real differences between classifiers. However, it was also the most likely to make Type I errors, *i.e.*, to claim there was a difference between classifiers when there really was not one.

Salzberg (1997) also warns that the paired t test has a high probability of making Type I errors because the trials in a cross-validation experiment are not independent. If more than two folds are used, each trial shares most of its training data with each of the other trials. Furthermore, even if the test sets of the cross-validation trials are disjoint, they are not really independent since the testing data for one trial is used as training data for all the other trials.

Dietterich (1998) recommends an alternative test, the "5x2cv paired t test" which uses five replications of twofold cross-validation and a carefully designed test based on it. He claims that it has less dependence between trials within a single cross-validation run because the training sets are non overlapping. The trials are not independent, since the training data of one trial is the testing data of the other, but they are less dependent than when three or more folds are used. Thus, they violate the assumptions of the statistical test to a lesser degree. In his study, he found that this test has a low probability of making Type I errors, which makes it a more reliable test than the t test applied to tenfold cross-validation.

However, the 5x2cv test has the strong disadvantage of using twofold cross-validation which we expect to be highly pessimistically biased (Kohavi, 1995a,b). The other disadvantage is that it is not a very powerful test. When a real difference exists between two algorithms, it is much less likely to detect it than the t test is. In fact, Dietterich (1998) finds that even when the performance of two learning algorithms differ by as much as ten percentage points, this test is still only able to detect the difference about one third of the time, making this test almost useless in practice.

When two algorithms are compared on the same test set and the results of each algorithm on each test example are available, Salzberg (1997) recommends considering the examples where

the algorithms differ and ignoring the ones where they agree. One can compare the proportion of examples Algorithm A got wrong but Algorithm B got right with the proportion of examples Algorithm B got wrong but Algorithm A got right, using a simple binomial test. Salzberg (1997) claims that this test is a much better test than the t test. Dietterich (1998) studies two tests that are based on a similar idea, including one called the McNemar test with continuity correction, and finds that they have a low probability of making Type I errors. However, he also finds that these tests are even less powerful than the 5x2cv test at detecting real differences. When comparing classifiers using our unseen test data, we use the McNemar test with continuity correction (see §6.1.1).

None of the studies we have found on the topic of comparing classification algorithms have discussed the use of statistical tests to compare repeated k-fold cross-validation runs using a $k > 2$. In our experiments, we always use repeated cross-validation runs because they yield more stable estimates of accuracy with a lower variance (Kohavi, 1995a,b). We also always use ten folds, because this works well with our training sets, whose size is always a multiple of ten. Kohavi (1995a,b) shows that using ten or 20 folds tends to yield unbiased or slightly pessimistically biased estimates of the accuracy, which confirms that this is generally a good estimate. He also shows that stratified cross-validation, where the class distribution within each fold is kept roughly the same, is even less biased. We always use stratified cross-validation, since that is what C5.0 does.

In principle, the lower variance of the accuracy estimates obtained using repeated cross-validation runs should allow for more powerful statistical tests to determine if two algorithms (or features sets) have significantly different accuracies. However, the t test and the ANOVA test both assume that the data they are comparing are independent. This is clearly not the case since each cross-validation run uses the same data, even if a different random split is used.

One convincing argument that there is a problem with these tests is that one can make any difference significant by simply running more repetitions of the cross-validation. If the difference between two algorithms is not significant when tested with tenfold cross-validation repeated 50 times, we could make it significant by using 500 or 5000 repetitions. However, if it takes that many repetitions, the difference is probably very small. When comparing different cross-validation results, it is therefore important to look at the size of a difference and consider whether it is meaningful or not for the application being studied, and not rely only on statistical tests. When designing his 5x2cv test, Dietterich (1998) found that five was the most appropriate number of repetitions for that test. With fewer repetitions, he had too much variance, but with more repetitions the risk of Type I errors was increased. With an appropriate experimental setup, one could determine power and the probability of Type I errors of the t test when used with different numbers of cross-validation repetitions, and determine the most appropriate

number of repetitions. However, we know of no study which has done such an experiment.

In previous studies on verb classification (*e.g.*, Merlo and Stevenson, 2001), an ANOVA test with a Tukey–Kramer post test was used on tenfold cross-validation repeated 50 times. This test was used to compare multiple classifiers at once. For comparing two classifiers, a two-tailed t test with Welch correction for continuity would be used on the repeated cross-validation results, with an adjustment for multiple tests when appropriate (see below). In either case, the independence assumptions of the tests are not respected, and the warnings of Salzberg (1997) and Dietterich (1998) about the risk of Type I errors are probably applicable. However, we still use this test (see §6.1.1), hoping that the lack of independence will not be too critical, because it is not clear that any other test will be better, and, as mentioned above, we have no way of knowing how many repeats are appropriate.

As an alternative test, we could take advantage of the fact that when we run repeated cross-validation for different subsets of features, we use the same random splits for each series of repeated cross-validation. Since `xval` controls the sampling seed used by C5.0, two sets of repeated cross-validation runs performed on the same data will use the same data splits, even if the feature subsets are different. Thus, we could use a paired t test or ANOVA instead of an unpaired one. A paired test would have the advantage of controlling for the variance that is due to the particular split used in each cross-validation run. It would therefore be more powerful at detecting existing differences, but the probability of Type I errors would likely be even greater than with the unpaired test, again because of the lack of independence between trials.

A second alternative would be to use the sign test, which has no distribution assumptions. This is also a paired test, but it would be used on the individual trials rather than on the average accuracy of the whole cross-validation run. When we run tenfold cross-validation repeated 50 times, we use the 50 average accuracies in the ANOVA or t test. In the sign test, each of the 500 trials would be considered individually. The number of trials where one feature subset performed better and the number of trials where the other feature subset performed better, would be counted, ignoring trials where they performed equally. The binomial test would then be applied to determine if the proportions are significantly different than one half each (Sokal and Rohlf, 1981, §13).

Being a non-parametric test, the sign test makes no assumptions about the distribution of the data. We could therefore be more confident in its conclusions than in those of the parametric tests. However, it also assumes independence between individual trials, and that assumption is still violated. Again, we hope that this violation is not too critical. Unfortunately, we know of no study which analyzes the behaviour of the sign test when used to compare repeated cross-validation runs as we suggest here.

Performing Multiple Classifier Comparisons

When statistical tests are performed to compare a number of different algorithms or feature subsets, and all accuracy estimates are based on the same training data, an adjustment is required in the significance level used. Salzberg (1997) shows that if n tests are performed at a nominal significance of α^*, the probability that at least one of them will be found to be significant by chance (*i.e.*, when the difference really is not significant) is

$$\alpha = 1 - (1 - \alpha^*)^n.$$

For example, if ten tests are performed at the significance level 0.05, we have $\alpha = 1 - (1 - 0.05)^{10} = .4012$, or a 40% chance that at least one of the tests will be significant by chance. If 200 tests are performed at that level, we have $\alpha = 1 - (1 - 0.05)^{200} = .9999649$ or a 99.996% chance that one of the tests will be significant by chance. Since a typical forward selection step in our experiments compares over 200 feature subsets, any statistical test would not be meaningful.

Salzberg (1997) mentions that one can compensate for this problem by an adjustment that is well known in the statistical community as the Bonferroni adjustment. With this adjustment, we set α to the desired significance level, and calculate the α^* required to get this α. This adjustment can be closely approximated by the simpler formula $\alpha^* = \alpha/n$ (where n is the total number of tests performed).

When we perform an ANOVA with a Tukey–Kramer post test, the test is properly adjusted to account for the multiple comparisons that it performs. However, if we use any test which is not already adjusted, such as the sign test discussed above, some appropriate adjustment will be required.

During feature selection, so many comparisons are performed that no test of significance can really be meaningful. Instead, as described in §4.2, we simply use a threshold improvement. Note, however, that we do not consider the accuracy estimate obtained by cross-validation on the selected feature subset to be representative of true generalization performance. Because of the procedure we use to select this subset, it is likely the one with the most optimistically biased estimated accuracy, and it is therefore no longer possible to rely on this estimate. To estimate the accuracy of a classifier built with this subset of features, it will be necessary to use previously unseen test data, as we do.

Chapter 5

Verb Classes

One of the important goals of this study is to design a feature space that can be general enough to apply to a wide range of class distinctions. (Following Levin, we focus on verb classes that exhibit alternations involving NP and PP arguments.) Hence, to evaluate our feature space, we want to run classification experiments over a number of different class distinctions. Section 5.1 describes the verb classes that we have selected for our experiments. Section 5.2 describes how we have established which verbs in those classes were usable for this study, and then how we selected which particular verbs to use for training and testing.

5.1 Class Descriptions

We have identified seven two- or three-way classification tasks which were selected to give us a good indication of how well our feature space can perform. To these seven tasks, we added three more, which grouped together larger numbers of classes, to see how well we can perform these more difficult classification tasks. Table 5.1 summarizes the ten classification tasks.

Table 5.2 gives more details about the two- and three-way classification tasks that we have selected. For each task, we show the immediate contrast, in terms of syntactic frames and/or thematic mapping, between the verb classes involved. The syntactic frames shown are those that apply to all or most verbs in the class. The thematic role mapping indicates how arguments are mapped to syntactic slots for each frame.

