Automatic Semantic Classification of Verbs According to their Alternation Behaviour

Sabine Schulte im Walde

Diplomarbeit

November 20, 1998

Supervisor: Prof. Mats Rooth

Institut für Maschinelle Sprachverarbeitung (IMS)

Universität Stuttgart
Azenbergstr. 12
70174 Stuttgart
Germany

schulte@ims.uni-stuttgart.de

Contents

1 Introduction **1**

1.1 Idea 1

1.2 Motivation 2

1.3 Background 3

1.3.1 Subcategorisation Frames of Verbs 4

1.3.2 Lexical Acquisition of Subcategorisation Frames 6

1.3.3 Verb Classes According to Subcategorisation Frames . 7

2 Automatic Acquisition of Semantic Verb Classes **11**

2.1 Induction of Subcategorisation Frames 11

2.1.1 Defining the Subcategorisation Frames 13

2.1.2 Interpreting the Subcategorisation Frames 22

2.2 Selectional Preferences for Subcategorisation Frames 31

2.2.1 The Lexical Database WordNet 35

2.2.2 Selectional Preferences Coded by WordNet 38

2.3 Clustering Verbs into Semantic Verb Classes 48

2.3.1 Clustering Algorithms 49

2.3.2 Verbs and Verb Classes 62

2.3.3 Experiments 66

3 Interpreting the Semantic Classification **68**

3.1 Quality of the Verb Classes 68

3.1.1 Baseline Clustering 69

3.1.2 One-Step Distance Clustering 69

3.1.3 Iterative Distance Clustering 70

3.1.4 Latent Class Clustering 71

3.1.5 Comparison 71

3.2 Interpretation of the Verb Classes 72

3.2.1 Baseline Clustering 72

3.2.2 One-Step Distance Clustering 72

3.2.3 Iterative Distance Clustering 89

3.2.4 Latent Class Clustering 108

3.2.5 General Interpretation 135

4 Conclusions **149**

A Subcategorisation Frames **152**

B WordNet Concepts **155**

B.1 File Numbers 155

B.2 (Top) Synset Numbers 157

B.3 Additionally Defined Nouns 158

C Distances between Verbs **159**

Chapter 1

Introduction

1.1 Idea

This thesis aims at an automatic acquisition of a semantic classification for verbs. As starting point, I assume that the diathesis alternation of verbs, i.e. the alternation in the expression of the verbs' arguments, is a basis for the comparison of the verbs' meanings. More specifically, I empirically investigate the proposition that verbs can be semantically classified according to their syntactic alternation behaviour concerning subcategorisation frames and their selectional preferences for the arguments within the frames.

The purpose of such a semantic classification system automatically acquired is to provide empirical evidence for the correspondence between the usage of a verb and its meaning. This is useful for various issues in the area of Natural Language Processing (NLP):

- Empirical support of the hypothesis that syntax and semantics interact with each other in the acquisition of language
- Definition of the verb's semantic class as part of its lexical entry, next to idiosyncratic information: the semantic class generalises as a type definition over a range of syntactic and semantic properties, to support further NLP tasks like lexicography (by the enrichment of lexical knowledge), word sense disambiguation (by the provision of context information provided by the semantic verb type), parsing (by the generalisation from verb tokens to verb types and the resulting restriction of syntactic structures)

- Basis for concrete considerations concerning the similarity of verbs, for instance, in the process of determining whether a verb participates in particle diathesis, i.e. whether a verb with and without a certain particle represents the same verb meaning; for example, does *climb* mean the same as *climb up*?
- Inter-lingual comparison of verbs, for example, when comparing the variety of verbs in multiple languages expressing a specific verb meaning; this presupposes similar semantic verb classifications in other languages than English

Concrete applications utilising such verb type information are, for example, [Lee, 1997] when trying to solve the sparse data problem: if data is organised into classes of similar events, then, if information about an event is lacking, the behaviour is estimated from information about similar events; and [Klavans and Kan, 1998] who discriminate documents by type and semantic properties.

1.2 Motivation

Section 1.2 briefly introduces into the theories underlying the idea of my thesis, before bringing them into contact with each other.

Theoretical Linguistics Traditional theoretical linguists as [Chomsky, 1965] state that the utterance of a verb in context requires the application of two kinds of rules: subcategorisation rules for choosing a subcategorisation frame, and selectional rules for selecting the arguments for the frame. An appropriate application of the rules prevents the human speaker from uttering a sentence like *Colourless green ideas sleep furiously*, unless it is meant metaphorically.

Lexical Acquisition Within the area of lexical acquisition an issue under discussion is the question whether children first learn the syntactic (see [Gleitman, 1990]) or semantic (see [Pinker, 1989]) properties of language – and especially verbs –, to infer further language features. Approaches like [Brent, 1994b] attempt to reconcile the positions for their use, by reducing them to their common denominator, an interaction between syntax and semantics in the child's learning process.

English Verb Classes [Levin, 1993, p. 1], puts the correspondence between syntax and semantics into concrete terms by investigating the hypothesis that "the behaviour of a verb, particularly with respect to the expression and interpretation of its arguments, is to a large extent determined by its meaning. Thus verb behaviour can be used effectively to probe for linguistically relevant pertinent aspects of verb meaning." She utilises the correspondence by defining semantic classes for English verbs based on their alternation behaviour, considering syntactic and semantic properties.

The idea of this thesis as outlined in section 1.1 is related to Levin's central assumption arisen from the field of lexical acquisition, that the verb behaviour can be used to probe for aspects of verb meaning. As she does, I attempt to derive verb classes from the verbs' behaviour. The information about the verbs' behaviour fed into the deduction process is referring back to Chomsky's demands for the utterance of verbs: the verbs' behaviour is defined by their subcategorisation rules and their selectional rules.

Means provided by the NLP-community allow to empirically investigate the verbs' behaviour and their meanings: I aim to automatically infer semantic verb classes by the help of data-intensive methods working on data from a large corpus, and by applying statistical methods proved useful for NLP-tasks. The inference process contains three main steps:

1. The induction of subcategorisation frames for verbs from a large corpus
2. The definition of selectional preferences for the subcategorisation frames
3. The clustering of the verbs into semantic verb classes, on account of the verbs' behaviour as defined in steps 1 and 2

1.3 Background

This section presents an introduction into the issue of subcategorisation frames as lexical properties of verbs, concerning the theories underlying the ideas of my thesis. Subsection 1.3.1 starts with a general description of subcategorisation frames, before subsection 1.3.2 informs about the discussion concerning lexically acquiring subcategorisation frames. Subsection 1.3.3 describes Levin's verb classification in more detail.

1.3.1 Subcategorisation Frames of Verbs

Subcategorisation Rules Each verb is associated with either a single subcategorisation frame or an alternation over a specific set of subcategorisation frames. The frames impose syntactic constraints on the number and the function (subject, direct object, etc.) of the arguments selected by the verb. In case the verb allows an alternation between a set of subcategorisation frames, it is said to undergo the linguistic phenomenon *diathesis alternation*.

For example, sentence (1.1) presents the typical usage of the verb *love*, demanding the specific subcategorisation frame consisting of a subject and a direct object:

(1.1) John loves Mary.

By contrast, the verb *give* in example (1.2) shows diathesis alternation between the two different subcategorisation frames, (1.2a) a subject, an indirect object and a direct object, and (1.2b) a subject, a direct object and a prepositional phrase headed by *to*:

(1.2) a. John gives Mary a book.

b. John gives a book to Mary.

[Chomsky, 1965] calls the specification of subcategorisation frames for verbs *subcategorisation rules* and demonstrates the strongly changing degree of grammaticalness when these rules are violated; a sentence like the one in example (1.3) is hardly interpretable:

(1.3) John found sad.

Selectional Rules In addition to the constraints imposed on the syntactic representation of the verbs' subcategorisation frames, Chomsky defines *selectional rules* to restrict the semantics of the arguments to specific semantic concepts. The semantic concepts propose generalisations of meanings as expected from the syntactically chosen arguments. Consider, for example, the difference between the sentences in (1.4):

(1.4) a. John sleeps well.

b. Colourless green ideas sleep furiously.

In both examples I identify the preferred subcategorisation frame of the verb *sleep*, a subject only, in these two sentences accompanied by an adverb. In example (1.4b), I have difficulties with the interpretation, however, because the subject is not represented as a living entity, the semantic concept we would have expected, and the choice of the adverb appears strange to us as well (how is it possible to sleep furiously?).

Violations of selectional rules decrease the degree of acceptability. But the decision whether a sentence is acceptable or not is not a *yes-or-no* decision, but rather within a range of acceptability. Selectional rules define semantic concepts for the arguments in subcategorisation frames by defining an ordering of preferences, so we rather talk about *selectional preference* for the selected arguments. The following sentences recited from [Allen, 1995] illustrate this observation:

(1.5)
1. I ate the pizza.
2. I ate the box.
3. I ate the car.
4. I ate the thought.

In example (1.5) sentence 1 is intuitively acceptable and sentence 4 is not. But how to judge about sentences 2 and 3? At this point it is important to refer to the context of the sentences, taking into account that context plays an important role in the interpretation of an utterance. The degree of acceptability is determined by the context of the sentences: maybe the proposition speaks about a chocolate car?
Widening the possibilities of context to the expressiveness of poetic licence might even enable to accept sentence 4 and the previously mentioned example (1.4b) as metaphorical expressions.

Summarising the above discussion, we note that there are two kinds of restrictions on the usage of verbs[1], syntactic restrictions in form of subcategorisation frames, and semantic restrictions in form of selectional preferences for the arguments in the subcategorisation frames. The restrictions are properties of the verbs, since each verb specifies its alternation behaviour (including the respective selectional preferences).

[1] Actually, there are more restrictions, of course, but I restrict myself to those two relevant for this work.

1.3.2 Lexical Acquisition of Subcategorisation Frames

Equipped with basic knowledge about the lexical properties of verbs concerning the application of subcategorisation frames, I now turn to the question of how to acquire this lexical knowledge.
In the area of lexical acquisition there has been much discussion about whether children first learn the syntactic or the semantic properties of language to infer further language features. The two hypotheses are called *syntactic bootstrapping hypothesis* and *semantic bootstrapping hypothesis*, respectively. Among other areas, this discussion also concerns the acquisition of subcategorisation frames, mainly driven by Gleitman and Pinker.

[Gleitman, 1990] argues in favour of the syntactic bootstrapping hypothesis, that in general the syntax acts as a kind of *mental zoom lens* for fixing on the interpretation the speaker is expressing. Children who understand the mapping rules for semantics onto syntax can use the observed syntactic structures as evidence for deducing the meanings. They first learn the subcategorisation frames and then exploit the correspondence to restrict their hypotheses about the possible meanings.
An example for this scenario can be given by differentiating the verb *put* from the verb *sleep*: the action of putting implies one who puts, a thing put, and a place into which it is put; a noun phrase is assigned to each of the participants. Sleeping, on the other hand, only involves one participant, the person who sleeps. Hence listening to sentences which contain either verbs and the respective syntactic arguments trains the child to acquire knowledge about the verbs' meaning.

[Pinker, 1989] claims that argument structures are projections of the verbs' semantic structures; subcategorisation frames are determined via so-called linking rules from the semantics of the verb and its arguments. Children first learn the meaning of a word and then exploit the regular correspondence between meaning and subcategorisation to infer the subcategorisation frames.
Pinker illustrates his statement with the difference between the examples (1.6) and (1.7):

(1.6) The ball rolled.
John rolled the ball.

(1.7) The baby cried.
*John cried the baby.

Observing the alternation between an intransitive and a causative transitive subcategorisation frame considering the verb *roll* is not sufficient to transfer the pattern to the verb *cry*. To know about the possible alternations of a verb, the lexical semantics must be taken into account.

I do not want to go into the discussion which of the two hypothesis is more plausible[2]. Instead, I reduce the seemingly contrary positions to their common denominator: the two opinions have in common, that there exists a correspondence between the syntax and the semantics in the acquisition of subcategorisation frames. Neither can be learned without interacting with the other.
With this view concerning the bootstrapping hypotheses I am on the line with, for example, [Brent, 1994b] who also assumes the interaction of syntax and semantics in the acquisition of subcategorisation frames as basis for his work.

Summarising the above discussion leads us to an interdependency between the verbs' demand for subcategorisation frames, i.e. the syntactic alternation behaviour of verbs, and their meanings.

1.3.3 Verb Classes According to Subcategorisation Frames

Exactly this interdependency between the alternation behaviour of verbs and their meanings is the basis for Levin's work [Levin, 1993].
As mentioned in the motivation, Levin investigates the syntactic and semantic properties of subcategorisation frames for English verbs and utilises the acquired knowledge to assign the verbs into classes. The resulting verb classes show meaning components shared by their members.

Levin splits the task of defining verb classes into two parts:

1. First, she defines 78 different diathesis alternations accompanied by the verbs showing the respective alternation behaviour.

   To give a concrete example, the alternation type TRANSITIVITY ALTERNATION generalises about alternations between the subcategorisation frames `NP-V-NP`/`NP-V` and `NP-V-NP`/`NP-V-PP`. One specific alternation of the latter kind is called LOCATIVE PREPOSITION DROP ALTERNATION, because the alternation is realised by "dropping" the preposition:

[2] See [Light, 1996] for a detailed discussion of this issue.

(1.8) a. Martha climbed up the mountain.
b. Martha climbed the mountain.

The verbs undergoing this kind of alternation typically appear intransitively with a directional phrase – as in (1.8a) –, or transitively with a path or goal – as in (1.8b). The direct object as in (1.8b) is often interpreted holistically. Specific verb examples are motion verbs like *climb, fly, jump, travel, walk.*

2. Having determined the diathesis alternations and their verbal participants, Levin defines 49 verb classes – partly divided into sub-classes – and assigns 3,104 verbs to them, according to which alternations the respective verbs undergo: verbs showing the same alternation behaviour are assigned to the same class.

   For example, the verb class *Vehicle Names*, sub-class of *Motion Verbs*, contains verbs like *balloon, bicycle, canoe, skate, ski* because they agree in the following properties:

   (1.9) Intransitive Use, possibly followed by a path:
   a. They skated.
   b. They skated along the canal/across the lake.

   (1.10) Induced Action Alternation (some verbs):
   a sub-type of Transitive Alternation, where the transitive use of the verb can be paraphrased as causing the action named by the verb; the causee is typically an animate volitional entity induced to act by the causer; in the transitive causative use, the verb must be accompanied by a directional phrase
   a. He skated Penny around the rink.
   b. Penny skated around the rink.

   (1.11) Locative Preposition Drop Alternation (some verbs):
   a. They skated along the canals.
   b. They skated the canals.

   (1.12) Resultative Phrase:
   an XP which describes the state achieved by the referent of the noun phrase it is predicated of as a result of the action named by the verb
   Penny skated her skate blades blunt.

In this example class only positive participation concerning the specific alternations is mentioned, i.e. which alternations the verbs are allowed to undergo. There might as well be explicit negative participation constraints on the verb classes like certain verbs not being allowed to take part in a certain alternation.

An important point to mention is the fact that the 3,104 verbs Levin investigates have 4,194 different verb senses. Levin assigns those verbs representing multiple verb senses to multiple verb classes, thereby accounting for the diversity of senses. This is a necessary act, since the classes are meant to represent verb meanings, and therefore different verb meanings (including the different senses of the same verb word-form) have to be assigned to different classes.

Levin's verb classification impressively illustrates the connection between a verb's alternation behaviour and its meaning: the verb classes are defined on the basis of common alternation behaviour concerning their members, and the result simultaneously represents common meaning of the verbs in one class.

An earlier investigation concerning the relationship between a verb's properties and its meaning has taken place by [Zwicky, 1971]. He is, like Levin, of the opinion, that certain combinations of the verbs' properties – he takes syntactic, semantic, and phonological properties into account – characterise certain classes of verbs.
For illustrating this relationship, Zwicky determines the properties of the specific class of *Manner-of-Speaking* verbs, i.e. verbs referring to intended acts of communication by speech and describing physical characteristics of the speech act. He invents a verb called *greem* and states that – presupposing that this verb referred to an intended act of communication by speech and described the physical characteristics of the act – one would know that it had all the properties defined for the class of *Manner-of-Speaking* verbs and could therefore use the verb in the same way.

These investigations by Levin and Zwicky present (i) evidence for an interdependency between the alternation behaviour of verbs – concerning a variety of properties – and their meanings, and (ii) possibilities to utilise the relationship.

This is the starting point for my work, since I attempt to follow the basic ideas by automatic means.

The structure of the thesis is as follows:
In chapter 2 I describe the three steps in the acquisition of semantic classes as mentioned above in detail, referring to possible approaches for their realisation and explaining the chosen variants.
Following the process of inferring the semantic classes, chapter 3 describes and interprets the resulting classification.
Chapter 4 concludes with considerations about the success of the classification process and the usefulness of the underlying assumptions.

Chapter 2

Automatic Acquisition of Semantic Verb Classes

Having introduced into the theoretical background of subcategorisation frames and verb classes, this section describes the three relevant steps necessary for the automatic acquisition of semantic verb classes. I briefly recite the steps:

1. The induction of subcategorisation frames for verbs from a large corpus
2. The definition of selectional preferences for the subcategorisation frames
3. The clustering of the verbs into semantic verb classes, on account of the verbs' behaviour as defined in steps 1 and 2

The parts of this chapter, sections 2.1 to 2.3, explain the respective ideas in detail, introduce into relevant approaches mentioned in literature, and describe the chosen approach.

2.1 Induction of Subcategorisation Frames

The first step in the course of my work was the induction of subcategorisation frames. Following is an extract of approaches concerning this issue.

[Brent, 1991] takes a raw, untagged text corpus and defines a three-step approach to assign a certain range out of five subcategorisation frames to verbs. He first identifies the verbs by applying a grammar rule defining that every noun-phrase has to appear either immediately to the left of a tensed verb,

immediately to the right of a preposition, or immediately to the right of a main verb. Then he uses a finite-state grammar for a fragment of English to find the subcategorisation frames for the verbs, and finally the frames are filtered for reliability by statistical models of the frequency distributions. Within a later approach [Brent, 1994a], he works on a partially parsed text and applies an algorithm with two components; he first identifies verbs which show a certain surface behaviour, i.e. he looks out for local surface cues which come with the verb, for example words with the suffix *-ing*. Afterwards he defines (further) surface cues to determine the argument phrases for each verb. As a second step, he tests the hypothetical subcategorisation frames by statistical modelling, calculating probabilities based on binomial distributions.

[Manning, 1993] uses a finite state parser to identify verbs and auxiliaries plus information about the verb modus. He adds context information by determining the complements of the verbs, not distinguishing between arguments and adjuncts. The output of this first step still contains wrong subcategorisation frames, so Manning also applies a filtering step, using the algorithm based on binomial distribution suggested by [Brent, 1994a].

[Briscoe and Carroll, 1997] extract subcategorisation frames from corpora by a system consisting of six components; they first tag and lemmatise the corpus, then they parse the text with their probabilistic LR parser (see [Briscoe and Carroll, 1994]) and extract subcategorisation patterns from the ranked parses. A pattern classifier assigns patterns to the subcategorisation classes, and finally an evaluator filters the subcategorisation entries by the degree of reliability, depending on the rank.

My decision for an approach was led by practical issues, though. The TCL (*Theoretical Computational Linguistics*) group at the *Institut für Maschinelle Sprachverarbeitung* (IMS) has developed a robust statistical parser[1] whose parse forest structures offer a useful basis for the extraction of subcategorisation frames.

The following subsection 2.1.1 gives a detailed description of the extraction of subcategorisation frame tokens from parse structures, before subsection 2.1.2 interprets the tokens and generalises them to a limited number of subcategorisation frame types which can be assigned to verbs in order to define their syntactic alternation behaviour.

During the description, the reader should bear in mind that the following

[1] The parser was developed by [Carroll and Rooth, 1998].

data was filtered automatically. Mistakes caused by the different tools were not corrected, so they are still included.

2.1.1 Defining the Subcategorisation Frames

As source for the data – verbs, subcategorisation frames, arguments – I chose the *British National Corpus (BNC)*[2], a 100 million word collection of written and spoken modern British English. 100 million words represent a sufficient amount of data to start with, and the corpus is freely available. In addition, the BNC has the important property of being heterogeneous, i.e. it contains language from various domains instead of concentrating on one specific area. This property creates a more general picture of the data, considering syntactic structures as well as the semantics. Homogeneous corpora, as the *Wall Street Journal*, by contrast, tend to exploit only a limited number of syntactic structures and also a limited number of words, depending on the specific domain (of economics, in this example corpus).

Following, I will go through the single steps of extracting subcategorisation frames from the BNC. To illustrate the effect of each step I refer to example sentences.

To begin with, I extracted the sentences of the BNC with a tool called `tbnc` that strips off the SGML information and leaves the words in the texts, annotated with their part of speech tags, one word-tag pair per line:

```
He                  PNP
argued              VVD
against             PRP
an                  AT0
excessively         AV0
formalist           AJ0
type                NN1
of                  PRF
analysis            NN1
of                  PRF
art                 NN1
,                   PUN
by                  PRP
pointing            VVG
out                 AVP
how                 AVQ
everyday            AJ0
emotions            NN2
```

[2]See `http://info.ox.ac.uk/bnc` for information about the corpus.

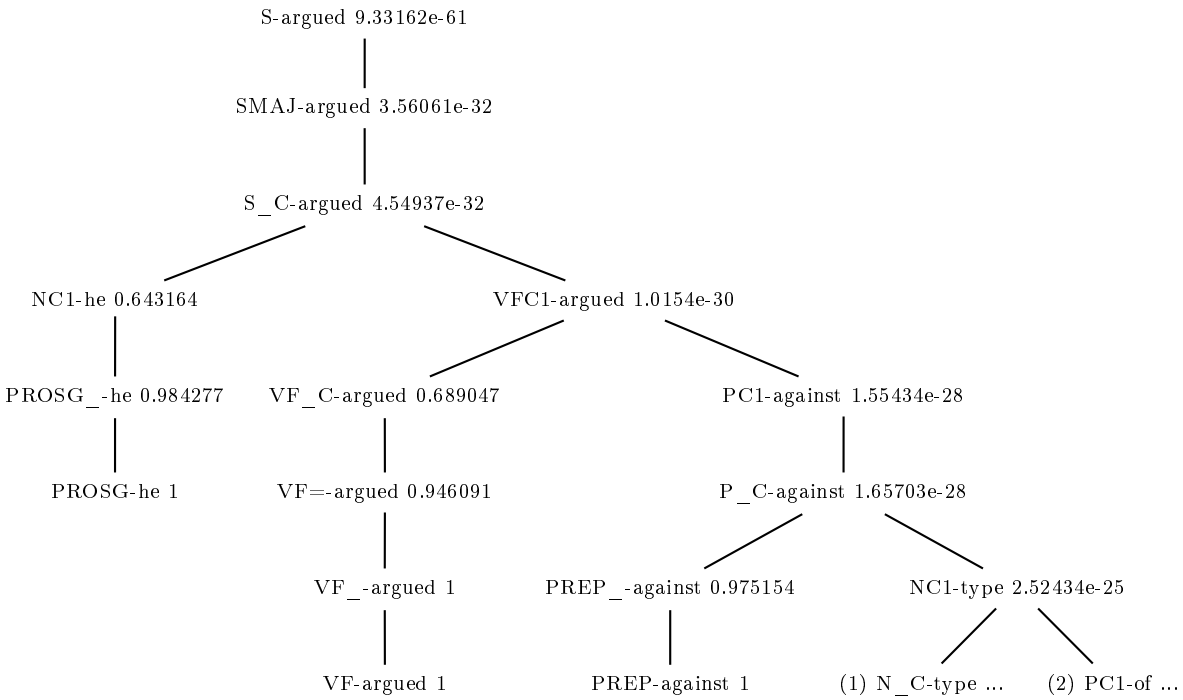
```
and                 CJC
ideas               NN2
also                AV0
affect              VVB
the                 AT0
viewer              NN1
of                  PRF
paintings           NN2
or                  CJC
sculpture           NN1
.                   PUN
Many                DT0
of                  PRF
his                 DPS
readers             NN2
approved            VVD
his                 DPS
sensitive           AJ0
and                 CJC
appreciative        AJ0
understanding       NN1
of                  PRF
paintings           NN2
,                   PUN
though              CJS
without             PRP
sharing             VVG
his                 DPS
political           AJ0
views               NN2
.                   PUN
```

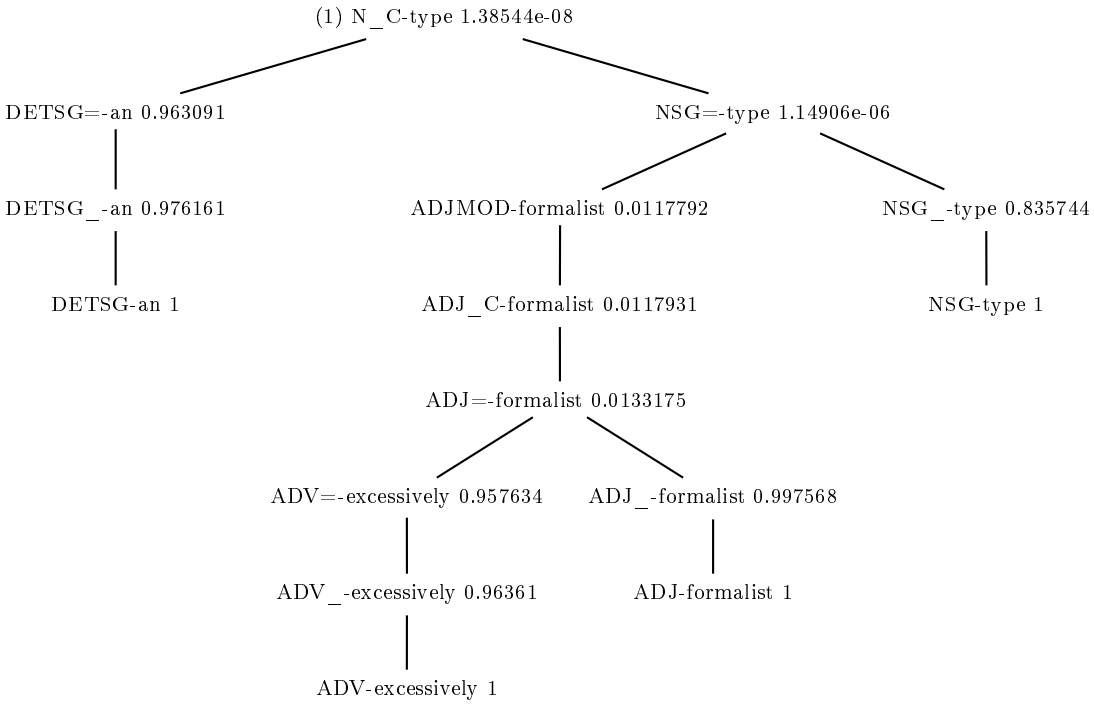
The so-structured sentences of the BNC were then parsed by the robust head-entity parser mentioned above. The parser utilises a lexicalised probability model to produce parse forests, annotated with information about the lexical head and the probability of each sub-tree. An additional option presents the *viterbi* parse in the parse forest, i.e. the most probable parse within the parse forest.
In this way I obtained the most probable parse for each sentence in the BNC. To present an example of the structure, following is the viterbi parse of a part of the sentence cited above:
He argued against an excessively formalist type of analysis of art, [...]

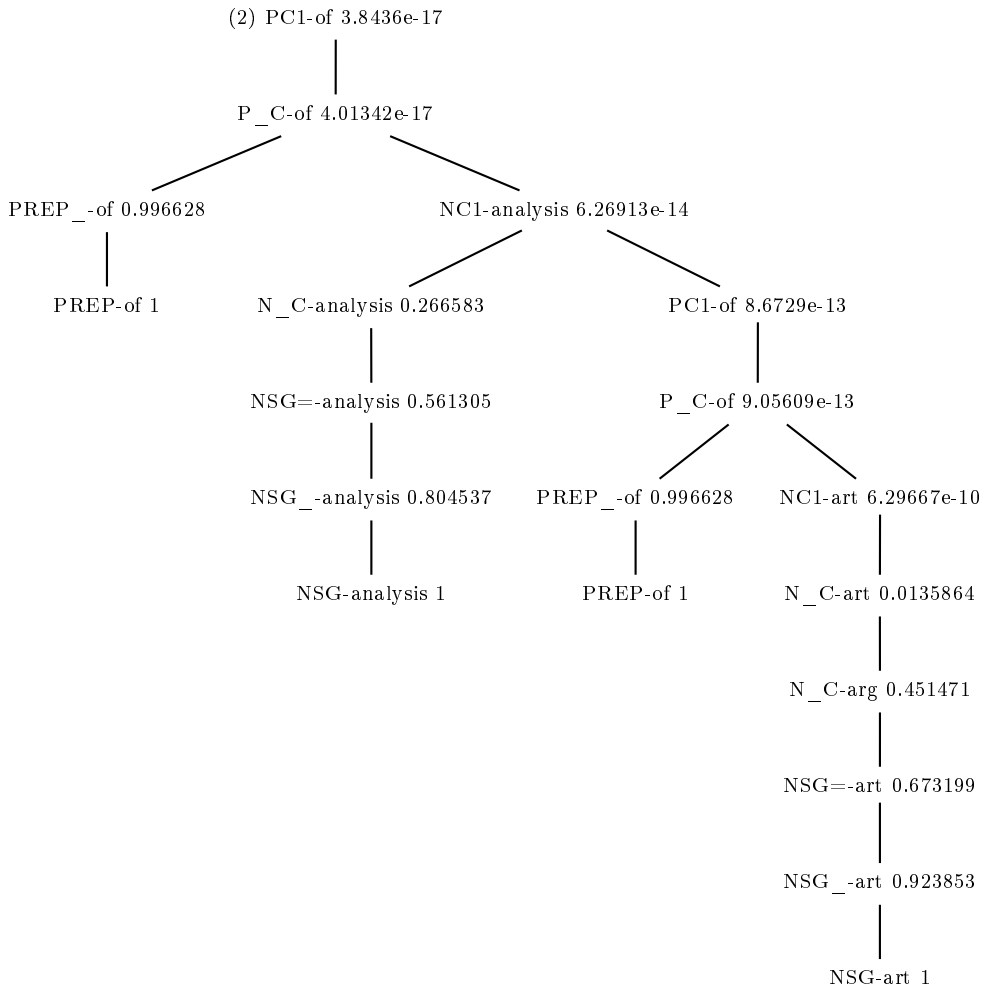
The parse tree is represented by nested structures, starting with the sentence symbol `S`. Sister leaves are arranged at the same line position. Each node is annotated by the head, followed by the probability for the sub-ordinated part of the tree.

```
{S-argued 9.33162e-61 {SMAJ-argued 3.56061e-32
    {S_C-argued 4.54937e-32
      {NC1-he 0.643164
      . {PROSG_-he 0.984277
      .   {PROSG 1 {he}}}}
      {VFC1-argued 1.0154e-30
        {VF_C-argued 0.689047
        . {VF=-argued 0.946091
        .   {VF_-argued 1
        .     {VF 1 {argued}}}}}
        {PC1-against 1.55434e-28
          {P_C-against 1.65703e-28
            {PREP_-against 0.975154
            . {PREP 1 {against}}}
            {NC1-type 2.52434e-25
              {N_C-type 1.38544e-08
              . {DETSG=-an 0.963091
              . . {DETSG_-an 0.976161
              . .   {DETSG 1 {an}}}}
              . {NSG=-type 1.14906e-06
              .   {ADJMOD-formalist 0.0117792
              .   . {ADJ_C-formalist 0.0117931
              .   .   {ADJ=-formalist 0.0133175
              .   .     {ADV=-excessively 0.957634
              .   .     . {ADV_-excessively 0.96361
              .   .     .   {ADV 1 {excessively}}}}
              .   .     {ADJ_-formalist 0.997568
              .   .       {ADJ 1 {formalist}}}}}}
              .   {NSG_-type 0.835744
              .     {NSG 1 {type}}}}}
              {PC1-of 3.8436e-17
                {P_C-of 4.01342e-17
                  {PREP_-of 0.996628
                  . {PREP 1 {of}}}
                  {NC1-analysis 6.26913e-14
                    {N_C-analysis 0.266583
                    . {NSG=-analysis 0.561305
                    .   {NSG_-analysis 0.804537
                    .     {NSG 1 {analysis}}}}}
                    {PC1-of 8.6729e-13
                      {P_C-of 9.05609e-13
                        {PREP_-of 0.996628
                        . {PREP 1 {of}}}
                        {NC1-art 6.29667e-10
                          {N_C-art 0.0135864
                            {N_C-art 0.451471
                              {NSG=-art 0.673199
                                {NSG_-art 0.923853
                                  {NSG 1 {art}}}}}}}}}}}}}}}}} ... }}}}
```

In a more clearly arranged manner the above nested structure represents the parse tree







I should explain the relevant grammatical categories underlying the parse tree[3]. The English context-free grammar provides three levels:

- The *chunk level* identified by the suffix `_C` (like `VF_C`): the idea of defining a chunk level in the grammar in addition to a phrase level goes back to [Abney, 1991]. He presents psychological evidence for the existence of chunks, defined as syntactic units which correspond in some way to prosodic patterns, containing a content word surrounded by some function word(s).

