Computational Measures of the Acceptability of Light Verb Constructions

by

Ryan North

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

Abstract

Computational Measures of the Acceptability of

Light Verb Constructions

Ryan North

Master of Science

Graduate Department of Computer Science

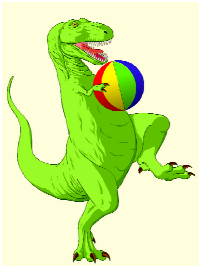
University of Toronto

2005

Light verb constructions are a semi-productive class of multiword expression which have not yet been studied computationally in great detail. These constructions combine a restricted set of light verbs (verbs used with a subset of their full semantic features) with a large set of complements; this combination determines the predicate meaning of the expression. In this work we investigate the (semi-)productivity of light verb constructions which employ a predicative noun (a noun that has an argument structure) as their complement. We show that the productivity of these constructions depends on the semantic class of the complement. We develop three novel computational measures for quantifying the acceptability of candidate light verb constructions. Most of these measures meet or exceed the performance of an informed baseline, and reflect distinct trends in productivity along semantic classes and across light verbs. Good correlation and agreement with human judgments of construction acceptability are achieved.

Dedication

My friends, sharing this work with you has been a pleasure.



Acknowledgements

I must first thank my supervisor, Suzanne Stevenson, for her apparently infinite knowledge, encouragement and patience! This work would not be here today without her guidance and support. Special thanks are also due to Afsaneh Fazly, my co-author on an earlier paper which formed the basis of this research, who was always open to discussing new ideas. The contributions of Suzanne and Afsaneh to this work cannot be underestimated.

I am indebted to my second reader, Graeme Hirst, who was invaluable in his criticism and suggestions for improvement. His command of the delightful small details of English is enviable, and our discussions about this work were both helpful and thoroughly enjoyable.

Further thanks go to Eric Joanis for supplying the code for the lexical transducer used in this work, and to Ben Bartlett, Afsaneh Fazly, Robert Swier and Vivian Tsang for their helpful comments in proofreading earlier drafts of this paper. Their suggestions and improvements have strengthened this thesis. The entire Computational Linguistics group at the University of Toronto has been very welcoming, and the working environment formed here by these talented people is nothing if not supportive, vibrant, and challenging.

I want to thank my family and particularly my parents for their encouragement and support, and thank my friends here in Toronto, Ottawa, and worldwide, for all the same reasons.

Contents

1 Introduction **1**

1.1 Background 1

1.2 Statement of Purpose 3

1.3 Outline of Study 4

2 Related Work **6**

2.1 Introduction 6

2.2 Light Verb Constructions 7

2.2.1 Light Verbs 7

2.2.2 Light Verb Constructions 9

2.2.3 Computational Work on LVCs 12

2.3 Statistical Measures of Collocation 16

2.3.1 Collocations 16

2.3.2 Pointwise Mutual Information 16

2.4 The Web as a Corpus 19

2.4.1 Introduction 19

2.4.2 Exploiting Web Data 20

2.4.3 Exploiting Search Engines 21

3 Computational Models Of LVC Acceptability **27**

3.1 Generalization across Semantic Classes 28

3.1.1 Motivation of Approach . . . 28
3.1.2 Sources Employed in Acquiring Semantic Classes . . . 30
3.2 Acceptability Measures . . . 31
3.2.1 The PMI Baseline . . . 31
3.2.2 The LVC-PMI Measure . . . 32
3.2.3 The LVC-Prob Measure . . . 34
3.2.4 The LVC-Freq Measure . . . 35
3.2.5 Summary of Acceptability Measures . . . 37

4 Materials and Methods **38**
4.1 Selection of Levin Verb Classes . . . 39
4.2 Generation of WordNet Classes . . . 39
4.2.1 Generating a Set of Related Words . . . 40
4.2.2 Choosing a Representative Seed Word . . . 42
4.3 Extracting Data from the Web . . . 45
4.3.1 The Web as a Corpus . . . 45
4.3.2 Collecting Data . . . 46
4.3.3 Removing Noise . . . 51
4.4 Statistical Measures of Association . . . 55
4.4.1 Spearman Rank Correlation . . . 56
4.4.2 The Kappa Statistic . . . 57

5 Experimental Results **61**
5.1 Human Acceptability Judgments . . . 62
5.2 Computational Measures of LVC Acceptability . . . 64
5.2.1 Correlation Between Computational and Human Ratings . . . 64
5.2.2 Agreement Between Computational and Human Ratings . . . 71
5.2.3 Analysis of Filtering Techniques . . . 73

5.2.4 Summary of Experimental Results 76

6 Conclusions **79**

6.1 Summary of Contributions 80

6.2 Limitations and Future Directions 81

A Semantic Classes and Human Acceptability Judgments **84**

B Agreement Scores by Semantic Class **89**

C Using Google as a Corpus **92**

Bibliography **94**

List of Tables

2.1 Examples of Light and Vague Action Verbs. 11

4.1 Levin classes chosen for development and test data. A '*' indicates a random subset of verbs was used in the class. 40

4.2 Trends identified for each light verb and class. A '*' indicates a random subset of verbs from that class were used. 44

4.3 Searches necessary for the LVC-PMI measure employed with the light verb *take* and verb *walk*, with no wildcards. 48

4.4 The average number of searches necessary for a non-dative candidate LVC. . . 50

4.5 Illustrative example data for use with Spearman Rank Correlation. 57

4.6 Thresholds established for placing the output of each computational measure into 'good', 'fair', and 'poor' buckets. 59

4.7 Two related 2×2 contingency tables. Both have observed agreement of .85. The left-hand table has $\kappa = .70$ while the right-hand table has $\kappa = .32$. 59

5.1 The proportion of constructions rated 'fair' or above in Levin test classes. A '*' indicates a random subset of verbs were used in the class. 63

5.2 The proportion of constructions rated 'fair' or above in WordNet test classes. . 63

5.3 Spearman Rank Correlation scores for each computational measure when tested against candidate constructions drawn from Levin classes. 65

5.4 Spearman Rank Correlation scores for each computational measure when tested against candidate constructions drawn from WordNet classes. 65

5.5 Spearman Rank Correlation scores for each computational measure across each light verb and Levin class. A '*' indicates a random subset of verbs were used in the class. 68

5.6 Spearman Rank Correlation scores for each computational measure across each light verb and WordNet class. 69

5.7 Weighted Kappa (κ_w) and weighted observed agreement (p_w) for each computational measure when tested against candidate constructions from Levin classes. 72

5.8 Weighted Kappa (κ_w) and weighted observed agreement (p_w) for each computational measure when tested against candidate constructions from WordNet classes. 72

5.9 Two 3×3 contingency tables for the Levin class 43.2 with the light verb *take*. The leftmost table reflects the PMI measure, and has a weighted Kappa score of .66. The rightmost table reflects the LVC-Prob measure, and has a $\kappa_w = -.03$. Both have high observed agreement of $\geq .94$. 73

5.10 Spearman Rank Correlation results using LVC-Prob with no filtering is applied, with Punctuation Filtering applied, and with Punctuation, Phrase, and MWE Filtering all applied. A '*' indicates a random subset of verbs were used in the class. Classes ending with '-wn' are generated with WordNet. 74

B.1 Measures of agreement for each computational measure across each light verb and Levin class. Observed agreement is measured by p_o and weighted observed agreement by p_w. 90

B.2 Measures of agreement for each computational measure across each light verb and WordNet class. Observed agreement is measured by p_o and weighted observed agreement by p_w. 91

Chapter 1

Introduction

1.1 Background

Recent work in computational linguistics has begun to address the problems presented by multiword expressions, or MWEs (Bannard et al., 2003; Sag et al., 2002). MWEs are a large class of constructions, loosely defined by Sag et al. (2002) as "idiosyncratic interpretations that cross word boundaries". MWEs include fixed and semi-fixed expressions such as *by and large* and *kick the bucket*, as well as more productive and syntactically-flexible expressions, such as light verb constructions. An unsolved problem posed by MWEs is whether or not (and how) they should be listed in a computational lexicon. As many MWEs are syntactically flexible, simple string-based storage methods are inappropriate. Further, fully compositional storage techniques lead to overgeneralization, as many classes of MWE are only semi-productive. Solving the problems presented by MWEs is "key... for the development of large-scale, linguistically-sound natural language processing technology" (Sag et al., 2002).

Our focus in this thesis is on light verb constructions (LVCs), a largely compositional and semi-productive class of multiword expression found cross-linguistically (Karimi, 1997; Miyamoto, 2000; Butt, 2003; Folli et al., 2003). Unlike some classes of MWEs such as verb-particle constructions (McCarthy et al., 2003; Baldwin and Villavicencio, 2002; Villavicencio

and Copestake, 2002), LVCs have not yet been explored computationally in great detail. LVCs combine a member of a restricted set of "light verbs" with one of a more open set of complements. Light verbs are verbs which are employed with a subset of their full semantic features (Butt, 2003); English light verbs include *take*, *give*, and *make*, among others. The sentences listed in (a–c) are examples of LVCs formed using these three light verbs.

a. Priya *took a walk* along the beach.

b. Allene *gave a smile* when she saw us.

c. Randy *made a joke* to his friends.

Most of the meaning of a (non-idiomatic) LVC comes from the complement of the construction. As shown below, the complement of the light verb employed in (a–c) contributes the main verb of the corresponding paraphrase listed in (d–f):

d. Priya *walked* along the beach.

e. Allene *smiled* when she saw us.

f. Randy *joked* to his friends.

The complements used in (a-c) (*walk*, *smile*, and *joke*) are all examples of *predicative nouns*: nouns that have an argument structure. Light verb constructions formed with predicative noun (PN) complements are called LV+PN constructions, and are the specific class of LVC focused upon in this research. These constructions are of interest because their productivity appears to be patterned. For example, the phrases *take a walk*, *take a stroll*, and *take a run* are all acceptable LVCs, and their complements all describe different manners of motion. Conversely, the phrases **take a groan*, **take a smile*, and **take a wink* are all unacceptable constructions, and their complements all describe methods of non-verbal expression.

As has been suggested, the lexical storage of LV+PN light verb constructions is an unsolved problem. Storing every possible LVC is inefficient and further has no predictive power,

as it fails to take advantage of the (semi-)productivity of the expression. However, storing generation rules for LVCs based on the semantic class of the complement may lead to over-generalization. The extent of the semi-productivity this class of LVCs exhibits must first be determined, so that an appropriate lexical storage method can be determined. Computational measures to quantify the acceptability of a candidate light verb construction would be useful in the development of an LVC lexicon, as they would allow quick and nuanced judgments of class productivity and acceptability.

1.2 Statement of Purpose

The aim of this work is to examine the (semi-)productivity of LV+PN light verb constructions and to develop computational measures for quantifying their acceptability.

We hypothesize that the semi-productivity of LVCs depends on the semantic class of its complement, and test this hypothesis by examining trends in productivity across semantic classes. Candidate LVCs are formed by combining each member of a class with three chosen light verbs—*take*, *give*, and *make*. The lexical semantic verb classes of Levin (1993) are used as a source for PN groupings, along with classes automatically extracted from the nominal and verbal hierarchies of WordNet 2.0 (Fellbaum, 1988). We hypothesize that these semantic classes will make nuanced-enough semantic distinctions to support generalization of LVC acceptability trends at the class level, and across the light verbs themselves.

Four diverse computational measures are developed to quantify the acceptability of candidate light verb constructions. These measures are pointwise mutual information (PMI), which is used as an informed baseline; LVC-PMI, a version of PMI enhanced with linguistic knowledge of LVCs; LVC-Prob, a probability formula which measures the likelihood of a given predicative noun and light verb forming an acceptable LVC; and LVC-Freq, a simpler measure which rates candidate constructions by how frequently they are attested in a corpus. Human acceptability judgments of each candidate LVC are gathered, and used as the standard against

which our computational measures are evaluated.

Since some acceptable LVCs are rare in smaller classical corpora, the World Wide Web is employed as a corpus in this work. We access web information through a general-purpose search engine, and develop three different techniques to filter the noise found in this data.

1.3 Outline of Study

The remainder of this work is divided into five chapters.

Chapter 2: Related Work introduces existing linguistic theory of light verbs and light verb constructions which is drawn upon in this thesis, and describes previous computational work on LVCs. As some of the models of LVC acceptability presented in this thesis rely on pointwise mutual information, an examination of this statistical measure of collocation is included. We also review the issues associated with using general-purpose search engines in linguistic research.

Chapter 3: Computational Models of LVC Acceptability introduces our approach of generalizing over semantic classes of complements, and briefly describes the two sources from which we extract these classes: Levin (1993) and WordNet 2.0. Our four acceptability measures are presented: the PMI baseline, the linguistically informed LVC-PMI measure, the LVC-Prob probability formula, and the simple LVC-Freq acceptability formula.

Chapter 4: Materials and Methods describes the realization of the models presented in Chapter 3. We present the Levin classes employed in this work, and describe our method for generating semantic classes from WordNet. Our technique of extracting information from the web is detailed, and methods developed to remove noise from this data specified. The statistical measures of association used to gauge the performance of our measures are also considered.

Chapter 5: Experimental Results describes the human acceptability judgments used as the standard in our experiments. Scores of correlation and agreement between these and each of our measures are presented, and performance at both a fine- and coarse-grained level of acceptability is considered. An analysis of the utility of our noise-filtering techniques concludes this chapter.

Chapter 6: Conclusions summarizes the contributions of this work and outlines its limitations. Possible future extensions of this research are also presented.

Chapter 2

Related Work

2.1 Introduction

There are three strands of research drawn upon in this work: the linguistic and computational analysis of LVCs, existing statistical measures of collocation, and past approaches to using the World Wide Web as a corpus.

The linguistics community has enjoyed a relatively long history of interest in multiword expressions (e.g., Bolinger, 1971), and there is a well-developed body of research examining LVCs in several languages (Karimi, 1997; Miyamoto, 2000; Butt, 2003). In contrast, in the computational linguistics community the study of MWEs is only a relatively recent trend, and while some MWEs (such as verb-particle constructions) have been studied in significant detail (Baldwin and Villavicencio, 2002; Bannard et al., 2003; Villavicencio, 2003), the study of light verb constructions has been, in comparison, largely overlooked. Nevertheless, there is a small body of computational research into LVCs available, including Sag et al. (2002), Grefenstette and Teufel (1995), and Dras and Johnson (1996).

In many cases, computational work on MWEs has used statistical measures of collocation in order to measure the mutual association between words. Statistical techniques such as point-wise mutual information (Church and Hanks, 1989; Church et al., 1991) have become standard

in measuring how well given elements of an MWE predict each other. Such measures are typically interpreted as providing evidence of a meaningful linguistic association. As they rely upon information extracted from corpora, statistical measures of collocation tend to perform optimally when a large body of evidence is available (Terra and Clarke, 2003).

Unfortunately, some LVCs, like many MWEs, appear relatively infrequently in classical corpora. The web, with its enormous size, can be thought of as a corpus with the potential to offer evidence of even very rare linguistic phenomena. Despite the inherent issues of noise, it is increasingly recognized that the unique properties (and challenges) the web presents can lead to approaches and opportunities unavailable with classical corpora (Kilgarriff, 2001; Turney and Littman, 2002; Kilgarriff and Grefenstette, 2004). Much work has focused on discovering the properties of web text, and on designing better techniques for web-based information extraction (Florescu et al., 1998; Lin, 2002).

In the next section, linguistic and computational linguistic studies of the properties of light verbs and light verb constructions are explored. In the section following, statistical measures of capturing collocations are studied, and in the final section, existing computational approaches to employing the web as a corpus are reviewed.

2.2 Light Verb Constructions

2.2.1 Light Verbs

In order to examine light verb constructions we must first consider the nature of light verbs. A light verb is a verb (usually a frequent verb with a very general meaning) which, when combined with certain complements, loses some of its normal semantics: such usages are said to be *semantically bleached* (Butt and Geuder, 2001). Light verbs are a cross-linguistic phenomenon, and are found in languages such as Persian, Urdu, and Japanese (Karimi, 1997; Butt, 2003; Miyamoto, 2000). Light verbs in English include *give*, *take*, *make*, *do*, and *have*, among others.

Individual light verbs can have semantics which range over a spectrum of meaning. Consider the following examples drawn from Butt and Geuder (2001), all of which employ the verb *give* at varying degrees of "lightness":

a. I gave the book to Emily.

b. I gave some advice to Emily.

c. I gave a kiss to Emily.

We can see how the meaning of the verb *give* ranges from its "heavy" literal usage in the first example (where something physical has been put in the recipient's possession), to more abstract in the second example (where something non-physical is transferred, but not possessed), to most abstract in the final example, where nothing is transferred, and nothing is possessed as a result of the action. A similar spectrum of semantic "lightness" is exhibited by all light verbs, indicating that rather than being a binary distinction, there are degrees of "heavy" and "light" usage of light verbs. The light verbs found in LVCs tend toward lighter, bleached semantics in most cases, but it can be unclear—especially with the light verb *make*—just how "light" a light verb is.[1]

Butt and Geuder (2001) note that the above examples employing *give* all share a vague sense of emission, which may explain why one can say *I gave the ball a good throw*, but not **I gave the ball a good catch.* However, *give* can also have a permissive quality in some LVCs, as in *She gave them a look into the room* (Butt and Geuder, 2001). It seems clear that light verbs can contribute different elements of their semantics to different LVCs.

Kearns (2002) argues that there are some cases where light verbs do *not* contribute semantic information to the LVC, citing the specific example of *give the roses a prune*. In this case she

[1]Consider the example *The child made a spill on the carpet* (i.e.: something was spilt onto the carpet). The light verb *make* seems to have a heavier usage here, especially when compared with similar constructions such as *The child took a spill on the carpet* (i.e.: the child fell over). While it can be difficult to compare degrees of "lightness" across light verbs, the fact that something physical is "made" in the first example suggests that *make* is being used with relatively heavier semantics.

argues that the light verb *give*, rather than adding semantic content, instead forces a specific meaning structure to be created. Although it is clear there is a semantic difference between *give the roses a prune* and *prune the roses*, it is largely irrelevant to this work whether the difference stems from semantic information contributed by the light verb, or rather is some artifact of the light verb's presence in the construction. In either case, the light verb and complement together determine the predicate's semantics.

2.2.2 Light Verb Constructions

Light verb constructions are those which combine a restricted set of light verbs with a large set of complements to determine the meaning of the predicate. We restrict our examination to light verb constructions employing NP complements; examples include the constructions *take a walk* and *give some advice*. These are referred to as LV+NP light verb constructions. Such LVCs comprise a heterogenous class of constructions, but also exhibit a number of important common properties. First, as the examples below illustrate, most of the distinctive meaning of a non-idiomatic LVC comes from the complement to the light verb, and not from the light verb itself.[2]

a. Everyone but Joey *took a ride* in the hot-air balloon.

b. Jeffrey would often *give a tour* of his house at the slightest provocation.

c. Please *give the show a try* before you change the channel.

d. Anna believes she *made a favourable trade* with her friend.

Secondly, LVCs are a semi-productive class of expressions: for example, in English one can *take a walk*, *take a stroll*, *take a run*, and so on. LVCs are even more productive in other languages: Khanlari (1973) notes that in contemporary Persian, LVCs have evolved to replace

[2]Since we are interested in productivity, we focus on non-idiomatic LVCs, as their behaviour is more regular and their semantics less idiosyncratic. Idiomatic LVCs generally tend not to be productive.

simple verbs, leaving only a handful of simple verbs in use. This (semi-)productivity raises the familiar problem of whether or not (and how) to store LVCs in a computational lexicon (Sag et al., 2002).

However, LVCs are not a uniform class of constructions, contrary to what is generally assumed in existing computational work on these constructions (Grefenstette and Teufel, 1995; Dras and Johnson, 1996; Sag et al., 2002). Rather, LVCs contain several distinct subclasses. Kearns (2002) and others divide the LV+NP class of LVCs we are interested in into two main subclasses, separating those in the LV+PN format from the remainder of the class. LV+PN constructions, such as the phrase *take a stroll*, are distinguished by employing a predicative noun[3] (PN) as their complement. While this predicative noun may superficially seem to be a noun object of the light verb, in many ways it is behaving as a verbal element and shares both nominal and verbal properties (Wierzbicka, 1982; Butt, 2003).

LV+PN Light Verb Constructions

LV+PN constructions have been the subject of detailed linguistic examination. Wierzbicka (1982) performs a comprehensive semantic analysis of one group of LVCs: those in the *have a V* format. Many of the complements employed by Wierzbicka (1982) would be labelled in this work as predicative nouns. She creates and examines 10 different subclasses of the construction and builds for each what she calls a "semantic formula" which begins to account for both the nuances in productivity and the differences in acceptability of individual LVCs.[4] These formulae offer some semantically-based predictive ability when constructing new LVCs, but unfortunately, as they are not rooted in any formal semantic system, their computational application is difficult. Nevertheless, the 10 different subclasses of *have a V* constructions which Wierzbicka identifies and justifies show that the complexity of LVCs is not idiosyncratic,

[3]The labelling of this complement varies in the literature, but as its salient property is that it features an argument structure, we will refer to it here as a *predicative noun*.

[4]Examples of Wierzbicka's subclasses include "Aimless Objectless Individual Activity Which Could Cause One To Feel Good" (which includes *have a walk*) and "Tentative Action Which Could Cause One To Come To Know Something and Which Would Not Cause One To Feel Bad If It Didn't" (which includes *have a try*).