The classification tasks we chose vary in difficulty. For most tasks, knowing exactly what PP arguments each verb can take would be sufficient to perform the classification. However, we do not have access to such knowledge, since we restrict ourselves to acquiring syntactic information from the corpus. With the simple tools we use, we cannot distinguish PP arguments either from PP adjuncts or from PPs that should be attached to nouns instead of the verb. For example, the PP_{for} argument in *I admired Jane for her honesty* is not distinguished from the PP_{for}

Task Name	Class Count	Verb Classes
Benefactive/Dative	2	• Benefactive: *Build* (26.1) and Preparation (26.3) verbs • Dative: Recipient verbs (13.1, 13.3)
Clear Alternation	2	• *Cheat* verbs† (10.6) • *Steal* (10.5) and *Remove* (10.1) verbs†
Admire/Amuse	2	• Verbs of Psychological State: *Amuse* verbs (31.1) • Verbs of Psychological State: *Admire* verbs (31.2)
Swarm Alternation	2	• Unergative: *Run* verbs‡ (51.3.2) • Verbs of Sound Emission‡ (43.2)
Wipe Alternation	2	• *Wipe* verbs† (10.4.1,2) • *Steal* (10.5) and *Remove* (10.1) verbs†
Spray/Load Alternation	3	• *Spray/Load* verbs† (9.7) • *Fill* verbs† (9.8) • Other verbs of Putting† (9.1–6)
Transitive Alternations	3	• Unaccusative: Change of State verbs (45.1–4) • Unergative: *Run* verbs‡ (51.3.2) • Object Drop: *Build* (26.1), Performance (26.7) and Preparation (26.3) verbs
Six Locative Classes	6	The six basic locative classes, marked with †
All Locative Classes	8	All locative classes, marked with † and ‡
All Classes	13	All classes (except Benefactive*)

Table 5.1: Summary of the ten classification tasks. Levin section numbers are in parentheses. * The Benefactive class is a subset of the Object Drop class, and therefore these two classes cannot be put together in a classification task.

Class	Frames	Thematic Role Mapping		
		Subj.	D. Obj.	Other
1. Benefactive/Dative task				
Benefactive	NP-V-NP-NP	agent	theme	I. Obj.: benefactor
(Object-Drop)	NP-V-(NP-(PP_{for}))	agent	theme	PP_{for}: benefactor
Dative	NP-V-NP-NP	agent	theme	I. Obj.: recipient
	NP-V-NP-(PP_{to})	agent	theme	PP_{to}: recipient
Contrast: these two classes differ by the thematic role that is assigned to the indirect object, and what prepositional argument it alternates with (dative and benefactive alternations, Levin §2.1 and §2.2).				
2. *Clear* Alternation task				
Cheat verbs	NP-V-NP-(PP_{of})	agent	location	PP_{of}: theme
Steal/Remove verbs	NP-V-NP-(PP_{from})	agent	theme	PP_{from}: location
Contrast: these two classes differ in what variant of the transitive *clear* alternation they can occur in (a locative alternation, Levin §2.3.2).				
3. *Admire/Amuse* task				
Amuse verbs	NP-V-NP-(PP_{with})	stimulus	exper.	PP_{with}: attribute
Admire verbs	NP-V-NP-(PP_{for})	exper.	stimulus	PP_{for}: attribute
	NP-V-NP-PP_{in}	exper.	attribute	PP_{in}: stimulus
Contrast: these two classes assign the experiencer and stimulus roles in opposite ways, and accept different PP arguments.				
4. *Swarm* Alternation task				
Run verbs	NP-V-(PP_{loc})	agent		PP_{loc}: loc./dest.
(Unergative)	NP-V-NP-(PP_{loc})	caus. ag.	agent	PP_{loc}: loc./dest.
Verbs of Sound	NP-V-(PP_{loc})	agent		PP_{loc}: location
Emission	NP-V-PP_{with}	location		PP_{with}: agent
Contrast: these classes differ in whether they can participate in the *swarm* alternation (a locative alternation, Levin §2.3.4).				

continued on next page

Table 5.2: Main characteristics of the semantic classes we consider for each task. Parts of syntactic frames in parentheses are optional. PP_{loc} represents a prepositional phrase headed by any locative preposition. (Role abbreviations on next page.)

Class	Frames	Thematic Role Mapping		
		Subj.	D. Obj.	Other
5. *Wipe* Alternation task				
Wipe verbs	NP-V-NP-(PP_{from})	agent	theme	PP_{from}: location
	NP-V-NP	agent	location	
Steal/Remove verbs	NP-V-NP-(PP_{from})	agent	theme	PP_{from}: location
Contrast: these classes differ in whether they can participate in the *wipe* alternation (a locative alternation, Levin §2.3.3).				
6. *Spray/Load* Alternation task				
Spray/Load verbs	NP-V-NP-PP_{loc}	agent	theme	PP_{loc}: loc./dest.
	NP-V-NP-(PP_{with})	agent	loc./dest.	PP_{with}: theme
Fill verbs	NP-V-NP-(PP_{with})	agent	loc./dest.	PP_{with}: theme
Other v. of Putting	NP-V-NP-PP_{loc}	agent	theme	PP_{loc}: loc./dest.
Contrast: these classes differ in which variants of the *spray/load* alternation they can occur in (a locative alternation, Levin §2.3.1).				
7. Transitive Alternations task				
Unergative (*Run* verbs)	NP-V-(PP_{loc})	agent		PP_{loc}: loc./dest.
	NP-V-NP-(PP_{loc})	caus. ag.	agent	PP_{loc}: loc./dest.
Unaccusative (Change of State v.)	NP-V	theme		
	NP-V-NP-(PP_{with})	caus. ag.	theme	PP_{with}: instrument
	NP-V-NP	instrument	theme	
Object-Drop (*Build*, Perf., Prep. v.)	NP-V-(NP-(PP_{for}))	agent	theme	PP_{for}: benefactor
	NP-V-NP-NP	agent	theme	I. Obj.: benefactor
Contrast: these classes assign roles differently for transitive and intransitive frames (transitivity alternations, Levin §1.1.2 and §1.2.1) and accept different PP arguments.				

Table 5.2 (*cont'd*): Main characteristics of the semantic classes we consider for each task. Role abbreviations: loc.: location, dest.: destination, exper.: experiencer, caus. ag.: causative agent.

adjunct in *I amused Jane for the money.* Thus, if two classes differ in that one can take a PP argument that the other cannot, we expect the features involving that PP to be useful, but imperfect, indicators. For the *Wipe* Alternation task (5), perfect information about syntactic frames and allowable PP arguments would not suffice since there is no frame or slot which is allowed by all verbs in one class and no verbs in the other. One might therefore expect this task to be one of the hardest ones we consider.

The Benefactive/Dative (1), *Admire/Amuse* (3), and Transitive Alternations (7) tasks were chosen because the verb classes involved assign different thematic roles to slots in the same syntactic frames. For example, consider the double object frame in the Benefactive/Dative task—benefactive verbs assign the benefactor role to the indirect object slot, whereas dative verbs assign the recipient role to that slot. The classes involved in these three tasks also differ in what PP arguments they can take, so we expect that PP argument information and role information will both contribute to these classification tasks. The *Clear* Alternation (2), *Swarm* Alternation (4), *Wipe* Alternation (5), and *Spray/Load* Alternation (6) tasks involve classes that differ in which locative alternations they can participate in. For example, the classes involved in the *Spray/Load* Alternation task differ in which variants of the *spray/load* alternation (Levin §2.3.1) they can occur in, namely the *with* variant (*e.g.*, *Leslie staffed the store with employees*), the locative variant (*e.g.*, *Cora coiled rope around the post*), or both (*e.g.*, *Jessica sprayed paint on the wall*, *Jessica sprayed the wall with paint*).

The tasks labelled Six Locative Classes and All Locative Classes group together six and eight classes, respectively, involving locative alternations. The All Classes task puts all thirteen classes together in one big classification experiment. These three tasks are meant to test how well our feature space can deal with larger numbers of classes, and give us an idea of how much we can expect to scale the approach to multiple class distinctions.

5.2 Verb Selection

For each verb class that we used, we determined which verbs were available for our study as follows:

1. We took the list of all verbs belonging to the particular class or classes in Levin.
2. We removed all verbs that did not occur at least 100 times in our corpus, the BNC. (Our study still considers a good number of infrequent verbs, since 100 occurrences is very low in a corpus of 100 million words.)
3. We removed verbs that belonged to more than one class under consideration.
4. We removed verbs that were highly polysemous, as determined by inspection, or for which

the sense under consideration was clearly a fairly rare sense.[1] For example, *perk*, in the sense of making coffee in a percolator, belongs to class 45.3, Cooking verbs, but that is a rare sense, so we removed *perk* from this class.

Table 5.3 indicates how many verbs were available in each class at different steps of the process described above.

Of the verbs available in each class, we selected 20 as training and development data, to be used to explore various classification approaches, and various feature sets, which we compare to each other using cross-validation.

The 20 training verbs were selected randomly for each class, with two exceptions, due to the need to keep as many verbs available for testing as possible. In the development of this work, we used the verbs and classes from (Merlo and Stevenson, 2001), as well as a number of other verbs, so these verbs cannot be considered as previously unseen data in our study. In selecting *Build*, Performance and Preparation verbs for training, we could not afford to let the random selection of training verbs "use up" all the previously unseen verbs, given that we had so few of them. Hence, for the Benefactive (*Build* and Preparation) verbs, we took the eleven previously seen verbs and added nine more randomly. For the Object Drop (*Build*, Performance and Preparation) verbs, we randomly chose 20 among the 21 previously used verbs and the nine that were chosen as Benefactive verbs. For all other classes, the selection of training verbs was strictly random, since either there was no concern about running out of new verbs (Unaccusative and Unergative verbs), or almost all the verbs were new data (all other classes).

For test data, we used all verbs remaining after the selection of training data, except those that had been used in the previous studies we build on. Thus, we used all verbs which were previously unseen to us as test data.

The training and testing verbs selected using this procedure are listed in Appendices A and B, respectively.

[1] We are aware that, by removing highly polysemous verbs, we make the classification task somewhat easier for ourselves. However, ambiguity is by no means removed—verbs that belong to a single Levin class can still be polysemous. Furthermore, we plan to use Levin-class disambiguation based on the work of Lapata and Brew (1999) to deal with polysemous verbs in the future.