- The *phrase level* identified by the suffix `C1` (like `VFC1`): phrases in the grammar are defined as chunks plus their complements and post-modifiers.

[3]Most of the description was provided by Glenn Carroll.

- The *machine level*, tri-grams consisting of a pair of categories separated by a colon (like `PC1:NC1`): the tri-grams contribute to the robustness of the parser.

All other categories are intermediate levels or terminal symbols.

Following is a list of the essential chunks in the grammar. Most of them have corresponding phrase levels.

```
ADJ_C          adjective chunk
AS_C           'as' clause
N_C            noun chunk
PART_C         particle
P_C            prepositional chunk
P_ST_C         stranded preposition
REL_C          relative clause
S_C            sentence chunk; phrase category: SMAJ
SUB_C          subordinated clause
THAT_C         'that' clause
VBASE_C        infinite verb chunk, active
VBASEP_C       infinite verb chunk, passive
VF_C           finite verb chunk, active
VFP_C          finite verb chunk, passive
VG_C           gerund, active
VGP_C          gerund, passive
VN_C           past-tense verb chunk
VPASS_C        stranded verb chunk, passive
VTO_C          infinitive verb chunk, active, including 'to'
VTOP_C         infinitive verb chunk, passive, including 'to'
```

On the basis of the viterbi parses I extracted the subcategorisation frames for all parsed BNC sentences. Since the grammar had imposed the structure on the viterbi parses, I worked hand in hand with the grammar rules. Having in mind that the subcategorisation frames should not only be the basis for the present thesis, but hopefully be re-used for further tasks, I tried to define a general pattern for the data:
The frames should represent the definition of the arguments as appearing in the syntactic deep structure of the sentences. From each sentence (chunk), I extracted the full verb form of the head of the sentence, accompanied by the verb modus (i.e. the subcategorisation frames distinguish between active and passive sentences) and all verbal arguments. It was possible to distinguish between internal and external arguments of the verb by defining the external argument as the sister of the finite verb phrase, and the internal arguments as the sisters of the finite verb chunk (compare the example parse tree above). Each argument was described by at least one feature, followed by its lexical head. Following you can find a detailed description of the definition of the arguments:

- The noun chunk sister of the finite verb phrase, i.e. the external argument, was identified as the subject (feature: `subj`) in active sentences and an object (feature: `obj`) in passive sentences. For example, *coach* is represented as `subj*coach` in the sentence *Our coach loves Mary*, and as `obj*coach` in the sentence *Our coach is loved*.
  In the case of proper names I added the feature `pn` to the description of the argument. In this way I left open the possibilities of either working with the names themselves (since they are following as the argument's head), or working by generalising over the class of proper names (by using the feature `pn`). For example, *John* instead of *coach* in the active example sentence above is represented as `subj*pn*john`.
  In the case of compound nouns I defined the last noun as head of the noun chunk.
- The noun chunk sisters of the finite verb chunk, i.e. the internal arguments representing a noun chunk, were identified as objects (feature: `obj`) in both active and passive sentences, which should generally be all right for active sentences, but include some noise for passive sentences: it was not possible to identify the subject among the chunks (if there was any). The subject in passive sentences is, generally said, often hidden in a *by*-phrase (again, if there is any), but there are exceptions to this rule - consider the sentence *The work is finished by tomorrow.*
  The cases of proper names and compound nouns were handled in the same way as for external arguments.
  To give an example, the proper name *Mary* in the sentence *John loves Mary* would be represented as `obj*pn*mary`.
- Prepositional chunks – when sisters of the finite verb chunk – were given the feature `pp`, followed by the preposition and the head of the subcategorised noun phrase, where the features for noun phrases were defined as for the noun chunks above. For example, the prepositional phrase *to Mary* in the sentence *John gives a present to Mary* is represented as `pp*to*pn*mary`.
- Infinite verb chunks starting with *to* – when sisters of the finite verb chunk – were given the identifier `to`, followed by the verb modus and the verbal head. For example, *to paint* in the sentence *John likes to paint* is represented as `to*act*paint`.
- *as*-chunks – when sisters of the finite verb chunk – were given the identifier `as`, an abbreviation for the subcategorised chunk (adjective: `ap`, noun: `np`, or gerund: `vger`), followed by the head of that chunk. For example, the chunk *as possible* is represented by `as*ap*possible`.

- *that*-chunks – when sisters of the finite verb chunk – were given the identifier `that` and the head of the subcategorised sentence, if there was any. For example, the chunk *that they leave early* is represented as `that*leave`.
- All other chunk sisters of the finite verb chunk were defined by the name of the chunk as the feature for that argument, followed by its head.

I conclude the description of the subcategorisation frame tokens by a complete list of the features

```
adv          adverb
ap           adjectival phrase
as           as-expression
part         particle
pp           prepositional phrase
ppart        stranded preposition
relp         relative clause
s            sentence
that         subordinated that-phrase
to           infinitive form of verb after 'to'
vbase        base form of verb
vger         gerund
vpast        past form of verb
vstrand      stranded verb
```

and additional identifiers

```
act          active verb
pas          passive verb
subj         subject of the sentence
obj          object of the sentence
pn           proper name
dummy        no head was available
```

Following I present some example subcategorisation frames tokens, extracted from the viterbi parses of the respective sentences. Each line represents one subcategorisation frame; the verb as well as the arguments are defined by a 2-/3-/4-tuple describing the features of the chunk. The frames start with the description of the verb, followed by all arguments, in the order they appeared in the parses. To give an example, the frame token
`act*excelled subj*nobody obj*him pp*in*judgement`
describes the sentence *Nobody excelled him in that judgement.*

```
pas*described obj*realism pp*by*pn*fischer
pas*made obj*attempt to*act*create
act*proved subj*distinction ap*difficult
```

```
pas*made obj*diversion to*act*emphasise
act*took subj*this obj*forms
act*argued subj*he pp*against*type
pas*called obj*type obj*compiler
act*is subj*pn*york ap*exemplary
act*chose subj*pn*barr obj*hugnet to*act*write
act*is subj*commentary pp*in*phrases
act*been subj*qualities ap*present
act*was subj*pn*barr vger*writing
act*intend subj*museum to*act*sponsor
act*were subj*men obj*tastmakers
act*were subj*judgements ap*important
pas*limited obj*writing pp*by*demands
pas*thrown obj*stress pp*on*modernism
act*has subj*critic obj*advantage
act*serve subj*comparison obj*us pp*as*example
act*have subj*works obj*character
act*seem subj*they to*act*proceed
act*excelled subj*nobody obj*him pp*in*judgement
act*united subj*he obj*observations
act*ought subj*which to*act*hold
act*demands subj*pn*michelangelo obj*preference
act*was subj*pn*reynolds adv*here
act*took subj*he obj*opportunity
act*was subj*pn*reynolds obj*conversationalist
act*know subj*we that*is
act*is subj*labour ap*unnecessary
act*finds subj*he obj*it ap*necessary
act*received subj*he obj*nothing pp*of*inspiration
```

At this point the extraction of the subcategorisation frame tokens from BNC sentences was finished. The result was a list of frame tokens in an as general as possible fashion, ready to be used for further applications. For my own work, I refined the frames in a further step, that of lemmatising the word tokens in the frames. The lemmatisation was carried out by using a morphological lexicon for English, built by [Karp et al., 1992], refined by a morphological stemmer for English, built by Steven Abney. The combination of the lexicon and the stemmer turned out to be most successful. Tests with only utilising the lexicon showed that 116,704 word tokens were not defined and therefore not lemmatisable by the lexicon; tests with only applying the stemmer showed that the morphological rules were used in too many cases, for example the noun *lens* would be lemmatised to *len*. By combining the lexicon with the stemmer, first the morphological database was exploited for a possible lemmatisation; if the token was not found, the stemmer was asked for the morphological stem.

2.1.2 Interpreting the Subcategorisation Frames

Once the subcategorisation frame tokens were formulated in an as general as possible form, I could start to interpret the syntactic information in order to (i) filter the information I needed, namely assign subcategorisation frame types to verbs, and (ii) gain some (statistical) insight into their properties. The two issues went hand in hand which each other. The following paragraphs describe some examples of the empirical properties of the data.

Active and Passive Sentences The extraction of subcategorisation frames from the 100 million words in the BNC resulted in a total of 5,419,708 frame tokens, representing the same number of parsed sentences: 4,852,656 active (90%) and 567,052 passive (10%) tokens/sentences. These frames still include sentences headed by auxiliaries, which will be disregarded from now on, since I am only interested in the properties of lexical verbs. This left a total of 3,428,273 subcategorisation frames to work with.

Verbs in the BNC This paragraph considers the questions which verbs appeared as head of the subcategorisation frames, with which particles they appeared, and how often the combinations appeared:
I only considered (and continued working with) those verbs which appeared at least 100 times in the BNC (with some syntactic function, so not necessarily as finite verb, as head of the sentence), which made a total of 3,186 different verbs. I created a list with all verbs and their different particles (no particle is indicated by '-'), alltogether 12,238 types, accompanied by their frequencies in the defined subcategorisation frames. The following example presents this information for the verb *give*.

```
give -          35855
give away         196
give back          63
give down          13
give in           182
give off           74
give on            13
give out          172
give over          71
give round          7
give through        1
give to             8
give up          1187
```

One point to mention is that it is not possible to distinguish between the different verb senses of a (polysemous) verb. A verb therefore represents all possible senses.

Subcategorisation Frame Types in the BNC In the course of defining a fixed set of types of subcategorisation frames, I created a sequence out of the categorical features appearing in the frames, separated by colons. For example, the frame type consisting of a subject and two objects is formulated as `subj:obj:obj`. I partly restricted the order of the arguments in the automatic process: the subcategorisation frame types had to start with the subject, followed by first all objects, then all prepositional phrases, and finally all other arguments. Leaving the order of the arguments completely undefined would have resulted in low frequencies for the single types, as it was the case for the sequence `subj:obj:adv:pp`, for example.

To put the information in more concrete terms, I specified the following refinements:

- For each prepositional phrase, I added information about the prepositional head to the definition of the category. A prepositional phrase is therefore indicated as `pp.`*preposition*, for example `pp.with` for a prepositional phrase headed by the preposition *with*.
- Concerning the use of *by*-phrases in passive sentences, I examined 100 such sentences and found out that in 95% of them the *by*-phrase contained the subject of the syntactic deep structure of the sentence. I generalised this by always assigning the role of the subject to the *by*-phrase in passive sentences (if there was any).

The above definitions resulted in 7,444 different types of subcategorisation frames, from which you find examples below. Each type is followed by its frequency.

`subj`	569525
`subj:adv`	86391
`subj:ap`	59206
`subj:ap:adv`	839
`subj:ap:that`	970
`subj:ap:that:adv`	14
`subj:ap:to`	1431
`subj:ap:to:adv`	18
`subj:obj`	836141
`subj:obj:adv`	54709
`subj:obj:ap`	21405

```
subj:obj:ap:adv                          180
subj:obj:as                            14776
subj:obj:obj                           89545
subj:obj:obj:adv                        2123
subj:obj:obj:obj                        1391
subj:obj:obj:obj:adv                      28
subj:obj:obj:pp.aboard                     3
subj:obj:obj:pp.about                    605
subj:obj:obj:pp.about:adv                  3
subj:obj:obj:pp.above                     46
subj:obj:obj:pp.above:pp.in                1
subj:obj:obj:pp.according_to              68
subj:obj:obj:pp.according_to:adv           1
subj:obj:obj:pp.across                    66
subj:obj:obj:pp.across:adv                 2
subj:obj:obj:pp.adjacent_to                4
subj:obj:obj:pp.after                    392
[...]
subj:obj:pp.across:s                       2
subj:obj:pp.across:to                      2
subj:obj:pp.adjacent_to                    5
subj:obj:pp.adjacent_to:pp.on              1
subj:obj:pp.after                       3847
subj:obj:pp.after:adv                     63
subj:obj:pp.after:pp.about                 1
subj:obj:pp.after:pp.according_to          2
subj:obj:pp.after:pp.after                 6
subj:obj:pp.after:pp.against               3
subj:obj:pp.after:pp.aged                  1
subj:obj:pp.after:pp.along_with            1
subj:obj:pp.after:pp.around                1
subj:obj:pp.after:pp.as                    4
[...]
subj:pp.by:s                              57
subj:pp.by:s:adv                           2
subj:pp.by:that                           97
subj:pp.by:to                             88
subj:pp.by:vger                           53
subj:pp.by:vger:adv                        3
subj:pp.by_means_of                       20
[...]
subj:s                                 93758
subj:s:adv                               692
subj:sub                                8551
subj:sub:adv                             174
subj:that                              96890
subj:that:adv                            633
subj:to                               187503
subj:to:adv                             3223
```

```
subj:to:to                          719
subj:vbase                        60118
subj:vbase:adv                     4449
subj:vbase:to                        32
subj:vger                         15561
subj:vger:adv                       959
```

A further step in the interpretation of the data was the combination of the verbs with the set of subcategorisation frame types they appeared with in the corpus. The tuples are followed by their frequencies and the total frequencies of the verb-particle type. The latter number supported the calculation of relative instead of absolute frequencies of the tuples later on, considering how often the verb appeared in total.
I stay with the example verb *give* (but only list a part of the frames):

```
give    -       subj                          758      35855
give    -       subj:adv                      105      35855
give    -       subj:ap                        58      35855
give    -       subj:ap:adv                     1      35855
give    -       subj:ap:to                      4      35855
give    -       subj:obj                     9982      35855
give    -       subj:obj:adv                  498      35855
give    -       subj:obj:ap                    60      35855
give    -       subj:obj:as                    53      35855
give    -       subj:obj:obj                13430      35855
[...]
give    away    subj                          16        196
give    away    subj:adv                       2        196
give    away    subj:obj                     115        196
give    away    subj:obj:adv                   2        196
give    away    subj:obj:pp.about              2        196
give    away    subj:obj:pp.as                 1        196
give    away    subj:obj:pp.at                 2        196
give    away    subj:obj:pp.during             2        196
give    away    subj:obj:pp.for                2        196
give    away    subj:obj:pp.in                 5        196
give    away    subj:obj:pp.in_return_for      1        196
give    away    subj:obj:pp.on                 2        196
give    away    subj:obj:pp.to                20        196
give    away    subj:obj:pp.to:adv             1        196
give    away    subj:obj:pp.with               6        196
give    away    subj:obj:pp.without            3        196
give    away    subj:obj:pp.worth              3        196
give    away    subj:pp.about                  2        196
give    away    subj:pp.during                 1        196
give    away    subj:pp.in                     3        196
give    away    subj:pp.on                     1        196
```

give	away	subj:pp.to	1	196
give	away	subj:pp.with	1	196
give	away	subj:s	2	196
give	back	subj	13	63
give	back	subj:obj	31	63
give	back	subj:obj:adv	2	63
give	back	subj:obj:pp.in	3	63
give	back	subj:obj:pp.to	9	63
give	back	subj:pp.at	1	63
give	back	subj:pp.through	1	63
give	back	subj:pp.to	3	63
give	down	subj	3	13
give	down	subj:obj	5	13
give	down	subj:obj:pp.of	1	13
give	down	subj:obj:pp.on	1	13
give	down	subj:obj:pp.to	1	13
give	down	subj:pp.at	1	13
give	down	subj:pp.beside	1	13
give	in	subj	92	182
give	in	subj:adv	10	182
give	in	subj:obj	3	182
give	in	subj:pp.about	1	182
give	in	subj:pp.after	1	182
give	in	subj:pp.at	1	182
give	in	subj:pp.for	1	182
give	in	subj:pp.for_fear_of	1	182
give	in	subj:pp.on	1	182
give	in	subj:pp.to	69	182
give	in	subj:pp.to:adv	1	182
give	in	subj:pp.under	1	182
give	off	subj	14	74
give	off	subj:obj	50	74
give	off	subj:obj:pp.for	1	74
give	off	subj:obj:pp.in	1	74
give	off	subj:obj:pp.into	1	74
give	off	subj:obj:pp.of	1	74
give	off	subj:obj:pp.on	1	74
give	off	subj:obj:pp.rather_than	1	74
give	off	subj:obj:pp.to	1	74
give	off	subj:pp.after	1	74
give	off	subj:pp.to	1	74
give	off	subj:s	1	74
give	on	subj	1	13
give	on	subj:obj	1	13
give	on	subj:pp.to	11	13
give	out	subj	42	172
give	out	subj:adv	3	172
give	out	subj:obj	94	172
give	out	subj:obj:adv	1	172

give	out	subj:obj:pp.at	3	172
give	out	subj:obj:pp.during	2	172
give	out	subj:obj:pp.for	1	172
give	out	subj:obj:pp.in	2	172
give	out	subj:obj:pp.on	1	172
give	out	subj:obj:pp.over	1	172
give	out	subj:obj:pp.than	1	172
give	out	subj:obj:pp.to	7	172
give	out	subj:obj:pp.towards	1	172
give	out	subj:obj:pp.with	1	172
give	out	subj:obj:pp.without	2	172
give	out	subj:pp.against	1	172
give	out	subj:pp.amid	1	172
give	out	subj:pp.in	2	172
give	out	subj:pp.rather_than	1	172
give	out	subj:pp.to	2	172
give	out	subj:pp.up_to	1	172
give	out	subj:pp.with	1	172
give	out	subj:that	1	172
give	over	subj	3	71
give	over	subj:obj	7	71
give	over	subj:obj:adv	1	71
give	over	subj:obj:pp.as	1	71
give	over	subj:obj:pp.in	1	71
give	over	subj:obj:pp.to	54	71
give	over	subj:pp.at	1	71
give	over	subj:pp.from	1	71
give	over	subj:pp.to	2	71
give	round	subj	5	7
give	round	subj:obj	1	7
give	round	subj:pp.in	1	7
give	through	subj:pp.with	1	1
give	to	subj	1	8
give	to	subj:obj	6	8
give	to	subj:obj:pp.to	1	8
give	up	subj	350	1187
give	up	subj:adv	27	1187
give	up	subj:obj	608	1187
give	up	subj:obj:adv	28	1187
give	up	subj:obj:pp.after	7	1187
give	up	subj:obj:pp.as	4	1187
give	up	subj:obj:pp.at	4	1187
give	up	subj:obj:pp.because_of	1	1187
give	up	subj:obj:pp.before	1	1187
give	up	subj:obj:pp.despite	1	1187
give	up	subj:obj:pp.for	23	1187
give	up	subj:obj:pp.for_fear_of	1	1187
give	up	subj:obj:pp.in	15	1187
give	up	subj:obj:pp.in_response_to	1	1187

```
give    up      subj:obj:pp.like                1       1187
give    up      subj:obj:pp.of                  6       1187
give    up      subj:obj:pp.on                  3       1187
give    up      subj:obj:pp.over                1       1187
give    up      subj:obj:pp.to                 17       1187
give    up      subj:obj:pp.with                2       1187
give    up      subj:obj:pp.without             2       1187
give    up      subj:pp.about                   1       1187
give    up      subj:pp.after                   6       1187
give    up      subj:pp.along                   1       1187
give    up      subj:pp.as                      3       1187
give    up      subj:pp.at                      8       1187
give    up      subj:pp.by                      1       1187
give    up      subj:pp.during                  1       1187
give    up      subj:pp.for                     3       1187
give    up      subj:pp.in                     11       1187
give    up      subj:pp.on                     37       1187
give    up      subj:pp.since                   1       1187
give    up      subj:pp.through                 1       1187
give    up      subj:pp.with                    2       1187
give    up      subj:pp.without                 7       1187
give    up      subj:s                          1       1187
```

Arguments within the Subcategorisation Frame Types Having considered the different types of subcategorisation frames themselves, I now turn to describe the arguments within the frames. I list all types of subcategorisation frames – again –, but this time each frame is followed by the words appearing in the different argument positions. So you will first find a line with the subcategorisation frame followed by its frequency, then an empty line, and after that a list of words (one per line) accompanied by their frequencies. An empty line marks the end of the word list for one argument slot, then the word list for the next argument slot follows. Such lists were created for all argument slots within a subcategorisation frame type, disregarding the verb types.

I should explain some strange appearance of arguments: if no subject at all appeared in the subcategorisation frame (usually: passive sentences), the count of the subject was nevertheless increased, and the subject was defined as not appearing ('-'). This treatment reflects the fact that a subject is obligatory in English sentences.
In addition, the word `dummy` might appear on the list. I chose that as argument in case I could not determine a head in the parse structures.

Here is one (incomplete, as the numbers tell, but nevertheless illustrative) example for the word tokens in two different subcategorisation frames:

subj:obj:pp.after	3847	
-	1007	[subj]
analyst	47	
angel	16	
band	110	
heads	48	
heroine	2	
herself	26	
humans	28	
i	663	
daughter	143	[obj]
day	851	
days	428	
deadline	19	
death	196	
defendant	90	
demise	12	
attending	3	[pp.after]
cuts	1	
family	1	
friend	2	
subj:obj:pp.at	22535	
-	6447	[subj]
bailiff	6	
band	110	
bank	357	
he	3293	
head	114	
heads	48	
crowd	80	[obj]
dancer	20	
days	428	
deadline	19	
death	196	
defendant	90	
demand	206	
party	69	[pp.at]
pass	3	
pennington	1	
pitts	1	
place	69	
pn	2770	

Finally, I created the same information for the subcategorisation frames in connection with a specific verb-particle type. Here is one example for the verb *give* with the particle *up*, when appearing with a subject and an adverb:

```
give up  subj:adv  27  1187

1950                  1                  [subj]
bastard               1
bean                  1
clegg                 1
generation            1
he                    2
i                     4
japanese              1
month                 1
padre                 1
pn                    6
reading               1
reporter              1
they                  2
you                   3

altogether            2                  [adv]
completely            4
easily               13
gracefully            1
half                  1
immediately           1
more                  1
much                  1
now                   1
soon                  1
then                  1
```

To summarise the first step of inducing subcategorisation frames of verbs I briefly list the relevant data we are provided with now:

- Frequency information about the verbs (defined as verb-particle types) in the BNC
- Frequency information about the subcategorisation frames (defined as 7,444 frame types) in the BNC
- Joint frequency information for the types of verbs and subcategorisation frames
- Token and frequency information about arguments in the subcategorisation frames

- Token and frequency information about the arguments in the subcategorisation frames depending on the verb types

In the following step of defining selectional preferences for subcategorisation frames the tokens in the frames will be generalised to conceptual classes.

2.2 Selectional Preferences for Subcategorisation Frames

Following the induction of subcategorisation frame types for verbs, the step of defining selectional preferences for the frames is divided into two sub-tasks:

1. Assigning the words which realise the verbs' arguments to conceptual classes in order to classify them

2. Identifying a preferential ordering on conceptual classes for the argument slots in the subcategorisation frames

This demand is illustrated by a short example: consider one of the possible subcategorisation frames for the verb *drink*, the transitive frame `subj:obj` which requires a subject noun phrase and an object noun phrase. In determining the semantically preferred class for the object slot, I consider all nouns which appeared in that slot, for example *coffee, milk, beer*. A preferred conceptual class for this argument would then be *beverage*.

Some implicit problems within the course of defining the selectional restrictions should be mentioned:

- A word may represent multiple senses belonging to different conceptual classes. For example, when considering the noun *coffee* isolated from its context, we do not know whether we are talking about the beverage *coffee*, the plant *coffee* or a *coffee* bean. This means that assigning a word to a conceptual class is closely connected with disambiguating the sense.

- Assigning words to conceptual classes presupposes that there is a system of conceptual classes available. So either it is possible to use an existing taxonomy, or the taxonomy has to be defined.

Following I introduce into approaches concerned with automatic classification. Some approaches work without a provided classification system:

[Hindle, 1990] classifies nouns according to the predicate-argument structures they appear in. Each noun is characterised by the variety of verbs it occurs with, and on this basis the nouns are grouped by the measurement of mutual information, according to the extend to which they appear in similar environments.

A similar syntactic background is used in [Pereira et al., 1993]; they classify nouns according to their distribution as direct objects of verbs. Words are represented by the relative frequency distributions of the contexts in which they appear, and relative entropy between those distributions is used as the similarity measure for clustering. The result is a hierarchical ordering of word clusters[4].

[Schütze, 1992] creats a high-dimensional space in which words and contexts are represented as vectors. The dimensions of the vectors are words, and the numbers which express the strength of the dimensions are determined by the co-occurrence of the word/context to be represented and the dimension words. Schütze's algorithm contains the following steps: first he automatically determines the words which are the dimensions of the space, then he calculates the co-occurrence values of the dimension words with the words he is interested in, and on this basis he calculates the context vector as normalised average of the vectors of some words appearing together in that context.
With this algorithm, Schütze clusters words by assigning vectors to them, since the vectors can be geometrically interpreted as points in space, and the points for similar words accumulate in a certain area in space. The approach can be utilised for word sense disambiguation by computing the context vector of the position of an ambiguous word and determining how close it is to the dimensions of the space which correspond to the different senses.

[Luk, 1995] uses the 2,000 word controlled vocabulary from the *Longman Dictionary of Contemporary English* to define conceptual sets for each sense of a word as well as for contexts – according to the words used in the dictionary definition. The similarity within the sets determines the similarity of word senses, and applying a measure based on mutual information provides the possibility of disambiguating a polysemous word.

[Yarowsky, 1995] bases his approach on two powerful constraints: (i) there is only *one sense per collocation* – nearby words provide strong and consistent clues to the sense of a target word – and (ii) there is only *one sense per*

[4]For the sake of my task of determining selectional preferences the terms *cluster* and *class* can be considered to be identical.

discourse – the sense of a target word is highly consistent within any given document. The second constraint may be overridden when local evidence is strong.
Based on these constraints Yarowsky provides an approach for sense disambiguation: he first identifies all appearances of a polysemous word; for each possible sense he then determines a small number of training examples representative of that sense; after training he is equipped with a decision list of (salient context) words which could be applied to disambiguate the polysemous word in further contexts.

Some approaches utilise already existing class taxonomies:

[Yarowsky, 1992] uses the 1,043 categories in *Roget's International Thesaurus*. He collects contexts which are representative of the categories by extracting the concordances for all occurrences of each member of the category, and identifies and weights salient words for the contexts by an estimate similar to mutual information. By this approach, Yarowsky provides salient words for each category which can be used to disambiguate a polysemous word.

[Ribas, 1994] and [Ribas, 1995] utilise the semantic taxonomy provided by WordNet (see [Beckwith et al., 1991]) to assign classes to arguments within subcategorisation frames. He provides a list of complement co-occurrence triples $\langle$*verb-lemma, syntactic-relationship, noun-lemma*$\rangle$ extracted from a corpus, creates a space of candidate classes from the WordNet taxonomy, evaluates the appropriateness by statistical means based on the measure of mutual information, and obtaines a set of syntactic subcategorisation frames in the pattern of $\langle$*verb-lemma, syntactic-relationship, semantic-class, weight*$\rangle$.

[Agirre and Rigau, 1996] also utilise WordNet to teach their system how words are clustered into semantic classes and how semantic classes are hierarchically organised. This is the basis for disambiguating a polysemous word; they assume that each sense of (i) a polysemous word and (ii) its context words belongs to a sub-hierarchy of WordNet. By measuring the density in the different parts of the hierarchy they find out the relevant sense of the word in the respective context.

[Abe and Li, 1996] define an association norm which measures the co-occurrence between two categories, for example a verb and a noun (in the case of assigning classes to the arguments of subcategorisation frames), by a norm similar to mutual information. They utilise an existing taxonomy; for each verb they calculate a cut within the tree which defines a partition over the set of all nouns represented by the leaf nodes. In this way they assign parts of the

tree as possible classes to the argument nouns within the subcategorisation frames.

[Resnik, 1993] and [Resnik, 1997] also use the WordNet taxonomy for a probabilistic model capturing the co-occurrence behaviour of predicates and conceptual classes. Resnik determines selectional preferences of predicates for certain classes by comparing the probability of the class occurring with an arbitrary predicate with the probability of the class occurring with the specific predicate. As measure he uses relative entropy.
For determining the relevant conceptual class in a predicate-argument relation, Resnik treats each occurrence of a word in the argument position as if it represents any of the classes to which the senses belong. Credit tends to accumulate in the taxonomy only in those classes for which there is real evidence of co-occurrence.

[Abney and Light, 1998] can define any semantic class hierarchy in form of a hidden Markov model, where the states and transitions of the HMM are identified with the nodes and arcs of the hierarchy. Training the HMM on predicate-argument relations results in an optimal path through the hierarchy for each predicate to identify the preferred conceptual class.