Construction	**True Light Verb (TLV)**	**Vague Action Verb (VAV)**
Indefinite Det.	Ro gave a groan just now.	Ro gave a demonstration today.
Definite Det.	* Ro gave the groan just now.	Ro gave the demonstration today.
Passive	* A groan was given by Ro.	A demonstration was given by Ro.
Wh-Movement	* Which groan was given by Ro?	Which demonstration was given by Ro?

Table 2.1: Examples of Light and Vague Action Verbs.

but rather, patterned.

Some of the general semantic properties that Wierzbicka identifies apply to the larger class of LV+PN light verb constructions. For instance, Wierzbicka notes that *have* LVCs are highly colloquial, and this applies to other light verbs more generally: one can *take a pee* but not **take a urination*. Like those in the *have a V* format, members of the full class of LV+PN constructions generally denote events which are brief (one can *take a walk* but cannot usually *?take a walk all day*), less goal-oriented (compare *give a knock on the door* with *knock on the door*), and repeatable (one cannot *take a bite* of a sandwich if one intends for the bite to consume it entirely) (Wierzbicka, 1982).

We adopt the terminology of Kearns (2002), who distinguishes LV+PN constructions by dividing them into the categories of True Light Verbs (TLVs) and Vague Action Verbs (VAVs). TLVs are defined as requiring an indefinite complement headed by a noun in a stem form identical to a verb; further, the NP complements of TLVs cannot undergo *wh*-movement. The complement of a TLV cannot be modified by a relative clause and cannot be pronominalized. In contrast, VAVs allow these properties, and do not require their complement have a verbal stem form. The light verb in VAV constructions additionally tends to make a more significant semantic contribution than in TLVs. Examples of TLV and VAV constructions are shown in Table 2.1, which is based on Stevenson et al. (2004). TLVs are the specific focus of this research.

While the division of LVCs into the two classes of TLVs and VAVs is useful, it is quite

categorical, and there exist constructions which do not fit comfortably in either category. For example, an LVC such as *I took the walk that was recommended* has all the properties of a TLV, except that it employs a definite article and is modified by a relative clause. Phrases such as these suggest that the classification of LV+PN light verb constructions into TLVs and VAVs is not a binary decision: rather, TLVs and VAVs represent two points in a continuum of LVCs. Even with these caveats, the distinction between TLV and VAV constructions is an informative one, as it distinguishes properties of LVCs which had before been blurred.

It is clear that LVCs have both complexity and nuance. The linguistic literature exploring the construction is well-developed; however, computational studies of LVCs are rare. This is perhaps due to the fact that, like many MWEs, the magnitude of LVC usage has only relatively recently become recognized in the field (Jackendoff, 1997). In the next section, we consider existing computational research into LVCs.

2.2.3 Computational Work on LVCs

While it is true that MWEs in general and LVCs in particular are a recent area of interest in the field of computational linguistics, there has been some previous work on LVCs. Sag et al. (2002) consider LVCs to be syntactically-flexible lexicalized phrases, and are interested in storing these constructions in a lexicon. The suggestion is made that all nouns which can combine with a given light verb have this information stored in their lexical entries via a semantic relation. Thus, for the light verb *make*, there would exist a semantic relation, $\mathrm{make\_arg\_relation}$, which would then be associated with each noun *make* can legitimately combine with (Sag et al., 2002). This approach requires a significant amount of storage, and offers no predictive power as it fails to exploit the (semi-)productivity of LVCs. For example, after encountering *give a sigh* and *give a groan*, the system would have no preconception as to the validity of *give a laugh*. Furthermore, this approach reduces the validity of every LVC to a binary decision: either a light verb is valid with a given complement, in which case the semantic relation is stored, or it is invalid, and the lexical entry is left unaltered. Our own experiments in rating

LVCs, described in Section 5.1, show that there is a continuum of acceptability for LVCs, and further, that the acceptability of a given construction can vary from person to person.

In this work we are interested in the acceptability of light verb constructions across a closed set of light verbs and a more open set of predicative nouns, but Grefenstette and Teufel (1995) take what is in some ways the opposite approach. Rather than trying to measure how good a predicative noun is with a candidate light verb, the problem of choosing which "support verb" is best for a given "deverbal noun" is considered. Support verbs (SVs) are a class of semantically impoverished verbs used for syntactic support: this class includes light verbs, but also more idiosyncratic and collocationally restricted verbs such as *sustain* (used in a supporting role in the phrase *sustain injury*), *lodge* (*lodge a complaint*), and *commit* (*commit murder*) (Fillmore et al., 2002). Deverbal nouns (DNs) are a subclass of predicative nouns which are derived from a corresponding verb. For example, the deverbal noun *proposal* is derived from the verb *propose*.

As deverbal nouns used in SV constructions have argument structure, Grefenstette and Teufel (1995) determine potential SV constructions by assuming that uses of a deverbal noun which are similar to the uses of the corresponding verb are likely to be occurring in a support verb construction. Similarity of context is determined by occurrence with common PP arguments. Grefenstette and Teufel (1995) first parse their corpus and extract the top three prepositions that follow the verb corresponding to each DN. Then, for each DN, they extract NPs headed by that DN and followed by a PP employing one of these top three extracted prepositions. For each occurrence of one of these extracted "NP PP" phrases in direct object position of a verb, the verb is considered a potential support verb, and these verbs are ranked by frequency. The most frequent verb on this list is chosen as the "best" support verb. Plausible choices (where plausibility depends on researcher intuition) are achieved seven out of ten times,[5] and the light verbs *make* and *have* are chosen 50% of the time.

[5]This estimate of plausibility comes from a review of Grefenstette and Teufel (1995) presented in Dras and Johnson (1996); Grefenstette and Teufel (1995) do not explicitly make these judgements.

While this approach gathers some interesting results, it has several flaws. The small class of English support verbs is left open, which allows the process to select both *reject* and *make* (the two are tied) as the best choice for the deverbal noun *suggestion*. Grefenstette and Teufel (1995) concede that the verb *reject* is not semantically impoverished. Additionally, frequent collocations which employ a DN as their head word can skew the choice of support verb. For example, the deverbal noun *order* was expected by the authors to have the support verb *give* chosen for it, but instead, *issue* is selected. As the news corpus employed by Grefenstette and Teufel (1995) includes many more instances of "restraining orders" (which are typically issued) than it does of "orders" (which are typically given), and as the word *order* and the collocated phrase *restraining order* are undifferentiated in their extraction process, the "wrong" support verb is chosen (Grefenstette and Teufel, 1995).

More critically, the idea of there being a single "right" support verb seems invalid. To *have a look* at something and to *take a look* at the same thing have quite similar semantics, the choice depending more on local dialect than anything else (Wierzbicka, 1982; Kearns, 2002). Worse, one can both *take a run* (around a field) and *make a run* (while playing sports, or "for the border"), and here the choice of light verb dramatically changes the semantics of the construction. While the process of Grefenstette and Teufel (1995) may sometimes determine the most common light verb used with a given deverbal noun, this does not imply that other less common uses are invalid. Finally, this process requires a corpus which can be parsed. This can be difficult, though not impossible, when using very large alternative corpora such as the web.

Dras and Johnson (1996), like Grefenstette and Teufel (1995) also attempt to determine which support verb is best for a given nominalization, and achieve results slightly more promising than those of Grefenstette and Teufel (1995). They consider more global information about candidate support verbs by including each candidate's corpus frequency in their model: this serves as a rough approximation of the frequency with which the candidate SV is used in a supporting capacity. The strong assumption is made that all verbs governing nominalizations

are acting as support verbs, and the formula employed is:

$$\mathrm{SV}(n) = \arg\max_{v \in V} m_{vn} \sum_{n} m_{vn} \tag{2.1}$$

where $\mathrm{SV}(n)$ is the most likely support verb for nominalization n, and m_{vn} is the number of times verb v has nominalization n as a complement.

This formula captures the intuition that (productive) light verbs are more ubiquitous than verbs which are unable to act in a light capacity, and favours these verbs. Eighteen example LVCs selected from the linguistic literature (such as *have a snooze*) are used as a test set, and the system is run with the nominal element (i.e., *snooze*) as input. The top two choices the system generates for a given complement are compared to the result that is expected for the construction. A match with either result is considered a success, and these (relatively relaxed) criteria generate a match 14 of 18 times, for success rate of 77%.[6] Of all choices made, the light verbs *make* and *have* appear as the first choice 72% of the time, which, when compared with the 50% rate of selection of these verbs in Grefenstette and Teufel (1995), suggests that more productive support verbs are being chosen using Dras and Johnson's method. However, it is very difficult to generalize from just this handful of test cases. The same concerns regarding the assumption of a single "correct" support verb that apply to Grefenstette and Teufel (1995) also apply here.

The computational work on light verb constructions thus far tends to oversimplify the problem, blurring classes of light verbs together and assuming that one "correct" light verb for a given complement can be chosen from an open set. More-appropriate measures are called for—measures which can recognize multiple light verbs as valid with a particular complement. Further, any such measures should be tested on more than a handful of examples, so that meaningful conclusions can be drawn.

[6] We count the three occasions in which sparse data prevents the measure from producing two candidate support verbs; Dras and Johnson do not, and measure their own success at 14/15, or 93%.

2.3 Statistical Measures of Collocation

We now examine existing statistical measures of collocation and word association, which will be the foundation of much of the statistically-based research into LVCs to follow in this thesis.

2.3.1 Collocations

There exists a large body of work concerning statistical collocation analysis: the quantification of how often given words occur with one another. As these statistical techniques are often interpreted as measuring a relationship between words, they have been used in many natural language tasks, such as the automatic creation of thesauri (Lin, 1998a) and synonym analysis and selection (Turney, 2001), among others. While there are many statistical measures (and variations thereof) which can be applied to a given linguistic problem (Terra and Clarke, 2003), pointwise mutual information (PMI) has been standardly employed in situations wherein one wishes to measure the association between two words (Lin, 1998a, 1999; Terra and Clarke, 2003; Turney and Littman, 2003). As a high mutual association between words is characteristic of several types of MWEs, including LVCs (Lin, 1999), we now explore the properties of PMI and examine how it has previously been adapted to various linguistic tasks.

2.3.2 Pointwise Mutual Information

PMI can be motivated by the observation of Firth (1957) that "You shall know a word by the company it keeps." PMI is defined in Church and Hanks (1989) as

$$\mathrm{PMI}(word_1, word_2) = \log_2 \left(\frac{\Pr(word_1, word_2)}{\Pr(word_1)\Pr(word_2)} \right) \tag{2.2}$$

PMI thus compares the probability of observing $word_1$ and $word_2$ together to the probability of observing $word_1$ and $word_2$ by chance. If $word_1$ and $word_2$ often do appear together, then $\mathrm{PMI}(word_1, word_2)$ will be > 0; however if they share no special relationship, then $\mathrm{PMI}(word_1, word_2)$ is expected to be 0. If $word_i$ usually appears only when its complement

word under consideration does not, then $\mathrm{PMI}(word_1, word_2)$ will be < 0. PMI thus defines a continuum of association. As written above, PMI applies only to two words, but it has been extended to three words or more (Alshawi and Carter, 1994).

One must be sure to categorize the nature of this associative relationship carefully. PMI measures the ability of the words in question to predict one another: no meaningful linguistic relationship is necessarily implied (Manning and Schüetze, 1999). However, if one is searching for evidence of a linguistic association between words, a key assumption made is that a high PMI score reflects such an association (Terra and Clarke, 2003).

As noted by Manning and Schüetze (1999), Lin (1998b), Pantel and Lin (2002) and others, mutual information is particularly sensitive to low frequency counts. In such cases where evidence is low, the measure tends to overestimate rare cases for which it has observed some evidence, and underestimate rare cases for which it has not observed evidence. Because of this, Manning and Schüetze (1999) characterize PMI as being a good measure of independence (since PMI scores close to zero often do indicate independence between the words in question) but a poor measure of dependence, as low-frequency words may be rated inappropriately high and pollute the set of scores $\gg 0$. Attempts have been made to correct this undesirable behaviour. Lin (1998b) adjusts some of the frequency counts used in calculating the PMI measure by subtracting a constant c from his counts of seeing the words in question together. This allows PMI to rate rare events (those with low frequency counts) uniformly $\ll 0$, rather than rating them inconsistently. But this is an unsatisfying solution, as it could introduce new 'rare' events into the measure. More-complicated adjustments of PMI values are presented by Fontenelle et al. (1994) and Pantel and Lin (2002), in which discounting functions, designed to combat the skew effects of PMI, are applied to the measure.

This idea of skewing (or rather, de-skewing) the output of the PMI measure is largely abandoned when it is applied to large corpora, as low frequency counts become a less pressing issue. When the web is employed as a corpus, the enormous size of the material available ensures that the likelihood of having low frequency counts is significantly reduced. Techniques

employing a vanilla PMI measure on web data tend to outperform more complicated measures designed for smaller classical corpora (Turney, 2001; Dumais et al., 2002; Lin, 2002; Keller and Lapata, 2003; Baroni and Bernardini, 2004). Furthermore, Terra and Clarke (2003) test PMI along with several other statistical word-association measures such as χ^2 and the log-likelihood ratio (Manning and Schüetze, 1999), and find that on their task of synonym selection, unmodified PMI operating on web data outperforms all other measures. For these reasons, research using the web as a corpus tends to focus less on altering the output of the PMI measure itself, and more on ensuring the data being used with the measure is accurate and appropriate.

Focusing as it does solely on the frequencies of word tokens and ignoring any linguistic intuition, PMI was recognized early on as being "extremely superficial" (Church and Hanks, 1989). Different approaches have been taken to inform PMI with linguistic knowledge. PMI has been adapted to include words of context by Turney (2001), Pantel and Lin (2002) and others, which allows some unrelated senses of a word to be filtered from the measure. PMI has also been altered so that, rather than employing data from two words, it employs data gathered from clusters of related words (Turney and Littman, 2003). In this way, idiosyncracies intrinsic from using a single word's frequency information are smoothed out. Finally, Turney (2001) explores alternative ways of gathering frequency counts in an effort to capture a more linguistically-appropriate idea of 'co-occurrence'. For his task of synonym similarity, the numerator of the PMI formula, $P(word_1, word_2)$, is re-interpreted as the likelihood of seeing $word_1$ *near*[7] $word_2$. As words that occur near each other are more likely to be describing the same concept than words which appear further away, this increases the accuracy of his measure.[8] All these approaches attempt to provide more linguistic information to PMI, so that it can better reflect the linguistic phenomenon under consideration.

Despite its limitations, PMI has been shown to be good at identifying LVCs, even when

[7]The operator 'near' (x NEAR y) is defined by the AltaVista search engine, which Turney employs, to return results wherein x and y co-occur within 10 words.

[8]In one experiment, accuracy increases by 41% to 62%. It is somewhat surprising that this alteration makes as large a difference as it does, given the relatively small window of 10 words that the 'near' operator provides.

they were not specifically sought out. Lin (1999) uses PMI to identify non-compositional phrases in text, hypothesizing that such phrases will have a high mutual association between their component words. However, this process also identifies many compositional LVCs, such as *take a bite*, *take a turn*, and *make a splash*. As Bannard et al. (2003) point out, PMI is better thought of as tapping into productivity than non-compositionality, explaining why such compositional LVCs were identified.

As long as there is sufficient evidence of rare instances, PMI is an appropriate measure for capturing the association between words and elements of phrases. For a sufficiently large corpus we turn to the web, and in the next section, the results of analyzing the web as a corpus are explored.

2.4 The Web as a Corpus

2.4.1 Introduction

There is a growing recognition that the web is not only a corpus, but one with properties suited to linguistic research. Keller and Lapata (2003), using the search engines Google (www.google.com) and AltaVista (www.altavista.com), have shown not only high correlations between web frequencies and corpus frequencies, but also correlation between web frequencies and plausibility judgements. Kilgarriff (2001) has gone so far as to claim, "The corpus of the new millennium is the web." What makes the web unique is its size: as much as 60 terabytes of text data—over sixty trillion words—is available to be searched online.[9] It is not only one of the largest corpora available, but also one of the easiest to access, as it is freely accessible and no further away than a search engine request. As Turney (2001), Dumais et al. (2002), Lin (2002), Keller and Lapata (2003), Baroni and Vegnaduzzo (2004) and others demonstrate,

[9]This estimate is based on Google's own estimate of their index size at just over 8 billion pages, an estimate from Lawrence and Giles (1999) of 7.5 kilobytes of non-markup text per page of web data, and a (generous) estimate from Kilgarriff and Grefenstette (2004) of 10 bytes per word.

simple web-based approaches to linguistic problems can often outperform more complicated approaches relying on smaller, classical corpora. The web promises new opportunities in corpus linguistics, as long as the problems and limitations inherent in the corpus are dealt with.

In the next section, we explore different techniques of exploiting the web as a corpus. We then look at the specific properties of web data acquired via general-purpose search engines, and explore previous work using such web extraction techniques.

2.4.2 Exploiting Web Data

Many different approaches have been suggested for exploiting web data. Perhaps the simplest approach is to use existing general-purpose search engines, such as Google or AltaVista. Turney (2001), Villavicencio (2003) and others employ such search engines, and this approach is adopted in this thesis. The advantage this technique offers is that of making an enormous corpus available effectively 'for free', as all of the work of developing and maintaining an index of the web is done by the search engine organization. However, there are several disadvantages inherent in employing a general-purpose tool (a chosen search engine) for a very specific task (linguistic research). As Kilgarriff (2004) notes, general-purpose search engines are "linguistically dumb," having no linguistic annotation and making no distinction between parts of speech.

Baroni and Bernardini (2004) seek to work around these limitations by employing a search engine only to gather pages of interest, which are retrieved, stored, and used to construct a smaller private corpus of web data with a specific linguistic focus. Depending on how specific this focus is, the corpus can be thought of as "single-use" (Baroni and Bernardini, 2004). One step further is the work of Terra and Clarke (2003), who use no search engine at all, but rather privately crawl[10] the web and use the terabyte of general-purpose web data gathered as their

[10] 'Crawling' the web is the standard way of discovering web data. Generally, an index is built by initializing a recursive algorithm with a collection of seed webpages. The algorithm then visits every link on each webpage, adding any new pages encountered to its index, and proceeds in this manner until stopped or until no new pages are discovered.

corpus: building, in effect, a private search index. These related approaches have several advantages over simple web searching. As more manageable subsets of web data are stored locally, access is quick, and many properties of the corpus (its size, for instance) can be directly measured instead of estimated. Furthermore, since the corpus can be computationally labelled and parsed, searches based on standard linguistic criteria – a word's part of speech tag, for instance – can also be run. Searches can also use more complex and customized interfaces than what are available through commercial search engines.

But these approaches are not without flaw. While Baroni and Bernardini (2004) do work towards presenting a general-purpose web extraction toolkit, a researcher employing such a kit is still splitting his or her focus between the linguistic problem at hand and the maintenance of a corpus of web data. The computational resources for storing, maintaining, and searching large amounts of data are still required, but have been shifted onto the shoulders of the researcher, rather than being invisibly handled by some general-purpose search engine. Kilgarriff (2003), Resnik and Elkiss (2003) and others have moved towards solving these problems by proposing large coverage, publicly accessible linguistic search engines: engines which are designed for linguists by linguists. While these efforts have resulted in some exciting early results, a stable and computationally accessible linguistic search engine with an index comparable to those of the large general-purpose search engines has not yet been developed. In their absence, we focus on using existing general-purpose search engines, and explore the issues associated with their use.

2.4.3 Exploiting Search Engines

Major general-purpose search engines (henceforth referred to as "search engines" for simplicity) have previously been used in the study of linguistic phenomena in general (Turney, 2001), and MWEs in particular (Villavicencio, 2003). Such research has had to deal with the noisy nature of web data, especially when compared to hand-built classical corpora such as the British National Corpus (BNC Reference Guide, 2000). Typographical errors, ungrammatical text,

and even results in other languages can all contribute noise which must be controlled for if accurate results are to be achieved. Worse, as researchers are constrained by a search interface not designed with linguistics in mind, it is often non-trivial to filter out results unrelated to the specific type of linguistic phenomenon in question. This leads to another source of noise in the data: that of the "false hit." As we will see, Villavicencio (2003) and Turney (2001) both use specially-designed searches in an attempt to minimize this sort of noise.

Villavicencio (2003) uses search engines in her exploration of the class-based behaviour of verb-particle constructions. Briefly, verb-particle constructions (VPCs) are MWEs that combine a wide range of verbs with a member from a small set of particles, such as *up, on, down, along,* etc.[11] Like LVCs, VPCs are semi-productive. The semantics of VPCs can range from being closely related in meaning to the component verb and/or particle, to being completely semantically distinct. Examples of VPCs with various levels of compositionality include *put up*, *look up*, *look out*, and *make out*.

Villavicencio automatically generates candidate VPCs by appending the particle *up* to each of the 3,100 verbs listed in Levin (1993), and examines trends of validity across classes. In order to determine if a candidate construction is valid, she searches a collection of classical corpora, which includes VPC dictionaries such as the Sinclair and Moon (1989) and the McCarthy and Walter (1997), along with a corpus of VPCs extracted from the BNC. If a candidate construction is not attested in these corpora, she attempts to validate it by searching for it on the web: if any results are found, she assumes the candidate is a valid verb-particle construction.

However, a search for the phrase "VERB up" can return results in which the verb and particle are not used in a VPC. When an NP follows the phrase "VERB up", the construction could either be a VPC (*run up a bill*) or a prepositional verb (*run up a hill*), and without linguistic markup, the two cases are difficult to distinguish. Therefore instead of searching for the phrase "VERB up", Villavicencio searches for the phrase "VERB up for", and thus

[11] VPCs are also referred to in linguistic literature as "phrasal verbs", "compound verbs", and "discontinuous verbs", among other labels.

excludes many (though not all) instances of this ambiguous case.[12] When the web-verified VPCs are combined with the existing dictionaries, the number of VPCs increases by 21%, for a total of about twelve thousand verb-particle constructions. Further, class-based behaviour is apparent: in some classes, such as 11.3 (Verbs of Bring and Take), it is found that all members form valid combinations with the particle *up*, while in others, such as 39.7 (Advice Verbs), no valid combinations are found. Levin (1993) is suggested as a good starting point for observing patterns in productivity in verb-particle constructions.