Verb Class	Levin classes	Number of Verbs				
		in Levin	≥ 100 in the BNC	Kept	Train	Test
Benefactive: *Build* and Preparation verbs	26.1, 26.3	58	52	35	20	15
Dative: Recipient verbs	13.1, 13.3	34	34	27	20	7
Cheat verbs	10.6	48	31	29	20	9
Steal and *Remove* verbs	10.5, 10.1	77	52	45	20	24*
Amuse verbs	31.1	220	144	134	20	114
Admire verbs	31.2	45	36	35	20	15
Unergative: *Run* verbs	51.3.2	124	87	79	20	45*
Verbs of Sound Emission	43.2	119	64	56	20	34*
Wipe verbs	10.4.1,2	56	44	35	20	15
Spray/Load Verbs	9.7	49	43	36	20	15*
Fill verbs	9.8	95	65	63	20	42*
Other verbs of Putting	9.1–6	70	63	48	20	28
Unaccusative: Change of State verbs	45.1–4	318	197	169	20	135*
Object Drop: *Build*, Performance and Preparation verbs	26.1, 26.3, 26.7	76	68	50	20	24*

Table 5.3: Number of verbs in each class at various phases of processing: number of verbs listed in Levin (1993); of those, number of verbs which occur at least 100 times in the BNC; of those, number of verbs which are not too polysemous and were kept for the study; number of verbs used for training and development; number of verbs used for testing. (*) In a number of classes, some of the verbs were used in previous studies and could not be considered unseen test data. In those cases, the number of training and testing verbs might not add up to the number of verbs kept.

Chapter 6

Experimental Results

In this chapter, we compare the accuracy of four versions of our feature space. The first version we consider is our core feature space—*i.e.*, all our features except for the syntactic collocation ones. Building this core feature space was one of the main goals of this study, and therefore we pay particular attention to it and compare other feature sets to this one.

The second version we consider is our whole feature space—our core features plus the syntactic collocation features. The syntactic collocation features are a crude approximation of selectional preference, but one that we expect should still provide some useful information for the classification tasks. However, since these features are somewhat experimental and not fine-tuned, our expectations for this set are limited.

One of the goals of this study is to investigate whether it is necessary for a linguist to devise a relevant set of features for each class distinction of interest, or if a general feature space can be adequate. In order to evaluate this, we have defined what we will call Levin-derived subsets for each task, the third version of our feature space. For each verb class, Levin gives a number of examples illustrating what alternations and syntactic frames are characteristic of that verb class. We have systematically identified which features within our core feature space are related to the syntactic frames and the alternations illustrated by each of her examples. In this way, for each verb class, we obtained a subset of relevant features from our core features, which we call the Levin-derived subset for that verb class. For each verb classification task, then, the Levin-derived subset is the union of the Levin-derived subsets for all the verb classes involved in the task. For example, the Levin-derived subset for the Benefactive/Dative task is the union of the Levin-derived subsets for classes 26.1 (*Build* verbs), 26.3 (Preparation verbs), 13.1 (*Give* verbs) and 13.3 (Verbs of Future Having). (The Levin-derived subsets are available from the author on request.)

The last version we evaluate are the subsets selected by stepwise feature selection (hereafter referred to as "stepwise"), as described in §4.2. In order to better understand how stepwise

works, we also look at its incremental behaviour—*i.e.*, how well the feature set retained at each iteration performs.

Finally, we compare these results to the baseline. In all the tasks we consider, the training data is balanced, having exactly 20 verbs from each class. If there were no information in the feature space, any algorithm trained on this data would not be expected to perform better than chance. Our baseline, therefore, is the chance baseline. For a k-way classification task, the chance baseline has an expected accuracy of $1/k$.

6.1 Methodology

In order to evaluate the different feature sets described above, we performed two kinds of experiments. First, we used tenfold cross-validation repeated 50 times to estimate the average accuracy on the training data only (20 verbs per class). (We will refer to these estimates as "mean cross-validation accuracy" in the rest of this chapter.) We then used the training data to build a classifier, which we evaluated on our previously unseen test data (described in §5.2).

We believe it is important to consider both the mean cross-validation accuracy, which is estimated using the training data, and the accuracy on the test data, because the number of test verbs is limited in many classes. When comparing two sets of results, we can be more confident that a difference is meaningful if the mean cross-validation accuracy and the test accuracy both increase or both decrease. If one increases and the other decreases, we should question whether there really is a difference between the classifiers compared.

In theory, the test accuracy should be more reliable, because it is estimated over previously unseen data and is therefore not biased by our attempts at improving the algorithm and the feature space, whereas the mean cross-validation accuracy might be. However, the test accuracy is based on a single classifier built by C5.0 over the training data, and does not take into account the variance due to the choice of training data—if we had used different training verbs, we would expect somewhat different results. Repeated cross-validation, on the other hand, has much lower variance because it averages accuracies over a number of training/test splits of the training data. This lower variance means that we can be confident in the significance of much smaller differences in the mean cross-validation accuracy than differences in the test accuracy.

In summary, because it is unbiased, the test accuracy is a more reliable indicator of the true accuracy of our feature space. But having lower variance, the mean cross-validation accuracy is a more powerful tool to detect differences between versions of our feature space.

6.1.1 Statistical Methodology

Given the small sample sizes in our test data, descriptive statistical tests on our results on unseen data lack power. We performed them, but few were significant, so we do not report them all here. Furthermore, since we are using all the verbs available as test data, subject to the filter described in §5.2, our test data cannot be considered a random sample, as most tests of statistical significance assume, so it is not clear that the tests are appropriate. In our tables, we report test accuracy both in terms of percentages and counts, in order to help the reader intuitively interpret how meaningful differences are. We report meaningful statistical tests in the main body of the text only.

When comparing cross-validation experiments, we used an unpaired t test, using the Welch correction for continuity (Uitenbroek, 1997c; GraphPad Software, 1998). When comparing test results on unseen data, we used a McNemar test with continuity correction (Dietterich, 1998; Uitenbroek, 1997b), which consistently produced almost identical significance levels as the binomial test recommended by Salzberg (1997). See §4.3.6 for a discussion of these tests.

Note that we did not apply any adjustments for multiple tests (see §4.3.6), since most of our analyses showed no significant differences; therefore, it was pointless to perform the adjustment. However, when we report significance tests, we provide the actual p value (rather than simply saying "significant at $p = 0.05$", for example) so that the reader can better assess the significance of the results.

6.2 Main Results

Tables 6.1 and 6.2 summarize our main results. Table 6.1 reports mean cross-validation accuracy, *i.e.*, accuracy estimated on our training verbs. Table 6.2 reports accuracy on the test verbs.

The first result we can see from these tables is that our feature space allows us to perform significantly better than the chance baseline in all cases, reducing the test error rate by 42% to 69% for the different tasks. Although there is room for improvement in accuracy, the feature space provides considerable information to the machine-learning system about the verb classification tasks we experimented with in this study.

6.2.1 Core Features *vs.* All Features

Comparing the accuracy of the classifiers built using our core features *vs.* using all our features (*i.e.*, adding syntactic collocation (SC) features to our core features), we see that accuracy is sometimes better when using all features, but more often worse. Looking at the test results,

Task Name	Baseline	Core features		All (with SC)		Levin-der.		Stepwise
Benefactive/Dative	50.0	74.1	0.6	82.4	0.6	79.5	0.6	(87.8)
Clear Alternation	50.0	88.9	0.4	84.4	0.4	83.0	0.5	(92.9)
Admire/Amuse	50.0	79.6	0.6	81.1	0.7	80.8	0.5	(92.5)
Swarm Alternation	50.0	79.9	0.7	89.2	0.6	71.9	0.8	(90.1)
Wipe Alternation	50.0	79.4	0.8	66.7	0.8	85.8	0.6	(92.2)
Spray/Load Alt.	33.3	78.3	0.5	72.6	0.6	79.2	0.5	(88.5)
Transitive Alts.	33.3	68.3	0.7	72.4	0.6	68.7	0.5	(80.1)
Six Locative Classes	16.7	69.0	0.3	66.4	0.4	72.8	0.4	(71.6)
All Locative Classes	12.5	69.3	0.4	66.4	0.4	64.7	0.3	(74.9)
All Classes	7.7	58.6	0.3	57.0	0.3	58.7	0.2	(58.6)

Table 6.1: Comparison of our main approaches: mean cross-validation accuracy on training verbs. The baseline column shows the expected accuracy of a random classifier. The next three columns show accuracy estimated using tenfold cross-validation repeated 50 times, for our core features, all our features—our core features plus the syntactic collocations (SC) features—and the Levin-derived subsets, respectively. The last column shows the mean cross-validation accuracy for the subsets of features selected by stepwise feature selection (stepwise). The stepwise figures are in parentheses because they are highly optimistic by definition, since stepwise retains the feature subset with the lowest mean cross-validation accuracy. Entries are percent accuracy with standard error (SE) where meaningful.

none of the differences is statistically significant (McNemar test, $p = 0.05$). Looking at the cross-validation results, adding SC reduces the mean cross-validation accuracy in six cases and increases it only in three cases (t tests: in each case, $N = 50$, $p \leq 0.0001$).

The *Swarm* Alternation task is especially noteworthy because adding SC significantly increases the mean cross-validation accuracy (t test: $N = 50$, $p < 0.0001$) while at the same time appreciably reducing the test accuracy (McNemar: $N = 79$, $p = 0.072$). Looking at the classifier C5.0 built for this task with SC, the first decision tree perfectly matched the training data, so boosting was disabled and the overall classifier used only two features in one decision tree with two decision nodes. Looking at the two features used, we find that the SC feature `%like_object%1` and the animacy feature `away_from_person` were deceivingly helpful in describing the training data, but did not classify the test data very well. In contrast, C5.0 used boosting when SC features were not available, thereby building a classifier with better generalization ability.