I decided to define an approach closely following Resnik's approach, with an extension supposed by Ribas. The reasons for this decision are as follows:

- WordNet is a lexical system already available, so it is not necessary to define a taxonomy of classes.
- The WordNet taxonomy is organised hierarchically. According to cognitive psychology, this is a plausible representation for semantic concepts, since hierarchical relationships between categories are one way in which words can be related in meaning (compare [Collins and Quillian, 1969], for example).
- WordNet does not provide an explicit measure of distance, so this has to be supported by the relevant approach.
- The approach does not require an explicit sense disambiguation; the disambiguation takes place by the tendency towards a certain class in the taxonomy.

For a detailed description of the approach subsection 2.2.1 introduces into the idea and implementation of WordNet, before subsection 2.2.2 explains the determination of selectional preferences with WordNet.

2.2.1 The Lexical Database WordNet

Concerning the description of WordNet I first briefly present the idea of the lexical database before I describe its design.

Idea of WordNet WordNet is an on-line lexical reference system ([Miller et al., 1990][5], [Beckwith et al., 1991]), whose design was inspired by psycholinguistic theories of human lexical memory, for instance [Caramazza and Berndt, 1978], [Collins and Quillian, 1969].

The lexicon distinguishes the categories nouns, verbs, adjectives, and adverbs. Within each of the categories, the words are organised into synonym sets (so-called *synsets*), sets representing a common underlying lexical concept. In addition, WordNet defines semantic relations as pointers between these sets.

So as not to be restricted to lemmatised word forms, WordNet provides an inflectional morphology.

Design in WordNet Concerning the design and implementation of WordNet, I will concentrate on the nouns in the WordNet hierarchy, since for us they represent the relevant part of speech.

The WordNet noun database (version 1.5) contains 87,642 nouns; the nouns are assigned to 60,545 synonymous sets, the sets uniting synonymous nouns. The noun synsets correspond to semantic classes, the items interesting for us.

The lexical relationships between the noun synsets are realised by *hypernymy/hyponymy* (super-/sub-ordination), *meronymy/holonymy* (part-of-/whole-of-relation), and *antonymy* (opposite-of-relation). The hypernymy/ hyponymy relation organises the nouns into a semantic hierarchy, an inheritance system where the sub-ordinated nouns inherit the properties from the super-ordinated ones.

Actually, there is not only one hierarchy of nouns, because the WordNet builders did not want to specify an artificial top level concept for a unique

[5]This reference is one out of five papers available at `ftp://ftp.cogsci.princeton.edu/pub/wordnet/5papers.ps`.

hierarchy. WordNet provides 25 beginners in separate files[6], which are partly united to higher instances, so 11 top level concepts (and therefore hierarchies) are defined: *{abstraction}, {act, human action, human activity}, {event}, {group, grouping}, {location}, {phenomenon}, {possession}, {psychological feature}, {shape, form}* and *{state}*. The hierarchies vary in size, but are all kept shallow. They are not mutually exclusive.

The pointer symbols, i.e. the symbols indicating the kind of lexical relation between the noun synsets, are the following:

- ! Antonym
- `@` Hypernym
- `~` Hyponym
- `#m` Member Meronym
- `#s` Substance Meronym
- `#p` Part Meronym
- `%m` Member Holonym
- `%s` Substance Holonym
- `%p` Part Holonym
- = Attribute

There are essentially two databases which organise the WordNet noun hierarchy:

1. Each noun is assigned
   - the part of speech `n`,
   - the number of senses it has,
   - the number of pointers it is involved in, followed by the different pointer symbols, and
   - the number of synsets it is member of, followed by the different synsets.

   For example, the entry for the noun *tree* looks as follows:

   ```
   tree n 8 5 @ ~ #m %s %p 2 07991027 08514899
   ```

   The interesting information for us is that *tree* is member of the two synsets `07991027` and `08514899`. It has two senses, since each synset defines one noun sense.

[6] These starting concepts are *act, animal, artifact, attribute, body, cognition, communication, event, feeling, food, group, location, motive, object, person, phenomenon, plant, possession, process, quantity, relation, shape, state, substance*, and *time*.

2. Each synset can be identified by
   - its (unique) synset number,
   - the beginner's file number (as explained above; see also appendix B.1),
   - the part of speech `n`,
   - the number of words in the synset, followed by the words themselves,
   - the number of pointers, followed by a list of 4-tuples (pointer, synset, part of speech, element concerning the pointer), and
   - possibly a gloss.

   Each synset except for those synsets at the top of the hierarchies has at least one super-ordinated synset (indicated by `@`).

   As an example, the synset containing the two words *climb* and *mount* is given the synset number `00182735`, member of file number 4, and has one hypernym, the synset number `00182471`:

```
00182735 04 n 02 climb mount 005 @ 00182471 n 0000
                                 ~ 00182896 n 0000
                                 ~ 00182998 n 0000
                                 ~ 00183210 n 0000
                                 ~ 00183326 n 0000
                  | the act of climbing something
```

To give an example of how the data is processed by WordNet, here is the output when asking for the hypernyms of the noun *tree*:

```
2 senses of tree

Sense 1
tree
       => woody plant, ligneous plant
           => vascular plant, tracheophyte
               => plant, flora, plant life
                   => life form, organism, being, living thing
                       => entity

Sense 2
tree, tree diagram
       => plane figure, two-dimensional figure
           => figure
               => shape, form
```

For each sense of the word the synset is printed, followed by the part of the hierarchy above the word, up to the top level.

2.2.2 Selectional Preferences Coded by WordNet

I start this section with the question of how it is possible to utilise the WordNet hierarchy as source for the definition of selectional preferences. As explained before, the selectional preferences in subcategorisation frames are defined by an ordering of preferences on semantic concepts. The semantic concepts again can be identified by WordNet synsets, the more general the closer they are to the top of the hierarchy. So the WordNet synsets are regarded as conceptual classes, concerning the approach I apply for the definition of selectional preferences.
The variety of subcategorisation frame types presented in subsection 2.1.2 contained 247 different syntactic categories (when distinguishing between the different kinds of prepositional phrases). In the step of semantic classification, however, I concentrated on the nouns within the argument slots for the subject, the objects and the prepositional phrases.
So the task concerning the definition of selectional preferences for the arguments in subcategorisation frames can be put in concrete terms as determining preferences concerning WordNet synset classes for the subject, objects and prepositional phrases in the subcategorisation frames obtained in the first step.

This is where Resnik's idea comes into play. Let us have a closer look at his approach and redefine the essential ideas for my usage. Resnik defines the term *selectional preference* as the amount of information a predicate (henceforth: verb, since I am only interested in verbal predicates) provides about its semantic argument classes. The more "extra-ordinary" the semantic arguments in a subcategorisation frame of a certain verb are, the more information is provided by the verb, i.e. the stronger the selectional preference is.
The degree of selectional preference is calculated by relative entropy ([Kullback and Leibler, 1951]), which measures the difference between two distributions, in this case called the *prior distribution* and *posterior distribution.*
The prior distribution determines how probable it is that a certain semantic class c appears as argument in a certain argument position of a subcategorisation frame s, without regarding the identity of the verb: $p(s, c)$.
The posterior distribution determines how probable it is that a certain semantic class c appears as argument in a certain argument position of a subcategorisation frame s of a certain verb v: $p(s, c|v)$.
The larger the difference between these two distributions is when accumulating it for all semantic classes, the more influence the respective verb has on

its arguments, and therefore the larger the selectional preference S of that verb is:

$$S(v) = \sum_c p(s,c|v) \ \ log\frac{p(s,c|v)}{p(s,c)} \tag{2.1}$$

Given the definition of selectional preference, Resnik defines the "semantic fit" of a particular semantic class by its relative contribution to the selectional preference of the verb, and calls it *selectional association* A:

$$A(v,s,c) = \frac{1}{S(v)} \ \ p(s,c|v) \ \ log\frac{p(s,c|v)}{p(s,c)} \tag{2.2}$$

This is almost what I needed. But in contrast to Resnik's approach, the selectional association I needed was independent of the overall selectional preference of the verb, since I only compared the selectional association of the same verb considering different classes to find the "best fitting" classes for the verb's arguments, so the normalisation factor $\frac{1}{S(v)}$ was not necessary. I therefore changed the selectional association A to A':

$$A'(v,s,c) = p(s,c|v) \ \ log\frac{p(s,c|v)}{p(s,c)} \tag{2.3}$$

With equation (2.3) it was possible to determine the selectional association of the verbs concerning the different conceptual classes (the WordNet synsets) in the argument slots of the subcategorisation frames. Determining the classes with the largest association values presented the selectionally most preferred concepts.

The first task for me in determining the maximally associated semantic noun classes for an argument position in a subcategorisation frame of a verb was to find out how to estimate the probabilities:
The probability of each class regarding a specific argument position within a certain verb-frame type was estimated as its maximum likelihood estimate (MLE): the relation between (a) how often the class appeared in that argument position of the verb, and (b) how often any class appeared in that argument position of the verb (i.e. the number of times the verb-frame type appeared in total):

$$p(s,c|v) = \frac{f(v,s,c)}{\sum_{c' \in class} f(v,s,c')} = \frac{f(v,s,c)}{f(v,s)} \tag{2.4}$$

The probability of each class regarding a specific argument position within a certain frame independent of the verb, so generalising over all verbs, was again estimated as the maximum likelihood estimate: the relation between

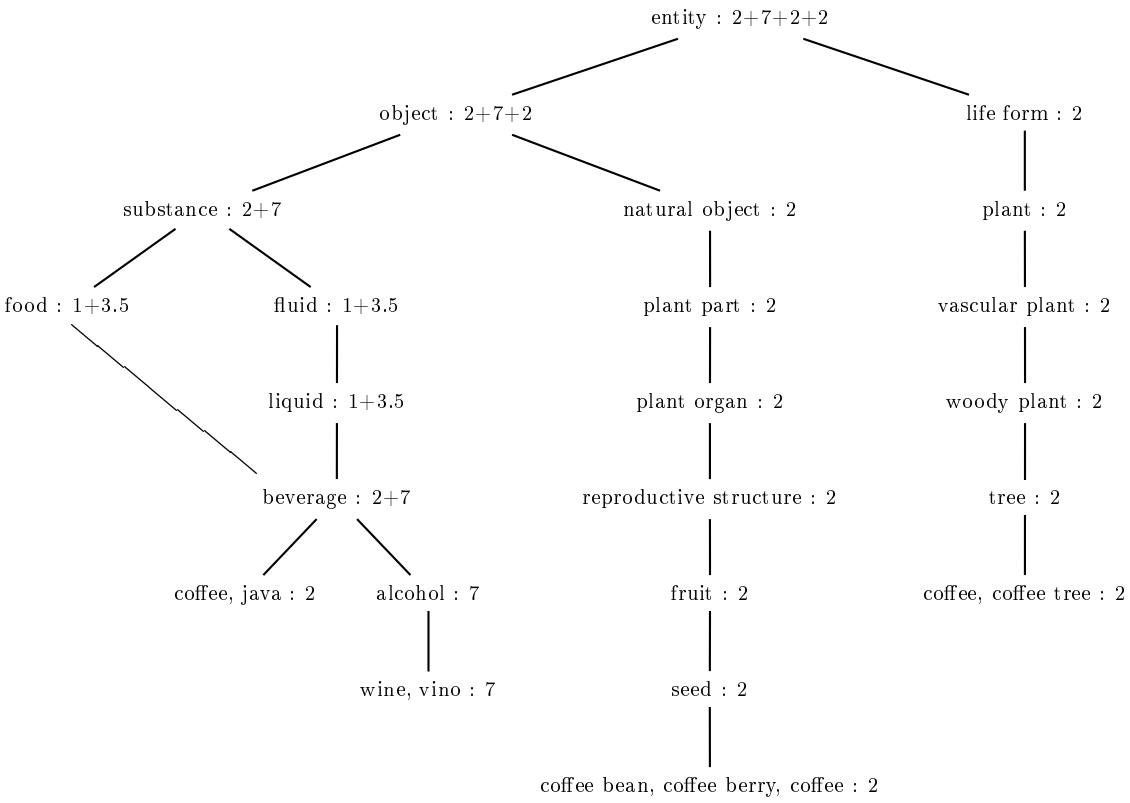
(a) how often the class appeared in that argument position, and (b) how often any class appeared in that argument position (i.e. the number of times the frame appeared in total):

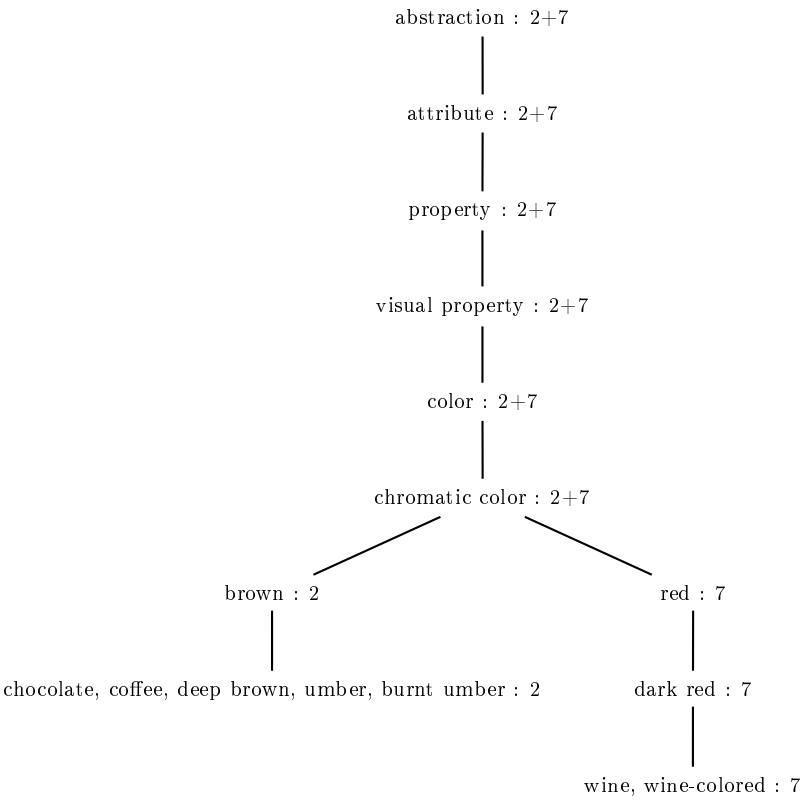
$$p(s,c) = \frac{f(s,c)}{\sum_{c' \in class} f(s,c')} = \frac{f(s,c)}{f(s)} \tag{2.5}$$

The next question then was how to estimate the frequencies. The frequencies of the verbs, the frames and the verb-frame types were already determined in the first overall step described in subsection 2.1.2. But how do we know which class in the WordNet hierarchy had to be assigned what value, concerning a specific argument position?
Each time a noun appeared in a certain argument position (dependent on or independent of the verb), first the number of senses of that noun was determined by looking up (cf. the structure of WordNet) the number of synsets the noun is member of. To each of the synset classes the value $\frac{1}{|senses|}$ was assigned. This division by the number of senses displays the uncertainty about the sense of the noun.
Afterwards, I followed upwards the hierarchy in WordNet from each synset representing a sense of the noun and added the same value to each node until a top node was reached. In case a class has several hypernyms, the value is divided by the number of hypernyms. Having followed this algorithm for all nouns appearing in the same argument position, I ended up with a numerical distribution over the WordNet classes. Each synset class was now assigned the following frequency:

$$f(v,s,c) = f(s,c) = \sum_{noun \in c} \frac{f(noun)}{|senses(noun)|} \tag{2.6}$$

For a better illustration of this algorithm, consider the following example: we are concerned with the direct object position of the verb *drink*, realised by the nouns *coffee* (8 times) and *wine* (14 times) in a training corpus. *coffee* has four senses in the WordNet hierarchy and belongs to a total of 29 classes; *wine* has two senses and belongs to 18 classes. Each of the classes containing the nouns *coffee*/*wine* was assigned $\frac{8}{4} = 2/\frac{14}{2} = 7$, respectively; and for each of the classes that value was projected upwards in the hierarchy:


entity : 2+7+2+2
object : 2+7+2
life form : 2
substance : 2+7
natural object : 2
plant : 2
food : 1+3.5
fluid : 1+3.5
plant part : 2
vascular plant : 2
liquid : 1+3.5
plant organ : 2
woody plant : 2
beverage : 2+7
reproductive structure : 2
tree : 2
coffee, java : 2
alcohol : 7
fruit : 2
coffee, coffee tree : 2
wine, vino : 7
seed : 2
coffee bean, coffee berry, coffee : 2



The algorithm behaves slightly different to Resnik's who splits the number of times a certain noun appears in an argument position by the total number of classes it appears in, up to the top of the hierarchy, in order to describe the degree of uncertainty about the word's sense. Intuitively, my approach (originally proposed by [Ribas, 1994]/[Ribas, 1995]) was an improvement to his idea, since the uncertainty arises from the different senses, not from the number of classes defined in WordNet, which is strongly dependent on the depth of the hierarchy.
But the basic idea which has turned out to make reasonable judgements about verb-argument relationships stays the same. The important bit is that the nouns in the subcategorisation frames may be ambiguous, but credit tends to accumulate in that semantic class to which most of the nouns belong.

Having determined the frequencies of the classes in this way led to the calculation of the prior and posterior probabilities, which again enabled to determine a distribution of selectional association over the WordNet conceptual classes. To restrict the size of the distribution, i.e. the number of classes, I kept to the already mentioned 23 WordNet top level concepts described in appendix B.2. Each verb for each frame type was therefore assigned a

distribution over the 23 general conceptual classes concerning the association with the subject, the objects and the prepositional phrases within the subcategorisation frames.

Finally, I should mention the constraints I placed on the data:

- Not constraining the variety of subcategorisation frames led – because of the explicit prepositions within the frames – to a total number of 7,444. I restricted the subcategorisation frames to those which appeared at least 2,000 times in the training corpus, which left a more usable number of 88.
- I considered only verb-frame types where the frequency of the verb was larger than 10 and the frequency of the verb-frame type larger than 5% of the verb's frequency.
- Not all of the nouns appearing in the subcategorisation frames are defined in WordNet. I filtered the nouns and skipped those not available.
- Nouns which are not defined in WordNet but appear quite often in discourse (mostly pronouns, but also proper names) I provided with an additional synset definition. You can find a list of them and their respective synsets in appendix B.3.
- In addition, numbers – usually integers – are insufficiently defined in WordNet. I could not cover all possible integers, but I created a definition for all integers between 1 and 10,000, assigning them to the synset `{integer, whole number}`.

I will now present an example to illustrate the enlarged information about the subcategorisation frames. Staying with the example verb *give* (here with no particle), the list is similar to the description of the subcategorisation frames followed by the different argument nouns. But this time each type of the frame is accompanied by a list of the 23 WordNet concepts for each argument. Each line defines the WordNet node abbreviation, the association of the verb-frame type for that node and the maximum likelihood estimate.

```
give -  subj:obj  9982  35855

LifeForm        0                       0                           [subj]
Cell            6.12437430162519e-05    0.00122279244763069
Agent           0                       0
PhysObject      0.0187478989352773      0.374320511744637
Thing           0                       0
Whole           7.10949124985579e-05    0.00141948087733848
Content         0.000219297706361356    0.00437849755607262
Unit            0                       0
Part            0.00135711329350237     0.0270961212385981
Essential       0                       0
Inessential     0                       0
Variable        0                       0
Anticipation    0                       0
Psycho          0.00297932455033738     0.0594851878701552
Abstract        0.0253305642285428      0.505749993508183
Location        0                       0
Shape           0.00131861251417423     0.026327414757385
State           0                       0
Event           0                       0
Action          0                       0
Group           0                       0
Possession      0                       0
Phenomenon      0                       0

LifeForm        0                       0                           [obj]
Cell            0                       0
Agent           0                       0
PhysObject      0                       0
Thing           0                       0
Whole           0                       0
Content         0                       0
Unit            0                       0
Part            0                       0
Essential       0                       0
Inessential     0                       0
Variable        0                       0
Anticipation    0                       0
Psycho          0.0132362280814078      0.0509724715909643
Abstract        0.167765324439536       0.646061187624506
Location        0                       0
Shape           0.000813299371709867    0.0031320009646601
State           0                       0
Event           0.0273558670665238      0.105346942370479
Action          0.0459318148670369      0.176882576670019
Group           0                       0
Possession      0.000236260897945409    0.000909836384996566
Phenomenon      0.0043352541943181      0.0166949843943748
```

```
give -  subj:obj:obj  13430  35855

LifeForm        0                       0                           [subj]
Cell            0                       0
Agent           0                       0
PhysObject      0                       0
Thing           0                       0
Whole           0                       0
Content         0                       0
Unit            0                       0
Part            0                       0
Essential       3.99417468035753e-05    0.00242973501075989
Inessential     0                       0
Variable        1.81208519975667e-06    0.000110232706495813
Anticipation    0                       0
Psycho          0.00618237657307974     0.376086125706674
Abstract        0                       0
Location        0                       0
Shape           0.000438182672600387    0.02665551374008
State           0.00294856942946037     0.179367277291309
Event           0.002521064297632       0.153361299355714
Action          0.00304612460268746     0.185301750338677
Group           0                       0
Possession      0                       0
Phenomenon      0.00126065406123865     0.0766880658502897

LifeForm        0.172565087237197       0.486057948364913           [obj]
Cell            0                       0
Agent           0.17513317888456        0.493291400839656
PhysObject      0                       0
Thing           0                       0
Whole           0                       0
Content         0.000241439940841205    0.000680055300741769
Unit            0                       0
Part            0                       0
Essential       0                       0
Inessential     0                       0
Variable        0                       0
Anticipation    0                       0
Psycho          0                       0
Abstract        0                       0
Location        0                       0
Shape           0                       0
State           0                       0
Event           0                       0
Action          0                       0
Group           0.00709015779972923     0.0199705954946895
Possession      0                       0
Phenomenon      0                       0
```

```
LifeForm        0                       0                           [obj]
Cell            0                       0
Agent           0                       0
PhysObject      0                       0
Thing           0                       0
Whole           0                       0
Content         0                       0
Unit            0                       0
Part            0                       0
Essential       0                       0
Inessential     0                       0
Variable        0                       0
Anticipation    0                       0
Psycho          0.0535767346905256      0.228887288260387
Abstract        0.0154121248675518      0.0658427484922524
Location        0                       0
Shape           0.00178290095264973     0.00761680170779466
State           0.0693752518281327      0.296380758458778
Event           0.000430499307456295    0.00183915312590095
Action          0.0701926341374568      0.299872729765613
Group           0                       0
Possession      0.0112655364894741      0.0481279442050461
Phenomenon      0.0120390673457691      0.0514325759842277

give -  subj:obj:pp.to  3735  35855

LifeForm        0                       0                           [subj]
Cell            1.48539478311714e-05    0.000126358987348643
Agent           0                       0
PhysObject      0.0236193982531406      0.200924581059053
Thing           0.000312793557805112    0.00266086010686602
Whole           0                       0
Content         8.34140342530556e-05    0.000709583271644619
Unit            0                       0
Part            0.000156357473868931    0.00133009569489729
Essential       0                       0
Inessential     0                       0
Variable        0                       0
Anticipation    0                       0
Psycho          0.0226934055353757      0.193047382119156
Abstract        0.0254121071735822      0.21617472777914
Location        0.00298741517534233     0.0254132275565193
Shape           0.000130307527635115    0.00110849502253704
State           0.0118805672583152      0.101065148804267
Event           0.000708506104706788    0.00602709225443812
Action          0.00873536793856        0.0743096892071955
Group           0.0157342145780905      0.13384720637308
Possession      0                       0
Phenomenon      0.00508484376767028     0.0432555517638575
```

LifeForm	0	0	[obj]
Cell	0	0	
Agent	0	0	
PhysObject	0	0	
Thing	0	0	
Whole	0	0	
Content	0	0	
Unit	0	0	
Part	0	0	
Essential	0	0	
Inessential	0	0	
Variable	0	0	
Anticipation	0	0	
Psycho	0.0695842595942418	0.187329656624309	
Abstract	0.119741227303197	0.322358578294586	
Location	0.0023279386472826	0.00626710623898046	
Shape	0.00322279057908443	0.00867616119036644	
State	0.0428860251360939	0.115454621628118	
Event	0.00142058281498528	0.00382438920079575	
Action	0.0919834645838808	0.247631158771984	
Group	0	0	
Possession	0.0291387899589887	0.0784453200953067	
Phenomenon	0.0111484373292352	0.0300130079555532	

LifeForm	0.0422524065603397	0.311667252838782	[pp.to]
Cell	0	0	
Agent	0.04009418135481	0.295747494046954	
PhysObject	0	0	
Thing	0.000135084068685891	0.000996423257678664	
Whole	0	0	
Content	0.000371745252216556	0.0027421117741243	
Unit	0	0	
Part	0	0	
Essential	0.000648507003756726	0.00478359489462282	
Inessential	0	0	
Variable	0	0	
Anticipation	0	0	
Psycho	0.0371362868833112	0.273929119209475	
Abstract	0	0	
Location	0	0	
Shape	0	0	
State	0	0	
Event	0	0	
Action	0.00745598734098288	0.0549977452395835	
Group	0.0055577978226769	0.040996092772926	
Possession	0	0	
Phenomenon	0.00191696764988322	0.0141401659658534	

To conclude the step of determining selectional preferences for the arguments within subcategorisation frames I cite some concrete examples. For that, I chose some verbs combined with subcategorisation frames and determined the (maximally) preferred WordNet nodes for all arguments positions:

- The verb *break* (without particle) when appearing with the subcategorisation frame `subj:pp.into` favours an *offender* as subject and a *smile* as pp-object. When regarding both preferences without connection to each other, the choices are pretty good.
- *drive* prefers a *person* as subject and an *artifact* as object in the `subj:obj` frame.
- The `subj:obj` frame for the verb *eat* prefers a living entity as subject and *food* as object.
- The verb *swim* appears with the frame `subj:pp.in` with a *fish* as subject and a *body of water* as pp-object.

Now we have reached the point to feed data into the clustering process for verbs: I have induced subcategorisation frames for the verbs and enriched the frames with selectional preferences in form of a distribution of associations over the top level WordNet classes.

2.3 Clustering Verbs into Semantic Verb Classes

Once equipped with information about the alternation behaviour of verbs concerning the usage of subcategorisation frames and the selectional preferences within the frames I could start clustering the verbs into semantic classes.

What does a process of clustering involve exactly? Generally said, clustering forms classes of items which are similar to each other in a certain property and to a certain extent. The properties relevant in a comparison of items and the definition of the degree of similarity necessary for items in order to belong to one class have to be defined according to the specific data and task.

I based the classification on the two informational versions concerning the data acquired in the preceding steps of my work, i.e. I classified the verbs twice: (i) according to their syntactic alternation behaviour only – the data

resulting from the first overall step –, and (ii) including the selectional preferences for the arguments within the alternating subcategorisation frames – the data resulting from the second overall step. Applying both versions allowed to (a) find out about the correspondence between semantic verb classes and the verbs' syntactic alternation behaviour, and (b) identify the importance of the selectional preferences for the arguments.
The classification was processed by two different algorithms: (i) clustering according to the similarity of the verbs' attributes describing the alternation behaviour, and (ii) clustering by latent classes. The algorithms are described in detail in subsection 2.3.1.
I defined a representative choice of verbs and semantic verb classes for the clustering experiments. The definitions are explained in subsection 2.3.2.
Finally, subsection 2.3.3 describes the experiments I carried out on the basis of the defined classes, processed by the two different algorithms considering the two versions of information.

2.3.1 Clustering Algorithms

My original idea for the classification of the verbs was an iterative clustering algorithm based on a definition by [Hughes, 1994]. Having adjusted the algorithm's notation to my domain, it contained the following steps:

1. Starting point: each verb represents a cluster containing a single element (= the verb). Build a matrix for the differences between the clusters. The differences represent the distances between the clusters.
2. Find the shortest distance in the matrix and therefore the two clusters which are closest to each other.
3. Merge the two clusters.
4. Update the distance matrix.
5. Go back to step 2.