There are however several issues with this approach. Most critical is the "failure-to-find fallacy," as defined by Church et al. (1991): "...when you don't have much evidence for something, it is very hard to know whether it is because it doesn't happen, or because you haven't been looking for it in the right way (or in the right place)." A VPC that is not found by Villavicencio's searches is marked as invalid, which might not be the case: the VPC might never be employed in the corpus with *for* after it, or it might simply not appear in the subsection of the web that is indexed by Google.

Additionally, the assumption that the validity of a given VPC is a binary decision also seems suspect. Like LVCs, VPCs seem to exhibit degrees of acceptability: some very common VPCs such as *cook up* are acceptable to most, while others such as *?sauté up* are less universally accepted, but still apparent online (Villavicencio, 2003; Villavicencio and Copestake, 2002). One can read Villavicencio (2003) as applying a threshold to this continuum of acceptability: if a VPC appears in Google's index at least once, it is considered valid. However, it is not clear that this (somewhat arbitrary) way of looking at the data is useful, and it is possible more meaningful thresholds could be determined and justified.

Turney (2001) also uses the web as a corpus, and for similar reasons: traditional resources (in this case, WordNet (Fellbaum, 1988)) were found to be lacking. Turney's task is slightly different: given a problem word p (such as *levied*), a set of four alternative words $Alternates(p)$

[12] Not all instances are removed because non-VPC phrases incorporating *wh*-movement, such as *Which hill did you run up for your mother?*, will still be returned by these searches.

(such as *imposed*, *believed*, *requested*, *correlated*), and, sometimes, an example gloss of the problem word in a sentence, his system is to chose the alternative word s most similar in meaning to the problem word. The assumption is made that two words close in meaning are more likely to co-occur frequently than two words with more distant semantics—i.e., that p will co-occur more often with s than with any other alternate. A more sophisticated approach to using the web is explored: rather than searching to see if a given candidate is attested or not, as Villavicencio (2003) does, he instead uses PMI to quantify the association of each $a \in Alternates(p)$ to the problem word p, and chooses the a with the strongest association.

Different techniques of searching the web are considered in order to gather more accurate data. The most straightforward method of gathering the information required for PMI is to simply determine the number of documents in which a given a and p co-occur. For each alternate word, Turney extracts from AltaVista the number of documents in which both the problem word and the alternative word are seen, using the $x \,\mathrm{AND}\, y$ operator. (He also gathers, as is necessary for PMI, the number of documents in which the alternative word is seen alone.[13])

AltaVista is somewhat unique in supporting an $x \,\mathrm{NEAR}\, y$ operator,[14] which returns links to documents wherein the words x and y co-occur within 10 words. Turney refines his initial approach by employing this operator instead of AND, and finds accuracy increases from 48% to 62%. In practice, both these scores tend to rank antonyms as highly as synonyms, so he extends the search by adding the rough qualifier "$(\mathrm{AND\ NOT}\ (p\ \mathrm{OR}\ a)\ \mathrm{NEAR}$ "not")" to his searches. This increases accuracy by another 4%, to 66%. Adding to the search some context words to the search extracted from the gloss (if available) brings accuracy to 74%.

These different techniques of searching AltaVista are each aimed at solving the problem of noise caused by false hits. Constrained, as he is, by the restrictions of AltaVista—which

[13] Since Turney (2001) is interested only in comparing the scores of each $a \in Alternates(p)$ to each other, and never across different problem words, the number of times the problem word p is observed is dropped from the PMI equation.

[14] AltaVista has supported this operator in the past, but revisions to its search engine technology have, at times, removed it. As of this writing, the NEAR operator is once again supported. This illustrates a major concern with relying on a third-party search engine: one has no control over the corpus or the way it is accessed.

provides little context for results—Turney must ensure that his queries remove as many of the unwanted results as possible.[15] Each successive iteration of his search technique is targeted at a certain kind of false hit, and with this noise removed, accuracy increases.

Turney and Littman (2002, 2003) continue work in this vein; here, PMI is used to measure semantic orientation of a given problem word w. Semantic orientation captures the degree to which a word is thought of as being 'good' or 'bad': *excellent* and *poor* are canonical examples of words with a positive and negative semantic orientation, respectively (Turney and Littman, 2002). Two exemplar sets of words are defined: a group of words with a positive semantic orientation, $Pwords$, and a group of words with a negative orientation, $Nwords$.[16] The number of times a problem word w is associated with each $pword \in Pwords$ and with each $nword \in Nwords$ is gathered, again using AltaVista's NEAR operator. These numbers are then combined to form what is called a semantic orientation PMI formula (SO-PMI), which subtracts the PMI of w with negative words from the PMI of w with positive words. This forms a measure of how 'good' w is:

$$\text{SO-PMI}(w) = \sum_{pword \in Pwords} \text{PMI}(w, pword) - \sum_{nword \in Nwords} \text{PMI}(w, nword) \tag{2.3}$$

This formula employs more data than previous examples of PMI we have seen, and can be seen as an effort to infuse PMI with linguistic knowledge: the measure 'knows' which words are 'good' and 'bad', and uses this information to make what is intended as a more informed calculation of the semantic orientation of a candidate. In order to evaluate SO-PMI, Turney and Littman (2003) simplify the orientation ratings generated to a binary positive/negative choice, and achieve 82% agreement with a three-thousand-word corpus of semantically labelled words.

An important idea advanced in this work is that of collecting web data across a whole class

[15] As retrieving each document returned with a search is not just infeasible (due to the large number of requests necessary) but impossible (AltaVista, like most search engines, limits the number of returned links to documents to 1000), one often is limited to the handful of words of context provided with each search result.

[16] The set of positive and negative words are *good*, *nice*, *excellent*, *positive*, *fortunate*, *correct*, *superior*, and *bad*, *nasty*, *poor*, *negative*, *unfortunate*, *wrong*, *inferior*, respectively. They are chosen by the authors for their perceived relative insensitivity to context (Turney and Littman, 2003).

of related words, rather than relying on one word for comparison. This extra data allows any idiosyncrasies associated with a specific word to be smoothed out, and may capture information not available otherwise (Turney, 2002). However, it is important that the sets of words all capture the same semantic notion: otherwise, collecting data across these words would be counterproductive.[17]

Another useful contribution of Turney and Littman (2003) is the examination of Laplace smoothing on PMI with different corpus sizes.[18] It is found that on large corpora, their PMI measure is not particularly sensitive to the value of the smoothing factor. Rather than providing resistance to noise, smoothing is used only to prevent division by zero. Two conclusions are drawn: first, that the benefit from optimizing the smoothing factor for noise resistance is small for large corpora, and second, that there is less need for smoothing when a large corpus is available (Turney and Littman, 2003). These conclusions are significant for other work employing PMI and the web as a corpus.

The web, and search engines in particular, present unique properties and issues when employed in corpus linguistics. Extracting information with search engines is a promising avenue of research, and offers a large corpus at very little cost, as long as issues of noise can be dealt with. Such issues of noise are encountered in our own computational exploration of LVCs, as the models we develop for quantifying the acceptability of candidate light verb constructions all employ web data acquired by means of a search engine. In the next chapter, we present and justify these models.

[17] Turney and Littman (2003) explore using classes of words which have a more context-sensitive semantic orientation. Their experiments are rerun employing $Pword$ and $Nword$ sets featuring words which are similarly frequent, but more semantically variable, to those that had previously been employed. Examples of these words include *classic* and *confidence* (in the positive set) and *guilt* and *lost* (in the negative set). When using these sets, accuracy is over 10% worse.

[18] PMI is tested with three corpora: the full AltaVista corpus; a limited AltaVista corpus accessed by instructing the search engine to only include pages from the .ca (Canadian) namespace; and a 10 million-word corpus called TASA, which is a collection of short documents culled from a wide variety of sources such as novels and newspapers (Turney and Littman, 2003). The limited AltaVista corpus is about 2% of the size of the full corpus, and TASA about 0.5% of the size of the limited AltaVista corpus.

Chapter 3

Computational Models Of LVC Acceptability

In this work we investigate the (semi-)productivity of LVCs in order to find a means to quantify how well particular light verbs and candidate complements go together. We consider constructions in which an LV occurs with a predicative noun (PN)—the LV+PN constructions discussed in Section 2.2. In particular, we focus on the subclass of True Light Verb constructions (Section 2.2.2), which are distinguished, among other properties, by employing an indefinite determiner with a PN complement in a stem form identical to a verb. As noted by Kearns (2002), Wierzbicka (1982), and others, the restrictions on which complements can occur with particular light verbs in these constructions appear to be semantically patterned: complements with similar semantics seem to have the same trends of co-occurrence across light verbs. We wish to examine this hypothesis by comparing the behaviour of the complement in an LVC across semantic classes.

Two methods of grouping potential complements into semantic classes are considered. As the PNs under consideration have an argument structure and are identical in form to a verb, we employ the lexical semantic verb classes of Levin (1993). However, it may be that semantic classes which incorporate nominal information are more appropriate for this task. We

therefore also generalize across semantic classes generated using both the nominal and verbal hierarchies of WordNet 2.0 (Fellbaum, 1988). For each Levin class we consider, we generate a corresponding set from WordNet grown from a representative seed word from the Levin class.

Four computational measures are developed to quantify the acceptability of the candidate LVCs generated via our semantic classes: PMI, LVC-PMI, LVC-Prob, and LVC-Freq. We hypothesize that comparisons of LVC acceptability between classes and ontologies, and across the light verbs themselves, will show distinct patterns of acceptability. Further, comparisons of these measures to human judgments of LVC acceptability will indicate which approach is best suited to our task.

3.1 Generalization across Semantic Classes

3.1.1 Motivation of Approach

The class of LV+PN light verb constructions[1] is interesting because the productivity of an LV appears to be related to the semantic class of the complement. Consider the following examples:

a. Scott gave a groan / gave a sigh / gave a laugh / gave a cry.

b. ? Scott made a groan / made a sigh / made a laugh / made a cry.

c. * Scott took a groan / took a sigh / took a laugh / took a cry.

The PNs *groan*, *sigh*, *laugh*, and *cry* all describe methods of non-verbal expression. Furthermore, they all combine to form natural-sounding LVCs with the light verb *give*. When combined with the LV *make*, the construction of LV and PN is less acceptable, but seems more acceptable than those formed with *take*, which seem completely unnatural. We believe, following Kearns (2002) and Wierzbicka (1982), that the way in which LVs combine with PNs to

[1] As our work focuses on LV+PN constructions, the label of "light verb construction" ("LVC") will henceforth be used, for simplicity's sake, to refer specifically to LV+PN constructions.

form acceptable LVCs is not fully idiosyncratic, but rather systematic. We propose employing semantic classes of PNs from Levin (1993) and WordNet as semantically similar groups over which to compare the acceptability of complements with a given light verb.

Our approach is related to the idea of substitutability in MWEs. Past work has examined substituting part of an MWE with a semantically similar word in order to determine the productivity of the expression—higher substitutability indicating higher productivity (McCarthy et al., 2003; Lin, 1999). Similarly, in this work we explore the (semi-)productivity of LVCs by substituting complements across a class of semantically related PNs. Like Villavicencio (2003), we examine trends across semantic classes, and expect to find clear distinctions of productivity between them. However, the approach in this thesis differs from that of Villavicencio (2003) not only in focusing on LVCs, but also in its goal of quantifying how good a given candidate is. As the acceptability of a candidate LVC is not a binary decision but instead spans a continuum of acceptability, we seek to measure the acceptability of each candidate using the (continuous) measures we develop in Section 3.2.

In addition to examining trends in LVC acceptability across semantic classes, we explore trends across the light verbs themselves, to determine if there are patterns in how individual LVs are used semi-productively in LVCs. We focus on three common English light verbs: *take*, *give*, and *make*. We select *take* and *give* because they have nearly opposite semantics but still occur in a wide range of LVCs. Interestingly, it seems that some types of PNs can occur equally well with both of these LVs: one can both *take a tour* and *give a tour*, depending on one's role in the event. The light verb *make* is chosen for its difference from *take* and *give*, as it seems to favour different types of PNs in light verb constructions. Additionally, the "light" and "heavy" uses of *make* seem more difficult to distinguish than those of *take* and *give*. We expect *make* to show different generalization behaviour when compared with the other two light verbs.

Finally, as it may be possible that classes extracted from one source form are more appropriate for use in LVCs than those of another, we compare the patterns of acceptability between the related semantic classes extracted from Levin (1993) and WordNet. The method in which

these classes are acquired is now described.

3.1.2 Sources Employed in Acquiring Semantic Classes

Since the PNs we employ have a stem form identical to a verb, the hand-constructed verb classes of Levin (1993) may be useful as a semantic grouping of potential complements for LVCs. Examples of Levin verb classes include "Verbs of Manner of Speaking" (class 37.3, which contains verbs such as *groan* and *cry*) and "*Hit* Verbs" (class 18.1, including verbs such as *bash* and *whack*). We hypothesize that these classes may make detailed enough semantic distinctions to allow generalization of LVC acceptability along class lines.

However, PNs are not verbs, and semantic classes drawn from an ontology incorporating nominal information may form more appropriate groupings. To this end, we also employ WordNet as a source of semantic classes of PNs. In order to meaningfully compare the LVCs generated using WordNet sets to those generated using the Levin sets, we require that both sets be semantically similar to one another. We develop a technique which, given a representative seed word from Levin (1993), generates a set of semantically related PNs from WordNet: this process is explained in more detail in Section 4.2. In brief, we examine WordNet's "is a" hypernym hierarchy of nouns (i.e., a *stroll* is a kind of *walk* is a kind of *action*) and the "is one way to" hypernym hierarchy of verbs (i.e., to *stroll* is one way to *walk* is one way to *travel*). In both hierarchies, we remove those words which do not appear in both the noun and verb trees, thereby excluding elements which are not guaranteed to be the particular type of predicative nouns we focus on. We extract coordinate terms—words which have a parent in common with the given seed word—from each hierarchy, and these are used to form sets of semantically related PNs corresponding to the Levin class.

3.2 Acceptability Measures

We develop and evaluate four different computational approaches to quantifying the acceptability of candidate light verb constructions. These measures are pointwise mutual information (PMI), which has been used to measure the productivity of different types of multiword expressions in the past (Lin, 1999); LVC-PMI, which can be seen as an extension of PMI incorporating linguistic knowledge of LVCs; LVC-Prob, which is a probability formula designed to measure how likely an LVC construction is given its component light verb and complement; and LVC-Freq, a simple measure which exploits an estimate of the noise found in the data and ranks candidates by their frequency of appearance in the corpus.

3.2.1 The PMI Baseline

As noted, we focus on LVCs in which generally only the indefinite determiner *a* or *an* is employed. Treating LV+PN constructions as a collocated phrase, we employ PMI to measure the association between the light verb and its phrasal complement, "a PN".[2] This is labelled as $\mathrm{PMI}(LV; aPN)$.

The PMI score of a candidate LVC is interpreted as its acceptability, making the assumption that co-occurrence of an LV and PN complement reflects shared participation in an LVC. If $\mathrm{PMI}(LV; aPN) \gg 0$, then "LV a PN" is understood to be a good LVC; conversely, if $\mathrm{PMI}(LV; aPN) \ll 0$, then "LV a PN" is interpreted as being an unacceptable construction. Since we employ the World Wide Web as a corpus, problems of sparse data are much less likely to skew the output of the PMI formula.

This usage of PMI is unchanged from our past work in employing the measure to quantify LVC acceptability (Stevenson et al., 2004), and is used here as an informed baseline. While PMI can detect a significant level of cooccurrence between a given LV and "a PN" complement,

[2]Since some PNs begin with vowels, the indefinite determiner may be either *a* or *an*, but it is written here and elsewhere in this chapter as *a* for simplicity's sake.

this does not necessarily imply that the association between these two words is due to frequent usage in light verb constructions.

3.2.2 The LVC-PMI Measure

The LVC-PMI measure employs both PMI and linguistic properties of LVCs in order to make what is intended as a more informed judgment of candidate construction acceptability. LVC-PMI is based on previous computational research into LVC constructions, and is a modified version of the DiffAll measure presented in Stevenson et al. (2004). The DiffAll measure is now explained, as it motivates LVC-PMI.

DiffAll is founded on the linguistic hypothesis that generally only the indefinite determiner *a* (or *an*) is allowed in LV+PN constructions (Kearns, 2002). Stevenson et al. (2004) theorize that a higher mutual information value should therefore be found for "LV a PN" than for "LV [det] PN" constructions, where [det] is any definite determiner. A measure is designed which incorporates both $\mathrm{PMI}(LV; aPN)$, which is interpreted as positive evidence for LVC usage, and $\mathrm{PMI}(LV; detPN)$, which is interpreted as negative evidence. While $\mathrm{PMI}(LV; aPN)$ should indicate if "LV a PN" is a good collocation, the difference between the two, $\mathrm{PMI}(LV; aPN) - \mathrm{PMI}(LV; detPN)$, should indicate if "LV a PN" is a good LVC.

To capture this intuition in a single measure, $\mathrm{PMI}(LV; aPN)$ and $\mathrm{PMI}(LV; detPN)$ are combined using a linear approximation of the lines $\mathrm{PMI}(LV; aPN) = 0$ and $\mathrm{PMI}(LV; aPN) - \mathrm{PMI}(LV; detPN) = 0$. The single line which approximates the combined effect of these two PMI equations is $2 \times \mathrm{PMI}(LV; aPN) - \mathrm{PMI}(LV; detPN) = 0$, from which the DiffAll measure is drawn:

$$\mathrm{DiffAll}(LV, PN) = 2 \times \mathrm{PMI}(LV; aPN) - \mathrm{PMI}(LV; detPN) \tag{3.1}$$

Like the measure of synonym similarity presented in Turney and Littman (2003), DiffAll captures more information about the linguistic phenomenon in question than PMI alone. It is hypothesized that DiffAll is a better measurement of LVC acceptability than $\mathrm{PMI}(LV; aPN)$.

On development data, Stevenson et al. (2004) find that contrary to linguistic claims, the definite determiner *the* is not always rare in LV+PN constructions. For example, the indefinite LVC *I took a walk* is acceptable, but so too is the definite *I took the walk that was recommended.* DiffAll is tested employing different sets of determiners including *the*, *this*, *that*, and the possessive determiners, and it is found that measures of $\mathrm{PMI}(LV; detPN)$ excluding *the* perform best on development data. The final measure therefore employs demonstrative determiners (*this*, *that*) as well as possessive determiners (*my*, *your*, etc.), but does not include the definite determiner *the*. On test data, DiffAll is found to perform roughly the same as PMI on candidate LVCs involving the light verb *take*, somewhat better than PMI for candidate LVCs with *give*, and worse than PMI for candidate LVCs with *make*.

Further research performed for this thesis suggests the reason DiffAll does not clearly outperform PMI is that actual LVC usage is more nuanced than suggested in the linguistic literature and accounted for in the measure. The phenomenon of definite determiners being used in LVCs is not limited to *the*: LV+PN light verb constructions seem possible with most definite determiners. Consider the following examples involving the light verb *take* and the predicative noun *walk*:

a. I took a walk into town.

b. I took the walk that was recommended in the brochure.

c. I took that walk yesterday.

d. We decided to take another walk along the beach.

e. Grandfather announced he was leaving to take his walk.

It seems that while "LV a PN" light verb constructions do occur more often than those using definite determiners, LVCs employing these determiners are still acceptable constructions. Further, corpus evidence shows that these determiners are used frequently enough in LVCs to adversely affect the DiffAll measure, which is predicated on such LVC usages being

rare. Our testing on development data indicates the best results occur when $\mathrm{PMI}(LV; aPN)$ and $\mathrm{PMI}(LV; detPN)$ are added (rather than subtracted) in a two-to-one ratio. This reflects the linguistic intuition that while the association between the light verb and the indefinite complement is the most important factor in a candidate LVC's acceptability, instances of the light verb employed with definite complements also capture positive information about LVC usage. $\mathrm{PMI}(LV; aPN)$ and $\mathrm{PMI}(LV; detPN)$ are therefore added to define our LVC-PMI measure:

$$\text{LVC-PMI}(LV, PN) = 2 \times \mathrm{PMI}(LV; aPN) + \mathrm{PMI}(LV; detPN) \tag{3.2}$$

LVC-PMI, unlike DiffAll, has no reason to exclude *the*, and so a full set of definite determiners is used.

3.2.3 The LVC-Prob Measure

We now describe an alternative approach which does not employ PMI. LVC-Prob is a probability formula which measures the likelihood that a given LV and PN form an acceptable LVC. This probability depends on both the light verb and the predicative noun, and on these elements being used in the context of an LVC. LVC-Prob is thus defined as the joint probability $\text{LVC-Prob}(LV, PN) = \Pr(LV, PN, LVC)$, which we factor as:

$$\text{LVC-Prob}(LV, PN) = \Pr(PN) \Pr(LVC|PN) \Pr(LV|PN, LVC) \tag{3.3}$$

We will examine each of the factors of our probability formula individually, and motivate their estimation.