Overall, we find that SC features generally do not help classification accuracy. Since they are also expensive to calculate and to use, we conclude it is not worth using them, at least as we

Task Name	N	Baseline	Core feat.	All (w. SC)	Levin-der.	Stepwise
Benefactive/Dative	22	50.0 (11.0)	72.7 (16)	86.4 (19)	86.4 (19)	81.8 (18)
Clear Alternation	33	50.0 (16.5)	72.7 (24)	72.7 (24)	72.7 (24)	69.7 (23)
Admire/Amuse	129	50.0 (64.5)	82.2 (106)	87.6 (113)	90.7 (117)	79.1 (102)
Swarm Alternation	79	50.0 (39.5)	78.5 (62)	64.6 (51)	77.2 (61)	78.5 (62)
Wipe Alternation	39	50.0 (18.5)	84.6 (33)	84.6 (33)	84.6 (33)	61.5 (24)
Spray/Load Alt.	85	33.3 (28.3)	65.9 (56)	62.4 (53)	64.7 (55)	52.9 (45)
Transitive Alts.	204	33.3 (68.0)	72.1 (147)	74.5 (152)	77.0 (157)	66.2 (135)
Six Locative Classes	133	16.7 (22.2)	56.4 (75)	59.4 (79)	54.9 (73)	53.4 (71)
All Locative Classes	212	12.5 (26.5)	59.9 (127)	55.7 (118)	59.0 (125)	50.5 (107)
All Classes	507	7.7 (39.0)	46.4 (235)	43.8 (222)	43.6 (221)	35.1 (178)

Table 6.2: Comparison of our main approaches: evaluation on test data. N is the number of test verbs for each task. The other columns are the same as in Table 6.1. Entries are percent accuracy, with the number of correctly classified verbs in brackets, or the expected number of correctly classified verbs in the case of the baseline.

have defined them in this study. In her work on clustering verbs semantically, Schulte im Walde (2000) also found that using similar syntactic collocation information reduced both precision and recall. In future work, we might consider a much simpler definition, with only a few high-level semantic categories of nouns. This would add a much smaller number of features to our feature space, thereby reducing the probability that one of them would fit the training data by chance, as was the case for the *Swarm* Alternation task. Note that the animacy features we use (see §2.7) are already a first step in that direction. A very simple set of semantic categories (*e.g.*, animate beings, physical objects, and possibly only a few more categories) might be more informative for verb classification than the 30-node WordNet cut we used in this study. (Schulte im Walde, personal communication, has also considered taking a similar approach in her work.)

6.2.2 Core Features *vs.* Levin-derived Subsets

Comparing the accuracy of the classifiers built using our core features *vs.* using the Levin-derived subsets, we find that there is no clear trend. The Levin-derived subsets have significantly higher mean cross-validation accuracy for three tasks, but significantly lower for three others (t tests: in each case, $N = 50$, $p < 0.0001$). On the test data, the Levin-derived subsets perform significantly better for the *Admire/Amuse* task only (McNemar+CC: $N = 129$, $p = 0.009$). Otherwise, accuracy is very similar to our core features.

This indicates that selecting a linguistically relevant subset of features can sometimes help,

but will usually not make a significant difference. In most cases, it will therefore be sufficient to use our core feature space as a whole and let the machine-learning system choose the most relevant features for a particular task, since investing in additional linguistic analysis may not yield any improvements.

6.3 Stepwise Feature Selection

Comparing the test accuracy of the classifiers built using our core features *vs.* using the subsets thereof selected by stepwise, we find accuracy of the latter to be generally worse. Accuracy is only improved for the Benefactive/Dative task, where the two features selected by stepwise increase the test accuracy from 72.7% (core features) to 81.8% (stepwise). For the *Wipe* Alternation, *Spray/Load* Alternation, All Locative Classes and All Classes tasks stepwise test accuracy is significantly worse (McNemar: $N = 39$, $p = 0.04$; $N = 85$, $p = 0.05$; $N = 212$, $p = 0.006$, and $N = 507$, $p < 0.001$, respectively).

The mean cross-validation accuracy of the feature subsets selected by stepwise is almost always significantly better than all other methods, but the test accuracy is generally worse. It is clear, therefore, that overfitting of the training data is a serious issue, and that if we are to use feature selection, more work will be required along the lines mentioned in §4.3.3 and §4.3.4.

Although we expected overfitting to be an issue (see §4.3.4), we still hoped that stepwise would improve predictive accuracy, or at least not degrade it, but our results show otherwise. Furthermore, it is a very expensive approach: it takes from several hours, for the two-way tasks, to several days, for the multi-way ones, to complete on hardware that is close to the fastest currently available on the market. Stepwise would have a use only if we could improve it enough so that it performed comparably to our core features. Then, even if it did not help building better classifiers, it would at least provide a list of the most useful features, and it would make interpretation easier, since only a small list of features would have to be analyzed. As it stands, though, there is no reason to use stepwise.

Since stepwise adds and removes features one at a time, we can also look at its incremental behaviour. To do so, we plotted the mean cross-validation accuracy and the test accuracy for the feature subset selected at each step of the process for the ten tasks we used in this study (Figures 6.1 and 6.2). In all graphs, each increment of the x axis corresponds to one full stepwise selection step, *i.e.*, a remove and an add step. The "base" line shows the chance baseline for the task. The "train" and "test" lines show the mean cross-validation accuracy and the test accuracy, respectively, for the feature subsets selected at that step.

From these graphs, we can see that stepwise systematically overfits the training data, since the mean cross-validation accuracy is consistently higher than the test accuracy (except in the

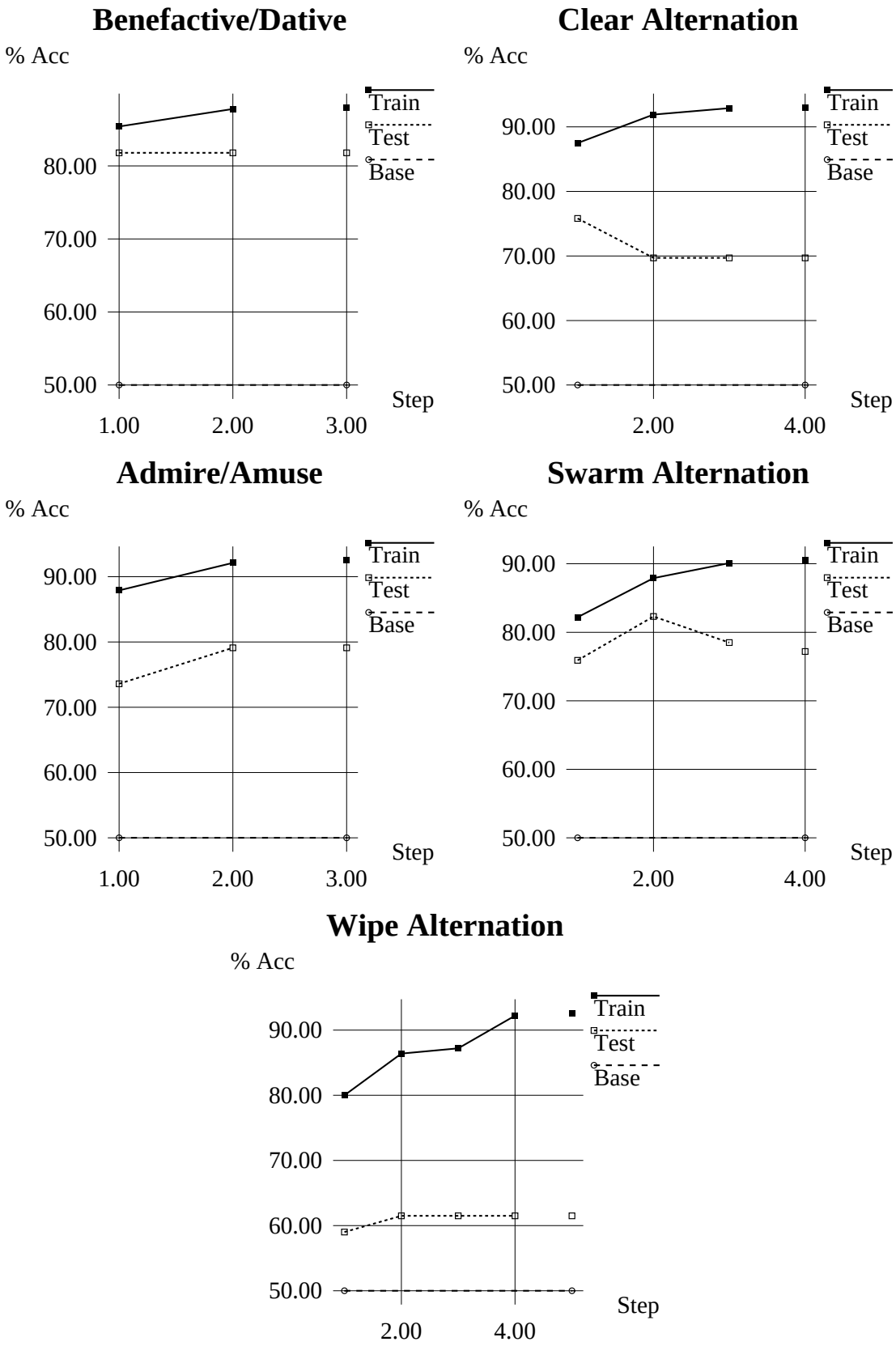

Figure 6.1: Detailed analysis of the stepwise selection process for the two-way tasks