The algorithm raised a number of questions concerning its application to the specific case of clustering verbs semantically. These issues had therefore to be defined by the specific properties of the data:

- *How are the verbs and clusters represented?*

  The first issue to consider was the representation of the verbs and the verb clusters. Before determining distances between clusters, as demanded in step 1 of the algorithm, I had to define some value for

them. The basis for the values was provided by the identification of the verbs' subcategorisation frames with their selectional preferences, as described in the preceding sections, so what was left was defining a representative form. For that, each verb was assigned a distribution over the different types of subcategorisation frames. Following I explain how the attributes in the distributions were determined, depending on the two versions of data I set the algorithms on:

– Version `A`: *Distribution over the subcategorisation frames only*

  In version `A` the verbs were identified by a distribution over the subcategorisation frames only, i.e. each attribute in the distribution characterising the verb was represented by a frame type. As mentioned before, I restricted the choice of subcategorisation frames to those which appeared at least 2,000 times in the BNC, 88 frames in total. For each verb, the distribution over these frames was determined by the maximum likelihood estimate of the verb v appearing with that frame sf, the relation between the number of times the verb appeared with the frame, normalised by the number of times the verb appeared in total:

$$p(sf|v) = \frac{f(v, sf)}{f(v)} \tag{2.7}$$

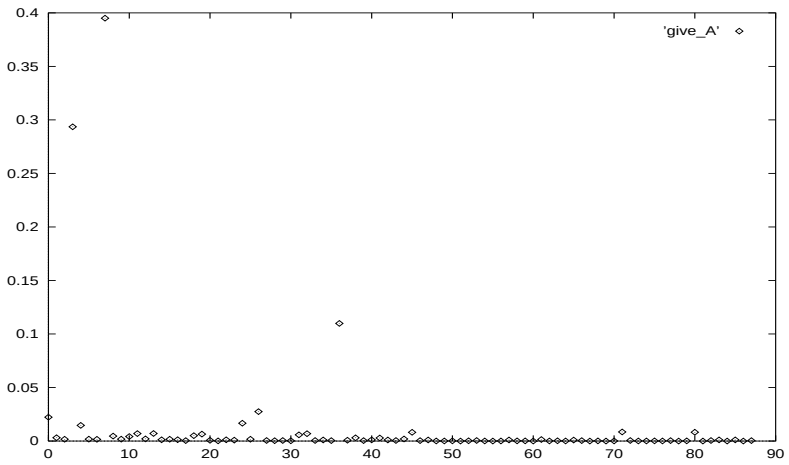
  Staying with the verb *give* as example, the distribution over the 88 subcategorisation frames looks as follows. The frames, i.e. the attributes in the distribution, are numbered from 0 to 87:

```
 0     subj                    0.0222941176470588
 1     subj:adv                0.00308823529411765
 2     subj:ap                 0.00170588235294118
 3     subj:obj                0.293588235294118
 4     subj:obj:adv            0.0146470588235294
 5     subj:obj:ap             0.00176470588235294
 6     subj:obj:as             0.00155882352941176
 7     subj:obj:obj            0.395
 8     subj:obj:obj:adv        0.00464705882352941
 9     subj:obj:obj:pp.at      0.00173529411764706
10     subj:obj:obj:pp.for     0.00423529411764706
11     subj:obj:obj:pp.in      0.007
12     subj:obj:obj:pp.on      0.002
13     subj:obj:obj:pp.to      0.00705882352941176
14     subj:obj:obj:pp.with    0.00114705882352941
15     subj:obj:pp.about       0.00167647058823529
16     subj:obj:pp.after       0.00123529411764706
17     subj:obj:pp.against     0.000411764705882353
18     subj:obj:pp.as          0.00502941176470588
```

19	subj:obj:pp.at	0.00647058823529412
20	subj:obj:pp.before	0.000705882352941177
21	subj:obj:pp.between	0.000147058823529412
22	subj:obj:pp.by	0.00117647058823529
23	subj:obj:pp.during	0.000882352941176471
24	subj:obj:pp.for	0.0166470588235294
25	subj:obj:pp.from	0.00164705882352941
26	subj:obj:pp.in	0.0275294117647059
27	subj:obj:pp.in:adv	0.000470588235294118
28	subj:obj:pp.in:pp.in	0.000323529411764706
29	subj:obj:pp.into	0.0005
30	subj:obj:pp.like	0.000235294117647059
31	subj:obj:pp.of	0.00582352941176471
32	subj:obj:pp.on	0.00688235294117647
33	subj:obj:pp.out_of	0.000470588235294118
34	subj:obj:pp.over	0.00102941176470588
35	subj:obj:pp.through	0.000441176470588235
36	subj:obj:pp.to	0.109852941176471
37	subj:obj:pp.under	0.000764705882352941
38	subj:obj:pp.with	0.00302941176470588
39	subj:obj:pp.within	0.000441176470588235
40	subj:obj:pp.without	0.00105882352941176
41	subj:obj:ppart	0.00288235294117647
42	subj:obj:s	0.00102941176470588
43	subj:obj:sub	0.000470588235294118
44	subj:obj:that	0.00197058823529412
45	subj:obj:to	0.00814705882352941
46	subj:obj:vbase	0.000441176470588235
47	subj:obj:vger	0.00102941176470588
48	subj:pp.about	0.000117647058823529
49	subj:pp.across	8.82352941176471e-05
50	subj:pp.after	0.000147058823529412
51	subj:pp.against	2.94117647058824e-05
52	subj:pp.as	0.000294117647058824
53	subj:pp.at	0.0005
54	subj:pp.at:adv	8.82352941176471e-05
55	subj:pp.between	2.94117647058824e-05
56	subj:pp.by	5.88235294117647e-05
57	subj:pp.for	0.001
58	subj:pp.for:adv	0.000176470588235294
59	subj:pp.from	0.000147058823529412
60	subj:pp.from:pp.to	2.94117647058824e-05
61	subj:pp.in	0.00147058823529412
62	subj:pp.in:adv	0.000117647058823529
63	subj:pp.into	0.000264705882352941
64	subj:pp.like	8.82352941176471e-05
65	subj:pp.of	0.000911764705882353
66	subj:pp.on	0.000411764705882353
67	subj:pp.on:adv	0

```
68      subj:pp.out_of          0.000176470588235294
69      subj:pp.over            2.94117647058824e-05
70      subj:pp.through         5.88235294117647e-05
71      subj:pp.to              0.00847058823529412
72      subj:pp.to:adv          0.0005
73      subj:pp.towards         5.88235294117647e-05
74      subj:pp.under           8.82352941176471e-05
75      subj:pp.up_to           0.000176470588235294
76      subj:pp.upon            8.82352941176471e-05
77      subj:pp.with            0.000411764705882353
78      subj:pp.with:adv        2.94117647058824e-05
79      subj:ppart              0.000176470588235294
80      subj:s                  0.00823529411764706
81      subj:sub                2.94117647058824e-05
82      subj:that               0.000529411764705882
83      subj:to                 0.00111764705882353
84      subj:to:adv             0
85      subj:vbase              0.00105882352941176
86      subj:vbase:adv          2.94117647058824e-05
87      subj:vger               0.000441176470588235
```

and as a more illustrative figure, where the peaks of MLE for the most probable frames are recognisable:



Distribution of the verb *give* – version `A`

The figure illustrates that only three subcategorisation frames are assigned a MLE greater than 0.05, several lie between 0 and 0.05, but most are zero.

– Version `B`: *Distribution over the subcategorisation frames and the selectional preferences*

Preparing the data for this version was more complicated, since the data is more complex and the amount of data enormous. In

version `B` the verbs were identified by a distribution over the subcategorisation frames including information about their selectional preferences, i.e. each attribute in the distribution characterising the verb was represented by a frame type combined with a tuple of WordNet nodes. For example, one attribute was defined as `subj:obj:pp.to::LifeForm:PhysObject:LifeForm`, meaning that the frame `subj:obj:pp.to` was combined with a living entity as subject and head of the prepositional phrase, and an inanimate entity as object. Since considering all possible combinations of subcategorisation frames with conceptual classes would have resulted in 2,321,528 attributes within the distributions, I restricted the combinations to those where the subcategorisation frame was followed by a class-combination which appeared at least once as favoured possibility for some verb-frame type. This left 2,192 attributes. The value of each attribute was determined in several steps:
First, the maximum likelihood estimate for each class c in a certain argument slot s was determined (for example, the MLE for the class `LifeForm` as subject in `subj:obj`) by relating the association of the class in the argument position to the overall association of all classes in the argument position:

$$p(s,c) = \frac{ass(s,c)}{\sum_{c' \in class} ass(s,c')} \tag{2.8}$$

Combining the classes c_{1-i} to a class combination cc for the different arguments positions s_{1-i} in a subcategorisation frame sf (compare the above example) and estimating its probability demanded to relate the specific combination to all possible combinations considering the argument frame with i arguments:

$$p(cc|sf) = p(s_{1-i}, c_{1-i}) = \frac{\prod_i p(s_i, c_i)}{\sum_{c' \in class} \prod_i p(s_i, c'_i)} \tag{2.9}$$

As in version `A`, the maximum likelihood estimate of the verb appearing with the specific subcategorisation frame was determined by:

$$p(sf|v) = \frac{f(v,sf)}{f(v)} \tag{2.10}$$

Finally, the maximum likelihood estimate of the verb appearing with the specific subcategorisation frame and the specific semantic classes was calculated by:

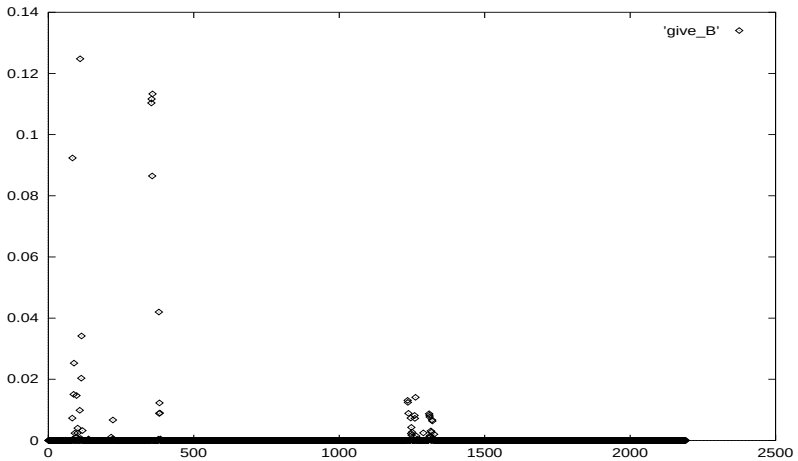
$$p(sf, cc|v) = p(sf|v) * p(cc|sf) \tag{2.11}$$

Compared to the distributions in version `A` it is striking how many zeroes appear. To give an example for a distribution, I list those frames for the verb *give* which are unequal to zero:

subj:obj	Cell:Shape	1.46267108747213e-06
subj:obj	Cell:Action	8.26056676521126e-05
subj:obj	PhysObject:Psycho	0.00728704347146904
subj:obj	PhysObject:Abstract	0.092361147350823
subj:obj	PhysObject:Shape	0.000447752021234279
subj:obj	PhysObject:Event	0.0150604380105458
subj:obj	PhysObject:Action	0.0252871988606566
subj:obj	PhysObject:Possession	0.000130070547541773
subj:obj	PhysObject:Phenomenon	0.00238672117007707
subj:obj	Psycho:Psycho	0.00115802136489394
subj:obj	Psycho:Abstract	0.0146775825253597
subj:obj	Psycho:Event	0.00239333126653585
subj:obj	Psycho:Action	0.00401851816221684
subj:obj	Psycho:Phenomenon	0.000379286073675188
subj:obj	Abstract:Psycho	0.00984563248006969
subj:obj	Abstract:Abstract	0.124790515634243
subj:obj	Abstract:Shape	0.000604964394756431
subj:obj	Abstract:Event	0.0203483810987631
subj:obj	Abstract:Action	0.0341659093166177
subj:obj	Abstract:Possession	0.00017574024535352
subj:obj	Abstract:Phenomenon	0.00322473436105953
subj:obj	Shape:Psycho	0.000512526056705487
subj:obj	Shape:Shape	3.14921378915355e-05
subj:obj	Whole:Psycho	2.76335881565075e-05
subj:obj	Content:Abstract	0.00108036574342879
subj:obj	Part:Psycho	0.000527490765743986
subj:obj	Part:Abstract	0.00668579136817707
subj:obj	Part:Possession	9.41548009040735e-06
subj:obj:obj	Psycho:LifeForm:State	0.110343168995819
subj:obj:obj	Psycho:LifeForm:Action	0.111643237131288
subj:obj:obj	Psycho:Agent:Psycho	0.0864833724399968
subj:obj:obj	Psycho:Agent:Action	0.113304697565445
subj:obj:obj	Event:Group:Abstract	0.000410709578929337
subj:obj:obj	Action:LifeForm:Psycho	0.0419864668957679
subj:obj:obj	Action:LifeForm:Possession	0.00882846029364335
subj:obj:obj	Action:Agent:Abstract	0.0122577597177466
subj:obj:obj	Action:Agent:Possession	0.00895984431518658
subj:obj:obj	Action:Group:Abstract	0.00049624777682631
subj:obj:pp.to	PhysObject:Abstract:LifeForm	0.0131031236578255
subj:obj:pp.to	PhysObject:Abstract:Agent	0.0124338246982715
subj:obj:pp.to	PhysObject:Action:Psycho	0.00884683496869106
subj:obj:pp.to	Psycho:Psycho:LifeForm	0.00731598809462888
subj:obj:pp.to	Psycho:Abstract:Action	0.00222156690339364
subj:obj:pp.to	Psycho:State:Agent	0.00427865900280351
subj:obj:pp.to	Psycho:Action:Action	0.00170657529725833

subj:obj:pp.to	Psycho:Possession:Psycho	0.00269265381077406
subj:obj:pp.to	Abstract:Psycho:LifeForm	0.00819245367344916
subj:obj:pp.to	Abstract:Psycho:Psycho	0.0072004729354521
subj:obj:pp.to	Abstract:Abstract:LifeForm	0.0140976488533991
subj:obj:pp.to	Abstract:Action:Group	0.00142450520802067
subj:obj:pp.to	Abstract:Possession:Group	0.000451258910933365
subj:obj:pp.to	Abstract:Phenomenon:Phenomenon	5.9549800989053e-05
subj:obj:pp.to	Location:Abstract:Group	0.000217998189969337
subj:obj:pp.to	Location:Location:LifeForm	3.22202880364481e-05
subj:obj:pp.to	Action:Psycho:Psycho	0.00247515013191056
subj:obj:pp.to	Action:Possession:Action	0.000208098396028427
subj:obj:pp.to	Group:Psycho:Action	0.000895101111831961
subj:obj:pp.to	Group:Abstract:LifeForm	0.00872873038787378
subj:obj:pp.to	Group:Abstract:Agent	0.00828287256653339
subj:obj:pp.to	Group:Abstract:Psycho	0.00767181474854556
subj:obj:pp.to	Group:Abstract:Action	0.00154029814093307
subj:obj:pp.to	Group:Abstract:Group	0.00114815989652989
subj:obj:pp.to	Group:Abstract:Essential	0.000133972079965587
subj:obj:pp.to	Group:State:LifeForm	0.0031262461497298
subj:obj:pp.to	Group:State:Psycho	0.00274770559443561
subj:obj:pp.to	Group:Action:LifeForm	0.00670528338967338
subj:obj:pp.to	Group:Action:Agent	0.00636278191342858
subj:obj:pp.to	Group:Action:Group	0.000881999688475466
subj:obj:pp.to	Group:Action:Essential	0.000102915398065361
subj:obj:pp.to	Group:Possession:Agent	0.00201562059625592
subj:obj:pp.to	Group:Possession:Group	0.000279402431541851
subj:obj:pp.to	Part:Possession:Group	2.77653887150829e-06

The same values can be found in the following figure, giving an overview of all 2,192 dimensions of the distribution:



Distribution of the verb *give* – version B

The figure illustrates that most attributes are assigned zero values;

the utilised frames concentrate on certain frame areas, i.e. the verb goes with a limited choice of subcategorisation frames (represented by the areas in the figure) equipped with various preferences.

The introduced distributions (in two versions, considering the different amount of information) describe the verbs concerning their use of subcategorisation frames and their preferences for the arguments and thereby form the verbs' relevant properties to cluster them.

- *How is the distance between two clusters measured?*

  Provided with the representation of the clusters' properties, we could then move to the next step, the comparison of the clusters, more concrete: the measure of difference/distance between the clusters. I used three measures to calculate and compare the distances: the information-theoretic measure *Relative Entropy*, and the two geometric measures *Euclidean Distance* and *Cosine*. To apply the geometric measures, the attributes within the distributions of the verbs were considered to be elements of a vector.
  I give a brief overview of the definition of the measures: relative entropy compares two distributions p and q concerning their i attributes by:

$$D_{Rel.Entr.} = \sum_i p_i * log(\frac{p_i}{q_i}) \tag{2.12}$$

  A general mathematical difficulty concerning relative entropy is the impossibility to apply the measure in case the distributions contain zeroes. That means that the estimates within the distributions had to be smoothed, which was realised by adding 0.5 to the frequencies of each verb-frame type.

  Euclidean distance measures the distance between the two points the vectors representing the respective distributions in i-dimensional space point at:

$$D_{Eucl.Dist.} = \sqrt{\sum_i (p_i - q_i)^2} \tag{2.13}$$

  Cosine measures the angle between the vectors representing the respective distributions:

$$D_{Cos} = \frac{\sum_i p_i * q_i}{\sqrt{\sum_i p_i^2} * \sqrt{\sum_i q_i^2}} \tag{2.14}$$

  Intuitively, relative entropy should be the suitable measure, since – differently to the geometric measures – it takes relative instead of absolute

differences of the attributes (i.e. the frame types) within the distributions into account. But test runs did not show significant differences, so I decided to apply all three measures and compare the results.
A more exhaustive comparison of the different measures can be found in Lillian Lee's dissertation [Lee, 1997].

- *How are the clusters merged?*

  When merging two clusters – because these clusters represented the two clusters closest to each other – two steps had to follow: first, the verbs of the two clusters were united in one common cluster, and secondly, the distributions of the clusters had to be merged. The first step was obvious to carry out, but the second caused difficulties: how are the distributions merged? This was realised by calculating the weighted average for each attribute in the distribution: assume sf_a^i to be the strength of a certain subcategorisation frame sf^i within the distribution over all frames, for a cluster a with m_a the number of verbs in that cluster, and sf_b^i and m_b the respective numbers for the other cluster b. Merging the distributions of the two clusters for that specific frame resulted in

  $$sf_{ab}^i = \frac{sf_a^i * m_a + sf_b^i * m_b}{m_a + m_b} \tag{2.15}$$

  By this, in addition to the values within the former distributions of the clusters the number of verbs within the clusters were taken into consideration. Geometrically viewed, the merged distribution for a certain number of verbs within one cluster can be considered as the centroid of that cluster of verbs.
  Once a cluster was assigned a new (merged) distribution, the matrix could be updated.

- *How many iterations of the algorithm are necessary to cluster the verbs?*

  The number of iterations determines the number of clusters resulting from the application of the algorithm, since each iteration decreases the number of clusters by one. So one possibility to infer the number of iterations was to specify the desired number of resulting clusters. This solution, however, turned out to behave in an insufficient way, since the verbs showed the tendency to cluster together in a few large clusters and leave a large number of verbs single. I decided to limit the maximum number of verbs within one cluster to four elements, which influenced the clustering algorithm in the following way: having clustered all verbs within a certain number of iterations, the resulting clusters were checked for their number of elements. Each time a cluster

contained more elements than the limit, the algorithm was run again on the verbs of that cluster, so a large cluster was split into several limited clusters. The repetitive method always started with the original distributions of the verbs, so the order and kind of verbs clustering together was not influenced.
Still, the number of iterations was an undetermined parameter. Test runs led to the following constraints which based the number of iterations (I) on the number of verbs to be clustered (V):

- If more than 100 verbs were to be clustered, the number of iterations was calculated by $I = V * 0.95$.
- Else: If more than 50 verbs were to be clustered, the number of iterations was calculated by $I = V * 0.9$.
- Else: If more than 20 verbs were to be clustered, the number of iterations was calculated by $I = V * 0.8$.
- Up to 20 verbs were clustered by $I = V * 0.7$.

The decreasing percentage of the number of verbs used to iterate was based on the observation that the number of iterations should be close to that of the verbs – to result in an expressive number of clusters – but not approach it, because that would have resulted in one large cluster plus several clusters containing only a single verb.

The outlined algorithm might seem arbitrary in certain decisions I made. To convince the reader about the practical use I illustrate a sample run I executed, based on the preceding data.
I defined three clearly distinguishable semantic classes, *reception, amusement*, and *motion* and assigned 13 verbs to these classes:

- *reception:* buy, collect, purchase, receive
- *amusement:* giggle, grin, laugh, smile
- *motion:* fly, move, run, swim, walk

Applying the algorithm in the way explained before, the clustering of the verbs resulted in exactly the way they had been assigned to classes, with exception of the verb *run*, which represented a cluster with a single member, because of the restriction of at most four verbs per cluster.
Compared to the amount of verbs I extracted from the BNC this was a simple and clear example. But it illustrates that the general idea of the clustering algorithm is applicable.

There are obvious disadvantages of the algorithm, however: the clearest is the fact that, before applying the algorithm to the data, several parameters had to be determined: how to define the distributions for the verbs, which measure to use for calculating the difference between the verbs, how to smooth, how to merge the distributions, how often the algorithm should be run, the cut-off for the number of verbs in a cluster. Test runs led to the described set which resulted in useful sample clusters. But how do we know if the parameters were set in the optimal way?

A way out of this problem was the application of an unsupervised instead of a supervised algorithm. The advantage of an unsupervised algorithm is the possibility to feed the data into the algorithm and initialise few parameters to make it organise itself to find the (local) optimum.
Consider *Kohonen Networks*, for example (a simple description of the algorithm can be found in [Beale and Jackson, 1990]). The unsupervised learning algorithm organises the nodes in a network into local neighbourhoods. Considering the nodes as the verbs I wanted to cluster, this is exactly what I needed, and the only requirement was the representation of the nodes.
Practical considerations, then, led to another unsupervised algorithm: the TCL group at the IMS provides a robust tool for *Latent Classes* (see [Rooth, 1996] for a description of the algorithm), based on the expectation-maximation algorithm. Generally said, latent class analyses identify categorical types among indirectly observed multinomial distributions, a background applicable to our problem of assigning semantic classes to verbs characterised by distributions over subcategorisation frames.
As input, the algorithm needs (i) a fixed number of classes to be built, and (ii) the absolute frequencies of the verbs appearing with the subcategorisation frames. For version `A`, this data was already provided by the results in section 2.1.2. The representation looked as follows for our usual example verb *give*. The frequency is followed by the verb which is itself followed by the frame:

```
13430     give     subj:obj:obj
 9982     give     subj:obj
 3735     give     subj:obj:pp.to
  936     give     subj:obj:pp.in
  758     give     subj
  566     give     subj:obj:pp.for
  498     give     subj:obj:adv
  288     give     subj:pp.to
  280     give     subj:s
  277     give     subj:obj:to
  242     give     subj:obj:obj:obj
  240     give     subj:obj:obj:pp.to
```

```
  238     give     subj:obj:obj:pp.in
  234     give     subj:obj:pp.on
  220     give     subj:obj:pp.at
  198     give     subj:obj:pp.of
  171     give     subj:obj:pp.as
  158     give     subj:obj:obj:adv
  144     give     subj:obj:obj:pp.for
  105     give     subj:adv
  103     give     subj:obj:pp.with
   98     give     subj:obj:ppart
   75     give     subj:obj:pp.to:pp.in
   68     give     subj:obj:obj:pp.on
   67     give     subj:obj:that
   60     give     subj:obj:ap
   59     give     subj:obj:obj:pp.at
   58     give     subj:ap
   57     give     subj:obj:pp.about
   56     give     subj:obj:pp.from
   55     give     subj:obj:pp.than
   53     give     subj:obj:as
   52     give     subj:obj:obj:pp.as
   50     give     subj:pp.in
   50     give     subj:obj:obj:pp.of
[...]
```

For version `B` the definition was slightly more complicated, since I was equipped with association values, not frequencies. The first row in the following list therefore contains the association value as calculated in the way described for the distance clustering algorithm and then multiplied by 10^6 to represent an integer:

```
339379     give     subj:obj::Abstract:Abstract
251184     give     subj:obj::PhysObject:Abstract
229030     give     subj:obj:obj::Psycho:Agent:Action
225672     give     subj:obj:obj::Psycho:LifeForm:Action
223044     give     subj:obj:obj::Psycho:LifeForm:State
174814     give     subj:obj:obj::Psycho:Agent:Psycho
102465     give     subj:obj:pp.to::Abstract:Abstract:LifeForm
 95237     give     subj:obj:pp.to::PhysObject:Abstract:LifeForm
 92917     give     subj:obj::Abstract:Action
 90372     give     subj:obj:pp.to::PhysObject:Abstract:Agent
 84870     give     subj:obj:obj::Action:LifeForm:Psycho
 68770     give     subj:obj::PhysObject:Action
 64301     give     subj:obj:pp.to::PhysObject:Action:Psycho
 63442     give     subj:obj:pp.to::Group:Abstract:LifeForm
 60202     give     subj:obj:pp.to::Group:Abstract:Agent
 59544     give     subj:obj:pp.to::Abstract:Psycho:LifeForm
 55760     give     subj:obj:pp.to::Group:Abstract:Psycho
 55339     give     subj:obj::Abstract:Event
```

```
 53174     give     subj:obj:pp.to::Psycho:Psycho:LifeForm
 52335     give     subj:obj:pp.to::Abstract:Psycho:Psycho
 48735     give     subj:obj:pp.to::Group:Action:LifeForm
 46246     give     subj:obj:pp.to::Group:Action:Agent
 40958     give     subj:obj::PhysObject:Event
 39917     give     subj:obj::Psycho:Abstract
 31098     give     subj:obj:pp.to::Psycho:State:Agent
 26776     give     subj:obj::Abstract:Psycho
 24777     give     subj:obj:obj::Action:Agent:Abstract
 22722     give     subj:obj:pp.to::Group:State:LifeForm
 19971     give     subj:obj:pp.to::Group:State:Psycho
 19817     give     subj:obj::PhysObject:Psycho
 19570     give     subj:obj:pp.to::Psycho:Possession:Psycho
 18182     give     subj:obj::Part:Abstract
 18111     give     subj:obj:obj::Action:Agent:Possession
 17990     give     subj:obj:pp.to::Action:Psycho:Psycho
 17845     give     subj:obj:obj::Action:LifeForm:Possession
 16146     give     subj:obj:pp.to::Psycho:Abstract:Action
 14650     give     subj:obj:pp.to::Group:Possession:Agent
 12403     give     subj:obj:pp.to::Psycho:Action:Action
 11195     give     subj:obj:pp.to::Group:Abstract:Action
 10928     give     subj:obj::Psycho:Action
 10353     give     subj:obj:pp.to::Abstract:Action:Group
[...]
```

Fed with this data the latent class analysis organised the verbs into classes. The output of the algorithm was a set of classes, with each class being assigned a certain probability. The classes contain all verb types and all subcategorisation frame types, ordered by the probability of their class membership. So for each class there are a couple of most probable verb-frame types. A threshold for the probability and/or a cut-off for the number of members within the classes limited the number of verbs within the classes: I set the threshold to 0.02 and the number of members to four, as for the preceding algorithm.
To give an example, a class could look as follows:

```
Class 3          0.0415648
==================================
think            0.283845
know             0.111733
say              0.0914999
believe          0.0454137

<><><><><><><><><><><><><><><><><><><><>

subj:s           0.66809
subj             0.230383
```

Since the verbs can be (probable) members of several classes, the approach is generally able to represent the polysemy of verbs.

Here I finish with the description of the two algorithms I used for the experiments. Subsection 2.3.3 will specify how they were set into play.

2.3.2 Verbs and Verb Classes

The final step before setting up the experiments was the definition of the verbs to cluster. I was provided with 12,238 different verb-particle types I could cluster, but I wanted to start with a neat amount of data. The main reason for that was the problem of finding a basis for evaluating the resulting verb classes. How is it possible to evaluate the results? Basically, there are two ways: (i) define an own classification system, optimally before running the experiments, otherwise based on the results, or (ii) use an existing classification system. The former possibility was rejected, since this would either result in an insufficiently defined classification system, or go beyond the scope of this thesis. So I decided in favour of the latter possibility and therefore had to make up my mind about which system to utilise.
As introduced in section 1.2, Levin's classification [Levin, 1993] already provides a semantic categorisation of verbs, so I decided to extract verbs and verb classes from there. The constraints I required for the verbs were (i) some verbs to be polysemous to investigate the realisation of the phenomenon by the algorithms, and (ii) to distinguish between high and low frequent verbs to see the influence of the frequency onto the algorithms.

I selected 153 different verbs with 226 verb senses: 103 verbs only have a single sense, 35 verbs have two senses, 9 verbs have three senses, and 6 verbs have four senses, according to Levin. Considering the (low) frequencies of the verbs, the data I had collected showed that 27 of the verbs appeared less than 500 times as heads of subcategorisation frames, and 4 even less than 100 times in total.
The 226 verb senses belong to 30 different semantic classes; I partly renamed the classes and I split four of them into sub-classes, to distinguish between the different parts of the classes. Here is the complete definition of the verb classes:

1. *Placing*
   arrange, place, position, put, situate

2. *Surfacing*
    - *Rubbing*
      brush, rub
    - *Loading*
      load, pack
    - *Spreading*
      spray, spread

3. *Change of Possession*
    - *Giving*
      allocate, entrust, give, guarantee, leave, offer, pass, pay, promise, provide, return, sell, supply, transfer
    - *Obtaining*
      accumulate, acquire, buy, collect, find, gain, get, leave, purchase, receive

4. *Sending*
   pass, return, send, transfer, transport

5. *Throwing*
   hit, kick, pass, smash, throw

6. *Contact*
    - *Impact*
      beat, brush, hit, kick, smash
    - *Touch*
      kiss, tickle, touch

7. *Removing*
   brush, delete, dismiss, eliminate, extract, remove, separate

8. *Disassemble*
    - *Separating*
      disconnect, distinguish, extract, part, separate
    - *Splitting*
      break, cut, kick, split, tear

9. *Destruction*
   break, crush, demolish, destroy, eliminate, execute, kill, murder, ruin, smash, split, tear, waste

10. *Change of State*
    break, cook, collect, climb, crush, gain, smash, split, tear

11. *Creation*
    arrange, build, collect, construct, cook, create, cut, develop, invent, pour, produce, roll

12. *Declaration*
    announce, believe, confess, declare, find, guarantee, show, suppose, think, want

13. *Telling*
    advise, announce, confess, declare, explain, instruct, propose, read, say, show, suggest, teach, tell, warn, write

14. *Learning*
    acquire, learn, read, study

15. *Characterisation*
    characterise, classify, describe, identify, offer, qualify, see

16. *Assessment*
    analyse, assess, evaluate, study

17. *Perception*
    feel, hear, notice, see, smell, study

18. *Admiration*
    admire, envy, hate, like, love

19. *Desire*
    desire, like, need, want

20. *Social Interaction*
    argue, communicate, correspond, fight, kiss, meet, play, visit

21. *Manner of Speaking*
    moan, scream, shout, whisper

22. *Ingesting*
    eat, drink, exist, live, survive

23. *Body*
    brush, cut, kick, part, roll, show

24. *Lodging*
    live, stay, stop

25. *Existence*
    climb, cut, exist, hit, live, meet, persist, run, stay, survive, touch

26. *Sliding*
    bounce, float, move, roll, slide

27. *Motion*
    climb, depart, exit, flee, leave, return

28. *Manner of Motion*
    bounce, climb, float, fly, jump, move, roll, run, slide, tear

29. *Aspect*
    begin, continue, end, finish, start, stop

30. *Weather*
    pour, rain, snow, storm

I added three verb senses to certain classes which I thought belonging there according to their alternation behaviour as well as their meaning: *extract* to *Separating*, *announce* to *Declaration*, and *like* to *Desire*.

There are some final remarks to mention before I conclude the description of the classes:

- As mentioned above, one should keep in mind that the definition of the Levin classes and their members is based on subjective judgement when deciding in the last instance. Comparing the classes with other conceptual categorisations (a dictionary, or WordNet, for example) would necessarily result in differences.

- In my opinion there is a difference in how closely related verbs in the same class are. Some verbs in the same class are more closely related to each other than others. That was the reason why I split some classes into sub-classes.
  In addition, the algorithms were defined in a way not allowing more than four members per class, so a successful application should be able to distinguish even more sub-classes within the classes than defined, and no more subjective opinions (mine, in this case) influenced the classes.

2.3.3 Experiments

Finally, I come to describe the clustering experiments I ran on the defined data with the specified algorithms. Summarising the above definitions, I had 153 different verbs with 226 verb senses to cluster into 30 semantic classes, according to information about the subcategorisation frames they occur with, partly accompanied by selectional preferences for the arguments within the frames. The clustering was performed by two different algorithms, one clustering according to the distances between the verbs, one clustering according to a latent class analysis.