The first factor, $\Pr(PN)$, reflects the linguistic observation that higher frequency words are more likely to be used as complements in LVCs than less frequent words (Wierzbicka, 1982). We estimate this probability by $freq(PN)/n$, where n is the number of words in the corpus. We do not attempt to distinguish nominal from verbal usages of the PN word token in these counts, since a higher verb use may contribute evidence as to the predicative nature of the noun.

The $\Pr(LVC|PN)$ factor captures the intuition that the probability of a given LV and PN combining to form an acceptable LVC depends on how often the PN forms LVCs in general.

If a PN forms LVCs with many light verbs, then it seems probable that it will form an LVC with a particular light verb. However, if a PN is not often employed in LVCs, then it is less likely a particular combination of LV and PN will form an acceptable LVC. The frequency with which a PN forms LVCs is estimated as the number of times we observe the prototypical LVC pattern "LV a PN" (or "LV an PN", for PNs beginning with a vowel) across possible LVs: $\sum_{i=1}^{v} freq(LV_i + aPN)$, where v is the number of light verbs in our study. (Note that this is an overestimate, since we cannot determine which of such usages are indeed LVCs.) Therefore:

$$\Pr(LVC|PN) \approx \frac{\sum_{i=1}^{v} freq(LV_i + aPN)}{freq(aPN)} \tag{3.4}$$

Since we are only counting usages of the PN in the context of an indefinite determiner in the numerator, we normalize over counts of "a PN".

Finally, the third factor, $\Pr(LV|PN, LVC)$, reflects that different LVs have varying degrees of acceptability when used with a given PN in an LVC. We similarly estimate this factor from corpus data with counts of the LV and PN in the typical LVC pattern: $freq(LV + aPN)/freq(aPN)$. (Note again that this is an overestimate, since we cannot know with certainty that a given usage is an LVC.)

Combining the estimation of the three factors gives the full estimation of LVC-Prob using corpus data:

$$\begin{aligned} \text{LVC-Prob}(LV, PN) &= \Pr(PN)\Pr(LVC|PN)(LV|PN, LVC) \\ &\approx \frac{freq(PN)}{n} \times \frac{\sum_{i=1}^{v} freq(LV_i + aPN)}{freq(aPN)} \times \frac{freq(LV + aPN)}{freq(aPN)} \end{aligned} \tag{3.5}$$

where v is the number of light verbs for which data is available, and n is the number of words in the corpus.

3.2.4 The LVC-Freq Measure

Our final measure, LVC-Freq, is specifically designed to employ web data accessed via linguistically naïve search engines. It assumes a vast but unannotated corpus, one which is costly to

access (each corpus search can take several seconds) and from which extracting accurate data is difficult. LVC-Freq ranks the acceptability of candidate LVCs by the frequency at which they are found in a corpus. However, rating these candidates by the number of results found in a search for the construction is problematic, as these searches are variously affected by noise. As detailed in Section 2.4.3, a web search for *take a walk* may return pages with phrases unrelated to the candidate construction. We therefore estimate the level of noise affecting each candidate construction, and remove this noise from its search results.

Our estimate of noise is based on the intuition that LVCs are more likely to be expressed without internal modifiers than with such modifiers, and that the likelihood of seeing an LVC with m modifiers approaches zero as m increases. For example, we expect the LVC *take a walk* is more likely to be expressed than internally modified LVCs such as *take a long walk*, which are in turn more probable than LVCs such as *take a long, relaxing walk* and *take a long but stimulating and relaxing walk*, and so forth. We assume there exists a threshold, t, at which the likelihood of producing an LVC involving $m > t$ words of internal modification is negligible. At this threshold, any results returned by a search for a candidate with t wildcards must include only noisy phrases unrelated to LVC usage.

To capture these internal modifiers, we employ wildcards ("$*$") which match one word. These wildcards are supported by the version of the Google search engine used in this work.[3] Thus, a search for *take a* $*$ *walk* may return documents including phrases such as *take a short walk* along with noise, such as *take a tip: walk with me*. As more wildcards are added to a search phrase, longer forms of noise are returned. We search for each candidate LVC at both the zero and t wildcard level (with t determined on development data), and label these searches $freq(0)$ and $freq(t)$ respectively. Evidence of LVC usage (if available) is expected to be found with the zero wildcard search, and only noise involving the LV and PN is expected to be found with the t wildcard search. Our estimate of noise is removed from the zero wildcard search via

[3] As detailed in Appendix C, past updates to the Google search engine have at times discontinued wildcard support.

a simple subtraction:

$$\text{LVC-Freq} = freq(0) - freq(t) \tag{3.6}$$

Scores are generally positive, but a very unacceptable construction may receive a negative score if $freq(0)$ is near zero. LVC-Freq is obviously a very basic measure: it is designed to measure the acceptability of LVCs with very few corpus samples, and should also illustrate whether the web is large enough to support such basic approaches to linguistic problems. Note that if non-noisy access to our corpus was available (i.e., if result sets contained only LVCs and no other constructions), then $freq(t)$ would be by definition equal to zero and we would simply be employing the number of instances of LVC usage, $freq(0)$, to rate the acceptability of each candidate. Section 4.3.2 describes the determination of the value of t through development testing and linguistic intuition.

3.2.5 Summary of Acceptability Measures

The four different computational measures developed for quantifying the acceptability of candidate LVCs each take a slightly different approach to the problem. PMI is an informed baseline, LVC-PMI is intended as an improvement on the baseline which incorporates linguistic knowledge of LVCs, LVC-Prob is a probability formula which measures the likelihood of a given LV and PN forming an acceptable construction, and LVC-Freq is a simple measure which rates candidates by their frequency of usage in corpora and attempts to estimate the noise found in corpus measurements.

Chapter 4

Materials and Methods

In this chapter we describe the realization of the computational models presented in Chapter 3 for quantifying the acceptability of candidate light verb constructions, along with the semantic classes of predicative nouns (PNs) against which these measures are tested. We employ two types of semantic classes in this work: a set of selected classes from Levin (1993), and corresponding classes automatically generated using WordNet 2.0. In the next section, we list the Levin classes selected; these classes are chosen to reflect a range of productivity across light verbs. We then focus on our method of generating corresponding classes of semantically related PNs from WordNet. We detail a technique which, given a seed PN, gathers a set of PNs which are semantically related to the seed. With this procedure in place, we describe a method for automatically selecting a representative seed word from each chosen Levin class. These seeds are used with the set generation algorithm to produce the WordNet semantic classes employed in this work. It is hoped that the trends in LVC productivity of the classes extracted from WordNet will reflect those of the classes selected from Levin (1993).

The web extraction techniques used to gather the data necessary for our four measures of LVC acceptability are then described. We consider the issues specific to using World Wide Web data acquired via general-purpose search engines, and detail the three techniques developed to filter noise. Each is aimed at a different type of "false hit" returned by the search engine

employed. In the final section, we consider the two statistical measures we employ to assess the performance of our computational measures of LVC acceptability. We employ Spearman Rank Correlation (SRC), a statistic which captures how often two annotators rank candidate constructions in the same relative order, along with Weighted Kappa, a measure of agreement which given a set of category labels (i.e., 'poor', 'fair', and 'good') captures how often two annotators assign the same labels to candidate constructions.

4.1 Selection of Levin Verb Classes

Levin (1993) is employed as a source of lexical semantic verb classes. Three development and four test classes are chosen to reflect a range of productivity of complements across light verbs: these classes are listed in Table 4.1. Members of these verb classes which are not form-equivalent to a noun are removed from consideration. Some classes, such as 51.4.2, are (according to researcher intuition) generally good with *take*, slightly worse with *give*, and poor with *make*. Others, such as 43.2, are better for *make*, but poor with *take*. Some classes, such as 18.1 and 18.2, allow the dative form with the light verb *give*, while others such as 43.2, do not. As our evaluation process employs comparisons with human acceptability judgments, and as these are costly to gather, Levin classes with more than 35 verbs (30 for development classes) have a random subset of their membership chosen. As a result, all test Levin classes have no more than 35 members, and all development Levin classes have no more than 30 members.

4.2 Generation of WordNet Classes

Our goal is to construct a semantic class of PNs from WordNet which roughly corresponds to the coarse-grained meaning of each class selected from Levin (1993), so that we can compare patterns of acceptability across the corresponding classes. Each Levin class has automatically extracted from it a single representative PN seed, which is used as a starting point for our set

Development Classes			
Levin #	Name	Count	Example Members
10.4.1*	*Wipe* Verbs, Manner	30	*dab*, *dust*, *scrub*, *wipe*
17.1	*Throw* Verbs	30	*bash*, *hit*, *shoot*, *throw*
51.3.2*	*Run* Verbs	30	*bolt*, *dash*, *hike*, *streak*
Test Classes			
Levin #	Name	Count	Example Members
18.1,2	*Hit* and *Swat* Verbs	35	*bang*, *kick*, *pound*, *slug*
30.3	*Peer* Verbs	18	*gaze*, *leer*, *peek*, *stare*
43.2*	Sound Emission	35	*burr*, *clap*, *plop*, *whistle*
51.4.2	Motion (non-vehicle)	10	*drive*, *paddle*, *row*, *tack*

Table 4.1: Levin classes chosen for development and test data. A '*' indicates a random subset of verbs was used in the class.

generation algorithm. This algorithm employs the nominal and verbal hierarchies of WordNet to collect a class of PNs semantically related to the seed provided.

For ease of explanation, in the next section we assume a method exists for choosing an appropriate PN seed from a given Levin class, and detail how a set of semantically related PNs can be extracted from WordNet. We then describe in the section following an algorithm for selecting an appropriate seed.

4.2.1 Generating a Set of Related Words

Given a predicative noun seed, pn_s, we wish to gather a set of semantically-related words from WordNet for use in LVCs. First, the noun and verb hierarchies are examined, and all the coordinate terms[1] of pn_s are gathered into two sets, respectively labelled $Coord_n(pn_s)$ and $Coord_v(pn_s)$. Non-verbs are automatically removed from $Coord_n(pn_s)$ and non-nouns from

[1] A coordinate term of pn_s is a word that shares at least one of the parents of pn_s.

$Coord_v(pn_s)$, to help ensure that only predicative nouns remain in the sets. The three light verbs we employ in this work, if found, are also filtered. The union of these two sets is labeled $Coord(pn_s)$. Note that since pn_s is likely polysemous, $Coord(pn_s)$ likely contains siblings of pn_s found under many different parents.

We measure three properties of each $pn \in Coord(pn_s)$ which can be used to select PNs most appropriate for the semantic classes we are generating. First, the number of syllables of pn is extracted from an electronic rhyming dictionary; if the word is not found in this lexicon, the syllable count is estimated. This syllable count can be used to exploit the linguistic observation that that shorter, simpler words are typically better in LVCs (Wierzbicka, 1982). Secondly, the number of times the stem form of the PN is used as a verb in the British National Corpus (BNC Reference Guide, 2000) is measured. This is labelled $Count_{BNC}(pn)$, and allows us to make use of the linguistic observation that words which are more frequently employed in general usage are more likely to to be employed in LVCs. A verb inflector is used to ensure that any tense of a verb with a stem form identical to a PN is included in $Count_{BNC}(pn)$. Finally, each PN has a WordNet count associated with it, $Count_{WN}(pn)$, which is defined as the number of times pn is found associated with a different sense of the seed pn_s.[2] This is used to capture the intuition that a word which is more often associated with pn_s likely has more in common with it than one which is not. Like $Count_{BNC}(pn)$, $Count_{WN}(pn)$ is also uniform across verb tenses.

It was our expectation that thresholds could be applied to these properties. By altering these thresholds, we could generate sets of PNs with varying appropriateness for LVCs. For example, if only monosyllabic words were allowed, but no restrictions were put on $Count_{BNC}$ and $Count_{WN}$, then we would expect to generate a set containing short, simple words which are likely better for LVCs but which are less related to pn_s. Similarly, if no ceiling were applied to the syllable counts, but several thousand appearances in the BNC and a high WordNet

[2]Word senses are listed in WordNet. For example, the verb *walk* has ten senses listed, including "using one's feet to advance" (i.e., *I walked instead of driving*), "make walk" (i.e., *I am walking the dog*), and "accompany or escort" (i.e., *I'll walk you to your car*).

frequency were required, we would expect to generate a small set including only very common (but potentially multi-syllable) words closely related to the seed pn_s. While we did find that applying such thresholds affected the sets as expected, $Coord(pn_s)$ would often became too small to be useful. In order for a meaningful comparison to be made with the Levin classes, we require that the WordNet sets have at least 35 members, and these heavily filtered sets were simply too small. Our set generation technique could be adapted to gather a larger collection of words, but we found extending our search beyond coordinate terms returned PNs which were intuitively much less related to the seed. In the final experiment, no syllable ceiling is applied, and PNs must appear in the BNC at least fifty times to be eligible for $Coord(pn_s)$. There are no restrictions on $Count_{WN}$, as we found $Count_{BNC}$ captured much of the same information. With these thresholds applied, the remaining membership of $Coord_n(pn_s)$ and $Coord_v(pn_s)$ is merged into the set $Coord(pn_s)$, which is used as our class of related words gathered from WordNet.

In all cases, $Coord_v(pn_s)$ is much larger than $Coord_n(pn_s)$, even without threshold ceilings applied. This is because WordNet's verb hierarchy is wider and shallower than its noun hierarchy (Resnik and Diab, 2000), implying a comparatively closer association between verbs. With the predicative noun *hop* as a seed, only six other words are selected by our process in the noun hierarchy: *bounce*, *leap*, *rave*, *spring*, *bound*, and *squash*. In contrast, ninety-eight verbs are available from the verb hierarchy, including words such as *run*, *flip*, *roll*, *jump*, *sail*, *waver*, and so on.

With a technique in place for generating classes of semantically related PNs from WordNet, we now explore the problem of choosing a representative seed word from each chosen Levin class.

4.2.2 Choosing a Representative Seed Word

We wish to select a representative seed word from each semantic class chosen from Levin (1993): a single PN which best captures the relevant semantics of the Levin class with respect

to LVC formation. In order to determine which members of a given Levin class are most representative, we measure in each class the general trend of acceptability across our three light verbs—*take*, *make*, and *give*—by examining the patterns in human acceptability judgments. Each candidate LVC generated by combining a member of a class with our light verbs has a human acceptability rating associated with it. These ratings range from 1–5 (for the development Levin classes) or 1–4 (for all other classes, as we found that the range of 1–5 was greater than seemed natural). A rating of 1 means the LVC seems unnatural to the human raters, while a rating of 4 or 5 means the LVC seems completely acceptable. Partial ratings, such as 2.5, are permitted. These ratings and the method with which they are gathered are explained in further detail in Section 5.1.

An approach is developed which allows each light verb with each class to have a label of either 'poor', 'fair', or 'good' applied to it. These labels capture the productivity of the light verb in LVCs with members of a given class. Predicative nouns which we assume reflect these labelled trends to the highest degree possible are automatically selected from each class, and used as the seed words necessary for the set generation algorithm previously described.

The Seed Selection Algorithm

For each Levin class and each LV, the human acceptability ratings are put into buckets: a 'poor' bucket contains ratings from $[1–2)$, a 'fair' bucket contains ratings from $[2–3)$, and a 'good' bucket contains ratings of 3 and above.[3] We determine the percentage of ratings each bucket contains and assigned a label to the LV for that class. If the 'poor' bucket contains more than 80% of the ratings, then the LV in question is labelled 'poor' for LVCs in that class. If the 'poor' bucket contains less than 50% of the ratings, then the LV is labelled 'good' for the class. Anything between 50% and 80% causes the light verb to be labelled 'fair'. The labels that were generated for each class are summarized in Table 4.2. The obvious limitation of this approach

[3]We found that the 1–5 rating scale was more discriminating towards high-end values when compared to the 1–4 scale. Therefore, the bucket thresholds are applied equally to all human judgments, regardless of the scale employed.

Development Classes					
		Acceptability Label			
Levin #	Class Description	*take*	*give*	*make*	Seed Word Selected
10.4.1 *	*Wipe* Verbs, Manner	poor	good	poor	*flush*
17.1	*Throw* Verbs	fair	good	fair	*flick*
51.3.2 *	*Run* Verbs	good	fair	poor	*hop*
Test Classes					
		Acceptability Label			
Levin #	Class Description	*take*	*give*	*make*	Seed Word Selected
18.1,2	*Hit* and *Swat* Verbs	fair	good	fair	*knock*
30.3	*Peer* Verbs	fair	fair	poor	*check*
43.2 *	Sound Emission	poor	good	fair	*ring*
51.4.2	Motion (non-vehicle)	good	fair	poor	*sail*

Table 4.2: Trends identified for each light verb and class. A '*' indicates a random subset of verbs from that class were used.

is that the relative frequency of 'poor' LVCs (which make up a large proportion of the membership in every Levin class) can affect which label an LV receives. More complex measures were explored, including those which considered only the values of the 'fair' and 'good' buckets, but these measures changed the labels applied to LVs in only one or two instances, and were considered overfitting.

We then search for predicative nouns which, for each of the three LVs, reflect these labelled trends. If a given LV is rated 'poor' for a class, we look for members which, when combined with the light verb under consideration, form a construction with a human rating of 1. If the LV is rated 'fair', PNs which have the LVC rated in the range of $[2–2.5]$ are chosen, and if the LV is rated 'good', PNs rated 3.5 or higher are selected. Ideally, a set of PNs reflecting the general trend of the class would be found by this algorithm, and would form a set of "perfect fit" PNs.

However, in several of the classes, no PN perfectly fitted the overall trend of the class. In these cases, we relax our search constraints, and select PNs which are, for each light verb, within one point of fitting our criteria. This forms a set of "near fit" PNs.

For each class, the set of perfect fit PNs has a candidate seed selected from it at random. If this candidate generates a set of at least 35 elements from WordNet, it is chosen as the seed, and 35 members from its set are randomly chosen as the WordNet set for the class. If the perfect fit set is empty, or if none of the perfect fit PNs generate a class with at least 35 elements, then the close fit set is used in the same manner. In only one class, 30.3, did none of the PNs in either the perfect or near fit sets generate a collection of at least 35 elements. In this case the remaining "poor fit" PNs were considered. There are four PNs in 30.3 which generate a set of a sufficient size, but only one, *check*, distinguishes itself by being within our near fit constraints for two of the three LVs. This PN is therefore chosen as the seed word for this class. The seed words selected for each class are listed in Table 4.2.

Using the above seed selection technique, a single representative PN is selected from each Levin class employed in this work. These PNs are used as seeds in the WordNet set generation technique described in Section 4.2.1 to form classes of semantically related PNs.

4.3 Extracting Data from the Web

In this section, we explore how information is gathered from the web about the candidate LVCs formed by combining the light verbs *take*, *give*, and *make* with the members of these semantic classes, along with those classes selected from Levin (1993). This data is employed by the computational measures presented in Chapter 3 to assess the acceptability of candidate LVCs.

4.3.1 The Web as a Corpus

The subsection of the web indexed by Google is used as a corpus, and Google's public search interface is used to access this corpus. There are several issues with employing general-purpose

search engines such as Google for linguistic research, and these issues must be considered by any extraction procedure if accurate data is desired. Critically, punctuation is ignored in search requests, which means that search results can cross phrase or sentence boundaries.[4] For example, a web page with the text *It was too much to take. A cry escaped his lips.* may be returned as a search result for the exact phrase "take a cry".

Additionally, Google, like most general-purpose search engines, limits the number of instances returned. No more than a thousand results can be examined for each search phrase: an unfortunate limitation, as it implies searches which return less than a thousand results can be examined in full, while those that do not can only be partially examined to varying proportions. Since this sample is not random, statistical prediction from it is difficult. However, as detailed in Section 3.2.4, Google offers the advantage of supporting a wildcard "$*$" operator, which matches for one word. A search for *take a* $*$ *walk* would return links to pages containing the LVCs *take a long walk*, *take a short walk*, along with links to pages containing non-LVC noise such as *take a look; walk over.* Similarly, one could search for *take a* $**$ *walk*, which would return results containing LVCs such as *take a long, tiring walk*, along with noise: *Take a bite! Then, walk over.* Further issues specific to using Google data as a corpus are explored in Appendix C. With these properties of the corpus in mind, we consider in the next section how information extraction is accomplished.

4.3.2 Collecting Data

Each search request is made via an exact-phrase search. The number of results returned with each search (which we treat as the frequency count of the searched pattern) is stored in a local database, as well as any context words included with the first thousand instances. Note that this frequency count is surely an underestimate, as an LVC may occur more than once in a single web page; however, examining each document to count the actual occurrences of the candidate

[4] As far as we are aware, all general-purpose search engines with a comparably sized index suffer from the same limitation.

is infeasible, given the number of results that could be returned and the limit on the number of instances returned. The size of the corpus, *n*, is estimated at 5.6 billion, the number of hits returned in a search for "the." This, too, is surely an underestimate.

We now explore the data required for each computational measure of LVC acceptability. As each request sent to a search engine can take several seconds to complete, the number of searches required can affect the relative merit of a measure. We therefore include the average number of search requests required by each measure.

Pointwise Mutual Information (PMI)

PMI is defined as $\mathrm{PMI}(LV; aPN)$ and measures the association between a given light verb and "a PN" complement. Several counts are required by PMI for each candidate LVC: the frequency of "LV a PN" (e.g., *take a walk*), the frequency of the light verb (*take*), and the frequency of the complement (*a walk*). Data from searches for the candidate with no wildcards (e.g., *take a walk*) and with one wildcard (e.g., *take a * walk*) are summed together. Using these wildcards allows us to capture adjectival use of a candidate LVC, such as *take a long walk*, at the expense of allowing in more noise. Development testing indicates that while including data with one wildcard improves the performance of PMI, employing data with more than one wildcard acts to decrease performance, as the noise found in such searches overwhelms any LVC usage detected.