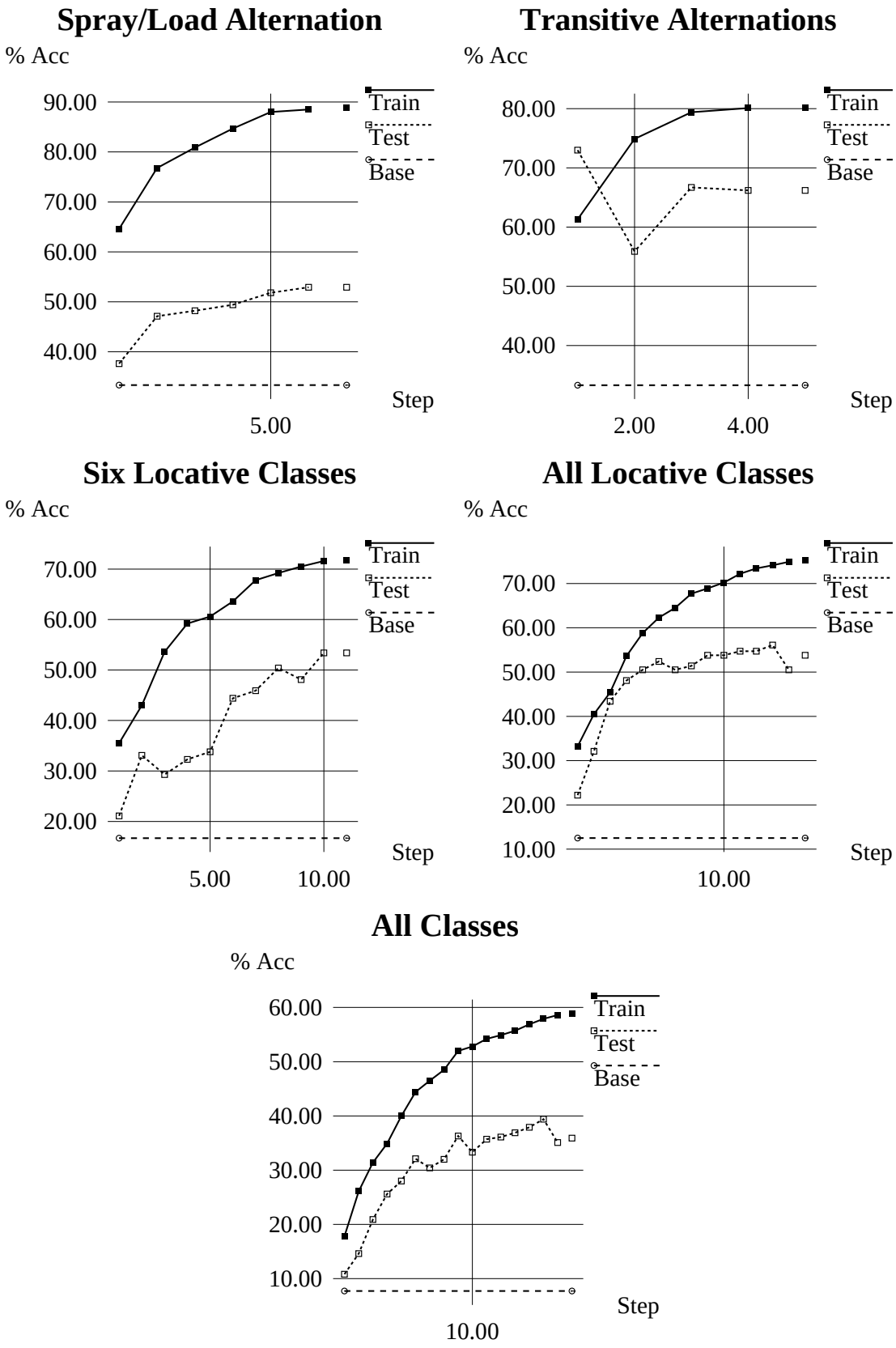

Figure 6.2: Detailed analysis of the stepwise selection process for the multi-way tasks

first step for the Transitive Alternations task). However, in most cases this overfitting is not harmful to the selection of features, since the test accuracy generally goes up with the mean cross-validation accuracy. Notable exceptions occur at the beginning of the selection process for the *Clear* Alternation, *Swarm* Alternation and the Transitive Alternations tasks, and at the end of the process for the All Locative Classes and All Classes tasks. Despite those cases, the graphs show that stepwise generally finds useful feature subsets, though not useful enough, as we discussed earlier.

6.4 Prior-Distribution Knowledge

In our experiments, the baseline accuracy is chance performance, because our training sets are balanced. Since C5.0 is not given prior class probabilities, it has no knowledge of which class is more frequent, and it therefore could not make use of such knowledge. If we provided class probabilities to C5.0, we would expect it to favour the majority class in uncertain cases, and then the relevant baseline would be the majority, or naive, classifier, rather than a chance classifier.

Table 6.3 shows the difference between the chance baseline and the naive baseline for the ten tasks we experimented with. Taking into account the prior class distribution sometimes yields a large improvement in expected classification accuracy over the chance baseline. We expect that

Task Name	Number of Classes	Chance Baseline (%Acc)	Number of verbs in each class (majority class in bold)	Majority Baseline (%Acc)
Benefactive/Dative	2	50.0	**35**, 27	56.4
Clear Alternation	2	50.0	29, **45**	60.9
Admire/Amuse	2	50.0	**134**, 35	79.3
Swarm Alternation	2	50.0	**79**, 56	58.5
Wipe Alternation	2	50.0	35, **45**	56.3
Spray/Load Alternation	3	33.3	36, **63**, 48	42.9
Transitive Alternations	3	33.3	**169**, 79, 50	56.7
Six Locative Classes	6	16.7	36, **63**, 48, 29, 45, 35	24.6
All Locative Classes	8	12.5	(above 6 and) **79**, 56	20.2
All Classes	13	7.7	(above 8 and) **169**, 50, 27, 134, 35	21.0

Table 6.3: Baselines: the chance baseline gives equal probability to each class. The naive baseline always chooses the majority class.

Feature Category	Defined in Section
Part of speech of the verb	2.1
Frequency of auxiliaries and modals	2.2
Combined feature: passive	2.3
Relative frequency of derived forms	2.4
Semantically empty constituents	2.5
Frequency of various syntactic slots	2.6
Animacy of NPs in various slots	2.7
Measure of slot overlap	2.8
Syntactic collocation features	2.9

Table 6.4: Summary of the feature categories

our accuracies would be similarly increased if we provided C5.0 with both our feature space and the prior class distribution. We leave it as future work to investigate how to do so.[1]

6.5 Feature Evaluation

To see how well the different types of features (summarized in Table 6.4) in our feature space perform, we examined what features were used by C5.0 for the ten tasks and the different sets of features we consider. There were not many striking patterns, but a few things are noteworthy. The first thing we noticed is that every type of feature was useful in at least one of the tasks. Thus, every part of our feature space has some utility in the classification of some of the verb classes we considered.

One surprising pattern is that the overlap features, which we expected to be very useful, were not used very much when we gave C5.0 access to our core feature space or our whole feature space, but they were used extensively when only the Levin-derived subset was available. Presumably, these features convey useful information for the classification tasks, but they are overshadowed by other features in our feature space.

One particular feature exhibited a rather unusual behaviour. The frequency of the slot PP_{as}, and the animacy of that slot, were the most useful features for the *Swarm* Alternation task. However, this preposition is not expected to be relevant for this task. To understand what was happening, we examined the sentences where SCOL identified this slot. We found

[1]Technically, C5.0 does not allow the specification of a prior distribution. However, it permits the specification of differential costs for each error type. If we take into account the size of each class when setting this cost function, we should be able to simulate the effect of prior class probabilities. How we should do so to correctly reflect the prior distribution is a difficult question left for future work.

that in many cases, what SCOL identifies a PP_{as} is really a subordinate clause introduced by *as* following the clause being analyzed. For example, in the sentence *Phones rang hot as Dave set the wheels in motion on the mission to rescue Des.* (BNC §A17), SCOL inaccurately marked *as Dave* as a PP attached to *rang* and did not identify a subject for *set*. Thus, our features `as` and `as_person` capture the frequency of such temporal relative clauses as well as the intended target, PP_{as}. Examining our corpus, we found that such temporal relative clauses are common with verbs of Sound Emission. They often describe situations where a sound emission event and some other event occur simultaneously (*e.g.*,, *The phone was ringing as he entered the door*, or *The house cracked as she was reading*). Thus, our features seem to tap into a semantic property of Sound Emission verbs that we had not anticipated. We leave it to future work to investigate this and other unexpectedly useful features, to determine the relation between the syntactic usages and the underlying semantic regularities that are being captured.

For the *Admire/Amuse* task, we noticed that one SC feature was particularly useful, namely `by_organism%1`, the frequency with which the object of *by* is a descendant of `organism%1` in the WordNet noun hierarchy. When given all features (including SC) for the *Admire/Amuse* task, C5.0 built the simplest classifier, using only two features, `by_organism%1` and `vbg_adjectival`, whereas with our core features and with the Levin-derived subset, C5.0 built much larger decision trees. It is noteworthy that `person%1` is a hyponym of `organism%1`, and that `by_organism%1` is in fact also an estimate of the animacy of the object of *by*. The animacy of the subject is expected to be relevant for this task, and consequently the animacy of the object of *by* would also be meaningful if passive frames were frequent. With the core features, and with the Levin-derived subset, C5.0 used the animacy of the subject, but `by_organism%1` was clearly a better feature in this case. This provides further evidence that a more restricted and better targeted set of SC features might be more useful in the future.

Our examination of the features used by C5.0 can answer the two questions we raised about the usefulness of the assumptions we used to identify indirect objects and to attempt to improve direct object identification (see §3.5). The first question was whether features involving the indirect object would be of any use, given that we identify them with only about 20% precision. Indeed, we found that features involving the indirect object were often used where they were expected to be relevant. We can therefore conclude that some useful information is successfully extracted from the corpus by our indirect object identification method, despite the noise.

The second question concerned direct objects, which we have two ways to identify. The first way is to simply use SCOL's identification of direct objects, which has good precision (high eighties) but low recall (about half) (see footnote 2, p. 27). The second way is to augment SCOL's identification of direct objects in the process of identifying indirect objects, which has a low precision, but should still improve overall recall. Since a number of features depend on

the identification of direct object, this gave us two redundant sets of features, the "original" set and the "augmented" set, and we let C5.0 decide which features were better. Again, we examined the features that C5.0 used in our experiments, and we found that features in the original set were used a lot more often than the augmented ones. We conclude that in this case, the noise added by our attempt to increase direct object recall (see §3.5) is significantly detrimental, and that only the original set of features should be kept. Thus, our technique for identifying indirect objects should not be used to modify the list of direct objects, even though not doing so leads to illogical analyses, such has having the same word labelled as both the direct and indirect object in a clause.