On this basis, four different experiments were carried out:

1. Distance Clustering according to subcategorisation frames only
2. Distance Clustering according to subcategorisation frames and their selectional preferences
3. Latent Class Clustering according to subcategorisation frames only
4. Latent Class Clustering according to subcategorisation frames and their selectional preferences

To investigate the background of the distance clustering in a concrete way, I added a further experiment for both informational versions: based on the distributions of the verbs over the frames as explained in subsection 2.3.1, I clustered the verbs in one step by assigning each verb into the same cluster as that verb most similar in the distribution. The similarity was measured by relative entropy, euclidean distance and cosine, as before.
In this way, clusters containing at least two and at most 153 verbs were created. To illustrate the idea, assume the five verbs *buy, purchase, fly, move* and *swim* pointing to the respective most similar verb in the following way:

```
buy       ->  purchase
purchase  ->  buy
fly       ->  swim
move      ->  fly
swim      ->  fly
```

Based on these distance formulations the two clusters *{buy, purchase}* and *{fly, move, swim}* would be created.
This experiment was also carried out for both informational versions, so the following two experiments were added:

1. One-Step Distance Clustering according to subcategorisation frames only
2. One-Step Distance Clustering according to subcategorisation frames and their selectional preferences

To distinguish the two kinds of distance clustering, the latter method gets the affix *one-step*, the former *iterative*, from now on.

For comparing the results of the experiments and determining their usefulness, I preceded a baseline experiment, where each verb points to an arbitrary other verb as most similar verb, in order to create the clusters in the same way as for one-step distance clustering.

I finish with an overview of the experiments as introduced above, according to the algorithm used for clustering and the amount of information the clustering was based on:

Algorithm	**Information**
Baseline	-
One-Step Distance Clustering	SFs
One-Step Distance Clustering	SFs and Prefs
Iterative Distance Clustering	SFs
Iterative Distance Clustering	SFs and Prefs
Latent Class Clustering	SFs
Latent Class Clustering	SFs and Prefs

Chapter 3

Interpreting the Semantic Classification

This chapter is concerned with the verb classes resulting from the experiments described in section 2.3.3. It contains two parts: section 3.1 presents the results of the clustering process, especially the recall and precision measures, and section 3.2 then describes and interprets the classification in detail.

3.1 Quality of the Verb Classes

In order to represent the results in a clear way, I build tables according to (i) the algorithms used for clustering, and (ii) the amount of information on which the clustering was based. Before displaying the results of the more sophisticated algorithms I start with the results of the baseline experiment, followed by those from the one-step distance clustering. Then the results for the iterative distance clustering are listed, followed by those for the latent class clustering. Each method is described for both using only information about the subcategorisation frames and using information about the subcategorisation frames and their selectional preferences.

The core of information about the results is the same for all tables. The important pieces are:

- the total number of clusters: the number of clusters obtained by the respective clustering method
- the number of correct clusters: a correct cluster is defined a cluster representing a subset of members within a Levin class

- the total number of verbs in the clusters: the total number of verbs appearing in all clusters
- the number of correct verbs: the total number of verbs appearing in a correct cluster

In addition, the tables are equipped with the respective recall and precision measures. The recall value is defined by the percentage of verbs within the correct clusters compared to the total number of verbs to be clustered:

$$recall = \frac{|verbs_{correct\_clusters}|}{153} \tag{3.1}$$

The precision value is defined by the percentage of verbs appearing in the correct clusters compared to the number of verbs appearing in any cluster:

$$precision = \frac{|verbs_{correct\_clusters}|}{|verbs_{all\_clusters}|} \tag{3.2}$$

3.1.1 Baseline Clustering

The baseline experiment where each verb was given an arbitrary closest neighbour resulted in one large cluster. Recall and precision are both zero in this case.

3.1.2 One-Step Distance Clustering

The first table shows the number of clusters and verbs resulting from one-step distance clustering done by the three measures *Relative Entropy*, *Euclidean Distance* and *Cosine*, according to the information about subcategorisation frames only. The total number of verbs is always 153, since all verbs appear in the clusters. This fact causes recall and precision to have the same value. The most successful measure for clustering was relative entropy which assigned 24% of the verbs to correct clusters.

Measure	Clusters		Verbs		Recall	Precision
	Total	Correct	Total	Correct		
Rel. Entr.	32	14	153	36	24%	24%
Eucl. Dist.	35	10	153	23	15%	15%
Cosine	29	8	153	18	12%	12%

One-Step Distance Clusters according to SFs

The next table is based on the same method, but includes information about the selectional preferences within the subcategorisation frames. With this basis, recall and precision strongly decrease; relative entropy assigned only 3% instead of 24% of the verbs to correct clusters, euclidean distance also deteriorated, but turned out to be the most successful measure, assigning 11% of the verbs to correct clusters.

Measure	Clusters		Verbs		Recall	Precision
	Total	Correct	Total	Correct		
Rel. Entr.	23	2	153	5	3%	3%
Eucl. Dist.	19	7	153	17	11%	11%
Cosine	25	5	153	14	9%	9%

One-Step Distance Clusters according to SFs and Prefs

3.1.3 Iterative Distance Clustering

The following two tables display the number of clusters and verbs resulting from iterative distance clustering. The first is based on the information about subcategorisation frames only. Compared to the respective experiment when clustering in one step, this method increased the number of verbs it assigned to correct clusters from 36 to 55 when using relative entropy as distance measure; precision increases to 61%.

Measure	Clusters		Verbs		Recall	Precision
	Total	Correct	Total	Correct		
Rel. Entr.	31	20	90	55	36%	61%
Eucl. Dist.	41	17	111	37	24%	33%
Cosine	38	18	98	43	28%	44%

Iterative Distance Clusters according to SFs

The second is based on the same method, but considers information about subcategorisation frames plus information about their selectional preferences. Again, using this more refined basis decreases recall and precision. Euclidean distance was most successful in measuring and resulted in clustering 44% of all verbs into correct clusters.

Measure	Clusters		Verbs		Recall	Precision
	Total	Correct	Total	Correct		
Rel. Entr.	30	14	81	31	20%	38%
Eucl. Dist.	15	8	45	20	13%	44%
Cosine	29	7	88	18	12%	20%

Iterative Distance Clusters according to SFs and Prefs

3.1.4 Latent Class Clustering

Finally, the numbers of clusters and verbs resulting from using latent classes as verb clusters are given, utilising information about subcategorisation frames only as well as adding selectional preferences for their arguments.
I briefly recall the differences to the preceding method, as far as the display in the table is concerned: (a) the number of clusters had to be determined before starting the clustering machinery, so it was not an effect of the algorithm's behaviour – as it was for distance clustering – and (b) the 153 different verbs have 226 different senses, represented in brackets in addition to the number of different verbs. Recall and precision are calculated on the basis of verb senses in addition to those for the verbs.
The precision of this method is below the most successful assignments when iteratively clustering by distance: 54% instead of 61%, and 31% instead of 44%.

Information	Clusters		Verbs(Senses)		Recall	Precision
	Total	Correct	Total	Correct		
SFs	80	36	107(159)	58(90)	38(40)%	54(57)%
SFs + Prefs	80	22	153(226)	47(56)	31(25)%	31(25)%

Latent Classes

3.1.5 Comparison

I briefly summarise the results from the different tables. Concerning precision, the assignment of verbs into semantic classes was most successful when using the iterative distance clustering method; 61% of all verbs were clustered into correct classes. Clustering the verbs into latent classes was with 54% comparably, but less successful.

The baseline experiment showed that no semantic clustering is possible without an algorithmic procedure, and the one-step distance clustering underlined the impression that sophisticated methods are needed to successfully cluster verbs.

With all clustering methods the results became worse when adding information about the selectional preferences for the arguments in the subcategorisation frames.

3.2 Interpretation of the Verb Classes

I have given an overview of the success of the different methods concerning the quality of clustering. But so far nothing has been said about how the clusters look like and why they look that way. For the goal of answering these questions I will describe, compare and interpret the clusters resulting from *Baseline Clustering* and the three clustering methods *One-Step Distance Clustering*, *Iterative Distance Clustering* and *Latent Class Clustering*. Concerning the distance clustering I concentrate on the measure of relative entropy.

3.2.1 Baseline Clustering

The baseline experiment shows that randomly clustering verbs does not result in any groups belonging together. Uniting each verb with an arbitrary most similar verb formed one large cluster with all verbs. With this experiment I illustrate that it is necessary to base the clustering of verbs on information making possible to find tendencies of clusters.

3.2.2 One-Step Distance Clustering

The results of this clustering method provide two sources of information about the distance clustering behaviour of the verbs: (i) the resulting clusters of this method, of course, and (ii) the basis for this clustering method, i.e. the identification of the closest and therefore most similar verb. I will first describe the correct clusters and then interpret their success and their weaknesses on account of the underlying linguistic model and the distance measure.

I start with a description of the correct clusters. Following you find the 14 clusters built on the basis of subcategorisation frames only (version `A`), each identified by `C(X)` – where `X` is the number of members in the cluster – and the respective class name as specified in section 2.3.2:

```
C(2) -- Placing :   place
                    situate

C(3) -- Surfacing :   load
                      pack
                      spray

C(2) -- Surfacing:Rubbing :   brush
                              rub

C(5) -- Change of Possession:Giving :   give
                                        offer
                                        guarantee
                                        pay
                                        sell

C(2) -- Declaration :   suppose
                        think

C(2) -- Telling :   advise
                    instruct

C(3) -- Telling :   confess
                    explain
                    write

C(2) -- Telling :   teach
                    tell

C(2) -- Admiration :   hate
                       love

C(4) -- Desire :   desire
                   like
                   need
                   want

C(2) -- Lodging / Existence :   live
                                stay

C(2) -- Existence :   exist
                      persist
```

```
C(2) -- Sliding / Manner of Motion :   bounce
                                       float

C(3) -- Aspect :   begin
                   continue
                   start
```

The same information is provided for the two correct clusters on the basis of subcategorisation frames and their selectional preferences (version B):

```
C(3) -- Telling :   advise
                    instruct
                    teach

C(2) -- Sliding / Manner of Motion :   roll
                                       slide
```

The description of the correct clusters only provides information about how the verbs clustered together (successfully). For an investigation about why the verbs clustered together in that way we are concerned with the underlying factors of the resulting clusters: (i) the way the verbs point to the respective most similar verb, and (ii) the clustering algorithm assigning the verbs to classes according to the pointers.
As a first step for investigating these two issues I present the pointers of the verbs which chose another verb of a class the verb belongs to as most similar verb. Considering the distribution over the subcategorisation frames only (informational version A), a remarkable percentage of verbs meets this conditions: 94 verbs (61%).
The verb pointers are accompanied by the distances between the verbs. The range of the distances shows that the closeness is not restricted by a certain threshold:

```
advise         ->   instruct       0.238852958099192
announce       ->   show           0.145183331411684
assess         ->   evaluate       0.115488424667221
begin          ->   continue       0.122438979008207
bounce         ->   float          0.207911911653362
brush          ->   rub            0.293856739744594
buy            ->   purchase       0.107678325089227
characterize   ->   identify       0.251898877231237
classify       ->   describe       0.32647933391065
climb          ->   run            0.161085018462758
collect        ->   receive        0.0930838718704027
confess        ->   explain        0.350904528766792
continue       ->   begin          0.122438979008207
create         ->   produce        0.062281119846764
declare        ->   show           0.287377119747368
```

depart	->	flee	0.303862995157347
describe	->	identify	0.321319755171558
desire	->	like	0.297218928250783
destroy	->	kill	0.0772081139176661
disconnect	->	extract	0.344415935546984
drink	->	eat	0.0709823041301391
eat	->	drink	0.0709823041301391
eliminate	->	destroy	0.160978731496061
entrust	->	transfer	0.641732143913148
envy	->	admire	0.244702341485668
evaluate	->	assess	0.115488424667221
exist	->	persist	0.240941972241492
explain	->	write	0.249442158678047
extract	->	separate	0.202433207625224
feel	->	notice	0.507429764923867
fight	->	play	0.284467694351444
find	->	show	0.294997762244098
float	->	bounce	0.207911911653362
fly	->	move	0.223868751079158
gain	->	acquire	0.100265273816536
give	->	offer	0.251940767125914
guarantee	->	offer	0.311146419996227
hate	->	love	0.0751317671516997
hear	->	notice	0.446658227203692
instruct	->	advise	0.238852958099192
invent	->	create	0.0997178331802089
jump	->	fly	0.240818182438295
kill	->	destroy	0.0772081139176661
kiss	->	touch	0.237324068838082
like	->	need	0.100615003520193
live	->	stay	0.277569635445438
load	->	pack	0.219579969265969
love	->	hate	0.0751317671516997
meet	->	play	0.101563004912523
moan	->	scream	0.219209058848646
move	->	fly	0.223868751079158
murder	->	kill	0.0946875501512546
need	->	like	0.100615003520193
offer	->	pay	0.23016781556807
pack	->	load	0.219579969265969
pay	->	sell	0.193750923225591
persist	->	exist	0.240941972241492
place	->	situate	0.347714173318143
play	->	meet	0.101563004912523
produce	->	create	0.062281119846764
purchase	->	receive	0.091472208760778
receive	->	purchase	0.091472208760778
return	->	flee	0.381486296844562
roll	->	climb	0.164845139679025

```
rub             ->   brush          0.293856739744594
ruin            ->   destroy        0.170729903674329
run             ->   climb          0.161085018462758
scream          ->   shout          0.14379024976656
see             ->   describe       0.391668825790375
sell            ->   pay            0.193750923225591
separate        ->   extract        0.202433207625224
shout           ->   scream         0.14379024976656
show            ->   announce       0.145183331411684
situate         ->   place          0.347714173318143
slide           ->   roll           0.361925305353889
smash           ->   break          0.290617681542671
split           ->   smash          0.462454177014069
spray           ->   load           0.317886328501611
start           ->   begin          0.227969177465212
stay            ->   live           0.277569635445438
stop            ->   finish         0.358016834934426
suppose         ->   think          0.389745446679773
survive         ->   eat            0.268766594586078
teach           ->   tell           0.366822326761819
tear            ->   cut            0.42485543708415
tell            ->   teach          0.366822326761819
think           ->   suppose        0.389745446679773
touch           ->   hit            0.181096658162722
transfer        ->   allocate       0.411858561344105
want            ->   need           0.156052537732552
warn            ->   suggest        0.332488889228512
waste           ->   destroy        0.275320906123404
whisper         ->   shout          0.195454739021813
write           ->   explain        0.249442158678047
```

Including information about the selectional preferences decreases the result to 55 verbs (36%):

```
admire          ->   envy           1.03689873105811
advise          ->   instruct       0.752304259439932
allocate        ->   offer          3.8281372496486
announce        ->   declare        2.15095017894666
assess          ->   evaluate       2.44672164605358
begin           ->   continue       0.406273360669858
believe         ->   think          0.999194493960703
brush           ->   touch          2.24051635347847
build           ->   invent         2.31796662630919
buy             ->   invent         2.28932351500408
climb           ->   cook           2.2617350639517
confess         ->   think          1.00445367918735
construct       ->   produce        2.1739038758978
continue        ->   begin          0.406273360669858
destroy         ->   demolish       2.76119094797651
```

```
drink          ->   eat            0.113938812685002
eat            ->   drink          0.113938812685002
envy           ->   admire         1.03689873105811
execute        ->   murder         2.14025445018575
exist          ->   survive        1.38292157560267
finish         ->   start          1.32579149894346
float          ->   bounce         1.373428162721
fly            ->   move           0.715873441817131
gain           ->   acquire        1.88591591077562
give           ->   offer          3.81174859761804
guarantee      ->   suggest        2.46119770645549
hate           ->   love           0.210547493965412
identify       ->   characterize   2.83856345292773
instruct       ->   advise         0.752304259439932
kill           ->   murder         1.64395338641121
love           ->   hate           0.210547493965412
move           ->   fly            0.715873441817131
notice         ->   feel           1.24064275601367
offer          ->   acquire        2.57196819696522
persist        ->   exist          1.89866951471403
place          ->   position       4.02370267798147
pour           ->   cook           2.13674380859853
produce        ->   construct      2.1739038758978
receive        ->   return         3.73964339166919
roll           ->   slide          1.9743774663702
scream         ->   shout          0.173030704772142
shout          ->   scream         0.173030704772142
show           ->   suggest        1.27762461113202
slide          ->   roll           1.9743774663702
start          ->   finish         1.32579149894346
stop           ->   finish         2.33408384272032
suggest        ->   show           1.27762461113202
suppose        ->   think          0.854520147950432
survive        ->   exist          1.38292157560267
teach          ->   advise         1.71554041129368
think          ->   suppose        0.854520147950432
touch          ->   brush          2.24051635347847
transfer       ->   allocate       4.0739942050971
want           ->   like           0.363226905493825
whisper        ->   moan           0.627836598794285
```

To get a feeling for the distances between the verbs the reader might have a look at appendix C. I selected a choice of verbs and present them in a matrix showing the distances to all of the 153 verbs, for both versions `A` and `B`.

These pointers are the basic data on which the one-step distance clustering as well as the iterative method work. Provided with the (correct) pointers of the verbs and the resulting (correct) clusters, we can now investigate the underlying factors of this simple clustering method.

1. *Why do the verbs choose the respective most similar verb in the way the pointers represent?*

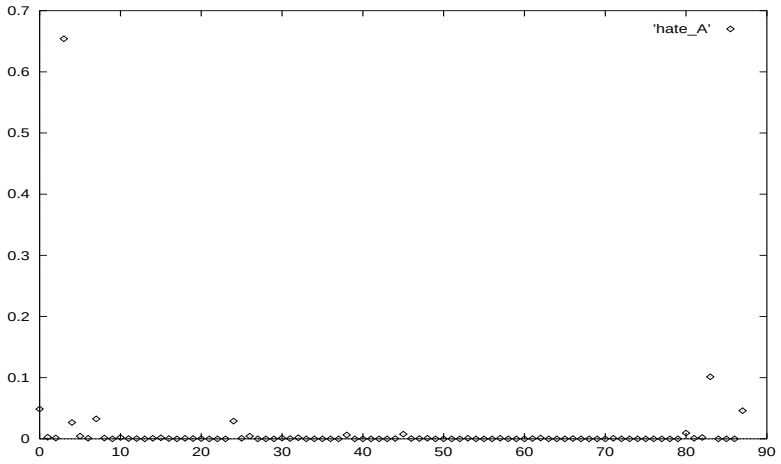
   61%/36% of the verbs in the respective versions `A` and `B` chose a verb in the same class as closest verb, so the remaining 39%/64% point at verbs from different classes. What are the reasons for this division? Why are there at all verbs which point to a verb from a different class? The answer is concerned with two issues underlying the determination of the pointers: (a) the representation of the verbs which describes the linguistic properties of the verbs, and (b) the distance measure which determines the differences between two verbs, based on their representations.

   (a) I start with an investigation of the linguistic representation of the verbs. How well are the verbs modelled, concerning their linguistic properties? For that, I have a look at the representation of some verbs, i.e. their distributions over subcategorisation frames: I take one positive (pointing to a verb from the same class) and one negative (pointing to a verb from a different class) example verb from both versions and the respective verbs they point at. For all verbs I present the distribution over the subcategorisation frames and example sentences[1] for the use of the frames – in case the usage is linguistically realisable (otherwise the verb is marked by a question mark). I only consider frames which are used with a probability greater than 0.02.

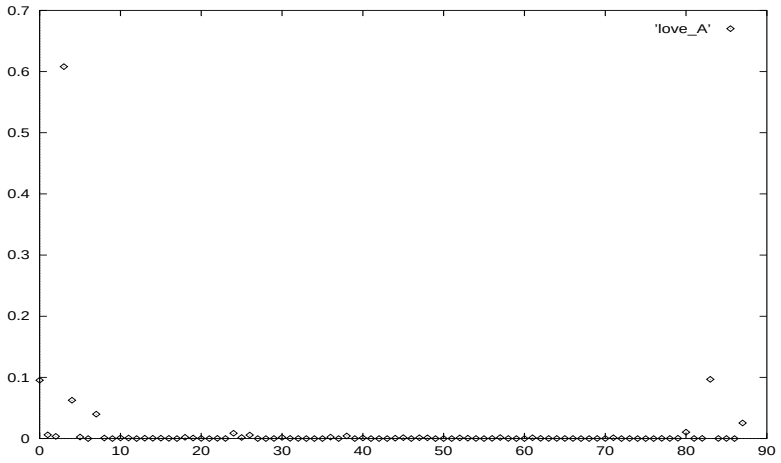
   - Version `A`, positive example:
     The verb *hate* chose the verb *love* as closest verb. Both verbs belong to the class *Admiration*.
     Compare the distributions in the following figures showing an enumeration of the 88 relevant frames on the x-axis, and the probability of the frames on the y-axis[2]:

[1] Evidence for the structure and content of the example sentences was taken from the BNC and WordNet.

[2] See appendix A for the relation between the numbering and the respective subcategorisation frames.



Distribution of the verb *hate*



Distribution of the verb *love*

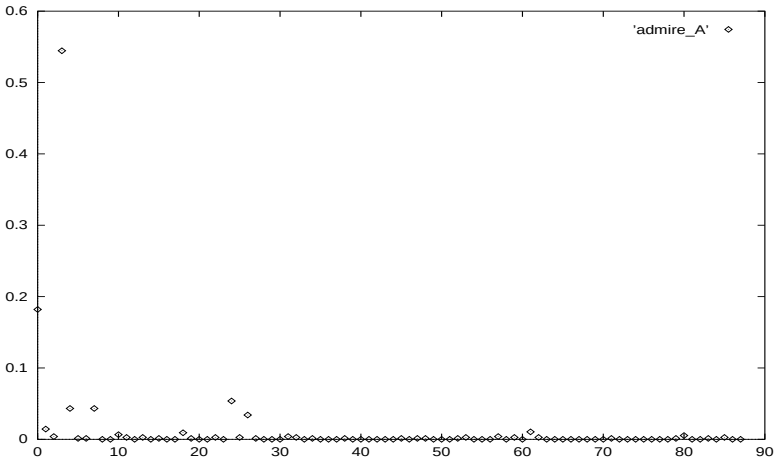
There is a strong preference (0.65/0.61) for a transitive use. The diathesis alternation with other frames as proposed by the distribution can be illustrated by the following example sentences:

(3.3)

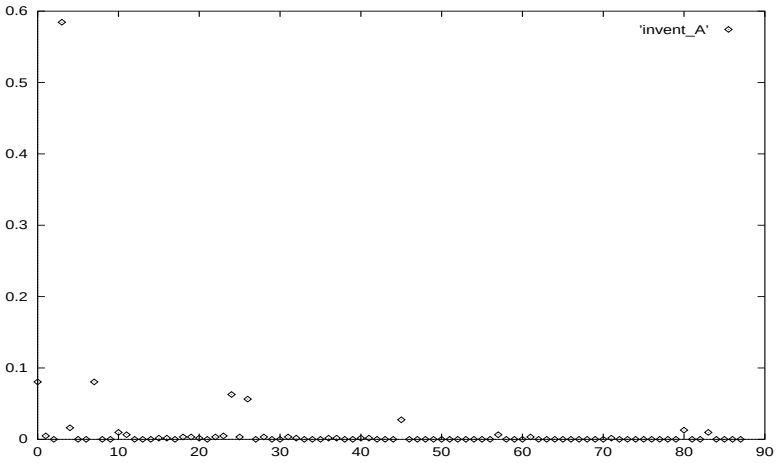
`subj`	hate ?
	John loves.
`subj:obj`	Susan's mother hates her husband.
	Fergie loves her status.
`subj:obj:adv`	He hates the cleric in particular.
	Alan loves his mother hopelessly.
`subj:obj:obj`	hate/love ?
`subj:obj:pp.for`	She hates school for the force.
`subj:to`	Elinor hates to be interrupted.
	People love to play.
`subj:vger`	People hate working.
	The sailor loves exploring.

There is no linguistic evidence for an intransitive use (of *hate*) or a ditransitive use (of either verb).

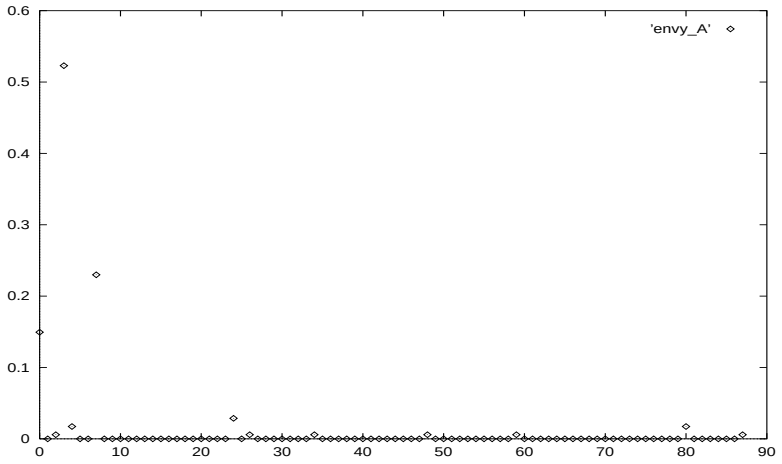
- Version `A`, negative example:
  The verb *admire* chose the verb *invent* as closest verb. There is no common class for the two verbs. Why did *admire* not choose *envy*, for example? These two verbs belong to the class *Admiration*.
  Compare the distributions of the three verbs:



Distribution of the verb *admire*



Distribution of the verb *invent*



Distribution of the verb *envy*

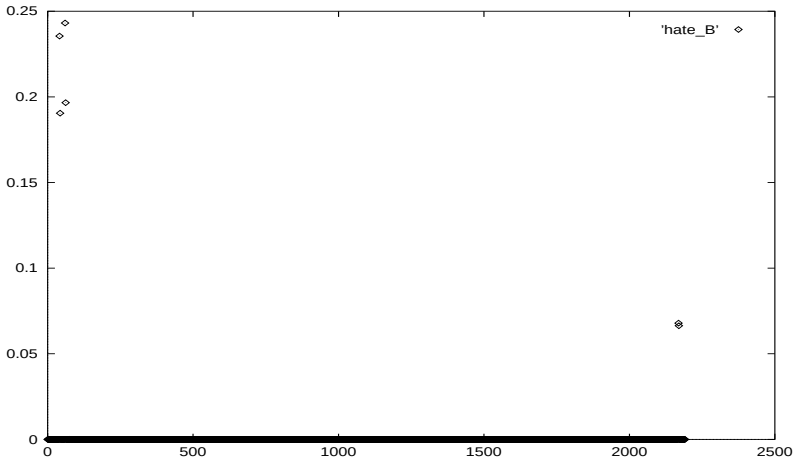
All three verbs have a strong preference for the transitive frame (0.54/0.58/0.52). The alternation behaviour as proposed by the distribution can be illustrated by the following example sentences:

(3.4)

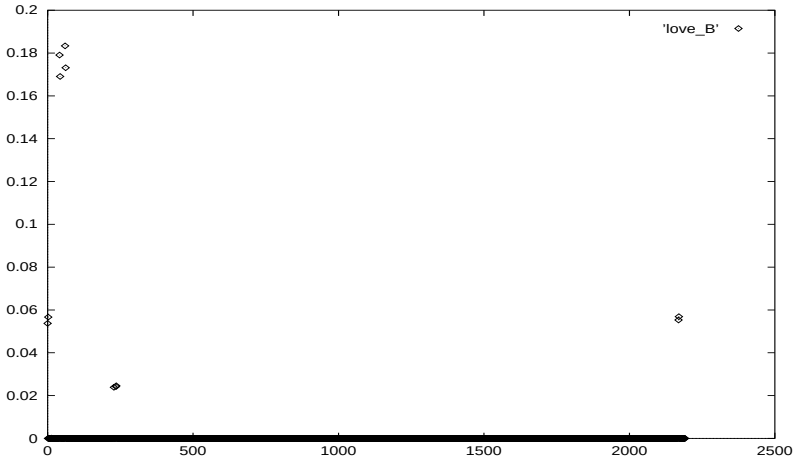
`subj`	admire/invent/envy ?
`subj:obj`	Everyone admires the guy.
	Dädalus invented the tyre.
	The girls envy the lifestyle.
`subj:obj:adv`	I admire Lewis immensely.
`subj:obj:obj`	admire/invent ?
	I envy Harvey the nice welcome.
`subj:obj:pp.for`	Adam admires her for her energy.
	He invents an excuse for {his wife / not coming}.
	I envy him for his tenaciousness.
`subj:obj:pp.in`	She admires {the qualities in Mary / the doctor in Hampstead}.
	An engineer invented the dye in 1856.
`subj:obj:to`	John invented a language to communicate.

There are no examples for an intransitive use of the verbs, and none for the verbs *admire* and *invent* with the ditransitive frame `subj:obj:obj`.

- Version `B`, positive example:
  As in version `A`, the verb *hate* chose the verb *love* as closest verb. Both verbs belong to the class *Admiration*.
  Following you find the verbs' distributions over the 2,192 subcategorisation frames including selectional preferences. Most probabilities are zeroes, and it is only possible to see a tendency of the exact frames which I will interpret afterwards.