In order to achieve broader coverage and to minimize any skew from a particular tense of the light verb, we search across three tenses of the light verb. This process is also performed with the LVC-PMI and LVC-Prob measures. Any search involving an LV is performed thrice, each employing one of three tenses: the base form (*give*), the present (*gives*), and the simple past (*gave*). Searching with additional tenses is possible, but as each new tense adds to the number of searches required for the measure, and as each search request can take seconds to complete, the three tenses we consider are felt to be sufficient to achieve a broader coverage while not unduly increasing the number of search requests required. All counts of the light

Determiner	Search Strings
Indefinite	*take/takes/took a walk*
Definite	*take/takes/took the walk*
Demonstrative	*take/takes/took this/.../which walk*
Possessive	*take/takes/took my/.../their walk*

Table 4.3: Searches necessary for the LVC-PMI measure employed with the light verb *take* and verb *walk*, with no wildcards.

verbs and the LVCs are collapsed across the three tenses.

Therefore, each candidate LVC involves a total of eight searches per wildcard: one for n, three for the candidate LVC, three for the LV, and one for the complement. However, as the results for each search are cached, the counts for n and for each LV are stored after the first search. Thus it takes an average of four searches to calculate PMI for each candidate LVC. We employ the NSP package (Banerjee and Pedersen, 2003) to calculate PMI.

LVC-PMI

LVC-PMI is a version of PMI that incorporates linguistic knowledge of LVCs, and is defined as $2 \times \mathrm{PMI}(LV; aPN) + \mathrm{PMI}(LV; detPN)$. Searches similar to those specified above for PMI's "LV a PN" are also performed for each possible "LV [det] PN" construction, as exemplified in Table 4.3. This includes searches for both the LVC (*take [det] walk*) and the complement alone (*[det] walk*) using the following determiners: *the*, *this*, *that*, *which*, *whose*, *what*, *each*, *every*, *one*, *no*, *any*, *my*, *our*, *your*, *his*, *her*, *its*, and *their*. Each candidate LVC requires the average four searches needed for PMI, multiplied by the nineteen determiners, for a total of seventy-six searches per predicative noun. As with PMI, LVC-PMI performs best when searches are run with both zero and one wildcard(s), and the counts combined.

LVC-Prob

LVC-Prob is defined as $\Pr(PN)\Pr(LVC|PN)\Pr(LV|PN,LVC)$. The $\Pr(PN)$ factor is estimated by $freq(PN)/freq(n)$. The number of results found in a search for the word *the* is again used as our estimate for n; therefore, this factor requires on average only the one search for the PN. We estimate $\Pr(LVC|PN)$ by $(freq(take + aPN) + freq(give + aPN) + freq(make + aPN))/aPN$, which requires on average ten searches: one for each of the three tenses considered for each of the three light verbs, plus the "a PN" search. Finally, the $\Pr(LV|PN,LVC)$ factor is estimated by $freq(LV + aPN)/freq(aPN)$, which requires no additional requests, as the searches for the candidate LVC and the "a PN" complement are already cached. Summing these counts together gives the total of an average of eleven searches per candidate LVC. Like both PMI and LVC-PMI, best performance is found with data gathered using both zero and one wildcard(s).

LVC-Freq

LVC-Freq is defined as the subtraction $freq(0) - freq(t)$. The number of results found in a search with t wildcards is used as an estimate of noise affecting the candidate LVC, and this noise is subtracted from the number of results found in a search with no wildcards, where we expect to find evidence of a valid construction. Development results indicate that the pattern in noise across values of t can vary, and this affects the accuracy of ratings assigned to candidate LVCs. Taking the average of the frequency counts between six and ten wildcards, inclusive, both smooths out fluctuations visible at individual wildcard levels, and gives the best agreement with development data. More than ten wildcards blurs the relationship between LV and complement, and moves more towards measuring the general level of noise in the corpus, rather than the specific level of noise affecting the LV and PN complement.

The LVC-Freq measure is slightly optimized by choosing the most frequent tense of the zero wildcard candidate LVC: on development data, this led to better agreement than simply merging counts across tenses of the light verb, as we do with the other measures. Therefore,

Measurement	Average Searches Per Non-Dative Candidate LVC
PMI	4 × (number of wildcards + 1)
LVC-PMI	76 × (number of wildcards + 1)
LVC-Prob	11 × (number of wildcards + 1)
LVC-Freq	8

Table 4.4: The average number of searches necessary for a non-dative candidate LVC.

LVC-Freq requires eight searches: three for determining the most frequent LV tense with zero wildcards, and five for the candidate LVC search with six through ten wildcards. The differences in the number of searches required for a candidate LVC between our four computational measures are illustrated in Table 4.4.

Capturing the Dative Form

Of the three light verbs we search for, only LVCs involving *give* can appear in the dative form (i.e. "give <indirect object> a PN"). However, capturing LVCs in this form is important, as previous research (Stevenson et al., 2004) indicates that while many PNs can appear in both non-dative and dative LVCs, there are some which can only appear in the dative form. For example, one can *give the door a knock* and *give something a try*, but while one can *give a knock on the door*, one cannot **give a try to something*.

In order to capture dative LVCs in our searches, we explored using wildcards. Searches like *give* $*$ *a kiss* and *give* $*$ *a* $*$ *kiss* promised to capture phrases like *give Mom a kiss* and *give Bailey a big kiss*, but we found that these search techniques introduced an overwhelming amount of noise into the measurement. We instead perform individual searches on a set of pronouns, which, while requiring many more search requests, return result sets which are significantly less noisy. A set $Pronouns$ is defined, containing fifty-six common pronouns.[5] For each predicative

[5]The full set of pronouns is *all*, *another*, *any*, *anybody*, *anyone*, *anything*, *both*, *each*, *either*, *everybody*, *everyone*, *everything*, *few*, *her*, *hers*, *herself*, *him*, *himself*, *his*, *it*, *itself*, *many*, *me*, *mine*, *myself*, *neither*, *nobody*, *none*, *nothing*, *one*, *others*, *ours*, *ourselves*, *several*, *some*, *somebody*, *someone*, *something*, *that*, *their*, *theirs*,

noun PN and for each $PRO \in Pronouns$, a search for *give PRO a PN* is performed, along with related searches at each wildcard level. While these searches do exclude any LVCs employing a proper or common noun, they still cover a large space of LVC usage: phrases such as *give her a kiss* (with zero wildcards) and *give me a big kiss* (with one wildcard) are detected by these searches. This follows Villavicencio (2003) in designing searches which, rather than considering a large collection of very noisy data, instead target a smaller collection of "cleaner" information. Such searches allow us to capture a large amount of data exhibiting the dative form while still keeping a relatively high signal/noise ratio.

LVCs that can appear in the dative form are first searched for with the usual non-dative search process, and then information specific to the dative form is gathered. As there are fifty-six pronouns to search for, the average number of searches required for candidate LVCs involving *give* (which may appear in the dative form) is fifty-six times those required for LVCs involving the LVs *take* and *make*. Searches for the dative form are run for each measure save LVC-Freq, as it is designed to explore rating LVCs using very little data. It is expected that the comparison of the performance of the simpler LVC-Freq to those of the more informed measures may prove interesting.

Finally, a very small smoothing factor of 10^{-5} is added to all counts employed by our measures. It is used, as in Turney and Littman (2003), simply to prevent division by zero: development testing indicates it has very little effect on results.

4.3.3 Removing Noise

While the web extraction techniques employed by our computational measures are designed to eschew unwanted information, constraints imposed by the corpus dictate that result sets will likely still contain significant elements of noise. We now describe the techniques developed for removing such noise from our search engine web data.

Our noise-filtering methods are constrained by what information is made available by the

them, *themselves*, *these*, *they*, *this*, *us*, *what*, *which*, *who*, *whom*, *whose*, *you*, *yours*, *yourself*, and *yourselves*.

search engine. Each Google exact phrase search for a phrase p is modeled as a function, $search(p)$, which returns the number of results found for p, $frequencycount(p)$, and a set of results, $results(p)$. Each $r \in results(p)$ represents a phrase result returned by the search engine, and has associated with it an ordered array of context words, $context(r)$. This context generally spans a range beginning with a few words before p and ending a few words after p, including p itself. By examining $context(r)$, we attempt to identify which results are noise, and design techniques to identify and remove these instances from $results(p)$.

There are two types of noise found in our data. The first is noise inherent to the corpus, including typographic errors and ungrammatical text. This noise is not targeted by our filtering methods, although it may be filtered incidentally. The second is "false hit" noise: results returned which are unrelated to the target of our search requests. We explore three different techniques for removing this noise from the web data. Punctuation Filtering is the first technique described, which removes results that are obviously non-LVC uses of the search phrase p from $results(p)$. A technique named Phrase Filtering is then considered, which can remove phrases unrelated to LVC usage, such as *give him a strip of paper*, from $results($*give him a strip*$)$. Finally, the similar technique of Multiword Expression (MWE) Filtering is described, which can remove non-LVC multiword expressions such as *give a slide show* from $results($*give a slide*$)$.

Punctuation Filtering

Given a set of results and their context, Punctuation Filtering extracts a filtered set in which all results involving internal punctuation are removed. Each $r \in results(p)$ is examined, and any r with phrase-ending punctuation (this includes periods, commas, question marks, colons, brackets, and so forth) between the words of p is removed from the set. This is an efficient technique, as noisy results are removed without incurring the cost of additional search requests. As an example, phrases like *Take a shower, shave, and begin your day* appear in the one-wildcard $results($*take a* $*$ *shave*$)$: with this filtering, all such phrases are removed. Even with searches

without wildcards, there are often results that cross sentence boundaries: phrases such as *take? A stroll* and *give, (a smile*. Punctuation Filtering removes this noise as well. Effectively, it acts as a linguistically aware front end to Google, removing its undesired behaviour of returning results involving punctuation for exact phrase and punctuation free search requests. This filtering was applied to every Google search performed, with the exception of those for the LVC-Freq measure: as LVC-Freq is based on capturing trends in noise, filtering out such noise would be counterproductive.

Phrase Filtering

In examining development results which disagreed with human ratings, we found many candidate LVCs had been rated quite highly by computational measures because of their use in phrases. As an example, the LVC *give a strip*, which had been rated poorly by our human judges, had achieved high computational rankings due to $search($*give a strip*$)$ returning many results involving strips of land and paper. In fact, collocated phrases such as *give a strip of paper* (to John) and *give her a strip of land* formed the majority of the result set. Phrase Filtering is designed to remove these phrases.

We employ PMI, which has been shown to work well at identifying significant collocations, given a large corpus. If a phrase such as *strip of paper* is indicated by PMI to occur significantly more often than chance allows, we assume this is an example of the non-LVC phrase usage of the PN, and remove the construction from result set. To accomplish this, the top twenty most frequent words occurring after the "LVC+of" construction are considered as candidates for filtering, and the information necessary to apply PMI to these words is gathered. We justify the choice of examining only the top twenty words by development results, which suggest that often only a few words follow the LVC with significant regularity, and by linguistic intuition, as we expect there will not be very many such collocated phrases. By considering the top twenty words, we have a window expected to be large enough to test and filter all significant collocations, while not requiring that a very large number of additional searches be performed.

However, we cannot simply examine each $r \in results(p)$, find which r are immediately followed by the word *of*, and rank the words following *of* by frequency. As we are limited by the thousand-instance ceiling applied to search results, there may not be sufficient examples of LVC usage followed by *of* in the results for a given candidate to form a representative sample. In order to gather more accurate data, we run a search for the candidate followed by the word *of*: i.e., *take a strip of*. The top twenty words following this construction are considered. These words are added to a set, $nextwords(\mathrm{LVC} + \mathrm{of})$, and for each $w \in nextwords(\mathrm{LVC} + \mathrm{of})$, we gather the information necessary for PMI. Thus, we record the frequency of "PN of w" (i.e., *strip of paper*), the frequency of PN (i.e., *strip*), and the frequency of "of w" (i.e., *of paper*). We calculate $\mathrm{PMI}(PN;$ *"of w"*$)$, and if it is sufficiently large, we mark that construction as noise. Development testing indicated that a PMI value of 1.0 was an appropriate threshold: higher thresholds tended not to filter out the targeted noise, and lower ones removed too many valid results. Collocations with a PMI score of 1.0 or higher are filtered.

To filter out these marked constructions from $results(\mathrm{LVC})$, we calculate the percentage of $results(\mathrm{LVC} + \mathrm{of})$ identified as noise, and label this $percent_{noise(\mathrm{of})}$. We then calculate the percentage of phrases in $results(\mathrm{LVC})$ which are followed by *of*, which is labelled $percent_{results(\mathrm{of})}$. Multiplying these two numbers together gives the percentage of results for the LVC under consideration which are followed by *of* and which are also identified as noise. This percentage, $percent_{results(\mathrm{of})} \times percent_{noise(\mathrm{of})}$, is removed from $frequencycount(\mathrm{LVC})$.

Obviously, the searches performed for "LVC+of" do not avoid the thousand-instance limit on Google search queries. However, they do allow us to partially relax this constraint as it applies to candidate LVCs. By gathering more detailed information on a specific subclass of expression (the "LVC+of" constructions), we can apply any knowledge gained about this specific construction to the data reflecting the candidate LVC itself; in effect, we focus on particular instances where noise may appear, and apply any knowledge gained to the whole.

MWE Filtering

Another instance of noise comes from words used in compound phrases and MWEs involving the stem form of the predicative noun as their first word. For example, $results(\textit{give a slide})$ includes many instances concerning how one can *give a slide show* and *give a slide presentation*. These false hits, like the phrases identified by Phrase Filtering, inflate the frequency counts for the candidate LVC and lead to computational ratings differing from human evaluations. A process very similar to Phrase Filtering, called MWE Filtering, is designed to filter out this noise.

Here, the top twenty words following the LVC itself are added to a set, $nextwords(LVC)$. For each $w \in nextwords(\text{LVC})$, we gather the necessary information for PMI: the frequency of "PN + w" (e.g., *slide show*), the frequency of PN (e.g., *slide*), and the frequency of w (e.g., *show*). PMI is again applied as in Phrase Filtering, and results which have a score greater than or equal to 1.0 are marked as noise. We determine the percentage of $results(p)$ identified as noise, and remove this percentage from $frequencycount(p)$. The targeted searches for "LVC+of" that were used in Phrase Filtering cannot be extended to MWE Filtering, and therefore this measure relies on slightly coarser data.

The above three techniques of Punctuation, Phrase, and MWE Filtering are applied as appropriate to each Google search performed. Excepting those for the LVC-Freq measure, punctuation filtering is applied to all searches, and both Phrase and MWE Filtering are additionally applied to all searches for candidate LVCs. The filtered number of results returned is used by our computational measures.

4.4 Statistical Measures of Association

The two statistical measures we employ for evaluation of our experimental results are Spearman Rank Correlation, which measures the linear relationship between two sets of ratings, and Weighted Kappa (Cohen, 1960, 1968; Carletta, 1996), which measures agreement. Spear-

man Rank Correlation (SRC) uses the raw ratings of candidate constructions, while Weighted Kappa uses categorized data, corresponding to labels of LVC acceptability of 'poor', 'fair', and 'good'. Weighted Kappa therefore reflects a coarser level of agreement than the correlation captured by SRC. We now consider issues associated with these two measures in detail, as both have idiosyncrasies which can affect the interpretation of their results.

4.4.1 Spearman Rank Correlation

Spearman Rank Correlation (SRC) is used to measure the degree at which two annotators rank candidate LVCs in the same relative order. SRC is based on Pearson's correlation coefficient (Altman, 1991), but differs in that it is a non-parametric rank statistic: the statistical ranking of data points, rather than their values, is considered. Since the majority of members in the Levin and WordNet semantic classes we employ are rated 'poor', a measure like SRC, which does not assume a normal distribution, is appropriate. To calculate SRC, the scores assigned to each candidate LVC by each annotator are ranked, from highest to lowest. If two or more candidates are ranked identically by an annotator, the mean rank of all tied candidates is used. A high SRC score is achieved if candidates frequently have the same ranking assigned to them by both annotators. Like Pearson's correlation coefficient, scores range from -1 to 1: a score of 1 indicates perfect agreement, while a score of -1 indicates perfect disagreement.

Unfortunately, there are situations in which SRC scores can seem not to accurately reflect the data. As ratings generated by our computational measures are continuous, while the human ratings are more discrete, there are many ties among the human ratings and very few, if any, among those of the computational measures. Such a situation can lead to SRC scores which vary significantly with slight changes to the data. For example, given the data in Table 4.5, the SRC score is .85, indicating a significant positive correlation. However, if Rater B's rating of .25 is changed slightly, to .20, the SRC shifts to .17, indicating a marginal positive correlation. This occurs because the average rank of the tied ratings (the '1's) has changed from being lowest ranked, when compared to the Rater B's rankings, to being ranked as mid-range entries, and

Raw Data		**Ranked Data**	
Rater A	Rater B	Rater A	Rater B
1	.21	2.5	1
1	.22	2.5	2
1	.23	2.5	3
1	.24	2.5	4
2	.25	5	5
3	.50	6	6

Table 4.5: Illustrative example data for use with Spearman Rank Correlation.

so the SRC shifts accordingly. This is obviously an extreme example, and not a fair demonstration of SRC, but similar behaviour is also apparent to a lesser degree in more realistic data drawn from an appropriately sized sample. As our experiments include many candidate LVCs with human ratings of 1, this behaviour can affect results. We therefore also include another measure, Weighted Kappa, which is used along with SRC to present a more complete picture of the data.

4.4.2 The Kappa Statistic

Kappa (Cohen, 1960; Carletta, 1996) compares agreement of two annotators using discrete labels and accounts for agreement due to chance. A Kappa value of 0 indicates no agreement better than allowed for by chance, while Kappa values of 1 and -1 indicate perfect agreement and perfect disagreement, respectively. Given an $m \times m$ contingency table, Kappa is defined as $\kappa = (p_o - p_e)/(1 - p_e)$, where p_o is observed agreement and p_e is expected agreement. Observed agreement is defined as the sum of the diagonal divided by the total number of ratings: $p_o = ((0,0) + (1,1) + ... + (m,m))/n$. Expected agreement is defined as $p_e = (p_1q_1 + p_2q_2 + ... + p_mq_m)/n^m$, where p_i is the sum of row p and q_i is the sum of column i.

A limitation of Kappa when applied to ordered categorical data is that all disagreements are

weighted equally. If a candidate is marked by one annotator as 'poor' and the other as 'fair', this disagreement is considered just as strong as one in which a candidate is rated 'poor' by one annotator and 'good' by another. Weighted Kappa (Cohen, 1968), a generalization of Kappa which treats near-agreements as being "more correct" than clear disagreements, is therefore employed. For our three categories of 'poor', 'fair', and 'good', a standard linear rating scheme is used, in which the weight for the row i, column j is given by $1 - (|i - j|/2)$: full agreement is credited at 100%, near agreement at 50%, and full disagreement is not credited.

To calculate Weighted Kappa, computational ratings and human judgments are put into buckets. Again, three buckets are employed, corresponding to 'poor', 'fair', and 'good' candidates. The human ratings the employ same thresholds established in Section 4.2.2 and used throughout this paper. As none of the computational measures save PMI are understood to have a point of no association, all computational thresholds are determined through experimentation on development data. A range of thresholds is tested, and each is judged by its bucket distribution: thresholds which generate a distribution of candidates most similar to those of the human judges are chosen. In cases where two or more thresholds have bucket distributions that are both optimal and comparable to one other, the threshold with the higher Kappa score is chosen.[6] The thresholds employed are listed in Table 4.6, and reflect a coarser level of LVC acceptability: rather than continuously quantifying the acceptability of each construction, we simply require that it be classified as 'good', 'fair', or 'poor'. It is believed that better agreement will be achieved with these coarser measures than with their more detailed versions. Note that as thresholds for the computational measures are chosen to reflect the bucket distribution of the human ratings, tests of marginal homogeneity (which compare one rater's propensity for using each rating category to another's) would not be revealing.

However, Kappa has "paradoxes" in its interpretation, as noted by Feinstein and Cicchetti

[6]This process favours a similar bucket distribution over a high Kappa score. However, the process of examining Kappa first, and then considering bucket distribution in order to break ties, results in very similar thresholds being chosen. The advantage of considering bucket thresholds first is that outlier cases—cases in which a high Kappa score is achieved, by chance, on data with an improbable bucket distribution—are not selected.

Computational Measure	**'Poor' Threshold**	**'Good' Threshold**
PMI	≤ 0.0	≥ 1.0
LVC-PMI	≤ -4.5	≥ -3
LVC-Prob	$\leq 0.5 \times 10^{-10}$	$\geq 1.5 \times 10^{-10}$
LVC-Freq	≤ 150	≥ 1000

Table 4.6: Thresholds established for placing the output of each computational measure into 'good', 'fair', and 'poor' buckets.

		Rater A		
		Yes	No	Total
Rater B	Yes	40	9	49
	No	6	45	51
	Total	46	54	100

		Rater A		
		Yes	No	Total
Rater B	Yes	80	10	90
	No	5	5	10
	Total	85	15	100

Table 4.7: Two related 2×2 contingency tables. Both have observed agreement of .85. The left-hand table has $\kappa = .70$ while the right-hand table has $\kappa = .32$.

(1990), Lantz and Nebenzahl (1996) and others: data with high observed agreement may receive a low Kappa score if one category of data is significantly more prevalent than others. Consider Table 4.7, drawn from Byrt et al. (1993). While both contingency tables have a high observed agreement of .85, the left-hand table has a Kappa score of .70 while the right-hand table has a Kappa score of only .32 (Byrt et al., 1993). This low score is due to high agreement on a single category causing very high observed agreement, which affects the measure of chance agreement. As Kappa is normalized so that chance agreement is scored at 0, lower Kappa scores than might be expected are returned.