6.6 Comparison with Previous Work

We compare the accuracy of our feature space with the results of Merlo and Stevenson (2001), whose work we expand on. Merlo and Stevenson (2001) looked at the Transitive Alternations task with a set of five features that were carefully selected as linguistically relevant for this specific three-way class distinction. We cannot directly compare their numerical results with ours, because we used a different set of verbs. Instead, we ran an experiment using their five features with our verbs (see Table 6.5). We found that in comparison, our core features, all our features (with SC), and the Levin-derived subsets all significantly improve predictive accuracy for this task. The mean cross-validation accuracy is increased in a statistically significant way, and the test accuracy is increased in an appreciable way. This shows that our general feature space can outperform a carefully selected set of features on this task.

Feature set	Xval		Test ($N = 204$)	
	%Acc	SE	%Acc	(#)
Merlo and Stevenson (2001)	61.6	0.5	64.2	(131)
Our core features	68.3	0.7	72.1	(147)
All our features (with SC)	72.4	0.6	74.5	(152)
Levin-derived subset	68.7	0.5	77.0	(157)
Stepwise	(80.1)		66.2	(135)

Table 6.5: Comparison with earlier work. We use our training and test verbs with the original 3-way task of classifying unergative, unaccusative and object drop verbs with various feature sets.

Chapter 7

Conclusions

We set out to investigate the possibility of developing a general feature space for the semantic classification of English verbs. This feature space was to be as general as possible, *i.e.*, it should be usable for the distinction of a wide range of verb classes, eliminating the need for class-specific linguistic analysis. It was also to be transferable, *i.e.*, the extraction of the features should be based on techniques that can be applied to corpora in other languages.

We presented a feature space that meets these goals to a large extent. Our feature space is comprised of nine categories of features which capture various syntactic and semantic usage statistics for target verbs. These features were devised by doing a high-level analysis of verb class distinctions, based largely on alternations in Levin. We extracted our features from a large, unparsed corpus of English relying only on two language-specific tools: POS tagging and partial parsing.

We showed that, using the C5.0 machine-learning system, our feature space performs well in a number of verb classification experiments, achieving accuracies comparable to linguistically informed feature selection. However, the syntactic collocation features we proposed and an alternative automatic feature selection approach we considered were not very successful. On the other hand, we have reason to believe that the approach we used to build our general feature space can be transferred to other languages, as we expect future studies will demonstrate.

7.1 A General Feature Space

7.1.1 Contributions

We experimented with ten classification tasks involving from two to thirteen verb classes. Our results show that we can reduce the chance error rate by more than 40% in all ten cases, and by more than 50% in half the cases.

For each task, we compared the performance of our core feature space with the performance

of the subset of those features that we would expect to be relevant, given Levin's description of the verb classes involved. Because of the careful class-by-class linguistic analysis performed by Levin, these Levin-derived subsets provided us with an objectively defined linguistically motivated subset of our core feature space for each task. Our results show that the Levin-derived subsets only occasionally outperformed our core features and that this linguistic selection of features generally made no significant difference. Thus we showed that, in most of the cases we tested, our feature space is general enough, and that it can be given to a machine-learning system without further task-specific linguistic analysis.

7.1.2 Limitations and Future Work—More Extensive Studies

Although we selected tasks that we believe are representative of some verb classification difficulties to test our feature space, we have used only a small subset of the classes defined by Levin. We have not verified, and therefore cannot claim, that our feature space is sufficient for all class distinctions. We expect that it will be applicable to distinctions involving many of the 191 classes defined by Levin, but a more extensive study will be required to confirm this. We also have not encoded features to capture every type of alternation in Levin. For example, alternations involving a possessor (Levin §2.12 and §2.13) would motivate features dealing with possessive constructions, and reciprocal alternations (Levin, §2.5) would motivate features capturing conjunct subjects and direct objects, as well as the occurrence of the adverbs *together* and *apart*. Before we can expect to apply our feature space to all Levin classes, a systematic analysis of the alternations she covers will be required to add missing features.

Furthermore, following Levin, we have focused on alternations involving NP and PP arguments of the verb. In future work, we would like to cover verb classes involving other complement types, such as sentential and infinitival complements, *e.g.*, *Coerce* verbs (*to force someone to do something*) and *Aspire* verbs (*to attempt to do something*) (Dorr, 1997).

7.2 Syntactic Collocations

7.2.1 Contributions

We have also attempted to enhance our feature space with syntactic collocation (SC) features, as a simple approximation to selectional preference. We used a simple encoding, where we assign nouns to one of 30 semantic categories based on WordNet. We then counted the occurrences of these categories in a number of syntactic slots. Like selectional preferences, these feature give us an indication of what kinds of nouns can occur as various arguments of the verb. Comparing classification performance with and without including the resulting SC features, we found that

accuracy was sometimes improved, but usually reduced. Although these features sometimes provided useful information to the machine-learning system, they proved inadequate for our purposes, at least as we have encoded them. This confirms the results of Schulte im Walde (2000), who found that using similar SC information reduced precision and recall in her work on clustering verbs semantically.

7.2.2 Limitations and Future Work—An Improved Encoding

We have only tried one encoding of SC and did not try to refine it. However, since this type of feature is not transferable to languages without a resource like WordNet, it was not a priority to do so. In future work on verb classification in English, it would be useful to experiment with an encoding involving only a few semantic classes that are expected to be relevant, such as animate beings, physical objects, and possibly a few more categories. This would provide us with a small number of targeted SC features, instead of the large number of SC features we used, most of which were irrelevant for most classification tasks. Having a smaller set of SC features would also reduce the likelihood of one of them fitting the training data by chance. Given how difficult it is to encode selectional preferences well, however, we expect that choosing appropriate semantic categories will be a challenging task.

7.3 Automatic Feature Selection

7.3.1 Contributions

We investigated various approaches to feature selection. Our feature space is intended to be general, and therefore, given any specific classification task, we expect a number of our features not to be relevant for that task. The machine-learning system we use, C5.0, performs feature selection internally. We also experimented with an alternative automatic-feature-selection approach, stepwise feature selection. Our results show that although this technique selects a useful subset of features, accuracy is almost always lower with this subset than when we let C5.0 select features from our core feature space on its own. We conclude that if a feature selection technique is to be applied to our feature space, it needs to be significantly better than stepwise feature selection. Otherwise, it is best to let the machine-learning system deal with all the features using its own internal mechanisms, at least when using C5.0.

7.3.2 Limitations and Future Work—Better Feature Selection Techniques

We experimented with only one feature selection approach. Important improvements to the technique have been proposed, though we have not experimented with them. The first improve-

ment would make more use of the information available at each step of the feature selection process, by considering adding together multiple features that are useful individually (Kohavi and John, 1997). The second improvement would reduce the chances of overfitting by splitting the training data into two sets, the first being used to identify the most useful features, the second to make sure the features selected do not simply overfit the first set (Ng, 1998). In future work, we would like to implement these modifications and investigate their usefulness when used with our feature space.

7.4 Transferability to Other Languages

7.4.1 Contributions

Since one of our long-term goals is to apply these techniques to other languages, we have to consider the transferability of our techniques. Except for the SC features, which did not prove useful anyway, the only language-specific tools that we rely on are a POS tagger and a partial parser for English. The use of a partial parser is a considerable source of noise in our features, but despite that, our results show good classification accuracy. Even the features involving the indirect object—probably our noisiest features—were useful. We therefore expect that imperfect partial parsers for other languages, which can be developed for many languages given the current state of the art in natural language processing, will be adequate.

7.4.2 Limitations and Future Work—Other Languages and Clustering

A potential limitation is that we used the state of the art in partial parsing for English, the partial parser of Abney (1991, 1997). One might think that this choice would weaken our claim that our approach is transferable to other languages. However, partial parsing is inherently noisy, and we have shown that our method works despite the noise. We do not believe that our choice of partial parser made much difference in obtaining our results.

The principal limitation with respect to transferability is that we have not yet attempted to transfer our feature space to another language. In future work, we would like to develop feature spaces for other languages, using a similar approach based on similar principles. A considerable challenge in such studies, however, will be to obtain training and testing data. When working in English, Levin provides the gold standard, even if it is not always perfect. In other languages, we will first have to manually develop at least a partial verb classification to have some training and/or testing verbs, like Schulte im Walde and Brew (2002) had to do for their clustering study of German verbs.

This, in conjunction with other work (Stevenson and Merlo, 2001; Schulte im Walde, 2000;

Brew and Schulte im Walde, 2002), suggests another future research direction, namely clustering, or unsupervised classification. In this study, we have only considered supervised classification, where the machine-learning system can select relevant features using the training data. Further work will be required to determine whether clustering techniques can yield useful results with our feature space, despite the presence of many features that may not be relevant for a given set of verbs being clustered. Given that we would like to expand our techniques to languages where a classification such as Levin's does not already exist, it will be important to develop unsupervised verb classification techniques in the future.

When attempting to transfer our feature space to other languages, we expect to need new feature types to reflect the characteristics of those languages. For example, features dealing with cases and morphology will be required for many languages. The way verbs are expressed in different languages may also present some challenges. For example, many verbs of hitting in Chinese are expressed using the same base verb with an accompanying adverb or PP to indicate what type of hitting is described (Gao Hong, personal communication). Persian would pose an even bigger challenge. In English, the verbs *take* and *make* are considered light verbs in expressions like *take a hit*, *take root*, or *make a decision*, because they convey little semantic content on their own (Trask, 1993). When classifying English verbs, we tend to ignore light verbs, but the phenomenon is too common to be ignored in Persian (Amtrup *et al.*, 2000; Dabir-Moghaddam, 1997). A way of identifying and dealing with multi-word verbal expressions would be required to classify verbs in Persian or in Chinese.

After having developed feature spaces for a number of languages, we might be able to generalize this line of work one level higher, building a language-independent general feature space. We could then specialize it for any language for which we want to develop a semantic classification of verbs using the help of automatic verb classification, at least in theory.