Distribution of the verb *hate*



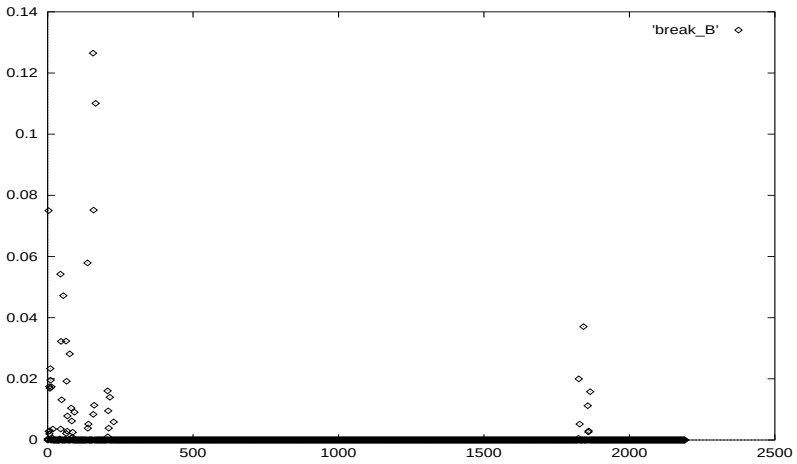
Distribution of the verb *love*

The strongest preferences (probabilities between 0.17 and 0.24) are for a transitive use of the verbs, demanding living entities as subject and object. The alternation behaviour as proposed by the distributions is illustrated by the following example sentences:

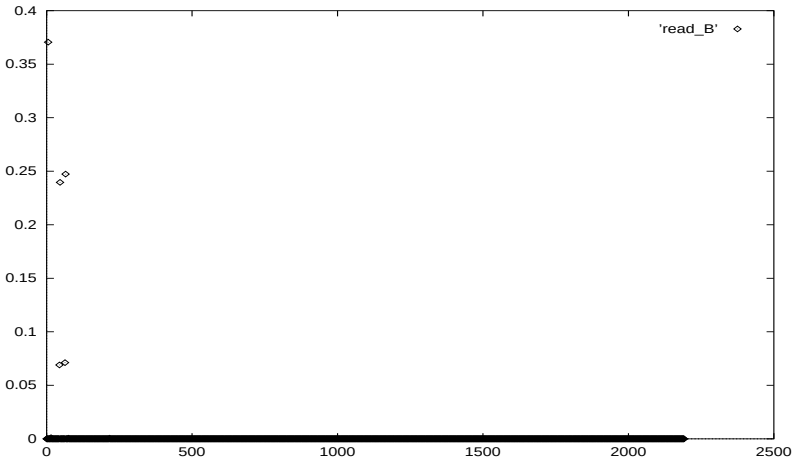
(3.5)

`subj::LifeForm`	John loves.
`subj::Agent`	ditto
`subj:obj::LifeForm:LifeForm`	Susan's mother hates her husband.
	Susan's mother loves her husband.
`subj:obj::LifeForm:Agent`	ditto
`subj:obj::Agent:LifeForm`	ditto
`subj:obj::Agent:Agent`	ditto
`subj:obj:adv::LifeForm:Agent`	Susan's mother loves her husband deeply.
`subj:obj:adv::Agent:LifeForm`	ditto
`subj:obj:adv::Agent:Agent`	ditto
`subj:to::LifeForm`	Vaughan hates to cook.
	Vaughan loves to cook.
`subj:to::Agent`	ditto

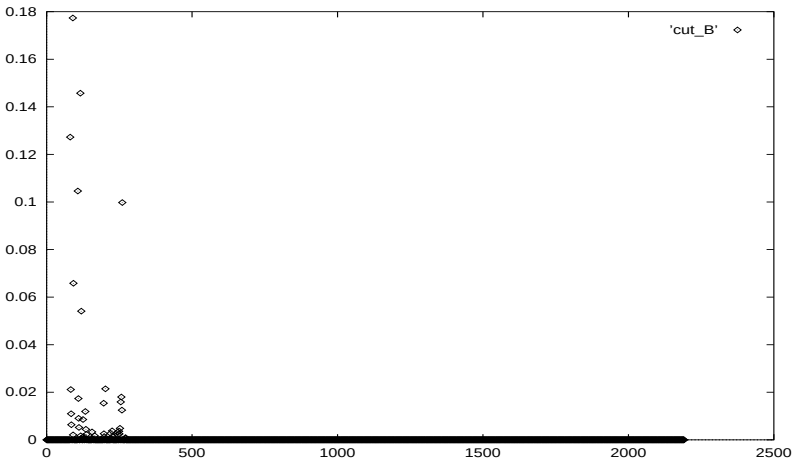
- Version `B`, negative example:
  The verb *break* chose the verb *read* as closest verb. There is no common class for the two verbs. Why did *break* not choose *cut*, for example? These two verbs belong to the subclass *Disassemble:Splitting*.
  Following is an overview of the distributions of the three verbs:



Distribution of the verb *break*



Distribution of the verb *read*



Distribution of the verb *cut*

At first glance, the distribution of *cut* actually looks more similar to *break* than *read* does. For a closer investigation, following is a display of the alternation behaviour as proposed by the distribution:

(3.6)

`subj::PhysObject`	The glass breaks.
`subj::Abstract`	read ?
`subj::Event`	break ?
`subj:obj::LifeForm:PhysObject`	John breaks the window.
	John reads a book.
`subj:obj::LifeForm:Abstract`	John reads a poem.
`subj:obj::LifeForm:Part`	break ?
`subj:obj::Agent:PhysObject`	John reads a book.
`subj:obj::Agent:Abstract`	John reads a poem.
`subj:obj::Agent:Part`	break ?
`subj:obj::PhysObject:PhysObject`	The knife cuts the silk.
`subj:obj::PhysObject:Abstract`	cut ?
`subj:obj::PhysObject:Possession`	The river cuts the land.
`subj:obj::PhysObject:Part`	The knife cuts a slice of bread.
`subj:obj::Abstract:PhysObject`	The carelessness cuts the ice.
`subj:obj::Abstract:Possession`	The carelessness cut his possessions.
`subj:obj::Abstract:Part`	cut ?
`subj:obj::Shape:PhysObject`	break ?
`subj:obj::Event:PhysObject`	The event broke the chain.
`subj:obj::Event:Abstract`	The cry broke the silence.
`subj:obj::Event:Part`	break ?
`subj:obj::Possession:Possession`	cut ?
`subj:obj:adv::Action:Action`	The design cuts amusement significantly.
`subj:pp.into::Abstract:Abstract`	Carefulness broke into carelessness.

The only overlap of subcategorisation frames is for the two verbs *break* and *read* used transitively with a living subject and an inanimate object. For each of the verbs there are some subcategorisation frames for which I could not find examples.

The examples show that the linguistic properties of the verbs are generally modelled sufficiently, since, on one hand, most of the subcategorisation frames in the modelled distribution can be identified within the diathesis alternation of the respective verb, and, on the other hand, one can identify the preferred linguistic alternations in the modelled alternation behaviour.

Problems arise when comparing verbs according to their linguistic representations: some verb pairs like *hate* and *love* are so similar in their usage and representation that they show overlap in most of their subcategorisation frames and are therefore easy to assign to a common class. But verb pairs like *break* and *cut* which are similar to each other to a certain extent show overlap in only a

part of their subcategorisation frames. And as the example above shows, another verb (like *read*) might show overlap in the usage, too, so that it is incidentally preferred as most similar verb. This means that the linguistic representation might be to the point, but not precise enough. The enlargement of the syntactic information in version `A` with the selectional preferences in version `B` already helped this problem to a certain extent. For example, *admire* which chose *invent* as most similar verb in version `A`, took *envy* in version `B`.

But the representation in the latter version is not sufficiently solved so far. Following I refer to specific parts of the representation which introduced noise into the linguistic model:

- The linguistic representation of the verbs over-generated the intransitive use: each verb demanded – among other frames – a `subj` frame. This took place in cases where the intransitive frame is actually part of the diathesis alternation (for the verb *love*, for example), but also for verbs that do not allow an intransitive use (for the verb *admire*, for example). It might have been caused by either underlying sentences containing an NP ellipsis (like *"Our responsibilities are as follows: you invent, I commercialize."*), by parsing mistakes, or while extracting the subcategorisation frames.
- In some cases too many objects were found in the transitive use of a verb. For instance, the frame `subj:obj:obj` appeared for verbs (like *hate* and *love* above) which do not alternate with a ditransitive frame. The extra object was caused by parsing mistakes: compound nouns were analysed as two objects, as in *He loves Welshness poetry*, determiner-noun pairs were not recognised as that, for example in *I admire that spirit*, and appositions were interpreted as two objects, as in *I love you, Gabriel.*
- As mentioned before, the verbs were not distinguished for their different senses. It is possible that one sense of a verb was used with specific frames, and another sense with different frames. But the distribution merged these uses together. For example, the verb *cut* bears two different senses when used with frame `subj:obj::PhysObject:Possession` – *Existence* – or frame `subj:obj ::PhysObject:PhysObject` – *Disassemble:Splitting*. Polysemy was therefore not recognised in the linguistic model of the verbs.

- Verbs with a low overall frequency (like *snow*, for example), appeared with only a small range of subcategorisation frame types in total – usually one or two –, and the joint frequency was also small. When smoothing the frequencies by adding 0.5 to all frame types (i) the frequencies for the observed frame do hardly differ from the smoothed frequencies, and (ii) most frequencies were equal (namely 0.5), since most used to be zero. The maximum likelihood estimate was therefore similar within the overall distribution over subcategorisation frames. Compare this with the distributions of verbs with higher frequencies where a certain number of frames was assigned high estimates, all others values are below 10^3. Measuring the distance between a low and a high frequency verb will always show a similarity which lies beyond the average. This is the reason why, for example, *snow* was chosen by 13 verbs in version `B` as most similar verb.
- WordNet provides two top level conceptual classes for living entities, `LifeForm` and `Agent`, which both sub-ordinate the conceptual class of persons, a frequently appearing concept. A part of the persons appearing in the context of a verb was assigned to the former, a part to the latter class, so in case the concept of an individual was required in a subcategorisation frame, generally both possible concepts were cited and therein provided redundant information.
- And finally, there were some subcategorisation frames for which I could not find example sentences. That is, their linguistic value is doubtful. Predominantly, this happened in the representation of the subcategorisation frames including selectional preferences.

The degree of noise in the linguistic representations is obviously stronger in version `B` than in version `A`. The underlying ideas for the representation and their influence on the success of the clustering process will be reassessed in section 3.2.5, since it concerns all clustering approaches.

(b) The second influence next to the linguistic representation of the verbs comes from the distance measures, since they determine the importance and the relations of the representations. As the results in section 3.1 show, different measures created different pointers between the verbs (assuming the same representation).

To answer the question of why the verbs point to the respective most similar verb in the way presented above and why this is only correct in 61%/36% of the pointers, this effect is due to a large extent to the linguistic representation of the verbs, and in addition depending on the distance measure when comparing the representational distributions.

2. *Why does the clustering algorithm cluster the verbs according to the pointers in the way the clusters represent?*

   Obviously, next to the definition of the pointers, the clustering algorithm had great influence on the resulting clusters. The influence is illustrated in the success of the clustering process: 61%/36% of the verbs in the respective version `A` and `B` chose a verb in the same class as closest verb, but only 24%/3% of the verbs were assigned to a correct cluster. The loss of precision must have be caused by the clustering algorithm.

   The lack of success can partly be explained by the inability of the algorithm to filter false pointers like those in the negative examples above: a large portion of the incorrect clusters (12 out of 18 in version `A`, and 11 out of 23 in version `B`) contains more than four verbs. Interestingly, the verbs in those clusters tend to belong to few classes included in the cluster, i.e. the clusters are incorrect, but their parts are correct classes. For example, the cluster

   ```
   C(5) :   allocate
            send
            transfer
            transport
            entrust
   ```

   merges verbs from only the two classes *Change of Possession:Giving* and *Sending*. By a wrong pointer the correct classes were merged to an incorrect one.

   The clustering shows that a more sophisticated approach is needed to cluster the verbs successfully. This leads to the following subsection 3.2.3 describing and interpreting the iterative method.

To conclude this section with a brief summary: I have described the correct clusters achieved by *One-Step Distance Clustering* and then investigated the problems of (i) this specific algorithm and (ii) the distance clustering in general. Generally said, the data fed into distance clustering was not fine-grained enough to model subtle tendencies; the model became worse in version `B`. Specifically said, the approach is too simple to model that part of the data which was linguistically well represented into successful clusters.

3.2.3 Iterative Distance Clustering

This section is concerned with distance clustering based on the more sophisticated iterative algorithm. As before, I start with the description of the correct clusters, presented in a similar way than above: each cluster is identified by `C(X)` – where `X` is the number of members in the cluster – and the respective class name. In addition, each verb in a cluster is now followed by the five subcategorisation frames with the highest probabilities in the overall verb's distribution. The additional information facilitates the interpretation of the clustering.
In the description of the clusters I concentrate on the striking facts, i.e. the most probable subcategorisation frames which are common for the verbs in the cluster.

Following are the correct clusters based on the verbs' alternation behaviour concerning the subcategorisation frames only (version `A`).

The first selection of verbs belongs to the specific sub-class of *Rubbing* verbs within the *Surfacing* class. They mainly alternate between using a `subj:obj` frame and adding an adverb or a prepositional phrase headed by *with* to that frame:

```
C(2) -- Surfacing: Rubbing

     brush          * subj:obj            0.25199203187251 *
                    * subj:obj:adv        0.192231075697211 *
                    * subj:pp.against     0.0727091633466136 *
                    * subj                0.0687250996015936 *
                    * subj:obj:pp.with    0.0408366533864542 *

     rub            * subj:obj            0.310964083175803 *
                    * subj:obj:pp.with    0.110586011342155 *
                    * subj:obj:adv        0.104914933837429 *
                    * subj                0.0482041587901701 *
                    * subj:obj:pp.over    0.0444234404536862 *
```

This next cluster contains the specific sub-class *Loading* of verbs of *Surfacing*. As the preceding sub-class belonging to the same class, the verbs prefer the use of a `subj:obj` frame. But differently, the verbs allow a prepositional phrase either headed by *with* or headed by *into*.

```
C(2) -- Surfacing: Loading

     pack           * subj:obj            0.305755395683453 *
                    * subj:obj:pp.with    0.156115107913669 *
                    * subj:obj:pp.in      0.0784172661870504 *
                    * subj                0.0769784172661871 *
                    * subj:obj:pp.into    0.0381294964028777 *
```

```
load          * subj:obj           0.331944444444444 *
              * subj:obj:pp.with   0.1375 *
              * subj:obj:pp.into   0.0875 *
              * subj               0.0597222222222222 *
              * subj:pp.with       0.0486111111111111 *
```

The verbs of *Giving* in the sub-class of verbs of *Change of Possession* also prefer the use of the `subj:obj` frame, possibly followed by a second object or a prepositional phrase headed by *to*:

```
C(3) -- Change of Possession: Giving

     sell          * subj:obj         0.342807579396851 *
                   * subj:obj:pp.to   0.113290632506005 *
                   * subj             0.0844675740592474 *
                   * subj:obj:pp.in   0.0535094742460635 *
                   * subj:obj:obj     0.0508406725380304 *

     pay           * subj:obj         0.355500354861604 *
                   * subj:obj:obj     0.0885024840312278 *
                   * subj:obj:pp.for  0.0738821859474805 *
                   * subj:obj:pp.to   0.0667849538679915 *
                   * subj             0.061958836053939 *

     offer         * subj:obj         0.387520085698982 *
                   * subj:obj:obj     0.173379753615426 *
                   * subj             0.0663631494376004 *
                   * subj:obj:pp.to   0.060149973219068 *
                   * subj:to          0.0544724156400643 *
```

The verbs of *Obtaining* form another sub-class of *Change of Possession* verbs. They demand `subj:obj` frames or, alternatively, an additional prepositional phrase headed by *from*:

```
C(4) -- Change of Possession: Obtaining

     receive       * subj:obj           0.468201391189135 *
                   * subj:obj:pp.from   0.124268521585514 *
                   * subj               0.076681020205366 *
                   * subj:obj:pp.in     0.053936181958706 *
                   * subj:obj:obj       0.0402451142762504 *

     purchase      * subj:obj           0.391456582633053 *
                   * subj:obj:pp.from   0.0987394957983193 *
                   * subj               0.069327731092437 *
                   * subj:obj:pp.in     0.0623249299719888 *
                   * subj:obj:pp.for    0.0567226890756303 *

     collect       * subj:obj           0.396492236917769 *
                   * subj               0.106095457159287 *
                   * subj:obj:pp.from   0.104945370902818 *
                   * subj:obj:pp.in     0.0526164462334675 *
                   * subj:obj:pp.for    0.0388154111558367 *
```

```
buy             * subj:obj          0.394256756756757 *
                * subj              0.0881756756756757 *
                * subj:obj:obj      0.0762387387387387 *
                * subj:obj:pp.for   0.0652027027027027 *
                * subj:obj:pp.in    0.0570945945945946 *
```

The next cluster contains the specific sub-class *Separating* within the class of *Disassemble* verbs. The verbs have a preference for the `subj:obj` frame, alternating with a subject only. Both frames also appear with an additional prepositional phrase headed by *from*:

```
C(3) -- Disassemble: Separating

     disconnect     * subj:obj          0.234126984126984 *
                    * subj:pp.from      0.170634920634921 *
                    * subj:obj:pp.from  0.0992063492063492 *
                    * subj              0.0674603174603175 *
                    * subj:obj:pp.at    0.0198412698412698 *

     extract        * subj:obj:pp.from  0.310473815461347 *
                    * subj:obj          0.275561097256858 *
                    * subj              0.0785536159600998 *
                    * subj:pp.from      0.0660847880299252 *
                    * subj:obj:pp.with  0.0261845386533666 *

     separate       * subj:obj:pp.from  0.288519637462236 *
                    * subj:obj          0.278449144008056 *
                    * subj              0.142497482376636 *
                    * subj:pp.from      0.0780463242698892 *
                    * subj:obj:pp.into  0.027693856998993 *
```

The verbs of *Destruction* have a strong preference for the `subj:obj` frame, partly allowing an additional prepositional phrase headed by `in`, or a subject only:

```
C(3) -- Destruction

     murder         * subj:obj          0.676287051482059 *
                    * subj:obj:pp.in    0.0663026521060842 *
                    * subj              0.0553822152886115 *
                    * subj:obj:pp.on    0.0179407176287051 *
                    * subj:obj:pp.for   0.0179407176287051 *

     destroy        * subj:obj          0.683620689655172 *
                    * subj:obj:pp.in    0.0738505747126437 *
                    * subj              0.057183908045977 *
                    * subj:obj:obj      0.0238505747126437 *
                    * subj:obj:adv      0.0192528735632184 *

     kill           * subj:obj          0.635327963176064 *
                    * subj:obj:pp.in    0.108285385500575 *
                    * subj              0.0546605293440737 *
                    * subj:obj:adv      0.0254315304948216 *
                    * subj:obj:obj      0.023590333716916 *
```

The verbs of *Declaration* agree in appearing with a subject followed by a whole sentence or a *that*-phrase:

```
C(3) -- Declaration

     believe       * subj:that     0.248414573489182 *
                   * subj:s        0.247233275304651 *
                   * subj          0.174365829395673 *
                   * subj:obj      0.131341706043273 *
                   * subj:obj:to   0.0681733399651828 *

     think         * subj:s        0.476226211202497 *
                   * subj          0.216835764743551 *
                   * subj:that     0.0593844429769266 *
                   * subj:adv      0.0423745772548078 *
                   * subj:obj      0.0378119309199336 *

     suppose       * subj:s        0.392576204523107 *
                   * subj          0.244428056374959 *
                   * subj:obj:to   0.211324156014422 *
                   * subj:adv      0.0440019665683382 *
                   * subj:that     0.03695509668961 *
```

These verbs of *Telling* mainly alternate between appearing with only a subject and a subject with an additional object:

```
C(2) -- Telling

     explain       * subj             0.477065026362039 *
                   * subj:obj         0.214499121265378 *
                   * subj:that        0.0927065026362039 *
                   * subj:obj:pp.in   0.0290861159929701 *
                   * subj:obj:pp.to   0.0287346221441125 *

     write         * subj             0.331352657004831 *
                   * subj:obj         0.204009661835749 *
                   * subj:pp.to       0.0583091787439614 *
                   * subj:obj:pp.in   0.0431400966183575 *
                   * subj:pp.in       0.0343478260869565 *
```

The following verbs of *Telling* prefer a `subj:obj` frame, possibly followed by a second object, a *that*-phrase, or an infinitival phrase:

```
C(4) -- Telling

     teach         * subj:obj         0.367608695652174 *
                   * subj:obj:to      0.150652173913043 *
                   * subj:obj:obj     0.126739130434783 *
                   * subj             0.0693478260869565 *
                   * subj:obj:that    0.0384782608695652 *

     tell          * subj:obj         0.593235143003939 *
                   * subj:obj:that    0.117023462921733 *
                   * subj:obj:obj     0.0597876348689844 *
                   * subj:obj:adv     0.0450248330193526 *
                   * subj:obj:to      0.0372495290289433 *
```

```
instruct      * subj:obj:to      0.504617414248021 *
              * subj:obj         0.213060686015831 *
              * subj             0.0718997361477573 *
              * subj:obj:pp.in   0.0402374670184697 *
              * subj:obj:obj     0.0217678100263852 *

advise        * subj:obj:to      0.35484693877551 *
              * subj:obj         0.239030612244898 *
              * subj             0.0875 *
              * subj:obj:that    0.0701530612244898 *
              * subj:that        0.0451530612244898 *
```

The verbs of *Characterisation* prefer a subject followed by an object, often accompanied by an *as*-phrase:

```
C(2) -- Characterisation

      classify      * subj:obj:as        0.345401174168297 *
                    * subj:obj           0.196673189823875 *
                    * subj:obj:pp.into   0.0792563600782779 *
                    * subj:obj:pp.in     0.0616438356164384 *
                    * subj:obj:pp.as     0.0596868884540117 *

      describe      * subj:obj           0.294258683255573 *
                    * subj:obj:as        0.214424572317263 *
                    * subj               0.11864955935718 *
                    * subj:obj:pp.in     0.0887117677553136 *
                    * subj:obj:pp.as     0.0784733022291343 *
```

The preferences for verbs in the *Desire* class is towards a subject followed by an infinitival phrase. Alternatively a `subj:obj` frame is used, partly followed by an additional infinitival phrase:

```
C(4) -- Desire

      need          * subj:to            0.382847629835582 *
                    * subj:obj           0.318590601723132 *
                    * subj               0.0962654034943192 *
                    * subj:obj:to        0.0536333367658669 *
                    * subj:obj:pp.for    0.0189647478804105 *

      like          * subj:to            0.344067278287462 *
                    * subj:obj           0.34302752293578 *
                    * subj               0.142110091743119 *
                    * subj:obj:adv       0.0364220183486239 *
                    * subj:obj:obj       0.0262691131498471 *

      want          * subj:to            0.533195075557434 *
                    * subj:obj           0.149146676529642 *
                    * subj               0.110892423121632 *
                    * subj:obj:to        0.102729049984149 *
                    * subj:to:adv        0.0163663742999049 *

      desire        * subj:obj           0.25 *
                    * subj               0.244535519125683 *
                    * subj:to            0.203551912568306 *
                    * subj:obj:to        0.069672131147541 *
                    * subj:s             0.0204918032786885 *
```

The verbs of *Admiration* strongly demand a `subj:obj` frame. Alternatively a subject only or combined with an infinitival phrase is possible:

```
C(2) -- Admiration

     hate          * subj:obj       0.641761612620508 *
                   * subj:to        0.0996932515337423 *
                   * subj           0.0479842243645925 *
                   * subj:vger      0.0453549517966696 *
                   * subj:obj:obj   0.0322085889570552 *

     love          * subj:obj       0.603842412451362 *
                   * subj:to        0.0962224383916991 *
                   * subj           0.0947632944228275 *
                   * subj:obj:adv   0.0623378728923476 *
                   * subj:obj:obj   0.0396400778210117 *
```

Both verbs of *Social Interaction* prefer a `subj:obj` frame, but also allow the subject only:

```
C(2) -- Social Interaction

     meet          * subj:obj         0.399157303370787 *
                   * subj             0.153876404494382 *
                   * subj:obj:pp.in   0.0467977528089888 *
                   * subj:obj:obj     0.0437640449438202 *
                   * subj:pp.in       0.0378089887640449 *

     play          * subj:obj         0.401702175628452 *
                   * subj             0.0870815015218127 *
                   * subj:obj:pp.in   0.0585616052305264 *
                   * subj:pp.in       0.0516852665990305 *
                   * subj:adv         0.0404125803178898 *
```

The verbs of *Manner of Speaking* strongly tend towards demanding a subject only. In addition, an adverb, an object or a prepositional phrase headed by either *at*, *in* or *to* are possible:

```
C(4) -- Manner of Speaking

     moan          * subj            0.478 *
                   * subj:adv        0.09 *
                   * subj:obj        0.05 *
                   * subj:pp.about   0.042 *
                   * subj:pp.in      0.038 *

     shout         * subj            0.534473094170404 *
                   * subj:obj        0.0871636771300448 *
                   * subj:pp.at      0.0675448430493274 *
                   * subj:adv        0.0395179372197309 *
                   * subj:pp.to      0.0355941704035874 *
```

```
scream        * subj            0.556396148555708 *
              * subj:pp.at      0.0749656121045392 *
              * subj:obj        0.0529573590096286 *
              * subj:pp.in      0.0433287482806052 *
              * subj:adv        0.0419532324621733 *

whisper       * subj            0.642141515341265 *
              * subj:adv        0.080463368816531 *
              * subj:pp.to      0.0654351909830933 *
              * subj:obj        0.0466499686912962 *
              * subj:pp.in      0.0335003130870382 *
```

The following two verbs *live* and *stay* both belong to the two classes of *Lodging* and *Existence*. The common preferences are towards demanding a subject only, often accompanied by an adverb or a prepositional phrase headed by *in*:

```
C(2) -- Lodging / Existence

     live          * subj:pp.in     0.290654831686485 *
                   * subj           0.173589013565567 *
                   * subj:adv       0.109278177859655 *
                   * subj:obj       0.0595377658683638 *
                   * subj:pp.with   0.0469770557695528 *

     stay          * subj           0.193655851680185 *
                   * subj:adv       0.136877172653534 *
                   * subj:pp.in     0.125 *
                   * subj:ap        0.0850231749710313 *
                   * subj:pp.at     0.0679316338354577 *
```

These verbs of *Existence* show a strong tendency to appearing with a subject only. Alternatively they take a prepositional phrase in addition, mostly headed by *in*:

```
C(2) -- Existence

     persist       * subj              0.558662280701754 *
                   * subj:pp.in        0.160635964912281 *
                   * subj:pp.for       0.0301535087719298 *
                   * subj:pp.with      0.0279605263157895 *
                   * subj:adv         0.0224780701754386 *

     exist         * subj              0.568633739576652 *
                   * subj:pp.in        0.127325208466966 *
                   * subj:pp.for       0.0458627325208467 *
                   * subj:to           0.0304682488774856 *
                   * subj:pp.between   0.0298268120590122 *
```

The verbs belonging to the cluster *Manner of Motion* preferably demand a subject or a subject with an additional adverb, object or prepositional phrase headed by *into*:

```
C(4) -- Manner of Motion

     fly           * subj             0.202803738317757 *
                   * subj:obj         0.105607476635514 *
                   * subj:pp.to       0.0937694704049844 *
                   * subj:adv         0.078816199376947 *
                   * subj:pp.into     0.0489096573208723 *

     run           * subj:obj         0.244731610337972 *
                   * subj             0.109675281643472 *
                   * subj:adv         0.0534791252485089 *
                   * subj:pp.into     0.0510934393638171 *
                   * subj:pp.out_of   0.0435387673956262 *

     climb         * subj:obj         0.25174520069808 *
                   * subj             0.110383944153578 *
                   * subj:pp.into     0.0920593368237347 *
                   * subj:pp.to       0.0571553228621291 *
                   * subj:obj:pp.to   0.0440663176265271 *

     roll          * subj:obj         0.211333333333333 *
                   * subj             0.166 *
                   * subj:adv         0.0673333333333333 *
                   * subj:pp.into     0.05 *
                   * subj:pp.in       0.034 *


C(2) -- Manner of Motion

     bounce        * subj             0.195754716981132 *
                   * subj:obj         0.0919811320754717 *
                   * subj:pp.on       0.0542452830188679 *
                   * subj:adv         0.0542452830188679 *
                   * subj:pp.into     0.0542452830188679 *

     float         * subj             0.184365781710914 *
                   * subj:adv         0.084070796460177 *
                   * subj:obj         0.0811209439528024 *
                   * subj:pp.in       0.0722713864306785 *
                   * subj:obj:pp.on   0.0427728613569322 *
```

The verbs of *Aspect* alternate between demanding a subject only and an object or a gerund in addition to the subject:

```
C(2) -- Aspect

     stop          * subj             0.287293244705519 *
                   * subj:obj         0.188823620343175 *
                   * subj:vger        0.186968619570258 *
                   * subj:adv         0.0427423094759623 *
                   * subj:to          0.0394960581233576 *
```

```
finish

                * subj:obj          0.406296627872277 *
                * subj              0.202178454192778 *
                * subj:vger         0.082214264398687 *
                * subj:pp.with      0.0443151298119964 *
                * subj:obj:pp.in    0.0320799761265294 *
```

Other *Aspect* verbs demand a subject and an infinitival phrase. Alternatively the subject only or with an additional object or gerund is possible:

```
C(3) -- Aspect

     begin          * subj:to         0.510654350907678 *
                    * subj            0.154670995473044 *
                    * subj:obj        0.0620741689149024 *
                    * subj:vger       0.0473044034935297 *
                    * subj:pp.with    0.0291508528053409 *

     continue       * subj:to         0.469665109034268 *
                    * subj            0.255101246105919 *
                    * subj:obj        0.0607866043613707 *
                    * subj:pp.in      0.0228582554517134 *
                    * subj:adv        0.0223909657320872 *

     start          * subj:to         0.248708698860375 *
                    * subj:vger       0.157784701156022 *
                    * subj            0.140567352627695 *
                    * subj:obj        0.118676723784537 *
                    * subj:pp.at      0.0373452488316799 *
```

The correct clusters based on the verbs' alternation behaviour concerning the subcategorisation frames and their selectional preferences (version `B`) are presented in a similar way. Each verb in a cluster is followed by the five combinations of a subcategorisation frame with its preferences, for the highest probabilities in the overall verb's distribution. The additional semantic information helps to get an idea about the semantic concepts of the arguments, especially concerning the variety of prepositional phrases.