Cicchetti and Feinstein (1990) claim this behaviour is a "desirable" quality of the measure: it can be seen as reflecting the difficulty in rising above chance agreement in situations where naïvely choosing a particular label makes agreement likely. However, as the thresholds for our

computational measures are chosen such that their bucket distribution reflects that of the human ratings—they are not chosen to unjustifiably favour a particular category—the "punishment" applied by Kappa for high agreement on a single category is undesirable. We agree with Lantz and Nebenzahl (1996), Byrt et al. (1993) and others in suggesting that the Kappa statistic alone, as an omnibus measure, does not fully reflect agreement. Along with Kappa scores, we report weighted observed agreement, p_w, so that a more complete picture may be shown. Note that we can compare Kappa scores between our own measures, since the same issues of prevalence apply, but comparing our Kappa scores to others drawn from more evenly distributed data is problematic.

Chapter 5

Experimental Results

Our experiments focus on several different aspects of light verb constructions. We hypothesize that semantically similar complements may have the same pattern of co-occurrence across light verbs, and wish to examine this hypothesis by comparing the behaviour of the complement in an LVC across semantic classes. We expect to find distinct patterns of acceptability among the semantic classes extracted from Levin (1993) and WordNet, and across the light verbs themselves. We believe the differences between the trends of acceptability of the WordNet and Levin classes may indicate whether grouping PNs according to both nominal and verbal senses is more appropriate for LVCs than grouping them by verbal information alone. We test these hypotheses by comparing human acceptability judgments, which we treat as a standard, to our four computational measures.

The results of our experiments are presented in this chapter. In the next section, the properties of the human acceptability judgments are considered. We then examine the performance of our computational measures, first at the more detailed level captured by Spearman Rank Correlation (SRC), and then at a coarser level of acceptability measured by the Weighted Kappa statistic. Finally, we evaluate the performance of our data filtering techniques, and examine results with and without these techniques applied.

5.1 Human Acceptability Judgments

As noted in Section 4.2.2, we employ pilot results in which two expert native speakers of English rate the acceptability of each potential "LV a PN" construction. On test data, the two sets of Levin class ratings yielded linearly weighted Kappa values of .72, .39, and .44, for *take*, *give*, and *make*, respectively, and .53 overall. WordNet class ratings yielded linearly weighted Kappa values of .79, .66, and .69, for *take*, *give*, and *make*, respectively, and .71 overall. Discussion of instances of disagreement when rating Levin classes led to more consistency when rating WordNet classes.[1]

We average both sets of ratings to form a consensus set, in order to judge the computational measures against a single standard. Before ratings were averaged, the two raters met and discussed the cases in which they disagreed by more than one point. In test data, this led to 4% of the ratings being changed. Note that this is 4% of ratings, not 4% of items, as in some cases both raters changed their ratings after discussion. For any remaining differences, the average of both ratings was used.

We examine trends in human ratings across light verbs and the semantic classes of their complements by placing the (consensus) human ratings in buckets of 'poor', 'fair', and 'good'. Again, the thresholds established in Section 4.2.2 are employed: the 'poor' bucket contains ratings from $[1–2)$, the 'fair' bucket contains ratings from $[2–3)$, and the 'good' bucket contains ratings of 3 and above. Table 5.1 shows the proportion of candidates rated 'fair' or above for each light verb across the Levin classes. The light verbs show very different acceptability with different Levin classes—for example, *give* is fairly good with 43.2, while *take* is very bad, and the pattern is reversed for 51.4.2. Overall, *give* allows more LVCs than the other two light verbs.

Table 5.2 shows a similar distribution of 'fair' or above ratings for each light verb across

[1]Trends were similar on development data. Agreement on development Levin classes was lower due to initial differences in interpretation of the ratings. The Levin classes yielded scores of .37, .23, and .56 for *take*, *give*, and *make*, respectively, and .38 overall. WordNet classes achieved linearly weighted Kappa scores of .77, .70, .63 for *take*, *give*, and *make*, with .71 overall.

Levin Classes					
		Proportion of Acceptable LVCs			
Class	Size	*take*	*give*	*make*	Overall
18.1,2	35	.23	.51	.26	.33
30.3	18	.28	.44	.17	.30
43.2 *	35	.03	.54	.26	.28
51.4.2	10	.70	.30	.10	.37
Overall	96	.21	.49	.22	.31

Table 5.1: The proportion of constructions rated 'fair' or above in Levin test classes. A '*' indicates a random subset of verbs were used in the class.

WordNet Classes					
		Proportion of Acceptable LVCs			
Class	Size	*take*	*give*	*make*	Overall
18.1,2	35	.26	.60	.17	.34
30.3	35	.14	.20	.11	.15
43.2	35	.09	.34	.34	.26
51.4.2	35	.46	.31	.23	.33
Overall	140	.24	.36	.21	.27

Table 5.2: The proportion of constructions rated 'fair' or above in WordNet test classes.

the WordNet classes. Again, *give* allows more LVCs than the other light verbs. Comparing Levin and WordNet classes, we find similar trends: class 43.2 is poor with *take* but better with *give*, and 51.4.2 has the opposite pattern. While the proportions of acceptable LVCs across light verbs are not identical to those of the Levin classes, they are very similar. The method we use to choose a representative seed for these WordNet classes may be responsible for their similarity to their Levin counterparts.

These experiments indicate that WordNet can be used as a source of semantic groupings of PNs. This is a significant result, as Levin classes appropriate for LVCs are infrequent. Interestingly, class 30.3 is the only class in which a fully representative seed could not be found, and this class exhibits the largest overall difference between Levin and WordNet versions. Examining individual light verbs, we see that classes 30.3 and 51.4.2 show the largest difference between Levin and WordNet versions; these are also the two classes with the largest difference in size between the two class types. Part of this difference in the proportion of acceptable LVCs may stem from the larger sample size in the case of the WordNet classes.

5.2 Computational Measures of LVC Acceptability

We focus on the analysis of our four computational measures on unseen test data: in most cases trends are similar on development data, and any differences between the two are noted. Recall that we are interpreting PMI as an informed baseline. We first consider SRC results, which illustrate the performance of our measures at a fine-grained level, and examine trends in correlation by light verb before considering more-detailed class-based behaviour. Weighted Kappa scores are then explored, which reflect the performance of our measures at a coarser level of acceptability. Finally, we examine the performance of our filtering techniques.

5.2.1 Correlation Between Computational and Human Ratings

Table 5.3 shows trends across light verbs for each measure when run against candidates drawn from Levin classes; Table 5.4 shows performance when tested against WordNet classes. Generally, reasonably good correlations with human ratings are seen across most measures. Considering performance across both Levin and WordNet, we see that while both LVC-PMI and LVC-Prob outperform the baseline, the latter does so by a wider margin. The only measure which is generally worse than the baseline is LVC-Freq.

The light verb *take* achieves the best correlations on both Levin and WordNet classes, fol-

Spearman Rank Correlation Scores: Levin Classes				
Light Verb	PMI	LVC-PMI	LVC-Prob	LVC-Freq
take	.50	.53	.55	.51
give	.27	.30	.47	.36
make	.24	.33	.46	.10
average	*.34*	*.38*	*.49*	*.32*

Table 5.3: Spearman Rank Correlation scores for each computational measure when tested against candidate constructions drawn from Levin classes.

Spearman Rank Correlation Scores: WordNet Classes				
Light Verb	PMI	LVC-PMI	LVC-Prob	LVC-Freq
take	.59	.61	.62	.55
give	.50	.52	.51	.24
make	.31	.35	.33	.31
average	*.47*	*.49*	*.49*	*.37*

Table 5.4: Spearman Rank Correlation scores for each computational measure when tested against candidate constructions drawn from WordNet classes.

lowed by *give* and *make*. Similar patterns were found on development data, although the SRC scores achieved here were generally slightly higher. The poorer correlations with *make* and *give* may be partly due to the difficulty in rating the constructions: both annotators reported having more difficulty with LVCs involving these light verbs than with *take*.

Comparison of correlation scores between Levin and WordNet classes

A comparison of the correlations of the Levin classes to those of the WordNet classes shows some interesting differences. While the general trends across light verbs are the same for

candidates generated using both Levin and WordNet classes (all measures perform best with candidates involving *take*, for instance), LVC-PMI and LVC-Prob share similar performance when run against WordNet classes, and do not appreciably outperform the baseline. However, the baseline PMI measure performs considerably better against the WordNet classes than it does against the Levin classes, which may indicate that these candidate constructions are easier to classify using PMI. (On development data, the average SRC scores achieved by all measures were very similar between groups of semantic classes, differing at most by .02.) This boost in the baseline also helps LVC-PMI achieve better correlations, as it is a measure based on PMI. LVC-Prob does not perform significantly worse against WordNet classes than it does against Levin classes, but neither does it perform significantly better: against a baseline achieving better correlation, it fails to distinguish itself.

LVC-Prob's failure to exceed the performance of the WordNet baseline can be attributed to literal usages of common complements. Any phrase with the same word order of an LVC is interpreted by all our measures as evidence of LVC usage. For example, the candidate *take a string* (WordNet class 30.3) is often used literally in a programming context: *This function takes a string and an integer*. While this affects both groups of semantic classes, those automatically extracted from WordNet are primarily affected: PNs with a stem form rarely used as a verb and frequently used as a noun seem more open to a literal interpretation, and these PNs are more often found in the WordNet classes than in those of Levin.

While all measures (incorrectly) treat such literal phrases as evidence of LVC usage, LVC-Prob has particular difficulty with literal uses of unacceptable constructions involving common complements. Candidate constructions such as *make a train* (WordNet class 51.4.2) and *take a link* (WordNet class 43.2) are rated anomalously high by LVC-Prob because the general frequency of the complement ($\Pr(PN)$) is included in the measure. When the complement of a candidate construction is frequently employed throughout the corpus, as *train* and *link* are, the significance of any evidence of LVC usage is overrated. Further, as literal usages of a light verb and complement are interpreted as evidence of LVC usage, this problem is compounded. Since

the PMI-based measures do not take the overall frequency of the complement into consideration, they do not rate these unacceptable candidates as highly, and perform better against these candidates than LVC-Prob. PNs with common nominal forms seem more open to this literal interpretation, and as PNs with relatively rare verb senses seem less likely to be included in the Levin classes, this can account for the less remarkable performance of LVC-Prob when tested against WordNet candidates.

Finally, both the Levin and WordNet classes include PNs with stem forms commonly used as adjectives and rarely used as nouns. For example, the PN *clear* has only two noun senses listed in WordNet, both of which seem relatively rare and specific: *clear* as in escaping punishment (*Looks like you're in the clear*), and *clear* as in being in the open: *We left the forest and into the clear*. However, while the candidate *give a clear* is rated 'poor' by both annotators, it is rated quite highly by the computational measures, which hurts correlation scores. This is primarily due to its use as an adjective: many phrases were found referencing clear explanations, signals, pictures, etc. As will be discussed in Section 5.2.3, while some of these phrases were filtered, not every instance appears with enough frequency to be flagged, and those instances flagged can be subject to underestimation.

While the classes automatically extracted from WordNet may be more noisy than those selected from Levin (1993), the WordNet sets nevertheless have average SRC scores exceeding those of the Levin classes. As there are a limited number of Levin classes appropriate for our task, the use of WordNet as a source for semantic groupings of predicative nouns is recommended.

Trends in acceptability by semantic class

Table 5.5 shows class-based SRC scores on unseen candidate LVCs generated with Levin (1993), while Table 5.6 shows the same information with unseen candidates generated using WordNet. Distinct trends in LVC acceptability are shown by these classes. Examining SRC performance on Levin classes alone, we see that, with two exceptions, LVC-PMI meets

Spearman Rank Correlation Scores: Levin Classes					
Light Verb	Class	PMI	LVC-PMI	LVC-Prob	LVC-Freq
take	18.1,2	.47	.56	.54	.49
	30.3	.56	.57	.60	.53
	43.2 *	.43	.43	.51	.31
	51.4.2	.54	.54	.55	.69
	average	*.50*	*.53*	*.55*	*.51*
	std. dev.	*.06*	*.06*	*.04*	*.16*
give	18.1,2	.26	.26	.54	.41
	30.3	.28	.45	.62	.72
	43.2 *	.39	.36	.45	.48
	51.4.2	.16	.13	.25	−.19
	average	*.27*	*.30*	*.47*	*.36*
	std. dev.	*.09*	*.14*	*.16*	*.39*
make	18.1,2	.29	.45	.52	.45
	30.3	.26	.47	.43	−.03
	43.2 *	.09	.07	.17	−.40
	51.4.2	.32	.32	.73	.38
	average	*.24*	*.33*	*.46*	*.10*
	std. dev.	*.11*	*.18*	*.23*	*.39*
all	average	*.34*	*.38*	*.49*	*.32*
	std. dev.	*.14*	*.16*	*.15*	*.35*

Table 5.5: Spearman Rank Correlation scores for each computational measure across each light verb and Levin class. A '*' indicates a random subset of verbs were used in the class.

or exceeds the baseline across all light verbs and classes. We see further that in every instance LVC-Prob outperforms the baseline. The only measure to perform generally worse than the baseline is LVC-Freq, which at worst achieves negative correlations on the semantic classes

Spearman Rank Correlation Scores: WordNet Classes					
Light Verb	Class	PMI	LVC-PMI	LVC-Prob	LVC-Freq
take	18.1,2	.55	.62	.69	.49
	30.3	.38	.39	.46	.45
	43.2	.63	.64	.59	.48
	51.4.2	.78	.79	.74	.78
	average	*.59*	*.61*	*.62*	*.55*
	std. dev.	*.17*	*.16*	*.12*	*.15*
give	18.1,2	.57	.60	.63	.41
	30.3	.57	.57	.51	.14
	43.2	.65	.66	.49	.25
	51.4.2	.23	.23	.42	.17
	average	*.50*	*.52*	*.51*	*.24*
	std. dev.	*.19*	*.19*	*.09*	*.12*
make	18.1,2	.44	.47	.45	.22
	30.3	.40	.34	.34	.45
	43.2	.13	.35	.14	.20
	51.4.2	.27	.25	.38	.37
	average	*.31*	*.35*	*.33*	*.31*
	std. dev.	*.14*	*.09*	*.13*	*.12*
all	average	*.47*	*.49*	*.49*	*.37*
	std. dev.	*.19*	*.18*	*.16*	*.18*

Table 5.6: Spearman Rank Correlation scores for each computational measure across each light verb and WordNet class.

found most difficult by the other measures. Again, against WordNet classes, PMI, LVC-PMI, and LVC-Prob perform similarly.

Some of the worst correlations, for all measures, are with the Levin and WordNet class

43.2 (Verbs of Sound Emission). Here, it seemed that several salient senses of the words were missed by the human raters. While it is acknowledged that using only two raters to reach consensus is not ideal, it was hoped that two raters with different backgrounds would be aware of most senses of a given candidate LVC. However, this was not always the case: class 43.2 in particular has many instances in which there are senses unfamiliar to the human raters. For example, the candidate *take a click* is rated '1' by both raters, yet it is among the highest-rated candidates of all measures. This is primarily due to phrases using the construction in an online sense, such as *take a click here* (with your mouse), *take a click on the wild side*, etc: such usages were unfamiliar to both annotators. The phrase *make a click* similarly revealed usages such as *make a click on the button* which were thought by human annotators to be largely unacceptable. This underlines the difficulty for humans in rating a semi-productive construction.

Levin class 51.4.2 is the semantic class most susceptible to the swings in SRC discussed in Section 4.4.1: with ten members, it is the the smallest class employed in our experiments, and further, 70% of its candidate LVCs involving the light verb *make* are rated 1 (unacceptable) by the human annotators. Comparing the performance of our four measures on this particular class and light verb, we see that LVC-Prob's correlation of .73 significantly outperforms the other measures, all of which have SRC scores less than .38. While these scores do reflect better correlation in the case of LVC-Prob, this difference is slightly exaggerated due to the high proportion of tied ratings. LVC-Prob benefits from this exaggeration because of the ratings it assigns to the three non-tied members in this class, while the other measures do not.

While the performance of LVC-Freq on *give* candidate LVCs drawn from WordNet is well below that of the baseline, LVC-Freq outperforms the baseline on candidate LVCs involving *give* drawn from Levin. This is somewhat surprising, as LVC-Freq does not detect the dative form, but this high SRC is accompanied by a very high standard deviation, indicating significant fluctuation in results. The class with which LVC-Freq achieves its best performance with *give*, Levin class 30.3, includes many PNs such as *peep* and *sniff* which can be employed in both dative and non-dative LVCs. Correspondingly, the members of Levin class 51.4.2, with

which LVC-Freq achieves its worst Levin *give* correlation, includes many candidate complements such as *ride* and *paddle* which seem to be employed most often in dative LVCs.

Finally, it is worth noting that common typographic errors form an unexpectedly strong source of noise in some test classes. This affects some candidates more than others: a particular example is the unacceptable candidate *make a think* from WordNet class 30.3, which is rated highly by computational measures due to a substantial number of results which are clear typographic errors. Phrases like *make a think sauce* are obviously intended either to be *make a thick sauce* or *make a thin sauce*, but as our measures have no way of knowing this, these phrases are considered to be evidence for LVC usage. This typo seems especially prevalent because the words *thin* and *thick* are each one erroneous keystroke away from the word *think*.

5.2.2 Agreement Between Computational and Human Ratings

We now inspect the performance of our computational measures when a coarser level of acceptability is considered, in which an acceptability label of 'poor', 'fair', or 'good' is applied to each candidate construction rather than a more precise rating. Table 5.7 shows Weighted Kappa and weighted observed agreement for the Levin classes, and Table 5.8 shows the same information for the WordNet classes. With the acceptability thresholds developed in Section 4.4.2 applied, all measures generally perform comparably well. LVC-Prob and LVC-Freq slightly outperform the PMI-based measures when tested against Levin classes, while again, against the WordNet classes, the differences between measures are less pronounced. Most interesting is the performance of LVC-Freq, which achieves the best performance of all measures when tested against Levin classes, and only moderately underperforms when tested against WordNet classes. It seems clear that while the computational measures do not all make the same fine-grained distinctions of LVC acceptability, they perform more similarly at the simpler task of merely distinguishing good candidate constructions from fair and poor ones.

Values of κ_w are generally low, for the reasons of prevalency discussed in Section 4.4.2. While the "paradox" of low Kappa scores with high observed agreement applies to all classes,

Linearly Weighted Kappa and Agreement **Levin Classes**								
Light Verb	PMI		LVC-PMI		LVC-Prob		LVC-Freq	
	κ_w	p_w	κ_w	p_w	κ_w	p_w	κ_w	p_w
take	.44	.77	.43	.79	.39	.85	.35	.80
give	.23	.59	.25	.59	.36	.77	.56	.86
make	.13	.81	.17	.79	.18	.82	.14	.82
average	*.27*	*.71*	*.26*	*.72*	*.28*	*.81*	*.30*	*.81*

Table 5.7: Weighted Kappa (κ_w) and weighted observed agreement (p_w) for each computational measure when tested against candidate constructions from Levin classes.

Linearly Weighted Kappa and Agreement **WordNet Classes**								
Light Verb	PMI		LVC-PMI		LVC-Prob		LVC-Freq	
	κ_w	p_w	κ_w	p_w	κ_w	p_w	κ_w	p_w
take	.57	.88	.52	.86	.50	.86	.42	.82
give	.35	.74	.36	.74	.42	.80	.27	.73
make	.26	.80	.22	.78	.22	.74	.27	.73
average	*.39*	*.82*	*.39*	*.81*	*.36*	*.80*	*.30*	*.75*

Table 5.8: Weighted Kappa (κ_w) and weighted observed agreement (p_w) for each computational measure when tested against candidate constructions from WordNet classes.

Levin class 43.2, when combined with the light verb *take*, illustrates this issue particularly well. The similar contingency tables generated here by the PMI and LVC-Prob measures are shown in Table 5.9. Both the PMI and LVC-Prob measures have approximately equal (and very high) weighted observed agreement of $\geq .94$, but while PMI has $\kappa_w = .66$, LVC-Prob

		Human Ratings			
		Poor	Fair	Good	Total
PMI	Poor	34	0	0	34
	Fair	0	0	1	1
	Good	0	0	0	0
	Total	34	0	1	35

		Human Ratings			
		Poor	Fair	Good	Total
LVC-Prob	Poor	33	1	0	34
	Fair	1	0	0	1
	Good	0	0	0	0
	Total	34	1	0	35

Table 5.9: Two 3×3 contingency tables for the Levin class 43.2 with the light verb *take*. The leftmost table reflects the PMI measure, and has a weighted Kappa score of .66. The rightmost table reflects the LVC-Prob measure, and has a $\kappa_w = -.03$. Both have high observed agreement of $\geq .94$.

has $\kappa_w = -.03$. Examples such as these make it clear that our results are susceptible to large swings in Kappa scores with only slight changes in agreement. For this reason, p_w is considered to be a more telling (and reliable) way of comparing performance between measures, although average Kappa scores are not discounted entirely. The trends across average κ_w and p_w are quite similar.