Appendix A

Training Verbs

Unaccusative: Change of State verbs: *advance, alter, awake, burst, cool, deteriorate, dissipate, expand, fill, flatten, flood, loosen, melt, moisten, multiply, overturn, scald, taper, tear, unfold.*

Unergative: *Run* verbs: *bolt, bound, creep, float, fly, gallop, hurtle, journey, meander, romp, rush, scamper, sidle, stagger, stomp, stray, stroll, travel, trundle, wade.*

Object Drop: *Build* Performance and Preparation verbs: *arrange, assemble, build, carve, cast, chant, compose, dance, develop, embroider, hack, hammer, perform, pound, produce, sculpt, sew, weave, whistle, whittle.*

Benefactive: *Build* and Preparation verbs: *arrange, assemble, build, carve, cast, chisel, develop, embroider, fashion, hack, hammer, knit, light, pound, sculpt, sew, spin, stitch, weave, whittle.*

Recipient verbs: *allocate, allot, assign, award, bequeath, cede, feed, give, issue, loan, offer, owe, pay, peddle, promise, rent, repay, sell, trade, will.*

***Amuse* verbs:** *alarm, antagonize, appall, astonish, astound, comfort, dazzle, demolish, devastate, disgust, enthrall, enthuse, infuriate, mystify, provoke, puzzle, ruffle, soothe, stun, tire.*

***Admire* verbs:** *adore, appreciate, deplore, detest, dislike, dread, enjoy, envy, favour, idolize, love, pity, prize, relish, resent, respect, revere, tolerate, treasure, value.*

Verbs of Sound Emission: *buzz, chatter, clap, clatter, click, cling, crack, crash, crunch, hiss, moan, murmur, purr, ring, rustle, splutter, swish, thud, thunder, whine.*

***Spray/Load* verbs:** *cram, crowd, cultivate, daub, drape, inject, load, pile, plant, pump, scatter, smear, sow, splash, splatter, spray, sprinkle, squirt, stack, string.*

***Fill* verbs:** *bandage, bind, carpet, deck, decorate, douse, encircle, enrich, garnish, infect, inundate, line, mask, pollute, replenish, riddle, shroud, spot, suffuse, taint.*

Other verbs of Putting: *curl, dip, dribble, hang, install, lay, lift, lower, perch, position, put, raise, ram, rest, scoop, slop, stow, tuck, wedge, whirl.*

***Cheat* verbs:** *bleed, burgle, cleanse, con, cure, deprive, disarm, divest, drain, ease, free, pardon, purge, purify, ransack, render, rid, rob, sap, strip.*

***Steal* and *Remove* verbs:** *abstract, disengage, dislodge, eradicate, evict, extract, grab, oust, reclaim, regain, remove, rescue, retrieve, separate, sever, sneak, steal, uproot, withdraw, wrest.*

***Wipe* verbs:** *bail, comb, dust, erase, filter, flush, lick, polish, prune, rake, rub, scour, scrape, shear, skim, suck, trim, wash, whisk, wring.*

Appendix B

Testing Verbs

Unaccusative: Change of State verbs: *abate, accelerate, age, attenuate, awaken, balance, bend, blacken, blast, blur, break, brighten, broaden, brown, burn, chill, chip, clog, close, collect, compress, condense, contract, crash, crease, crumble, crumple, crush, crystallize, darken, decompose, decrease, deepen, deflate, degenerate, degrade, detonate, dim, diminish, disintegrate, double, drain, dry, dull, ease, empty, enlarge, evaporate, fade, fatten, freeze, fuse, halt, harmonize, heal, heat, heighten, hush, ignite, improve, increase, incubate, inflate, intensify, lengthen, lessen, level, lighten, loop, magnify, mature, mellow, narrow, neutralize, operate, pale, pop, proliferate, propagate, quadruple, quicken, quieten, redden, reopen, reproduce, rip, ripen, rupture, sharpen, shatter, shorten, shrink, shrivel, shut, sicken, singe, sink, slacken, slow, smash, smooth, snap, soak, sober, soften, sour, splay, splinter, split, sprout, steady, steam, steep, stiffen, straighten, strengthen, stretch, submerge, subside, tan, tense, thaw, thicken, thin, tighten, tilt, topple, triple, vary, vibrate, waken, warm, weaken, worsen, wrinkle.*

Unergative: *Run* verbs: *amble, bowl, canter, charge, clamber, climb, crawl, dart, dash, file, flit, fly, hobble, inch, limp, lumber, lurch, nip, pad, plod, prowl, ramble, roam, run, saunter, scuttle, shuffle, slither, sneak, speed, streak, stride, strut, stumble, stump, swim, tack, totter, tramp, trek, troop, trudge, walk, whiz, zoom.*

Object Drop: *Build* Performance and Preparation verbs: *blend, brew, chisel, churn, compile, draw, fashion, forge, grind, hatch, hum, intone, light, make, mix, pour, prepare, scramble, set, shape, sing, spin, toss, write.*

Benefactive: *Build* and Preparation verbs: *blend, brew, churn, compile, forge, grind, hatch, make, mix, pour, prepare, scramble, set, shape, toss.*

Recipient verbs: *concede, grant, guarantee, lease, lend, refund, yield.*

***Amuse* verbs:** *affect, afflict, aggravate, agitate, alienate, amaze, amuse, anger, annoy, appease, arouse, assuage, baffle, bore, bother, bug, calm, captivate, charm, cheer, chill, concern, confound, confuse, console, content, convince, delight, demoralize, depress, disappoint, disarm, discourage, disillusion, distract, distress, disturb, electrify, embarrass, enchant, encourage, engross, enlighten, enliven, enrage, entertain, entice, entrance, exasperate, excite, exhaust, fascinate, flatter, fluster, frighten, frustrate, gall, galvanize, gratify, grieve, harass, haunt, horrify, humiliate, hypnotize, impress, inspire, insult, interest, intimidate, intrigue, irritate, jolt, lull, mesmerize, muddle, numb, offend, overwhelm, perplex, perturb, placate, plague, please, reassure, refresh, relax, relieve, repel, revitalize, revolt, sadden, satisfy, scare, shame, shock, startle, stimulate, stump, surprise, tease, tempt, terrify, terrorize, threaten, thrill, torment, trouble, try, unnerve, unsettle, upset, worry, wound.*

***Admire* verbs:** *abhor, admire, cherish, despise, distrust, fancy, fear, hate, lament, like, loathe, mourn, regret, trust, worship.*

Verbs of Sound Emission: *bellow, blare, blast, boom, bubble, chime, clash, crackle, creak, cry, groan, growl, hoot, howl, hum, jangle, pipe, pop, rasp, rattle, roar, rumble, scream, screech, shriek, splash, squeak, squeal, thump, tick, ting, trumpet, wail, wheeze.*

***Spray/Load* verbs:** *dab, hang, heap, jam, plaster, seed, settle, shower, spatter, spread, stick, stock, strew, stuff, wrap.*

***Fill* verbs:** *adorn, anoint, bathe, block, blot, bombard, choke, clog, clutter, coat, contaminate, cover, dirty, dot, drench, edge, embellish, emblazon, endow, entangle, face, fill, frame, imbue, intersperse, interweave, lash, litter, ornament, pad, pave, plug, ripple, saturate, smother, soak, soil, staff, stain, stud, surround, veil.*

Other verbs of Putting: *arrange, channel, coil, dangle, drip, dump, hoist, immerse, lean, lodge, loop, mount, place, pour, push, sit, situate, sling, spew, spill, spin, spoon, spurt, squash, squeeze, suspend, twirl, wind.*

***Cheat* verbs:** *absolve, acquit, cheat, defraud, deplete, milk, plunder, relieve, wean.*

***Steal* and *Remove* verbs:** *abduct, capture, confiscate, delete, discharge, dismiss, eject, eliminate, excise, expel, kidnap, liberate, omit, partition, pry, recover, redeem, repossess, rustle, seize, smuggle, snatch, subtract, wrench.*

***Wipe* verbs:** *distill, leach, mop, pluck, rinse, scrub, shave, shovel, smooth, soak, sponge, strain, wear, weed, wipe.*

Bibliography

Steven Abney. 1991. Parsing by chunks. In Robert Berwick, Steven Abney, and Carol Tenny, editors, *Principle-Based Parsing*. Kluwer Academic Publishers: Boston, MA. Originally appeared in Carol Tenny (ed.), *The MIT Parsing Volume*, 1988–89. Center for Cognitive Science, MIT. Available at `http://www.research.att.com/~abney/`.

Steven Abney. 1997. The SCOL manual version 0.1b. Available with the CASS package at `http://www.research.att.com/~abney/`.

ACL. 1999. *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*, College Park, Maryland, June 20–26.

Jan W. Amtrup, Hamid Mansouri Rad, Karine Megerdoomian, and Rémi Zajac. 2000. Persian–English machine translation: An overview of the Shiraz project. Memoranda in Computer and Cognitive Science MCCS-00-319. Computing Research Laboratory, New Mexico State University.

Chinatsu Aone and Douglas McKee. 1996. Acquiring predicate–argument mapping information in multilingual texts. In Branimir Boguraev and James Pustejovsky, editors, *Corpus Processing for Lexical Acquisition*, pages 191–202. MIT Press.

Avrim L. Blum and Pat Langley. 1997. Selection of relevant features and examples in machine learning. *Artificial Intelligence*, 97(1–2):245–271. Available at `http://www.isle.org/~langley/`.

BNC. 2000. British National Corpus user reference guide. Available at `http://www.hcu.ox.ac.uk/BNC/World/HTML/`.

Chris Brew and Sabine Schulte im Walde. 2002. Spectral clustering for German verbs. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP-02)*, pages 117–124, Philadelphia, PA, July 6–7.

Mohammad Dabir-Moghaddam. 1997. Compound verbs in Persian. *Studies in the Linguistic Sciences*, 27(2):25–59.