The first cluster contains verbs of *Giving*, a sub-class of *Change of Possession* verbs, showing strong preferences for a `subj:obj` frame, mostly accompanied by a prepositional phrase headed by `to`. Preferably the subject is a group, the object a possession, and the nominal head of the prepositional phrase a group[3] or inanimate entity:

[3]Most groups in WordNet refer to living entities.

```
C(2) -- Change of Possession: Giving

     allocate      * subj:obj:pp.to   Group:Possession:Group   0.224852016680507 *
                   * subj:obj         Group:Possession         0.172679355461681 *
                   * subj:obj:pp.to   Group:Abstract:Group     0.159894302929161 *
                   * subj:obj         Group:Abstract           0.0425168928907968 *
                   * subj:obj         PhysObject:Possession    0.040112032971137 *

     transfer      * subj:obj:pp.to   Group:Group:PhysObject   0.21840864822257 *
                   * subj:obj:pp.to   Group:State:PhysObject   0.10581066708609 *
                   * subj:obj:pp.to   Group:Group:Group        0.105662063929036 *
                   * subj:obj:pp.to   Group:Possession:Group   0.0886785169300222 *
                   * subj:obj         Abstract:Possession      0.0808659643973325 *
```

For the verbs of *Obtaining*, also sub-class of verbs of *Change of Possession*, the preferred frame is `subj:obj`. *find* mostly chooses an agentive subject and object, *leave* also takes an agent as subject, but varies between an animate and inanimate object:

```
C(2) -- Change of Possession: Obtaining

     find          * subj:obj      LifeForm:LifeForm     0.0984983676279855 *
                   * subj:obj      Agent:LifeForm        0.0980725887287475 *
                   * subj:that     LifeForm              0.0754720681253665 *
                   * subj:obj      LifeForm:Agent        0.0752089591469579 *
                   * subj:obj      Agent:Agent           0.0748838533750595 *

     leave         * subj          LifeForm              0.109476082345808 *
                   * subj:obj:ap   Action:LifeForm       0.106014337383049 *
                   * subj          Agent                 0.104817824945699 *
                   * subj:obj      LifeForm:PhysObject   0.0697924636617648 *
                   * subj:obj      Agent:PhysObject      0.0697627126835695 *
```

The common property of the *Destruction* verbs is a `subj:obj` frame with a phenomenon in the subject role and an inanimate object in the object role. Both verbs vary in the selection for the subject:

```
C(2) -- Destruction

     demolish      * subj:obj         Location:PhysObject         0.110303456242118 *
                   * subj:obj:pp.in   Event:PhysObject:Abstract   0.0894097605486901 *
                   * subj:obj         Phenomenon:PhysObject       0.0886125157889064 *
                   * subj:obj         Agent:PhysObject            0.0868696806557967 *
                   * subj:obj         LifeForm:PhysObject         0.0767890126060299 *

     smash         * subj:obj         PhysObject:PhysObject       0.252852148136339 *
                   * subj             PhysObject                  0.122313235238738 *
                   * subj:pp.into     PhysObject:PhysObject       0.104886857520264 *
                   * subj:obj         Phenomenon:PhysObject       0.0675156408956261 *
                   * subj:obj         Group:PhysObject            0.0552273617129487 *
```

Both *Creation* verbs tend to have an agentive subject and an inanimate object:

```
C(2) -- Creation

     pour          * subj:obj           Agent:PhysObject                0.163118021042506 *
                   * subj:obj           LifeForm:PhysObject             0.158340191499267 *
                   * subj:obj:pp.into   Agent:PhysObject:PhysObject     0.0851066519746377 *
                   * subj:obj:pp.into   LifeForm:PhysObject:PhysObject  0.0835640013683956 *
                   * subj:obj:obj       Agent:Agent:PhysObject          0.0746459915168309 *

     cook          * subj:obj           Agent:PhysObject                0.224287578890969 *
                   * subj:obj           LifeForm:PhysObject             0.221619026254064 *
                   * subj               PhysObject                      0.156583821828932 *
                   * subj               LifeForm                        0.0807303473650241 *
                   * subj:obj:pp.in     Abstract:PhysObject:PhysObject  0.0628334084486924 *
```

The verbs of *Declaration* preferably appear with a subject and a whole sentence, *think* and *believe* also with a *that*-phrase instead of the sentence. The subject is in all cases an agent, for *believe* it might be a group:

```
C(3) -- Declaration

     think         * subj:s         Agent            0.31749876059606 *
                   * subj:s         LifeForm         0.311281241098789 *
                   * subj           Agent            0.145654646708915 *
                   * subj           LifeForm         0.140895646331977 *
                   * subj:that      Agent            0.0392119513706955 *

     suppose       * subj:s         Agent            0.231321847728741 *
                   * subj:s         LifeForm         0.226458717562727 *
                   * subj           Agent            0.144727508609334 *
                   * subj           LifeForm         0.14017916851983 *
                   * subj:obj:to    Agent:LifeForm   0.0474329214626571 *

     believe       * subj:s         Group            0.147782947841849 *
                   * subj:that      LifeForm         0.13407986323598 *
                   * subj:that      Agent            0.132386385523092 *
                   * subj           Agent            0.09488952789685 *
                   * subj           LifeForm         0.0926174403475393 *
```

The tendency of this *Declaration* cluster is towards a group (alternatively: a living entity) as subject in the frames `subj`, `subj:that` and `subj:obj`. In case of an object it is generally a state or an activity:

```
C(2) -- Declaration

     announce      * subj:that        Group               0.179744393151009 *
                   * subj:obj         Group:Action        0.147890244956638 *
                   * subj:obj         Group:Psycho        0.0792090913729706 *
                   * subj             Group               0.076671817985555 *
                   * subj:obj:pp.on   Group:Action:Agent  0.0684631006196161 *

     declare       * subj             Agent               0.171078270346521 *
                   * subj             LifeForm            0.159608496880677 *
                   * subj:that        Group               0.120889118507728 *
                   * subj:obj:ap      Group:Action        0.0610505946272457 *
                   * subj:obj         Group:State         0.0608675025337214 *
```

The striking preferences in the *Telling* cluster is the overlap between the verbs *advise* and *instruct* which both use `subj:obj:to` with an agent or a group for the subject as well as for the object. The similarity with *warn* is based on the frame `subj:obj`, also with agents in both argument slots:

```
C(3) -- Telling

     advise        * subj:obj:to   LifeForm:Agent      0.1115078480514 *
                   * subj:obj:to   Agent:Agent         0.110717465475416 *
                   * subj:obj:to   LifeForm:LifeForm   0.105408271085895 *
                   * subj:obj:to   Agent:LifeForm      0.104661123129165 *
                   * subj:obj      Agent:Agent         0.061602393218947 *

     instruct      * subj:obj:to   Group:Agent         0.125098162005585 *
                   * subj:obj:to   Group:LifeForm      0.115472710013593 *
                   * subj:obj:to   Agent:Agent         0.0736885741688757 *
                   * subj:obj:to   Agent:LifeForm      0.0680187400030531 *
                   * subj:obj:to   LifeForm:Agent      0.067177532917295 *

     warn          * subj          Agent               0.118193245100582 *
                   * subj          LifeForm            0.111186392672282 *
                   * subj:that     Group               0.104569647755627 *
                   * subj:that     Agent               0.0697110174361806 *
                   * subj:obj      Agent:Agent         0.0697049715927629 *
```

These verbs of *Telling* show a preference for a subject alone or accompanied by a *that*-phrase; the subject is mostly realised by a psychological feature or an abstraction:

```
C(2) -- Telling

     suggest       * subj:that   Psycho              0.187084402063971 *
                   * subj:that   Abstract            0.178048044068513 *
                   * subj        Abstract            0.136319248896456 *
                   * subj:that   Action              0.0852568091186142 *
                   * subj        Psycho              0.0840625556416116 *

     show          * subj:that   Psycho              0.104034522759806 *
                   * subj:that   Action              0.0808689697416139 *
                   * subj:that   Abstract            0.0560109170977127 *
                   * subj        Abstract            0.0560008696588554 *
                   * subj:obj    Abstract:Abstract   0.054634233343515 *
```

Both *Perception* verbs tend to appear with a living subject, either alone or accompanied by an adjectival phrase (*feel*) or a *that*-phrase (*notice*):

```
C(2) -- Perception

     feel          * subj:ap     Agent         0.148957743908689 *
                   * subj:ap     LifeForm      0.144651883061671 *
                   * subj        Agent         0.106890183977064 *
                   * subj        LifeForm      0.101779437040005 *
                   * subj:s      LifeForm      0.086322731855946 *
```

```
notice        * subj         Agent          0.166969225709522 *
              * subj         LifeForm       0.161198702000427 *
              * subj:that    Agent          0.117709084266079 *
              * subj:that    LifeForm       0.116839827853374 *
              * subj:obj     Agent:Event    0.0577001467369464 *
```

The *Admiration* verbs choose a living being for the subject as well as for the object:

```
C(2) -- Admiration

     envy          * subj:obj:obj   Agent:LifeForm:State   0.202422062386625 *
                   * subj:obj       LifeForm:Agent         0.113387120464967 *
                   * subj:obj       LifeForm:LifeForm      0.112901377570876 *
                   * subj:obj       Agent:Agent            0.112565961049389 *
                   * subj:obj       Agent:LifeForm         0.112083735947702 *

     admire

                   * subj:obj       Agent:Agent            0.152353351495087 *
                   * subj:obj       Agent:LifeForm         0.149670971445564 *
                   * subj:obj       LifeForm:Agent         0.146368312962172 *
                   * subj:obj       LifeForm:LifeForm      0.143791307345169 *
                   * subj           LifeForm               0.111411846548935 *
```

The specific frame alternation the *Manner of Speaking* verbs show preferably chooses a living subject only. Alternatively a prepositional *at*-phrase is added, also pointing to a living object:

```
C(2) -- Manner of Speaking

     scream        * subj         LifeForm            0.377416933492826 *
                   * subj         Agent               0.374180270097413 *
                   * subj:pp.at   LifeForm:Agent      0.0285056802388912 *
                   * subj:pp.at   LifeForm:LifeForm   0.0262331496884849 *
                   * subj:pp.at   Agent:Agent         0.0176874737178993 *

     shout         * subj         Agent               0.376252766088892 *
                   * subj         LifeForm            0.372624124441519 *
                   * subj:obj     LifeForm:LifeForm   0.0274048928918853 *
                   * subj:obj     Agent:LifeForm      0.0261478660539234 *
                   * subj:pp.at   Agent:LifeForm      0.0247400464890067 *
```

The verbs of *Ingesting* vary between taking only a subject and adding an object to the subject. In any case the subject is an agent, in case the object appears it is an inanimate object:

```
C(2) -- Ingesting

     drink          * subj:obj        LifeForm:PhysObject   0.248898695359965 *
                    * subj:obj        Agent:PhysObject      0.228022950478341 *
                    * subj            LifeForm              0.152151056363092 *
                    * subj            Agent                 0.145982134304283 *
                    * subj:adv        Agent                 0.0621115942868485 *

     eat            * subj:obj        LifeForm:PhysObject   0.330343558399857 *
                    * subj            LifeForm              0.207481487631302 *
                    * subj:obj        Agent:PhysObject      0.172514864040348 *
                    * subj            Agent                 0.118209426914264 *
                    * subj:adv        LifeForm              0.0502738866426838 *
```

The *Manner of Motion* verbs preferably appear with a subject only, partly followed by an adverb. The subject in both frames is an inanimate object, for *move* it might also be a piece or a group. *roll* and *fly* alternatively use the frame `subj:obj`, preferably with a living entity as subject, followed by an inanimate object:

```
C(3) -- Manner of Motion

     roll           * subj          PhysObject            0.241451670685337 *
                    * subj:adv      PhysObject            0.104624830989344 *
                    * subj:obj      Agent:PhysObject      0.0722786755339997 *
                    * subj:obj      LifeForm:PhysObject   0.0680756190652667 *
                    * subj:obj      Agent:Part            0.0525121359227189 *

     fly            * subj          PhysObject            0.335013432064644 *
                    * subj:adv      PhysObject            0.123622741498 *
                    * subj:obj      LifeForm:PhysObject   0.0657165877759204 *
                    * subj:pp.to    LifeForm:LifeForm     0.0452314211355251 *
                    * subj:pp.to    LifeForm:Agent        0.0438113663530466 *

     move           * subj          PhysObject            0.200321615821647 *
                    * subj:adv      PhysObject            0.11363088866625 *
                    * subj          Part                  0.0925972119246233 *
                    * subj:adv      Group                 0.0442911091963341 *
                    * subj:adv      Part                  0.0395279510615529 *
```

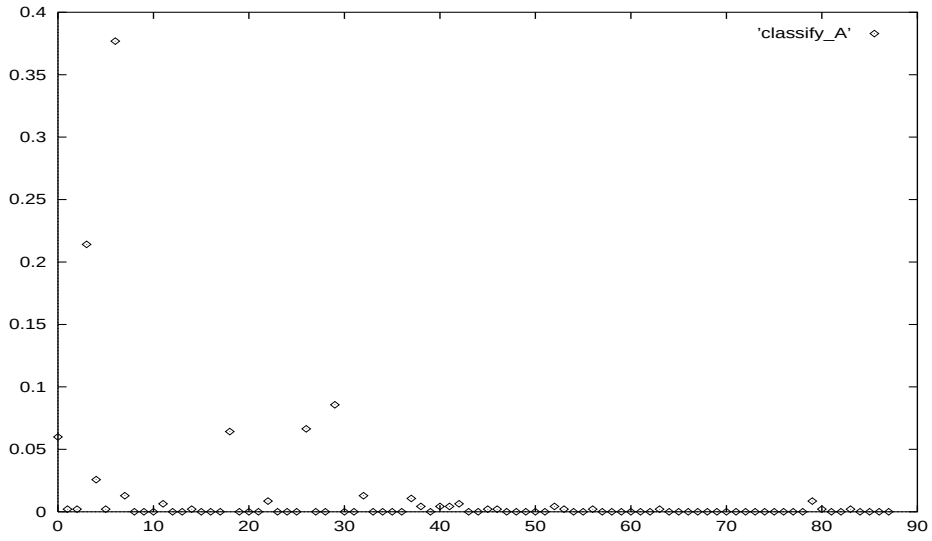
The common tendency in the *Aspect* cluster is the subject frame, represented by an activity:

```
C(2) -- Aspect

     finish         * subj          Action                0.213044306748117 *
                    * subj:obj      Agent:PhysObject      0.190640508016503 *
                    * subj:obj      LifeForm:PhysObject   0.190575345671984 *
                    * subj:obj      Agent:Action          0.0608076134363279 *
                    * subj:obj      LifeForm:Action       0.060786828941483 *

     start          * subj:to       PhysObject            0.272426787458104 *
                    * subj          Action                0.0985457254904005 *
                    * subj:vger     LifeForm              0.0702453716376685 *
                    * subj:vger     Agent                 0.0561457300534095 *
                    * subj:vger     PhysObject            0.0547956330207566 *
```

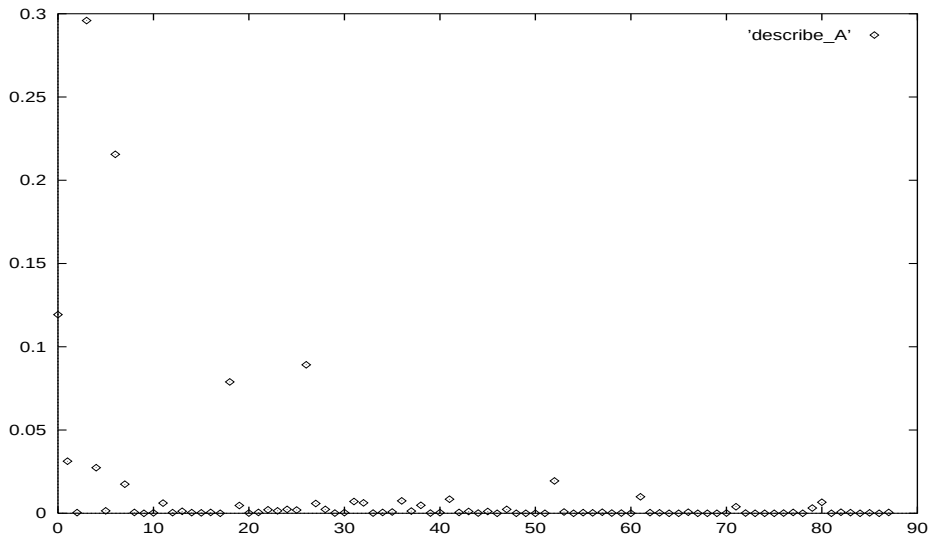
Now we are provided with a rough definition of the correct clusters as obtained for both informational versions. The clusters representing the semantic verb classes clearly show common alternation behaviour of the included verbs. The common ground of the verbs in one the same cluster is justified by a similar usage of (a part of) the five preferred frames and in addition – especially in the cases where these specific frames do not differ strongly to those from other clusters – a similar percentage of preference for these frames and a similar alternation with further frames.

Let me pick out an example to illustrate this dependency of the clusters on the similarity of both the frame types and their probability: the *Characterisation* verbs in version `A` can be distinguished from the verbs in other clusters by only paying attention to the specific choice of frames; but for differentiating the verbs of *Destruction* from those of *Social Interaction*, the percentage of preference has to be taken into account as well as the distribution over further frames, since when considering only the five preferred frames both clusters show a strong preference for the `subj:obj` frame and similar additional preferences for the alternative frames `subj`, `subj:obj:pp.in` and `subj:obj:obj`, so this information is not enough to distinguish the two clusters.

I underline these assumptions with an illustration of the complete distributions concerning the two example clusters. First compare the verbs *classify* and *describe*:



Distribution of the verb *classify* – class: *Characterisation*



Distribution of the verb *describe* – class: *Characterisation*

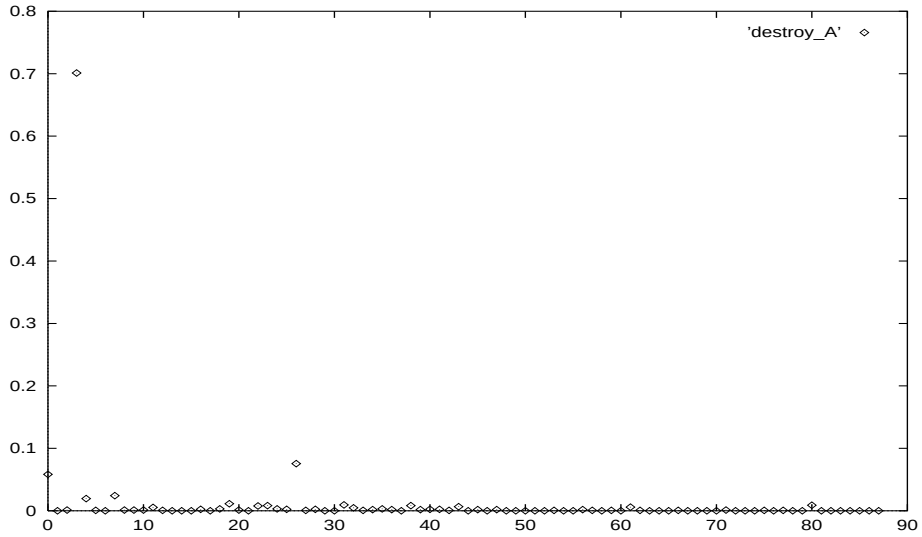
The verbs agree in the striking preferences for the frames `subj`, `subj:obj`, `subj:obj:as`, `subj:obj:pp.as` and `subj:obj:pp.in`. Following are example sentences to describe the usage:

(3.7)

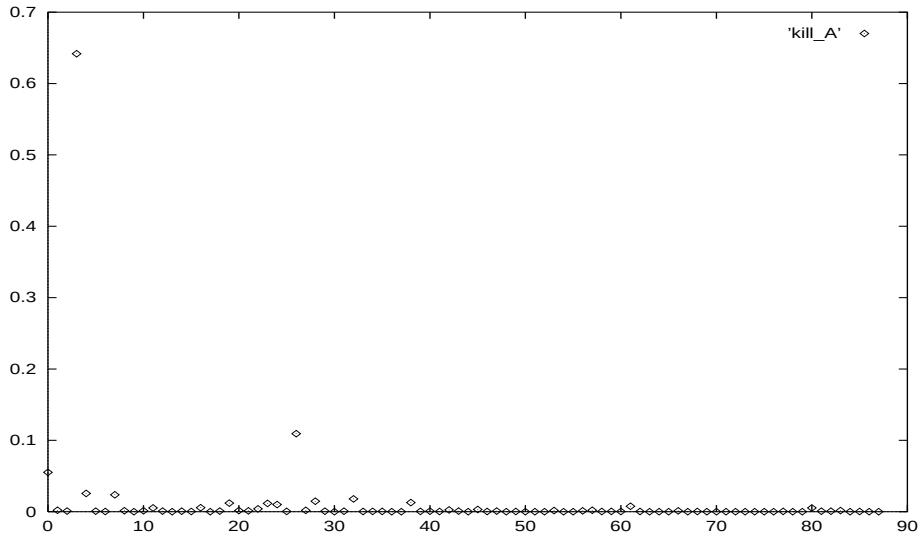
`subj`	The system classifies.
	The tool describes.
`subj:obj`	We classify the topic.
	The chapter describes the procedure.
`subj:obj:as`	Wordsworth classifies it as poem.
	The statement describes the system as impractical.
`subj:obj:pp.as`	The office classified the cobalt as mineral.
	The staff describes Gareth as a stupid boy.
`subj:obj:pp.in`	The system classifies the patients in groups.
	Mandeville described it {in detail / in Stockholm / in 1985}.

The five frames are exactly those frames appearing in the cluster, and they actually justify the similarity of the verbs and delimit them from other verbs. All other frames within the distributions – except for one more frame for *classify* – have probabilities below 0.05.

Now compare the distributions of the verbs *destroy* and *kill*:



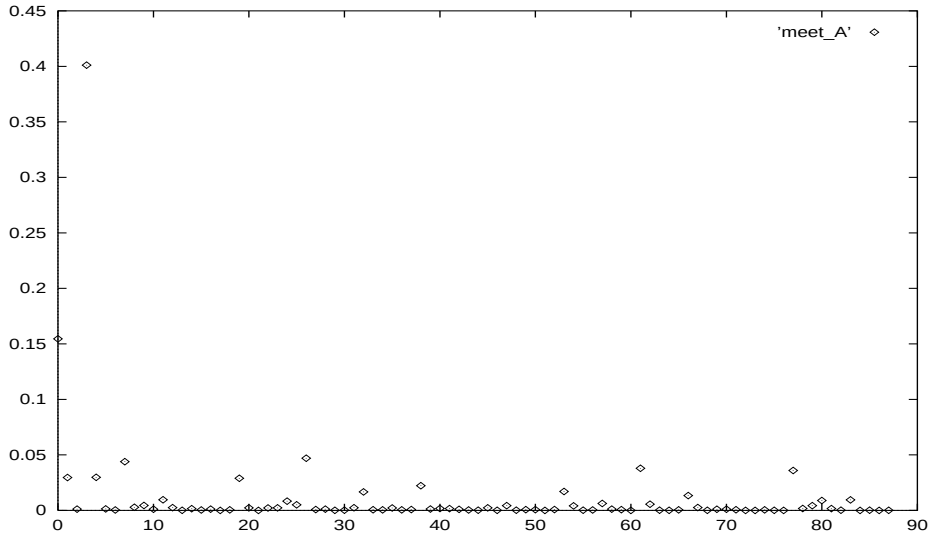
Distribution of the verb *destroy* – class: *Destruction*



Distribution of the verb *kill* – class: *Destruction*

Only the three frames `subj`, `subj:obj` and `subj:obj:pp.in` justify the similarity of the verbs. But compared to the preceding example, the probabilities of these frames are very close, 0.058 and 0.055 for the `subj` frame of the verbs *destroy* and *kill*, respectively, 0.701 and 0.642 for the `subj:obj` frame, and 0.075 and 0.109 for the `subj:obj:pp.in` frame.
Confronting this analysis with the distribution of the verb *meet* shows that

the striking three frames are the same as for the verbs of *Destruction*, but *meet* differs in the probabilities (0.155, 0.401 and 0.047) of these three frames, and it uses a variety of additional frames, for example `subj:obj:obj`, `subj:pp.in` and `subj:pp.with`.



Distribution of the verb *meet* – class: *Social Interaction*

Following are example sentences underlining the common ground and the differences of the three verbs:

(3.8)

`subj`	destroy ?
	David killed.
	Their eyes met.
`subj:obj`	The culture destroyed their souls.
	The policy killed the man.
	The society meets the needs.
`subj:obj:obj`	meet ?
`subj:obj:pp.in`	Russia destroyed the fleet {in the port / in 1957}.
	The gunman killed the people {in the forest / in the end / in 1983}.
	She met him in the pub.
`subj:pp.in`	People meet {in strange situations / in London}.
`subj:pp.with`	The government meets {with the minister / with resistance}.

I have only chosen examples from version `A` to illustrate the common ground of the verbs in the clusters, but the phenomena are the same for version `B`. Additionally, compared to version `A`, version `B` displays the importance of the selectional preferences in the subcategorisation frames. To give an example, take verbs of *Declaration* (*believe, think, suppose*). The three verbs

are clustered because they show common alternation behaviour concerning the frames `subj`, `subj:s` and `subj:that`, so the subcategorisation frames determine the cluster properties. Now compare the correct clusters *Destruction*, containing the verbs *demolish* and *smash*, and *Admiration*, containing the verbs *admire* and *envy*. Both clusters tend to use a transitive frame, so they cannot be differentiated clearly on the basis of the subcategorisation frames only. But including the selectional preferences creates a demarcation between the two clusters: the former includes verbs with an inanimate object, the latter includes verbs with an animate object.
As the examples show, often it is possible to distinguish verbs from others because of their specific syntactic usage of subcategorisation frames. But in other cases it is necessary to add information about the selectional preferences within the frames.

What are the influences onto the clusters? Clearly one issue concerning the clusters is the underlying linguistic model of the data. In section 3.2.2 I illustrated the linguistic reliability of the verbs' representation, their strength and their weaknesses. The observations can be transfered to the current approach, since the underlying data are the same: the distributions representing the verbs and the measures to calculate the distances.

Therefore the clustering algorithm itself determines the difference and thus the success of this method. What are the central issues responsible for the improvement?
One fact concerns the process of merging verbs together. Obviously, the calculation of the clusters' centroids reflects the distances between the verbs and clusters better than only taking the distances between verb-pairs into account. An argument in favour is the following investigation: I wanted to know how many of the verbs in the correct clusters were clustered together with the most similar verb, i.e. is the similarity of verbs preserved in the clusters? With version `A`, 93% of the verbs meet this condition, version `B` decreases this number to 77%. This point illustrates that the verbs in one cluster are not always together with the respective most similar verb, but still the overall success of the algorithm is increased. The mathematical explanation of the phenomenon is as follows: when merging the distributions of the verbs clustered together, instead of the original distribution for each verb the centroid of the cluster was calculated, so it is possible that the centroid was closer to a certain verb than each of its members was and also closer than the originally most similar verb. This is the background of the difference between the two distance clustering algorithms.
The second fact concerns the limit on the size of the clusters. As the clusters resulting from one-step distance clustering show, not limiting the number of

verbs per cluster led to larger (incorrect) clusters containing correct classes. Limiting the number had the disadvantage that correct clusters had to be split into smaller clusters (often clusters with only one element, for example: the verb *jump* was split into a singleton cluster, separated from the cluster containing *fly, run, climb* and *roll*), but the advantages exceed the disadvantage, as the results prove.

Two main problems affected the success of the method:
First, the algorithm could not filter the multiple senses of a verb. The negative sides of this problem are that (a) it was not possible to represent the polysemy of verbs, and (b) as said in section 3.2.2, the presence of polysemy in the representation of the data, that is, the representation of the alternation behaviour merged the alternation behaviour of several verb senses and thereby increased the noise in the representation.
Secondly, verbs with a low frequency falsified the clusters, since they tended to cluster together with verbs with which they only share a single property. The effect was strengthened in version `B`, since the restrictions increased the number of attributes which were left undefined within the distribution of low-frequent verbs. For example, version `B` clusters the low-frequent verbs *rain* and *snow* together with the two verbs *bounce* and *float* belonging to the class of *Manner of Motion* verbs.

The final point to mention is the fact that – as for the one-step distance clustering – the clustering success was worse when adding selectional preferences to the information about the subcategorisation frames. Though the additional information helped to form certain clusters where the differences of the verbs depend on the differences of the selectional preferences (for example, *admire* and *envy* clustered together in version `B`, but not in version `A`), the overall results decreased. As mentioned before, I postpone the discussion about this issue to section 3.2.5, since it concerns all clustering methods.

Summarising the interpretation of the iterative distance method, the resulting clusters are an improvement to those based on one step. Since the underlying data are the same as for the simpler approach, the success must be justified by the different and more sophisticated algorithm.

3.2.4 Latent Class Clustering

Now I turn to describe the latent classes. As for the preceding clusters I start with listing the correct clusters and their assignment to the defined Levin classes. First I describe those 36 out of 80 clusters which were created

on the basis of information about the subcategorisation frames only. Each cluster is defined in a table identified by the number of the cluster – accompanied by the respective class name and the cluster's probability – in the top left corner. The following lines list the verbs with the highest probabilities for that cluster, according to cluster membership and combination with the subcategorisation frames in the columns. The bullets show which of the verbs go with which frames.