5.2.3 Analysis of Filtering Techniques

Three techniques for filtering World Wide Web data acquired via a general purpose search engine, described in Section 4.3.3, are explored in this work. Briefly, these methods are Punctuation Filtering, which removes internally-punctuated phrases such as *take. A walk* from search results; Phrase Filtering, which removes non-LVC phrases like *give a strip of land* from the search results for the candidate *give a strip*; and Multiword Expression (MWE) Filtering, which removes multiword expressions like *give a slide presentation* from the search results for the candidate LVC *give a slide*. In Table 5.10 we compare the SRC scores of the LVC-Prob measure when no filtering is applied, when only Punctuation Filtering is applied, and when all three filtering techniques are applied. While the data we consider here reflects only the correlation scores of the LVC-Prob measure, the same trends are found when filtering is applied to

LVC-Prob Spearman Rank Correlation Scores Across Filtering Levels												
	No Filtering				Punctuation Filtering				Full Filtering			
Class	*take*	*give*	*make*	avg.	*take*	*give*	*make*	avg.	*take*	*give*	*make*	avg.
18.1,2	.50	.55	.56	.53	.52	.54	.55	.54	.54	.54	.52	.53
30.3	.60	.61	.41	.54	.62	.61	.41	.55	.60	.62	.43	.55
43.2 *	.49	.46	.13	.36	.49	.44	.10	.34	.51	.45	.17	.38
51.4.2	.62	.29	.79	.57	.57	.29	.79	.55	.55	.25	.73	.51
18.1,2-wn	.66	.61	.40	.55	.69	.61	.38	.56	.69	.63	.45	.59
30.3-wn	.40	.49	.31	.40	.46	.54	.31	.44	.46	.51	.34	.44
43.2-wn	.56	.45	.15	.39	.57	.45	.12	.38	.59	.49	.14	.41
51.4.2-wn	.76	.48	.43	.56	.78	.45	.40	.54	.74	.42	.38	.51
average	*.58*	*.50*	*.41*	*.49*	*.59*	*.49*	*.40*	*.49*	*.59*	*.49*	*.40*	*.49*

Table 5.10: Spearman Rank Correlation results using LVC-Prob with no filtering is applied, with Punctuation Filtering applied, and with Punctuation, Phrase, and MWE Filtering all applied. A '*' indicates a random subset of verbs were used in the class. Classes ending with '-wn' are generated with WordNet.

the PMI and LVC-PMI measures as well.[2]

Punctuation Filtering

Punctuation Filtering has little effect on correlation scores. Average SRC scores with and without Punctuation Filtering applied are nearly identical; however, this is because noise due to internal punctuation tends to apply approximately equally to all candidates considered. No semantic class or light verb is especially sensitive to this source of noise, although all suffer from it. On average, Punctuation Filtering removed 4% of instances from result sets, with a standard deviation of 1%. While correlation results fail to change significantly, Punctuation Filtering is an efficient filtering method, as many instances of noise are removed at very little

[2]Recall that no filtering was applied to the LVC-Freq measure, as filtering out noise in this measure would be counterproductive.

cost: no additional searches are required. On searches with no wildcards, Punctuation Filtering has very few false positives, as it is rare for acceptable LVCs without internal modification to have internal punctuation. On searches with wildcards, Punctuation Filtering continues to remove the targeted noise, although it does begin to filter out more valid LVCs. We therefore consider Punctuation Filtering to be beneficial, even if results do not change considerably.

Phrase and MWE Filtering

Very little overall change is observed when Phrase and MWE Filtering are applied. There is a slight improvement in some classes, but this is coupled with a decrease in correlation in other classes; overall, results are very similar. Three reasons are apparent for this behaviour: first, some LVCs appear more often with phrase and MWE modification than without. Secondly, while instances of noise are filtered from results, so are instances of valid LVC usage. Finally, the sample size available to the filtering techniques is typically insufficient to correctly estimate the prevalence of noise. We now consider these issues in more detail.

We assume with filtering techniques that any LVC capable of appearing with phrase or MWE modification is just as capable of appearing without it, but there are cases where this is not an appropriate assumption. The LVC *take a leap* (WordNet class 51.4.2) is a good example of this. While the LVC *take a leap* (off a building) seems acceptable, the collocated phrase and near idiom *take a leap of faith* is so prevalent that it forms the majority of the context returned with the phrase *take a leap*. Additionally, though to a lesser extent, the phrases *take a leap forward* and *take a leap backward* are also very common. These constructions are flagged (incorrectly) as noise and filtered; with them removed, evidence of LVC usage of *take a leap* is reduced by 54%. The LVC is judged by computational measures to be much poorer than it otherwise would be.

Both Phrase Filtering and MWE Filtering are clearly capable of erroneously flagging collocated phrases associated with an LVC as noise. Like *take a leap forward*, other very common phrases are removed from different candidates: *give a knock* has filtered the phrase *knock any-*

way, along with such noise as *knock out* (punch). The PMI thresholds employed by our filtering techniques are established to minimize these false positives, but they do still occasionally occur.

Most critically, the thousand-instance ceiling on Google searches severely constrains the Phrase and MWE filtering techniques. While we attempt to work around this constraint with focused searches for specific phenomena (the "LVC+of" searches detailed in Section 4.3.3), it is most often the case that the filtering measures are extrapolating from a relatively small and non-random sorted sample. As a result, the amount of noise affecting candidates tends to be underestimated to various degrees, and an insufficient number of results are removed to appreciably alter candidates' scores. We did attempt various techniques for increasing the amount of results removed when a noisy result is identified, but these techniques would often increase the skew: some classes improved, others got worse, and the overall effect was minimal. The failure of these filtering techniques to have a large impact on improving correlation can be partially attributed to a lack of data, stemming from the limited methods available for accessing our corpus. As an example, WordNet class 43.2 has the candidate *give a phone* rated quite high by computational measures—as high as second place by LVC-Prob. Most of the evidence comes from phrases like *phone call*, *phone number*, etc. When MWE Filtering is applied these phrases are identified and filtered, but as extrapolation from the thousand instances available is unreliable, MWE Filtering tends to underestimate. After MWE Filtering is applied, the candidate *give a phone* is reduced in rank but still remains rated inappropriately high, appearing in the middle range of candidates and above candidates that seem more acceptable, such as *give a click*.

5.2.4 Summary of Experimental Results

Our experiments show that LVC-PMI is a slightly improved measure over PMI at the task of rating candidate light verb constructions, and that LVC-Prob is more markedly improved over both PMI and LVC-PMI. The LVC-Freq measure was found to be poor at making fine-grained distinctions of acceptability, but comparable to the other measures at making coarse-grained

distinctions.

We hypothesized that while PMI should indicate a collocation, LVC-PMI should focus on LVCs in particular, and LVC-PMI does show slightly stronger correlations. Further, LVC-PMI is an improvement over previous work on LVCs, as unlike the earlier DiffAll measure (Stevenson et al., 2004), it generally exceeds the performance of PMI across all light verbs. LVC-Prob has the best performance and is the preferred measure, as it requires only slightly more data than PMI, and far less than LVC-PMI.

Distinct patterns of acceptability were found across the semantic class of the complement to an LVC, as well as across light verbs themselves. Semantically similar complements are shown to have the same pattern of co-occurrence across light verbs. In general, the results confirm our hypothesis that the semantic class of the complement is highly relevant to measuring the acceptability of "LV a PN" LVCs.

WordNet classes are shown to reflect the trends in LVC usage of the Levin class from which their seed is taken. While the WordNet classes include more members which are used literally than those of the Levin classes, they do provide a useful way of grouping together semantic classes of LVC complements. This is an important result, as Levin classes appropriate for LVCs are infrequent.

The LVC-Prob measure has particular difficulty in rating the acceptability of literal candidates with very common complements, and many instances of such candidates are found in the WordNet classes. This accounts for the worse performance of LVC-Prob against the WordNet classes, when compared to the baseline. However, despite this behaviour LVC-Prob performs just as well as the baseline when tested against WordNet classes, and outperforms it when tested against Levin classes, where literal candidates are less common.

Finally, a need to look in more detail at the properties of different light verbs is indicated. The light verb *make*, which we hypothesized to have behaviour different from *take* and *give*, does perform worse than *take*, but at about the same level as *give*. Gathering human ratings for *give* and *make* is difficult, however, which may explain the low correlation scores. Candidates

involving the light verb *take*, with which we achieve our best results, were easiest for our annotators to rate.

Chapter 6

Conclusions

Light verb constructions are a semi-productive class which, like most multiword expressions, pose the familiar problem of whether or not (and how) they should be listed in a computational lexicon. Our goal in this work has been to investigate the (semi-)productivity of light verb constructions and to develop computational measures for quantifying their acceptability. We focused on the particular class of light verb construction (LVC) which employs a predicative noun (PN) as its complement. We predicted that distinct patterns of acceptability would be found across semantic classes and light verbs, and have shown this to be the case. Pilot experiments which capture human acceptability judgments of light verb constructions have been performed and used as a gold standard with which to judge computational approaches capturing their acceptability.

Candidate LVCs were formed by combining the English light verbs *take*, *give*, and *make* with complements drawn from two different ontologies. The semantic verb classes of Levin (1993) were employed, along with automatically constructed semantic classes extracted from both the nominal and verbal hierarchies of WordNet 2.0 (Fellbaum, 1988). A novel approach was employed to extract appropriate complements from WordNet for use in LVCs, and the semantic classes generated by this approach are shown to reflect the trends in acceptability found in corresponding Levin classes.

We developed four different computational measures for quantifying LVC acceptability, and we have compared the performance of these measures across ontologies, light verbs, and semantic classes. We have examined the performance of these measures both at a fine-grained level of acceptability, at which all candidate constructions have a unique acceptability score assigned, and at a coarser level, at which only rank labels of 'poor', 'fair', and 'good' are assigned. Each computational measure employs World Wide Web data gathered via a general-purpose search engine, and we have tested the performance of each with and without various novel approaches to filtering web data applied. In this chapter, we discuss contributions and limitations of our work, and detail possible extensions.

6.1 Summary of Contributions

Statistical measures of LVC acceptability. Three novel measures are suggested for quantifying LVC acceptability. LVC-PMI is an extension of pointwise mutual information (PMI) incorporating linguistic knowledge, LVC-Prob is a probability formula which measures the likelihood of a given light verb and predicative noun complement forming an acceptable construction, and LVC-Freq is a simple measure which rates candidate LVCs by their frequency of usage in a noisy corpus. While LVC-PMI and LVC-Prob both outperform the baseline PMI measure, LVC-Prob does so by a wider margin, and requires less data than LVC-PMI.

Fine- and coarse-grained evaluation of measures. The PMI, LVC-PMI, LVC-Prob and LVC-Freq measure all perform at different levels when nuanced distinctions of acceptability are desired. The best measure is LVC-Prob, which outperforms the baseline PMI measure when tested against Levin classes, and performs just as well when tested against WordNet classes. However, our results show that at a coarser level of acceptability, at which only a label of 'good', 'fair', or 'poor' is desired, all measures perform about equally well. LVC-Freq can be used as a quick, inexpensive measure for roughly quantifying LVC acceptability, while LVC-Prob can be employed when more-detailed distinctions are called for.

Generation of semantic classes of predicative nouns. Our previous work focused on classes from Levin (1993), which has only a small number of semantic classes appropriate for our task. We have detailed here an approach which allows sets of semantically related predicative nouns to be extracted from WordNet. A process for choosing a representative seed from a Levin class is also specified. When such a seed is employed, the WordNet sets are shown to reflect the trends of LVC acceptability found in the corresponding Levin class from which the seed is taken.

An exploration of wildcards in web data. As some LVCs are rare in smaller classical corpora, our computational measures employ data gathered from public web search engines. This give us access to a vast but noisy corpus, access to which is limited by the search engine interface. Wildcards, which match for one word, were tested in our experiments. For example, a search for *take a * walk* may return pages containing phrases like *take a long walk*. While these searches allow more complex LVC usage to be detected, they also allow in more noise. Various levels of wildcards were tested, and results indicated that using data with no wildcards and one wildcard, combined, gave the highest consensus between computational and human ratings. This result is significant for future work on LVCs employing web data, and may also apply to other constructions which can appear in similar internally-modified forms, such as verb-particle constructions.

A survey of the state of the art. A survey of the state of the art is provided both in the field of linguistic analysis of light verb constructions, and the field of computational approaches to LVCs. Additionally, various web extraction techniques are considered and evaluated.

6.2 Limitations and Future Directions

More-detailed web data. While a large corpus is required for sufficient evidence of LVC usage to be found, using a general-purpose search engine such as Google is limiting, as the

information returned with queries is often too sparse for linguistic application. As a result, all of the computational measures are capable of considering non-LVC usage of a candidate construction as evidence of LVC usage. Unfortunately, without part-of-speech tags and with very little context available, it is difficult to distinguish the phrase *give a clear* (explanation) from the LVC *give a tour*. This limitation negatively affects the accuracy of all measures, and is tied to our decision to employ a search engine as our means of accessing the web. Approaches employing a private crawl of web data may help solve these issues, as linguistic markup could be automatically applied; further, the development of robust, public, linguistically-aware search engines is encouraged.

Different approaches to filtering web data. It is clear that our approach to filtering web data is generally too conservative. The focus on extrapolating from the first 1000 results is likely inappropriate, and it is possible that measures which rely on further search engine requests are called for. While this does incur an additional cost, it would allow a more detailed process to take place. Two approaches suggest themselves. One is to re-run searches after a phrase is identified by the noise filtering techniques. For example, if *take a leap* has *of faith* identified as noise by Phrase Filtering, then a search could be run for *"take a leap" −"of faith"* (or *"take a leap"* NOT NEAR *"of faith"*, if such operators are supported by the search engine), and new sources of noise searched for in these results. These searches would remove all documents in which the construction and the targeted word or phrase co-occur. While they clearly overestimate the prevalence of the targeted noise, such an overestimation may be an acceptable one.

A second technique may be to simply perform additional searches for the candidate LVC combined with each noisy phrase identified. If a phrase p is to be removed from the results of a search for s, rather than examining the thousand instances returned by the search for s for instances of p and extrapolating to the whole, we could instead search for $s + p$ and subtract the number of results found here from those found for s. For example, if *phone*

call and *phone number* were flagged as noise affecting the candidate LVC *give a phone*, $frequencycount(\textit{give a phone})$ could have subtracted from it $frequencycount(\textit{give a phone call})$ and $frequencycount(\textit{give a phone number})$. This process might filter the targeted phrases more accurately, but requires additional searches for each element of noise identified. It may also be that correlation with human rankings would drop using such a technique, as these searches are themselves susceptible to the issues of "false hit" noise they are trying to eliminate. Future work exploring these approaches is called for.

Removing inappropriate members from semantic classes. Both the Levin and WordNet classes include members which are more commonly used as adjectives than as nouns or verbs, or have only very rare nominal senses. We could alter our method of choosing PNs from Levin and WordNet, requiring that candidate members be frequently employed as both a noun and a verb. This would move to exclude such inappropriate members. The British National Corpus, which features part-of-speech tags, is used in this work to ensure that a candidate is frequently used as a verb: requiring that a candidate also be frequently used as a noun would be a simple change that would likely increase the quality of our semantic classes.

More extensive human judgments of LVC acceptability. While the human ratings of candidate LVC acceptability used in this work enabled us to gauge the performance of our computational measures, there were salient senses of complements that were not considered by our two judges. It could be that using judgments of acceptability taken from many raters would solve some of the problems stemming from using only two judges. Additionally, as the set of average ratings across several raters would likely be more fine grained than the set of average ratings across two raters, the issues of comparing discrete to continuous data associated with Spearman Rank Correlation would be lessened. However, such ratings are expensive to gather.

Appendix A

Semantic Classes and Human Acceptability Judgments

This appendix contains a full listing of the semantic classes and the human acceptability judgments used in evaluating this work. Each numerical entry represents the consensus human judgment of the acceptability of the candidate construction formed by the predicative noun (PN) listed to the left and light verb listed above. Ratings range from 1 (completely unacceptable) to 4 (completely acceptable), and partial ratings are permitted. A '*' after a Levin class number indicates a random subset of the verbs in the class were used.

Levin Classes 18.1,2			
PN	*take*	*give*	*make*
bang	1	2.5	2
bash	1	2.75	1.5
batter	1	1	2
beat	1	2	2.5
bump	2	2.25	1.75
butt	1	1.5	1
dash	2.25	1	3.5
drum	1	1	1
hammer	1.5	1.5	1
hit	3.5	3.5	3.25
kick	3.75	4	3
knock	2	3.5	1.75
lash	1	1	1
pound	1	1	1
rap	1	2.25	2.5
slap	1.5	2.5	1
smack	1.5	2.5	1
smash	1	1.5	1
strike	1	1.5	1
tamp	1.5	1.5	1
tap	1	3	1.75
thump	1	1.75	1.75
thwack	1.5	2	1.5
whack	1.75	2	1
bite	4	4	1
claw	1	1	1
paw	1	1	1
peck	1.5	1.5	1
punch	3.25	3.5	1.75
scratch	1.5	2.75	1.75
shoot	1	2.5	1
slug	1	1.5	1
stab	4	2.5	3.25
swat	1	1.5	1
swipe	1.5	1	1.25

Levin Class 30.3			
PN	*take*	*give*	*make*
check	1.5	2	3.25
gape	1	1	1
gawk	1	1	1
gaze	1	1	1
glance	3	2.75	1
glare	1	2.5	1
goggle	1	1	1
leer	1	2.25	1.5
listen	2.5	3	1
look	4	4	1.75
ogle	1	1	1
peek	3.5	2	1
peep	1	1.75	3.25
peer	1.5	1	1
sniff	3.25	2.75	1.5
snoop	1.25	1	1
squint	1	1	1.25
stare	1	1	1

Levin Class 43.2 *			
PN	*take*	*give*	*make*
beat	1	1.5	2.25
bellow	1	3	2.25
boom	1	2.5	2.25
bubble	1	1	1.25
burr	1	1	1
clang	1	2.25	2
clank	1	1.5	1.5
clap	1	2	1.5
click	1	3	2
crunch	1	1	1.5
cry	1.5	4	1.5
hum	1	2.75	1.75
jingle	1	2	1.75
knock	2	3.5	1.75
peal	1	2.5	1
pipe	1	1	1
plink	1	1	1.5
plonk	1	1	1.5
plop	1.25	2	2
putter	1	1	1
ring	1	3.25	1.5
screech	1	3.25	2.25
shriek	1	3.25	1.75
shrill	1	1	1
snap	1	1	1
sputter	1	2	1.5
strike	1	1	1
thunk	1	1.75	1.5
tick	1	2	1.5
ting	1	1.25	1.5
trill	1	2	2
trumpet	1	1.25	1
vroom	1	1.5	1.25
whir	1	2.5	2
whistle	1	3	1.75

Levin Class 51.4.2			
PN	*take*	*give*	*make*
cruise	4	2.5	2.25
drive	4	1.75	1.5
fly	1	1	1
oar	1	1	1
paddle	2.5	1.25	1
pedal	1.5	1	1
ride	4	4	1
row	2	1.25	1
sail	3.5	2	1
tack	2	1.25	2

WordNet Classes 18.1,2			
PN	*take*	*give*	*make*
bang	1.25	2.5	2.5
bash	1	2.75	1.5
batter	1	1	1
beat	1.5	2.25	1.75
blow	3	2	1
box	1	1	1
clash	1	1	1
click	1	2.5	2.5
crack	1.5	2.5	1.5
cry	1.5	4	1.5
drip	1	1	1.5
drum	1	1	1
hum	1	2	1.5
lash	1	1	1
lick	3.5	4	1
pat	1	3.5	1
pick	2.25	1.5	1
poke	2.25	2.75	1
pound	1	1.5	1
rattle	1	2.75	2.75
roll	2.5	3	1.25
shame	1	1	1
shoot	1	1	1
sigh	1	4	1.5
slap	1.75	3	1.25
smack	1.25	2.5	1
snap	1	2.5	1
splash	1.5	1.5	4
stab	4	4	3
step	4	1	1
swing	4	2	1.5
thump	1	2.25	2.5
tip	3	3.5	1
trample	1.25	1	1
whip	1	1	1

WordNet Class 30.3			
PN	*take*	*give*	*make*
add	1	1	1
bluff	1	1	1.5
chill	2.5	3.25	1
chip	1	1.5	1
clean	1	1	1
clear	1	1	1
control	1	1	1
cook	1	1	1
cool	1	1	1
dawn	1	1	1
exchange	2	2	3.75
fill	1	1.25	1
fit	1.5	2.5	1
fly	1	1	1
freeze	1	1	1
grace	1	1	1
heat	1	1	1
increase	2.25	2.25	2.25
name	1.5	3.5	4
open	1	1	1
parallel	1	1	1
pin	1	1	1
plot	1	1	1.5
poison	1	1	1
project	1	1	1
put	1	1	1
redress	1	1	1
rest	4	4	1
run	4	2.25	4
separate	1	1	1
sketch	1.25	1.5	1
spike	1	1.5	1.75
string	1	1	1
temper	1	1	1
think	1	1	1

WordNet Class 43.2			
PN	*take*	*give*	*make*
bang	1.25	2.5	2.5
beat	1.5	2.25	1.75
chatter	1	1	1
click	1	2.5	2.5
couple	1	1	1
drip	1	1	1.5
drum	1	1	1
echo	1	1.75	3
fax	1	1	1
glue	1	1	1
harness	1	1	1
hitch	1	1.75	1
insert	1	1	1
institute	1	1	1
knock	2.25	3.75	2
land	1	1	1
lead	3.5	3	1
link	1	1	2.75
lodge	1	1	1
mutter	1	2.25	1.5
phone	1	1	1
play	1	1	3.5
pop	1	2.5	2.5
rap	1	2.5	2.5
rattle	1	2.5	2.5
round	1	1	1
sigh	1	4	3
surround	1	1	1
tack	2.75	1	1
tackle	1	1	1
tap	1	3.5	2
telephone	1	1	1
tether	1	1	1
thump	1	2.25	2.25
top	1	1	1

WordNet Class 51.4.2			
PN	*take*	*give*	*make*
budge	1	1.5	1
bustle	1	1	1.5
chop	1.5	2.5	1
click	1	2.5	2.5
climb	2.5	2.25	2
crash	1.75	2.5	3.75
cruise	4	1.5	1
dance	1	1.5	1.25
dodge	1	1	1
duck	1	1	1
falter	1	1	1
feed	1	1	1
flex	1	1.5	1
flinch	1	1	1
hitch	1	1.75	1
hop	3.5	2.5	1
hurl	1	1.5	1
kick	2.75	4	2
leap	4	3	3
leave	4	2.75	1
mount	1	1	1
roll	2.5	2.5	1.25
row	2	1.5	1
run	4	2	3.75
sail	4	1.25	1
shrink	1	1	1
slip	3	1	1
spring	1	1	1
startle	1	1	1
sweep	2.5	2.5	2.5
train	4	1	1
trip	4	1	1
tumble	4	1	1
twist	2	2	2
wobble	1.25	1.5	1

Appendix B

Agreement Scores by Semantic Class

This appendix contains the full agreement scores (including Weighted Kappa, observed agreement, and weighted observed agreement) achieved by each computational measure against the Levin and WordNet classes. A '*' after a Levin class number indicates a random subset of the verbs in the class were used.