Thomas G. Dietterich. 1998. Approximate statistical tests for comparing supervised classification learning algorithms. *Neural Computation*, 10(7):1895–1924.

Bonnie J. Dorr. 1997. Large-scale dictionary construction for foreign language tutoring and interlingual machine translation. *Machine Translation*, 12(4):1–55. Available at `http://www.umiacs.umd.edu/~bonnie/`.

Bonnie J. Dorr and Doug Jones. 1996. Role of word sense disambiguation in lexical acquisition: Predicting semantics from syntactic cues. In *Proceedings of the 16th International Conference on Computational Linguistics (COLING-96)*, pages 322–327, Copenhagen, Denmark, August.

Daniel Gildea and Daniel Jurafsky. 2002. Automatic labeling of semantic roles. *Computational Linguistics*, 28(3):245–288.

GraphPad Software. 1998. Instat guide to choosing and interpreting statistical tests. Available at `http://www.graphpad.com`.

George H. John, Ron Kohavi, and Karl Pfleger. 1994. Irrelevant features and the subset selection problem. In William Cohen and Haym Hirsh, editors, *Machine Learning: Proceedings of the Eleventh International Conference (ICML-94)*, pages 121–129, New Brunswick, NJ, July. Morgan Kaufmann. Available at `http://robotics.stanford.edu/~ronnyk/`.

Leslie Pack Kaelbling. 1993. *Learning in Embedded Systems*. MIT Press, Cambridge, Massachusetts.

Kenji Kira and Larry A. Rendell. 1992. A practical approach to feature selection. In *Machine Learning: Proceedings of the Ninth International Conference (ICML-92)*, pages 249–256, Aberdeen, Scotland, July. Morgan Kaufmann.

Judith L. Klavans and Min-Yen Kan. 1998. The role of verbs in document analysis. In *Proceedings of the 36th Association for Computational Linguistics and the 17th International Conference on Computational Linguistics (COLING-ACL-98)*, pages 680–686, Montreal, Canada, August 10–15.

Ron Kohavi. 1995a. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence (IJCAI-95)*, pages 1137–1143, Montreal, Canada, August 20–25. Morgan Kaufmann. Available at `http://robotics.stanford.edu/~ronnyk/`.

Ron Kohavi. 1995b. *Wrappers for Performance Enhancement and Oblivious Decision Graphs*. Ph.D. thesis, Department of Computer Science, Stanford University, September. Available at `http://robotics.stanford.edu/~ronnyk/`.

Ron Kohavi and George H. John. 1997. Wrappers for feature subset selection. *Artificial Intelligence*, 97(1–2):273–324. Late draft available at `http://robotics.stanford.edu/~ronnyk/`.

Ron Kohavi and George H. John. 1998. The wrapper approach. In Huan Liu and Hiroshi Motoda, editors, *Feature Selection for Knowledge Discovery in Databases*. Kluwer Academic Publishers: Boston, MA. Available at `http://robotics.stanford.edu/~gjohn/`.

Ron Kohavi and Dan Sommerfield. 1995. Feature subset selection using the wrapper model: Overfitting and dynamic search space topology. In *Proceedings of the First International Conference on Knowledge Discovery and Data Mining (KDD-95)*, Montreal, Canada, August. Available at `http://robotics.stanford.edu/~ronnyk/`.

Maria Lapata and Chris Brew. 1999. Using subcategorization to resolve verb class ambiguity. In P. Fung and J. Zhou, editors, *Joint SIGDAT Conference on Empirical Methods in NLP and Very Large Corpora*, pages 266–274, College Park, Maryland, June. Association for Computational Linguistics. Available at `http://www.hcrc.ed.ac.uk/Site/LAPAM99.html`.

Lillian Lee. 1999. Measures of distributional similarity. In (ACL, 1999), pages 25–32.

Beth Levin. 1993. *English verb classes and alternations: a preliminary investigation*. Chicago University Press.

Hang Li and Naoki Abe. 1995. Generalizing case frames using a thesaurus and the MDL principle. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing*, pages 239–248.

Hang Li and Naoki Abe. 1998. Generalizing case frames using a thesaurus and the MDL principle. *Computational Linguistics*, 24(2):217–244.

Diana McCarthy. 2000. Using semantic preferences to identify verbal participation in role switching alternations. In *Proceedings of the First Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-00)*, Seattle, WA, May 1–3. North American Chapter of the Association for Computational Linguistics. Available at `http://www.cogs.susx.ac.uk/lab/nlp/mccarthy/mccarthy.html`.

Diana McCarthy. 2001. *Lexical Acqusition at the Syntax-Semantics Interface: Diathesis Alternations, Subcategorization Frames and Selectional Preferences*. Ph.D. thesis, University of Sussex. Available at `http://www.cogs.susx.ac.uk/lab/nlp/mccarthy/mccarthy.html`.

Paola Merlo and Suzanne Stevenson. 2001. Automatic verb classification based on statistical distributions of argument structure. *Computational Linguistics*, 27(3):373–408.

Radford M. Neal. 1997. Monte Carlo implementation of Gaussian process models for Bayesian regression and classification. Technical Report 9702, Dept. of Statistics, University of Toronto. Available at `http://www.cs.toronto.edu/~radford/`.

Robert G. Newcombe. 1998. Two-sided confidence intervals for the single proportion: comparison of seven methods. *Statistics in Medicine*, 17(8):857–872.

Andrew Y. Ng. 1998. On feature selection: learning with exponentially many irrelevant features as training examples. In *Machine Learning: Proceedings of the Fifteenth International Conference (ICML-98)*, pages 404–412, Madison, Wisconsin, July. Morgan Kaufmann. Available at `http://www.ai.mit.edu/people/ayn/`.

Akira Oishi and Yuji Matsumoto. 1997. Detecting the organization of semantic subclasses of japanese verbs. *International Journal of Corpus Linguistics*, 2(1):65–89. Available at `http://citeseer.nj.nec.com/oishi97detecting.html`.

Julian D. Olden and Donald A. Jackson. 2000. Torturing data for the sake of generality: How valid are our regression models? *Écoscience*, 7(4):501–510.

Steven Pinker. 1989. *Learnability and cognition: the acquisition of argument structure*. MIT Press, Cambridge, Massachusetts.

Paul Procter, editor. 1978. *Longman dictionary of contemporary English*. Longman, London.

J. R. Quinlan. 1993. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, San Mateo, California.

Randolph Quirk, Sidney Greenbaum, Geoffrey Leech, and Jan Svartvik. 1985. *A Comprehensive Grammar of the English Language*. Longman, New York.

Philip Resnik. 1997. Selectional preference and sense disambiguation. In *Tagging Text with Lexical Semantics: Why, What, and How? Proceedings of the Workshop*, pages 52–57, Washington, D.C., April 4–5. Association for Computational Linguistics.

Philip Resnik. 1998. WordNet and class-based probabilities. In Christiane Fellbaum, editor, *WordNet: an electronic lexical database*, pages 239–263. MIT Press, Cambridge, Massachusetts.

Douglas L. T. Rohde. 2002. TGrep2 user manual version 1.3, February 27. Available with the TGrep2 package at `http://tedlab.mit.edu/~dr/Tgrep2/`.

Steven L. Salzberg. 1997. On comparing classifiers: Pitfalls to avoid and a recommended approach. *Data Mining and Knowledge Discovery*, 1(3):317–328.

Beatrice Santorini. 1995. Part-of-speech tagging guidelines for the Penn Treebank project (third revision, second printing). Available at `ftp://ftp.cis.upenn.edu/pub/treebank/doc/`.

Sabine Schulte im Walde. 1998. Automatic semantic classification of verbs according to their alternation behaviour. Master's thesis, Institut für Maschinelle Sprachverarbeitung, Universität Stuttgart. Available at `http://www.ims.uni-stuttgart.de/~schulte/`.

Sabine Schulte im Walde. 2000. Clustering verbs semantically according to their alternation behaviour. In *Proceedings of the 18th International Conference on Computational Linguistics (COLING-2000)*, pages 747–753, Saarbrücken, Germany, August. Available at `http://www.ims.uni-stuttgart.de/~schulte/`.

Sabine Schulte im Walde and Chris Brew. 2002. Inducing German semantic verb classes from purely syntactic subcategorisation information. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 223–230, Philadelphia, PA, July 7–12. Available at `http://www.ims.uni-stuttgart.de/~schulte/`.

Eric V. Siegel. 1999. Corpus-based linguistic indicators for aspectual classification. In (ACL, 1999). Available at `http://www.cs.columbia.edu/~evs/`.

Robert R. Sokal and F. James Rohlf. 1981. *Biometry; The Principles and Practice of Statistics in Biological Research*. W. H. Freeman, New York, second edition.

Manfred Stede. 1999. *Lexical Semantics and Knowledge Representation in Multilingual Text Generation*. Kluwer Academic Publishers: Boston, MA.

Suzanne Stevenson and Paola Merlo. 2001. Indicator-based learning of argument structure: Computational experiments. Abstract in *Proceedings of the 14th Annual CUNY Conference on Human Sentence Processing*. Philadelphia, PA, March 15–17.

Suzanne Stevenson, Paola Merlo, Natalia Kariaeva, and Kamin Whitehouse. 1999. Supervised learning of lexical semantic verb classes using frequency distributions. In *Proceedings of SigLex99: Standardizing Lexical Resources*, College Park, Maryland, June. Available at `http://www.latl.unige.ch/personal/paola.html`.

R. L. Trask. 1993. *A Dictionary of Grammatical Terms in Linguistics*. Routledge, London.

Daan G Uitenbroek. 1997a. SISA One Mean. Available at `http://home.clara.net/sisa/onemean.htm`. Accessed July 2002.

Daan G Uitenbroek. 1997b. SISA Pairwise. Available at `http://home.clara.net/sisa/pairwise.htm`. Accessed July 2002.

Daan G Uitenbroek. 1997c. SISA T-Test. Available at `http://home.clara.net/sisa/t-test.htm`. Accessed July 2002.