Linearly Weighted Kappa and Agreement Scores - Levin Classes													
Light Verb	Class	PMI			LVC-PMI			LVC-Prob			LVC-Freq		
		κ_w	p_o	p_w	κ_w	p_o	p_w	κ_w	p_o	p_w	κ_w	p_o	p_w
	18.1,2	.29	.51	.67	.34	.57	.69	.46	.74	.83	.30	.57	.70
	30.3	.31	.61	.67	.51	.72	.78	.64	.78	.86	.72	.78	.89
take	43.2 *	.66	.97	.99	.38	.91	.96	−.03	.94	.97	−.04	.89	.90
	51.4.2	.49	.60	.75	.49	.60	.75	.51	.60	.75	.40	.50	.70
	avg.	*.44*	*.67*	*.77*	*.43*	*.70*	*.79*	*.39*	*.77*	*.85*	*.35*	*.68*	*.80*
	std. dev.	*.17*	*.20*	*.15*	*.08*	*.16*	*.12*	*.29*	*.14*	*.09*	*.31*	*.18*	*.11*
	18.1,2	.07	.26	.46	.11	.29	.47	.28	.54	.71	.30	.63	.79
	30.3	.19	.44	.53	.17	.39	.50	.38	.67	.75	.67	.78	.89
give	43.2 *	.18	.34	.51	.22	.40	.56	.19	.60	.73	.40	.63	.80
	51.4.2	.48	.80	.85	.48	.80	.85	.60	.80	.90	.87	.90	.95
	avg.	*.23*	*.46*	*.59*	*.25*	*.47*	*.59*	*.36*	*.65*	*.77*	*.56*	*.73*	*.86*
	std. dev.	*.18*	*.24*	*.18*	*.17*	*.23*	*.17*	*.19*	*.11*	*.09*	*.25*	*.13*	*.08*
	18.1,2	.25	.71	.77	.22	.69	.76	.46	.77	.84	.54	.71	.84
	30.3	.37	.89	.92	.53	.89	.92	.47	.83	.86	.44	.89	.89
make	43.2 *	−.10	.46	.66	−.08	.46	.60	−.12	.57	.79	−.26	.51	.71
	51.4.2	.00	.80	.90	.00	.80	.90	−.11	.70	.80	−.15	.70	.85
	avg.	*.13*	*.72*	*.81*	*.17*	*.71*	*.79*	*.18*	*.72*	*.82*	*.14*	*.70*	*.82*
	std. dev.	*.22*	*.19*	*.12*	*.27*	*.19*	*.15*	*.34*	*.11*	*.04*	*.40*	*.15*	*.08*
all	avg.	*.27*	*.60*	*.71*	*.26*	*.61*	*.72*	*.28*	*.70*	*.81*	*.30*	*.69*	*.81*
	std. dev.	*.22*	*.23*	*.17*	*.21*	*.22*	*.17*	*.27*	*.12*	*.07*	*.32*	*.13*	*.08*

Table B.1: Measures of agreement for each computational measure across each light verb and Levin class. Observed agreement is measured by p_o and weighted observed agreement by p_w.

Linearly Weighted Kappa and Agreement Scores - WordNet Classes													
Light Verb	Class	PMI			LVC-PMI			LVC-Prob			LVC-Freq		
		κ_w	p_o	p_w	κ_w	p_o	p_w	κ_w	p_o	p_w	κ_w	p_o	p_w
take	18.1,2	.42	.63	.79	.45	.66	.79	.69	.86	.90	.54	.74	.81
	30.3	.63	.89	.94	.51	.89	.93	.31	.77	.83	.34	.71	.79
	43.2	.53	.86	.91	.48	.81	.90	.37	.83	.89	.19	.77	.84
	51.4.2	.71	.80	.87	.65	.77	.84	.61	.71	.83	.62	.74	.83
	avg.	*.57*	*.79*	*.88*	*.52*	*.78*	*.86*	*.50*	*.79*	*.86*	*.42*	*.74*	*.82*
	std. dev.	*.13*	*.12*	*.07*	*.09*	*.10*	*.06*	*.19*	*.06*	*.04*	*.19*	*.02*	*.02*
give	18.1,2	.42	.54	.73	.39	.51	.70	.48	.60	.79	.38	.54	.73
	30.3	.44	.71	.81	.45	.74	.81	.27	.66	.76	−.02	.54	.64
	43.2	.43	.60	.77	.52	.66	.81	.37	.69	.80	.19	.57	.70
	51.4.2	.09	.43	.63	.11	.43	.63	.55	.74	.87	.32	.60	.79
	avg.	*.35*	*.57*	*.74*	*.36*	*.59*	*.74*	*.42*	*.67*	*.80*	*.22*	*.56*	*.71*
	std. dev.	*.17*	*.12*	*.08*	*.18*	*.14*	*.09*	*.12*	*.06*	*.05*	*.18*	*.03*	*.06*
make	18.1,2	.46	.77	.86	.41	.77	.83	.31	.66	.79	.27	.66	.76
	30.3	.29	.80	.86	.11	.77	.80	.14	.66	.70	.27	.66	.69
	43.2	.05	.51	.71	.17	.63	.77	.12	.57	.71	.17	.57	.71
	51.4.2	.24	.63	.76	.19	.60	.70	.31	.66	.77	.36	.66	.77
	avg.	*.26*	*.68*	*.80*	*.22*	*.69*	*.78*	*.22*	*.64*	*.74*	*.27*	*.64*	*.73*
	std. dev.	*.17*	*.13*	*.07*	*.13*	*.09*	*.06*	*.10*	*.04*	*.04*	*.08*	*.04*	*.04*
all	avg.	*.39*	*.70*	*.82*	*.39*	*.71*	*.81*	*.36*	*.70*	*.80*	*.30*	*.65*	*.75*
	std. dev.	*.18*	*.13*	*.08*	*.17*	*.11*	*.07*	*.17*	*.09*	*.06*	*.18*	*.08*	*.06*

Table B.2: Measures of agreement for each computational measure across each light verb and WordNet class. Observed agreement is measured by p_o and weighted observed agreement by p_w.

Appendix C

Using Google as a Corpus

In Section 4.3.1 the issues with using web data as a corpus—specifically, web data indexed by and accessed through a general-purpose search engine—are detailed. These issues include the ceiling on results returned which is applied by most search engines, along with issues of noise, both to intrinsic to the corpus (e.g., ungrammatical text, typographic errors) and due to the limitations of the search interface (e.g., results with internal punctuation, non-LVC usages, and so on.). However, there are further issues more closely associated with our particular choice of search engine, and in this appendix, we explore the concerns related to using Google in linguistic research.

Changes to the corpus Shortly after the data needed for our measurements was gathered, Google dramatically increased the size of its index, from 5.6 to 8 billion pages. The Google index is updated often as new data is made available, but it is not often such a large increase in size is observed. This highlights what is simultaneously an advantage and disadvantage of using search engines: while the size of the corpus available to this experiment has doubled with no cost to the researcher, had we been gathering information at the time of the shift in the index, the numbers would have been inconsistent. The existing data would have to have been thrown out, and new information gathered. This change in the Google corpus also highlights the importance, when using search engine data as a corpus, of gathering data over a short time

period: if the corpus changes significantly, frequency counts will not be comparable. The data required for the measures presented in this thesis was gathered in a five-week period beginning in October, 2004, and the number of results found for our baseline search (*the*) did not change during this time period.

Accessing the corpus A primary concern with using the data available in a search engine as a corpus is that the interface employed to access this corpus (either through the public web, or through a more private API) is subject to change. This issue is highlighted by Google's support of wildcards[1]: as this document was being prepared, Google dropped support for this search operator, only to resume support a few weeks later. Wildcards had been removed temporarily from Google's search interface in the past, and while they are now once again available, it is nevertheless disappointing to have such a useful tool for corpus research removed without warning.

Inflated counts Véronis (2005b) notes that in early January, the number of results returned by a Google search request appear to have been significantly overestimated, while further research shows that this problem appears to have been fixed (Véronis, 2005a). These anomalous results may be associated with Google's shift to a larger index (Véronis, 2005c). It is difficult to apply these observations to our own data, as our measurements were gathered before Google's increase in index size. However, this count inflation appears to have been systematic, and not particular to any search phrase: in other words, if counts were inflated, they were inflated uniformly across searches. Because of this, if count inflation did affect the data gathered for the measures presented in this thesis, it would not have affected our results. Scores of correlation and agreement would be identical.

[1] Wildcards are search terms which match for one word. For example, an exact-phrase search for *take a * walk* may return links to pages containing the phrases *take a short walk*, *take a long walk*, and so forth.

Bibliography

Alshawi, H. and Carter, D. (1994). Training and scaling preference functions for disambiguation. *Computational Linguistics*, 20:635–648.

Altman, D. (1991). *Practical Statistics for Medical Research*. Chapman and Hall, London.

Baldwin, T. and Villavicencio, A. (2002). Extracting the unextractable: A case study on verb-particles. In *Proceedings of the Sixth Conference on Computational Natural Language Learning (CoNLL 2002)*, pages 98–104. Association for Computational Linguistics.

Banerjee, S. and Pedersen, T. (2003). The design, implementation, and use of the Ngram Statistic Package. In *Proceedings of the Fourth International Conference on Intelligent Text Processing and Computational Linguistics*, pages 370–381, Mexico City.

Bannard, C., Baldwin, T., and Lascarides, A. (2003). A statistical approach to the semantics of verb-particles. In *Proceedings of the ACL-2003 Workshop on Multiword Expressions: Analysis, Acquisition and Treatment*, pages 65–72, Sapporo, Japan. Association for Computational Lingistics.

Baroni, M. and Bernardini, S. (2004). Bootcat: Bootstrapping corpora and terms from the web. In *Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC)*, Lisbon, Portugal.

Baroni, M. and Vegnaduzzo, S. (2004). Identifying subjective adjectives through web-based

mutual information. In *Proceedings of Konferenz zur Verarbeitung Natrlicher Sprache (KONVENS) 2004*, pages 613–619.

BNC Reference Guide (2000). *Reference Guide for the British National Corpus (World Edition)*. http://www.hcu.ox.ac.uk/BNC, second edition.

Bolinger, D. (1971). *The Phrasal Verb in English*. Harvard University Press, Camebridge, Massachusetts.

Butt, M. (2003). The light verb jungle. http://edvarda.hf.ntnu.no/ling/tross/Butt.pdf.

Butt, M. and Geuder, W. (2001). On the (semi)lexical status of light verbs. In Corver, N. and van Riemsdijk, H., editors, *Semi-lexical Categories: On the content of function words and the function of content words*, pages 323–370. Mouton de Gruyter, Berlin.

Byrt, T., Bishop, J., and Carlin, J. B. (1993). Bias, prevalence and Kappa. *Journal of Clinical Epidemiology*, 46(4):423–429.

Carletta, J. (1996). Squibs and discussions: Assessing agreement on classification tasks: The Kappa statistic. *Computational Linguistics*, 22:399–404.

Church, K. W., Gale, W., Hanks, P., and Hindle, D. (1991). Using statistics in lexical analysis. In Zernik, U., editor, *Lexical Acquisition: Exploiting On-line Resources to Build a Lexicon*, pages 115–164. Lawrence Erlbaum Associates.

Church, K. W. and Hanks, P. (1989). Word association norms, mutual information, and lexicography. In *Proceedings of the 27th Annual Meeting of the Association for Computational Linguistics*, pages 76–83, Vancouver, B.C.

Cicchetti, D. V. and Feinstein, A. R. (1990). High agreement but low Kappa: II. Resolving the paradoxes. *Journal of Clinical Epidemiology*, 43(6):551–558.

Cohen, J. (1960). A coefficient of agreement for nominal scales. *Educational and Psychological Measurement*, 20:249–254.

Cohen, J. (1968). Weighted Kappa: Nominal scale agreement with provision for scaled disagreement or partial credit. *Psychological Bulletin*, 70:213–220.

Dras, M. and Johnson, M. (1996). Death and lightness: Using a demographic model to find support verbs. In *Proceedings of the Fifth International Conference on the Cognitive Science of Natural Language Processing*, pages 165–172, Dublin, Ireland.

Dumais, S., Banko, M., Brill, E., Lin, J., and Ng, A. (2002). Web question answering: Is more always better? In *Proceedings of the 25th annual international ACM SIGIR conference on research and development in information retrieval*, pages 291–298.

Feinstein, A. R. and Cicchetti, D. V. (1990). High agreement but low Kappa: I. The problems of two paradoxes. *Journal of Clinical Epidemiology*, 43(6):543–549.

Fellbaum, C., editor (1988). *WordNet: An Electronic Lexical Database*. MIT Press, Cambridge, MA.

Fillmore, C. J., Baker, C. F., and Sato, H. (2002). Seeing arguments through transparent structures. In *Proceedings of the Third International Conference on Language Resources and Evaluation (LREC)*, pages 787–791, Las Palmas.

Firth, J. (1957). A synopsis of linguistic theory 1930-1955. In Palmer, F., editor, *Selected papers of J.R. Firth 1952–1959*.

Florescu, D., Levy, A. Y., and Mendelzon, A. O. (1998). Database techniques for the world-wide web: A survey. *SIGMOD Record*, 27(3):59–74.

Folli, R., Harley, H., and Karimi, S. (2003). Determinants of event type in Persian complex predicates. In Richards, M., editor, *Cambridge Working Papers in Linguistics*, pages 101–126.

Fontenelle, T., Adriaens, G., and Braekeleer, G. D. (1994). The lexical unit in the Metal MT system. *Machine Translation*, 9:1–19.

Grefenstette, G. and Teufel, S. (1995). Corpus-based method for automatic identification of support verbs for nominalizations. In *Proceedings of the Seventh Conference on European Chapter of the Association for Computational Linguistics*, pages 98–103, Dublin, Ireland.

Jackendoff, R. (1997). *The Architecture of the Language Faculty*. MIT Press, Cambridge, MA.

Karimi, S. (1997). Persian complex verbs: Idiomatic or compositional? *Lexicology*, 3(1):273–318.

Kearns, K. (2002). Light verbs in English. http://www.ling.canterbury.ac.nz/documents/lightverbs.pdf.

Keller, F. and Lapata, M. (2003). Using the web to obtain frequencies for unseen bigrams. *Computational Linguistics*, 29:459–484.

Khanlari, P. (1973). *Tarikh-e Zaban-e Farsi (The History of Persian Language)*. Bonyad-e Farhang.

Kilgarriff, A. (2001). Web as corpus. In *Proceedings of the Corpus Linguistics 2001 Conference*, pages 342–344, Lancaster, UK.

Kilgarriff, A. (2003). Linguistic search engine. In Simov, K., editor, *Shallow Processing of Large Corpora: Workshop Held in Association with Corpus Linguistics 2003*, Lancaster, England.

Kilgarriff, A. (2004). Web as corpus (presentation). http://corpus.ling.sinica.edu.tw/result/tsil2004/materials/workshop_Adam.ppt.

Kilgarriff, A. and Grefenstette, G. (2004). Introduction to the special issue on the web as corpus. *Computational Linguistics*, 29:333–348.

Lantz, C. A. and Nebenzahl, E. (1996). Behavior and interpretation of the Kappa statistic: Resolution of the two paradoxes. *Journal of Clinical Epidemiology*, 49(4):431–434.

Lawrence, S. and Giles, C. L. (1999). Accessibility of information on the web. *Nature*, pages 107–109.

Levin, B. (1993). *English Verb Classes and Alternations, A Preliminary Investigation*. University of Chicago Press, Chicago, IL.

Lin, D. (1998a). Automatic retrieval and clustering of similar words. In *Proceedings of the 17th International Conference on Computational Linguistics (COLING) and the 36th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 768–774, Montreal, Canada.

Lin, D. (1998b). Extracting collocations from text corpora. In *Workshop on Computational Terminology*, pages 57–63, Montreal, Canada.

Lin, D. (1999). Automatic identification of non-compositional phrases. In *Proceedings of ACL-99*, pages 317–324, University of Maryland, College Park, Maryland.

Lin, J. (2002). The web as a resource for question answering : Perspectives and challenges. In *Proceedings of the Third International Conference on Language Resources and Evaluation (LREC)*, Canary Islands, Spain.

Manning, C. D. and Schüetze, H. (1999). *Foundations of Statistical Natural Language Processing*. MIT Press, Cambridge, Massachusetts.

McCarthy, D., Keller, B., and Carroll, J. (2003). Detecting a continuum of compositionality in phrasal verbs. In *Proceedings of the ACL-SIGLEX Workshop on Multiword Expressions: Analysis, Acquisition and Treatment*, pages 73–80.

McCarthy, M. and Walter, E., editors (1997). *Cambridge International Dictionary of Phrasal Verbs*. Cambridge University Press.

Miyamoto, T. (2000). *The Light Verb Construction in Japanese: the role of the verbal noun*. John Benjamins Publishing Company.

Pantel, P. and Lin, D. (2002). Discovering word senses from text. In *Proceedings of ACM SIGKDD Conference on Knowledge Discovery and Data Mining 2002*, pages 613–619, Edmonton, Canada.

Resnik, P. and Diab, M. (2000). Measuring verb similarity. In *Proceedings of the 22nd Annual Meeting of the Cognitive Science Society (CogSci 2000)*, pages 399–404.

Resnik, P. and Elkiss, A. (2003). The linguist's search engine: Getting started guide. Technical Report LAMP-TR-108/CS-TR-4541/UMIACS-TR-2003-109, University of Maryland, College Park.

Sag, I. A., Baldwin, T., Bond, F., Copestake, A. A., and Flickinger, D. (2002). Multiword expressions: A pain in the neck for NLP. In *Proceedings of the Third International Conference on Intelligent Text Processing and Computational Linguistics (CICLING)*, pages 1–15.

Sinclair, J. M. and Moon, R., editors (1989). *Collins Cobuild Dictionary of Phrasal Verbs*. Harper Collins, London.

Stevenson, S., Fazly, A., and North, R. (2004). Statistical measures of the semi-productivity of light verb constructions. In *Proceedings of the ACL 2004 Workshop on Multiword Expressions: Integrating Processing*, pages 1–8, Barcelona, Spain.

Terra, E. L. and Clarke, C. (2003). Frequency estimates for statistical word similarity measures. In *Proceedings of the 2003 Human Language Technology Conference*, pages 165–172, Edmonton, Canada.

Turney, P. D. (2001). Mining the web for synonyms: PMI-IR versus LSA on TOEFL. In *Proceedings of the Twelfth European Conference on Machine Learning (ECML-2001)*, pages 491–502, Freiburg, Germany.

Turney, P. D. (2002). Thumbs up or thumbs down? Semantic orientation applied to unsuper-

vised classification of reviews. In *In Proceedings 40th Annual Meeting of the Association for Computational Linguistics (ACL'02)*, pages 417–424, Philadelphia, Pennsylvania.

Turney, P. D. and Littman, M. L. (2002). Unsupervised learning of semantic orientation from a hundred-billion-word corpus. Technical report, National Research Council, Institute for Information Technology. Technical Report ERB-1094.

Turney, P. D. and Littman, M. L. (2003). Measuring praise and criticism: Inference of semantic orientation from association. *ACM Transactions on Information Systems (TOIS)*, 21(4):315–346.

Véronis, J. (2005a). Google adjusts its counts? http://aixtal.blogspot.com/2005/03/web-google-adjusts-its-counts.html.

Véronis, J. (2005b). Google's counts faked? http://aixtal.blogspot.com/2005/01/web-googles-counts-faked.html.

Véronis, J. (2005c). Google's missing pages: mystery solved? http://aixtal.blogspot.com/2005/02/web-googles-missing-pages-mystery.html.

Villavicencio, A. (2003). Verb-particle constructions in the World Wide Web. In *Proceedings of the ACL-SIGSEM Workshop on the Linguistic Dimensions of Prepositions and their use in Computational Linguistics Formalisms and Applications*, Toulouse, France.

Villavicencio, A. and Copestake, A. (2002). Verb-particle constructions in a computational grammar of English. In *Proceedings of the 9th International Conference on Head-Driven Phrase Structure Grammar*, pages 357–371, Seoul, Korea.

Wierzbicka, A. (1982). Why can you Have a Drink when you can't *Have an Eat? In *The Semantics of Grammar*, pages 293–358. John Benhamins Publishing Co.