The verbs of *Placing* clearly prefer a `subj:obj` frame with a prepositional phrase, determining the *on* and *in*:

Cluster 1 – Placing – PROB 0.0081		0.2989	0.2126	0.0798	0.0403
		subj:obj:pp.on	subj:obj:pp.in	subj:obj	subj:obj:pp.at
0.4481	place	•	•	•	•
0.2035	put	•	•	•	•

The following three clusters represent *Change of Possession* verbs. Cluster 2 contains verbs preferably appearing with a subject and an object, possibly followed by a prepositional phrase headed by *for*:

Cluster 2 – Change of Possession – PROB 0.0103		0.4579	0.2879	0.0563	0.0482
		subj:obj	subj:obj:pp.for	subj:obj:pp.at	subj:obj:pp.in
0.0850	buy	•	•	•	•
0.0830	pay	•	•	•	•
0.0579	offer	•	•	•	•

Differently, cluster 3 prefers a subject only, mainly followed by a prepositional phrase headed by *from*:

Cluster 3 – Change of Possession – PROB 0.0038		0.3688	0.1545	0.0949	0.0335
		subj:pp.from	subj	subj:pp.in	subj:obj:pp.from
0.1830	return	•	•	•	•
0.1269	receive	•	•	•	•
0.0800	gain	•	•	•	•

The strong preference in cluster 4 is in favour of a `subj:obj` frame:

Cluster 4 – Change of Possession – PROB 0.0206		0.7438	0.0541	0.0425	0.0325
		subj:obj	subj:obj:adv	subj:obj:ap	subj:obj:obj
0.4413	get	•	•	•	•
0.1388	leave	•	•	•	•
0.0425	give	•	•	•	•

The following four clusters are as well verbs of *Change of Possession*, but in addition they belong to the more specific sub-class of *Giving* verbs. The two verbs *give* and *offer* show preferences for a `subj:obj` and a `subj:obj:obj` frame:

Cluster 5 – Change of Possession: Giving – PROB 0.0361		0.5477	0.3408	0.0358	0.0190
		subj:obj	subj:obj:obj	subj:obj:pp.for	subj:obj:pp.in
0.3806	give	•	•	•	•
0.1108	offer	•	•	•	•

The clusters 6 and 7 agree in a preference for a `subj:obj` frame and often an additional prepositional phrase. They differ, however, in the kind of preposition they choose. For the verbs *leave* and *supply* in cluster 6 it is the preposition *with*, for *provide*, *offer* and another sense of *supply* it is the preposition *for*:

Cluster 6 – Change of Possession: Giving – PROB 0.0107		0.4656	0.3535	0.0712	0.0238
		subj:obj	subj:obj:pp.with	subj:obj:pp.in	subj:obj:pp.for
0.1414	leave	•	•	•	•
0.0679	supply	•	•	•	•

Cluster 7 – Change of Possession – PROB 0.0094		0.3728	0.1958	0.0838	0.0536
		subj:obj	subj:obj:pp.for	subj:obj:pp.with	subj
0.7820	provide	•	•	•	•
0.0824	offer	•	•	•	•
0.0340	supply	•	•	•	•

The fourth sub-class shows a strong preference for the frame `subj:obj:pp.to`:

Cluster 8 – Change of Possession – PROB 0.0042		0.6306	0.0681	0.0356	0.0351
		subj:obj:pp.to	subj:pp.to	subj:obj:pp.from:pp.to	subj:obj:obj:pp.to
0.2568	transfer	•	•	•	•
0.1739	offer	•	•	•	•
0.1059	pay	•	•		•

One more cluster belongs to the class of *Change of Possession* verbs, to the sub-class *Obtaining*. The verbs mainly alternate between a `subj:obj` frame and that frame with an additional prepositional phrase headed by the preposition *from*:

Cluster 9 – Change of Possession: Obtaining – PROB 0.0095		0.4945	0.1746	0.0587	0.0308
		subj:obj	subj:obj:pp.from	subj:obj:pp.in	subj
0.6843	receive	•	•	•	•
0.0862	collect	•	•	•	•
0.0554	buy	•	•	•	•

Following are two clusters containing verbs of *Creation*. Cluster 10 strongly prefers the `subj:obj` frame, possibly followed by a prepositional phrase indicating the *in*:

Cluster 10 – Creation – PROB 0.0041		0.3644	0.0990	0.0328	0.0327
		subj:obj	subj:obj:pp.in	subj:obj:pp.into	subj:obj:pp.for
0.6474	build	•	•	•	•
0.1591	develop	•	•	•	•
0.0611	construct	•	•		•

Cluster 11 chooses an additional prepositional phrase headed by *for*, or an additional object:

Cluster 11 – Creation – PROB 0.0014		0.2755	0.1264	0.0921	0.0737
		subj:obj:pp.for	subj:obj:obj	subj:obj	subj
0.4927	create	•	•	•	•
0.0570	develop	•	•	•	•

The following five clusters contain verbs of *Declaration* in combination with different alternating subcategorisation frames. The verbs *find* and *show* in cluster 12 demand a subject plus an object, preferably with an additional prepositional phrase headed by *in*:

Cluster 12 – Declaration – PROB 0.0176		0.4797	0.3320	0.0351	0.0221
		subj:obj:pp.in	subj:obj	subj	subj:obj:obj:pp.in
0.3011	find	•	•	•	•
0.1218	show	•	•	•	•

In the following cluster the additional argument is an adjectival phrase:

Cluster 13 – Declaration – PROB 0.0169		0.2683	0.2651	0.0729	0.0669
		subj:obj:ap	subj:obj	subj:that	subj:obj:to
0.9362	find	•	•	•	•
0.0352	declare	•	•	•	•

The verbs *suppose* and *think* alternate between demanding a subject only and demanding a sentence (which is the preferred use) or an object plus infinitival phrase in addition:

Cluster 14 – Declaration – PROB 0.0141		0.4087	0.2490	0.2145	0.0456
		subj:s	subj	subj:obj:to	subj:adv
0.4243	suppose	•	•	•	•
0.3395	think	•	•	•	•

The following verbs are similar in two of these frames, but with differing preferences:

Cluster 15 – Declaration – PROB 0.0071		0.3916	0.2497	0.1770	0.0359
		subj:s	subj:obj:to	subj:pp.in	subj:obj
0.5741	believe	•	•	•	•
0.2195	find	•	•	•	•
0.0610	see	•	•	•	•
0.0462	show	•	•	•	•

announce and *declare* mainly demand a prepositional phrase headed by *on*, either directly combined with a subject and followed by a *that*-phrase, or following an object:

Cluster 16 – Declaration – PROB 0.0038		0.1440	0.1060	0.0903	0.0815
		subj:obj:pp.on	subj:pp.on:that	subj	subj:obj:pp.in
0.8767	announce	•	•	•	•
0.0858	declare	•	•	•	•

The verbs of *Characterisation* correspond in the preferred use of an *as*-phrase in addition to a `subj:obj` frame:

Cluster 17 – Characterisation – PROB 0.0133		0.3913	0.1672	0.1569	0.0586
		subj:obj:as	subj:obj:pp.as	subj:obj	subj:obj:pp.in
0.4879	see	•	•	•	•
0.3230	describe	•	•	•	•
0.0578	identify	•	•	•	•

The verbs of *Perception* typically appear with a `subj:obj` frame or that frame with an additional verb in base-form:

Cluster 18 – Perception – PROB 0.0119		0.3255	0.1398	0.0673	0.0635
		subj:obj	subj:obj:vbase	subj:s	subj:obj:vger
0.7282	hear	•	•	•	•
0.2275	see	•	•	•	•
0.0173	feel	•	•	•	•

Some verbs of *Perception* prefer a subject only, mostly accompanied by an adjectival phrase, partly by a prepositional phrase headed by *like*:

Cluster 19 – Perception – PROB 0.0108		0.2818	0.1903	0.0854	0.0653
		subj:ap	subj	subj:pp.like	subj:sub
0.9620	feel	•	•	•	•
0.0108	smell	•	•	•	

The verbs of *Admiration* strongly prefer a `subj:obj` frame, possibly exchanged by a subject followed by an infinitival phrase:

Cluster 20 – Admiration – PROB 0.0035		0.6517	0.1398	0.0539	0.0285
		subj:obj	subj:to	subj:obj:obj	subj:pp.about
0.7632	like	•	•	•	•
0.0768	love	•	•	•	•

The class of *Desire* verbs, asking for a subject either accompanied by an infinitival phrase or an object, is represented by the two verbs *want* and *need*:

Cluster 21 – Desire – PROB 0.0271		0.4972	0.2408	0.0835	0.0669
		subj:to	subj:obj	subj:obj:to	subj
0.5992	want	•	•	•	•
0.1983	need	•	•	•	•

The two clusters of *Social Interaction* verbs clearly distinguish two senses, with cluster 22 showing a strong preference for a `subj:obj` frame, and cluster 23 emphasising the prepositional phrase headed by *against*, in addition to either a subject only, or a subject plus an object:

Cluster 22 – Social Interaction – PROB 0.0095		0.5545	0.0468	0.0366	0.0340
		subj:obj	subj	subj:obj:pp.with	subj:obj:pp.at
0.4947	meet	•	•	•	•
0.1954	play	•	•	•	•

Cluster 23 – Social Interaction – PROB 0.0018		0.1829	0.1297	0.0894	0.0693
		subj:pp.against	subj:obj	subj:obj:pp.against	subj:obj:adv
0.2212	fight	•	•	•	•
0.1959	play	•	•	•	•

The following eight clusters contain verbs of *Telling*. The verbs in these clusters combine in different ways, depending on the preferred subcategorisation frames.

Cluster 24 has an almost exclusive preference for the `subj:obj` frame:

Cluster 24 – Telling – PROB 0.0442		0.9700	0.0161	0.0043	0.0018
		subj:obj	subj:obj:obj	subj:obj:pp.by	subj:obj:pp.during
0.0840	tell	•	•	•	•
0.0613	show	•	•	•	•

In the following cluster the verbs of *Telling* take a subject only, possibly accompanied by a *that*-phrase:

Cluster 25 – Telling – PROB 0.0368		0.6635	0.1271	0.0580	0.0392
		subj	subj:that	subj:obj	subj:adv
0.8150	say	•	•	•	•
0.0750	suggest	•	•	•	•
0.0583	explain	•	•	•	•
0.0105	write	•	•	•	•

In this cluster the sense of the verb *say* is paired with *read*, preferably asking for a subject only or a subject with an additional object:

Cluster 26 – Telling – PROB 0.0257		0.5694	0.3400	0.0223	0.0126
		subj	subj:obj	subj:adv	subj:obj:obj
0.3770	say	•	•	•	•
0.0548	read	•	•	•	•

Following *say* is paired with *write*, with similar preferences to the preceding cluster, but possibly alternating with a prepositional phrase headed by *in* in addition to either a subject only or a subject and an object:

Cluster 27 – Telling – PROB 0.0211		0.4474	0.4082	0.0711	0.0253
		subj	subj:obj	subj:pp.in	subj:obj:pp.in
0.1710	say	•	•	•	•
0.1331	write	•	•	•	•

Compare the clustering with the combination of *say* with *show*, where you can also find the `subj` and `subj:obj` frames, but the preference lies on an infinitival phrase following the latter:

Cluster 28 – Telling – PROB 0.0081		0.5664	0.2090	0.1185	0.0600
		subj:obj:to	subj	subj:obj	subj:obj:pp.of
0.1930	say	•	•	•	•
0.1502	show	•	•	•	•

Some verbs of *Telling* were clustered according to their similar use of a `subj:obj` frame alternatively followed by a *that*-phrase:

Cluster 29 – Telling – PROB 0.0206		0.5320	0.1978	0.0605	0.0525
		subj:obj	subj:obj:that	subj:obj:to	subj:obj:pp.about
0.8574	tell	•	•	•	•
0.0317	warn	•	•	•	•
0.0215	teach	•	•	•	•
0.0197	advise	•	•	•	•

A similar group of verbs strongly prefers an infinitival phrase in addition to a `subj:obj` frame:

Cluster 30 – Telling – PROB 0.0040		0.7455	0.0857	0.0482	0.0158
		subj:obj:to	subj	subj:obj	subj:pp.on
0.1734	advise	•	•	•	•
0.1213	teach	•	•	•	•
0.1198	instruct	•	•	•	•

And the last sub-class chooses a prepositional phrase headed by *about* either in combination with only a subject or with a `subj:obj` frame:

Cluster 31 – Telling – PROB 0.0016		0.1718	0.1351	0.1220	0.0557
		subj:pp.about	subj:obj:pp.about	subj:pp.in	subj:obj:pp.to
0.5014	write	•	•	•	•
0.4258	read	•	•	•	•
0.0181	explain	•	•	•	•
0.0087	teach	•	•	•	•

The verbs of *Lodging / Existence* prefer, next to their subject, a prepositional phrase headed by *in*, or an adverb:

Cluster 32 – Lodging / Existence – PROB 0.0118		0.3107	0.1173	0.0872	0.0546
		subj:pp.in	subj:adv	subj	subj:pp.with
0.4857	live	•	•	•	•
0.1415	stay	•	•	•	•

The verbs restricted to the *Lodging* class prefer a subject only, possibly accompanied by a prepositional phrase headed by *at*:

Cluster 33 – Lodging – PROB 0.0052		0.3716	0.1678	0.0946	0.0489
		subj	subj:pp.at	subj:adv	subj:pp.in
0.3287	stop	•	•	•	•
0.2140	stay	•	•	•	•

The verbs of *Manner of Motion* demand a subject as argument, possibly accompanied by an adverb, a prepositional phrase with *to*, or an object:

Cluster 34 – Manner of Motion – PROB 0.0100		0.1372	0.1005	0.0855	0.0720
		subj	subj:adv	subj:pp.to	subj:obj
0.9145	move	•	•	•	•
0.0247	fly	•	•	•	•

Comparing the two clusters of verbs of *Aspect*, cluster 35 preferably chooses a gerund in addition to the subject, whereas cluster 36 prefers a prepositional phrase, either headed by *with* or by *in*:

Cluster 35 – Aspect – PROB 0.0092		0.3224	0.0636	0.0626	0.0482
		subj:vger	subj:pp.with	subj:pp.at	subj
0.6376	start	•	•	•	•
0.3310	begin	•	•	•	•
0.0235	stop	•	•	•	•

Cluster 36 – Aspect – PROB 0.0069		0.2198	0.1444	0.0748	0.0746
		subj:pp.with	subj:pp.in	subj:adv	subj
0.3625	begin	•	•	•	•
0.2866	end	•	•	•	•
0.0972	continue	•	•	•	•

Following you can find the same kind of information about the 22 out of 80 correct clusters based on the relationship between verbs and the subcategorisation frames plus their selectional preferences.

The first three clusters contain verbs belonging to the class *Change of Possession*. Though they belong to the same class, they strongly differ in the preferred selection of their subcategorisation frames and selectional preferences. Cluster 1 preferably appears with a subject representing either a location or a group:

Cluster 1 – Change of Possession – PROB 0.0167		0.1065	0.0672	0.0603	0.0565
		subj::Location	subj::Group	subj:obj:pp.to::Group:PhysObject:Group	subj:obj:pp.with::Group:LifeForm:PhysObject
0.4999	supply	•	•	•	•
0.3310	acquire	•	•		
0.0763	offer	•	•	•	

The verbs in cluster 2 only agree in the usage of a subject only, indicating a group:

Cluster 2 – Change of Possession – PROB 0.0111		0.1365	0.1301	0.0947	0.0792
		subj:obj:pp.for::Agent:Possession:PhysObject	subj:obj:obj::Group:Possession:Possession	subj::Group	subj:obj:pp.to::Group:Possession:Agent
1.0000	pay	•	•	•	•
0.0000	receive			•	
0.0000	collect			•	

Cluster 3 prefers a group as subject and a possession in the role of the object within a `subj:obj` frame, possibly accompanied by a prepositional phrase headed by *to* and indicating a group:

Cluster 3 – Change of Possession – PROB 0.0067		0.2157	0.1647	0.1180	0.0983
		subj:obj::Group:Possession	subj:obj:pp.to::Group:Possession:Group	subj:obj:obj::Group:Agent:PhysObject	subj:obj:pp.to::Group:Abstract:Group
0.8257	allocate	•	•	•	•
0.0707	offer	•	•	•	•
0.0564	receive	•			
0.0422	transfer	•	•		

The following two clusters also contain verbs of *Possession*, but belong to the more specific sub-class of *Giving*. The verbs in cluster 4 agree only in using a `subj:obj:pp.to` frame with a group as subject, an activity as object and an agent as object within the *to*-phrase:

Cluster 4 – Change of Possession: Giving – PROB 0.0094		0.1821	0.0974	0.0914	0.0800
		subj:pp.with::Group:Action	subj:obj:pp.with::Agent:Action	subj:obj:pp.with::LifeForm:Agent:Action	subj:obj:pp.to::Group:Action:Agent
0.7858	entrust	•	•	•	•
0.2141	offer				•
0.0002	transfer				•

Cluster 5 preferably demands an abstract or inanimate subject combined with an abstract object:

Cluster 5 – Change of Possession: Giving – PROB 0.0063		0.1100	0.0930	0.0681	0.0672
		subj:obj::Abstract:Abstract	subj:obj::PhysObject:Abstract	subj:obj:obj::Psycho:Agent:Action	subj:obj:obj::Psycho:LifeForm:Action
0.8753	give	•	•	•	•
0.1089	offer	•	•	•	•
0.0158	provide	•	•		
0.0000	guarantee	•	•	•	•

There are two more clusters belonging to the *Change of Possession* verbs, this time to the sub-class of *Obtaining*. The verbs in cluster 6 agree in demanding an abstract subject only:

Cluster 6 – Change of Possession: Obtaining – PROB 0.0254		0.1429	0.0719	0.0675	0.0535
		subj::Abstract	subj:obj:pp.for::Group:PhysObject:Psycho	subj:pp.through::PhysObject:Location	subj:obj:pp.from::Group:PhysObject:Agent
0.3707	purchase	•	•		•
0.2667	gain	•			

The verbs in cluster 7 agree only in the use of a `subj:obj:pp.from` frame where the subject and the source within the prepositional phrase are agents, the object represents an inanimate entity:

Cluster 7 – Change of Possession: Obtaining – PROB 0.0097		0.1496	0.1083	0.0994	0.0938
		subj::Possession	subj:obj:pp.in::Agent:PhysObject:Agent	subj:obj:pp.for::LifeForm:PhysObject:Agent	subj:obj:pp.from::Agent:PhysObject:Agent
0.9992	buy	•	•	•	•
0.0008	collect	•			•
0.0000	purchase		•		•

The verbs of *Removing* demand an abstract subject and an object representing a state:

Cluster 8 – Removing – PROB 0.0122		0.0893	0.0768	0.0618	0.0391
		subj:obj:pp.from::Action:State:Action	subj:obj:::Action:State	subj:pp.from::PhysObject:Abstract	subj:obj:::Abstract:State
0.3780	extract			•	
0.3023	eliminate	•	•		•

Clusters 9 and 10 contain verbs of *Creation*. Within cluster 9 the verbs agree in an agentive subject, either followed directly by an inanimate entity, or followed by an agentive object and an inanimate entity:

Cluster 9 – Creation – PROB 0.0156		0.1249	0.0933	0.0650	0.0636
		subj:pp.into::PhysObject:PhysObject	subj:obj:pp.in:::Abstract:PhysObject:PhysObject	subj:obj:obj::Agent:Agent:PhysObject	subj:obj:::Agent:PhysObject
0.5430	pour	•		•	•
0.3129	cook		•	•	•

In cluster 10 the two verbs agree in a psychological feature as subject:

Cluster 10 – Creation – PROB 0.0128		0.1433	0.0955	0.0813	0.0802
		subj::Psycho	subj::Location	subj:obj:obj::Agent:Abstract::PhysObject	subj:obj:pp.from::LifeForm:PhysObject:PhysObject
0.6212	invent	•	•	•	
0.3503	collect	•			•

The verbs of *Declaration* agree in the use of a groupal subject, possibly followed by an object indicating an activity or a psychological feature:

Cluster 11 – Declaration – PROB 0.0136		0.1949	0.1585	0.1510	0.0948
		subj:that::Group	subj:obj::Group:Psycho	subj::Group	subj:obj::Group:Action
0.4578	announce	•	•	•	•
0.1649	propose		•	•	•
0.1513	declare	•	•	•	•

Two clusters contain verbs of *Telling*. The verbs in cluster 12 preferably demand an agentive subject only, mostly followed by a *that*-phrase; concerning the verbs in cluster 13, the subject is possibly followed by an agentive object and an infinitival phrase:

Cluster 12 – Telling – PROB 0.0248		0.3681	0.3650	0.1342	0.1233
		subj:that::LifeForm	subj:that::Agent	subj::Agent	subj::LifeForm
0.1097	confess	•	•	•	•
0.1027	explain	•	•	•	•

Cluster 13 – Telling – PROB 0.0141		0.2088	0.1986	0.0820	0.0794
		subj::Agent	subj::LifeForm	subj:obj:to::Agent:Agent	subj:obj:to::Agent:LifeForm
0.2970	instruct	•	•	•	•
0.2703	advise	•	•	•	•
0.2182	warn	•	•	•	•

The next two clusters contain verbs of *Characterisation*. They agree in choosing an abstract subject and object, followed by an *as*-phrase:

Cluster 14 – Characterisation – PROB 0.0111		0.1667	0.1267	0.0822	0.0476
		subj:obj:pp.as::Action:PhysObject:PhysObject	subj:obj:pp.into::Abstract:Psycho:Group	subj::Abstract	subj:obj:as::Abstract:Abstract
1.0000	classify	•	•	•	•
0.0000	characterize				•

Cluster 15 differs only in the choice of an inanimate entity as object:

Cluster 15 – Characterisation – PROB 0.0098		0.1363	0.0831	0.0811	0.0579
		subj:obj:as::Abstract:PhysObject	subj::Psycho	subj:obj:pp.with::Group:Agent:Group	subj::Abstract
0.9412	identify	•	•	•	•
0.0588	characterize	•			

The most probable common frame for the verbs of *Assessment* is an activity in the subject role, accompanied by an abstract object and an agent in the role of a prepositional phrase headed by *in*:

Cluster 16 – Assessment – PROB 0.0136		0.0996	0.0525	0.0440	0.0413
		subj:obj:pp.on::Psycho:Possession:Abstract	subj:obj:pp.in::Action:Abstract:Agent	subj:obj::Psycho:Psycho	subj:obj::Psycho:Abstract
0.4072	assess	•	•	•	•
0.2713	evaluate		•	•	•

The verbs of *Perception* have an agentive subject, preferably followed by a sentence:

Cluster 17 – Perception – PROB 0.0195		0.2860	0.2826	0.1389	0.1352
		subj:s::LifeForm	subj:s::Agent	subj::Agent	subj::LifeForm
0.2293	notice	•	•	•	•
0.1544	feel	•	•	•	•

The verbs *stay* and *live* belong to the classes *Lodging* and *Existence*, mainly used with an agentive subject and an adverb:

Cluster 18 – Lodging / Existence – PROB 0.0119		0.0730	0.0713	0.0611	0.0608
		subj:adv::LifeForm	subj:adv::Agent	subj:ap::LifeForm	subj:ap::Agent
0.9989	stay	•	•	•	•
0.0011	live	•	•		

The following two verbs appear both in the two clusters of *Sliding* and *Manner of Motion*. They preferably appear with an inanimate entity as subject, possibly followed by an adverb:

Cluster 19 – Sliding / Manner of Motion – PROB 0.0143		0.2140	0.1671	0.0592	0.0558
		subj::PhysObject	subj:adv::PhysObject	subj:obj::Agent:PhysObject	subj:obj::LifeForm:PhysObject
0.6292	slide	•	•	•	•
0.3652	roll	•	•	•	•

Following are two clusters whose verbs only belong (differently to the preceding, more specific sub-class) to the *Manner of Motion* verbs. *move* and *fly* mainly have an inanimate or grouped subject, followed by an adverb:

Cluster 20 – Manner of Motion – PROB 0.0135		0.1742	0.1156	0.0468	0.0464
		subj:adv::PhysObject	subj:adv::Group	subj:pp.to::LifeForm:LifeForm	subj:pp.to::LifeForm:Agent
0.4005	move	•	•	•	•
0.3556	fly	•		•	•

jump and *climb*, however, prefer an agentive subject followed by a prepositional phrase headed by *into* and defined by an inanimate entity:

Cluster 21 – Manner of Motion – PROB 0.0107		0.2209	0.1255	0.0668	0.0467
		subj:pp.into::LifeForm:PhysObject	subj:pp.into::Agent:PhysObject	subj:pp.to::Possession:PhysObject	subj:obj::Action:PhysObject
0.5130	jump	•	•	•	•
0.4407	climb	•	•	•	

The verbs of *Aspect* alternate between a subject only, realised by an activity, an inanimate subject followed by an infinitival phrase, and an living subject followed by a gerund:

Cluster 22 – Aspect – PROB 0.0208		0.2203	0.1032	0.0942	0.0863
		subj::Action	subj:to::PhysObject	subj:vger::LifeForm	subj:vger::Agent
0.3382	start	•	•	•	•
0.1945	finish	•		•	•
0.1846	stop	•		•	•
0.1584	begin	•	•		

The clusters illustrate the relation between the verbs' alternation behaviour and their semantic classes; the verbs in one cluster overlap in the usage of the preferred subcategorisation frames. The overlap in version `A` is generally larger than in version `B` – noticeable by the more regular numbers of bullets in the verb-frame matrix –, since the frame specification is less specific. It is obvious that, differently to the distance clustering, only a partial overlap of all frame types a verb uses is necessary for verbs clustering together, since some verbs appear in multiple clusters with a different choice of preferred subcategorisation frames, representing the multiple verb senses.
Consider, for example, the class of *Social Interaction* verbs in version `A`, wherein two sub-classes were automatically created, one containing the verbs *meet* and *play*, the other the verbs *fight* and *play*. Investigating the respective subcategorisation frames, the sub-class *{meet, play}* shows a strong preference for a transitive use; the sub-class *{fight, play}* also possibly uses a transitive frame, but tends to include a prepositional phrase headed by *against*. Disregarding further possibilities, following are example sentences illustrating these uses:

(3.9)

`subj:obj`	She meets her grandson.
	Concentration plays an important role.
`subj:obj`	England plays Pakistan.
	Tarzan fights a lion.
`subj:pp.against`	The woman fights against the supremacy.
	They play against a superior opponent.
`subj:obj:pp.against`	The applicant fights a battle against the authorities.
	England plays a tournament against the USA.

Obviously, *play* is clustered with *meet* illustrating a general meeting, and it is clustered with *fight* when illustrating a more aggressive meeting like a match or a fight.
The negative side of the possibility to express multiple senses of a verb is the over-interpretation of the senses' variability. The verbs' senses were determined by filtering the different kinds of alternation behaviour out of the overall distribution. By over-interpreting them too many combinations were deduced. For example, the verb *place* in version `B` was assigned to nine different clusters, representing nine different senses based on the demand for the alternation of the frame types.

The additional information about the selectional preferences version `B` provided helps to disambiguate the subcategorisation frames concerning the arguments' concepts. For example, version `A` clustered the verb *assess* together with the verbs *explain* (class *Telling*), *describe* (class *Characterisation*) and *analyse* (same class *Assessment*), since all show a strong preference for a transitive frame. In version `B`, *assess* was clustered together with *evaluate* which was not clustered at all in version `A`. Both verbs agree – in one sense – in a transitive frame with an additional prepositional phrase headed by *in*. Concerning the concepts, the subject is held by an activity, the object by an abstraction, and the prepositional phrase points to a living entity. The following two sentences illustrate this usage:

(3.10)

`subj:obj:pp.in`	The research assesses the effect in patients.
`::Action:Abstract:Agent`	The report evaluates the risk in patients.

A further point to mention concerning the information within the clusters is the probability value accompanying the cluster. First, we are provided with a probability for the cluster itself, secondly with a probability for the verbs being member of that cluster, and in the third place with a probability for the subcategorisation frames being member of that cluster. The probabilities are an interesting additional source concerning how definite a verb or a type is member of a cluster. For example, a cluster containing the verbs *classify, characterize, provide* and *gain* determines a probability of 1 for *classify* and 0 for the other three verbs. How should one consider such a cluster? As a cluster containing only one verb? Or as a cluster containing four verbs with different (and by chance mostly zero) probabilities? I do not interpret the probabilities, which equalises the interpretation with the latter possibility.

Investigating the linguistic reliability of the data fed into the clustering algorithm, we must distinguish between the two versions `A` and `B`: in version `A`

I used the frequencies of the subcategorisation frame types appearing with the verbs, so the data was even more pure than the maximum likelihood estimate when clustering by distance. In version `B` we prepared the data in the same way as for the distance clustering approaches, since no frequencies for the specifically defined frame-preference combinations were available.
As a whole, the linguistic representation, its strength and its weaknesses, is strongly comparable to that described in section 3.2.2. As said before, the difference in accuracy of the linguistic representation in both versions is viewed as main cause for the difference in precision. But this discussion is postponed to subsection 3.2.5.

The peculiarities of this approach do not lie in the data, however, but in the algorithm: latent classes are not built on the definitions and differences of the verbs' overall distributions, but model the positive, i.e. available, data. So the similarity of the verbs within one cluster is justified by a partial overlap within the distributions over subcategorisation frames. If there is evidence for the co-occurrence of a verb and certain subcategorisation frames, the verb will appear in the same cluster with other verbs showing evidence for the same frames. It is possible to distinguish different verb senses, since a verb might appear with different sets of frame types, representing different senses. Concerning the above example, the verb *meet* has a partial overlap with the distribution of the verb *play*, and a partial overlap with the distribution of the verb *fight*, so it appears in both clusters, illustrating two different senses.

An overall problem seems to be the problem of data sparseness. Concerning the fact that only a certain small percentage of the frames appeared with each verb was not the relevant part of the problem, since the latent class algorithm modelled the data which actually appeared. But in total there were only 6,873 verb-frame types for version `B` which was a narrow basis for training. For version `A` I had 27,016 verb-frame types, but differently to `B` only 88 different frames, so creating 80 different clusters had the tendency to result in some classes where only one frame was favoured.
As a result, low frequent verbs increase the noise in the clusters. For example, the verb *rain* which altogether appeared only 460 times, is found in six different clusters in version `B`. By smoothing the frequencies all formerly zeroes were changed to 0.5, which resulted in an almost uniform distribution concerning the probabilities, so the verb had partial overlap in the distribution with several verbs and was therefore assigned to several clusters.

Summarising the overall representation of clusters achieved by latent classes, the method is similarly successful in illustrating the relationship between the alternation behaviour and the semantic classes of the verbs than iterative

distance clustering. In addition, the latent classes are able to distinguish the different senses of a verb.

3.2.5 General Interpretation

This final part of the interpretation is concerned with a summary of the preceding parts. In the subsections 3.2.2 to 3.2.4 I described and interpreted the clusters resulting from the application of the different clustering approaches. Now I summarise these insights to a general interpretation. For that, I concentrate on the results from *Iterative Distance Clustering* and *Latent Class Clustering* and refer to the simpler approach for explanations.

The classifications of both approaches illustrate the close relationship between the verbs' alternation behaviour and their affiliation to semantic classes: the resulting clusters which can be annotated by semantic class names show common alternation behaviour of their verbal elements. The two to four verbs united in a cluster agree in the usage of a certain set of subcategorisation frames. Sometimes the demarcation of one cluster is justified by the common usage of only one frame type (especially in the `B` versions), in which case the frequency/probability of using this type is nearly identical concerning the included verbs. But mostly the common usage is justified by more than one frame type, in which case the probabilities of using these types may be less similar, since the similarity of the verbs is established by the range of frames.
Both approaches show that the relationship between alternation behaviour and semantic class can already be established when only considering information about the syntactic usage of the subcategorisation frames (versions `A`). The refinement by the frames' selectional preferences allows further demarcations by the identification of conceptual restrictions on the use of the frames. With the information obtained in version `B` it is possible to deduce a precise definition of the verbs' usage and the typical semantic background of its arguments, which is especially useful when distinguishing between the different semantic roles (location, source, etc.) described by prepositional phrases.
An advantage of the latent classes is the further distinction into verb senses. Instead of correlating verbs with their alternation behaviour the semantic classes distinguish between the different verbs' senses and the respective uses of subcategorisation frames.

In subsection 3.2.2 an investigation of the linguistic reliability of the verbs' and clusters' subcategorisation frames showed that the characterising usages

can actually be underlined by example sentences. This means that the linguistic properties as modelled for the approaches agree with (a part of) the verbs' properties. The clusters were therefore created on a reliable linguistic basis, an important fact to ensure, since an unreliable representation would question the successful relation between alternation behaviour and semantic classes.
The quality of the linguistic basis must be differentiated concerning the two informational versions, though. Concerning version `A` there was little noise in the descriptions of the verbs' subcategorisation frames. Concerning version `B` the problems increased. Since the increase of noise correlates with the decrease of precision concerning the clustering success, this seems an important factor to investigate: considering each argument slot within a subcategorisation frame on its own, the preferred conceptual classes illustrate linguistic reliable possibilities to insert arguments. But by the combination of the classes too many combinatorial possibilities were created, so the combinations are not always possible to underly with examples. In addition, the conceptual classes do not necessarily rely on a certain subcategorisation frame; for example the subject role might be a living entity independent on the frame types `subj`, `subj:obj`, etc. The solution to this problem seems to be a different formulation of the conceptual classes like: a representation where only those parts of the subcategorisation frames which depend on each other are combined and specialised by conceptual classes. For example, for the mainly used subcategorisation frames of the verb *give*, `subj:obj:obj` and `subj:obj:pp.to`, I would formulate conceptual classes for `subj` – the subject in both frames –, the pair `obj:obj`, and the pair `obj:pp.to` – because those roles depend on each other, since this is where the diathesis actually takes place. This procedure would require a preliminary investigation of the importance of such functional combinations realising the diathesis alternation.

A further issue to investigate is the applicability of the two algorithms. Comparing the iterative distance clustering with the simple, one-step approach, I obviously succeeded in the idea and formulation of the method. Considering the underlying information as illustrated by the pointers in subsection 3.2.2, the algorithm allowed to find out and model the relevant distances between the verbs in order to cluster them together. There is no guarantee that this was the optimal solution (as mentioned in section 2.3.1), but it was a successful start into the right direction. The clustering on basis of latent classes was slightly less successful, but one should keep in mind that this algorithm is generally able to distinguish between the different verb senses – an essential feature –, and it needs less manual preparation and restrictions to work. In addition, the evaluation basis for this approach is different because of the

sense distinction, so it is difficult to directly compare the results with those obtained by distance clustering.
The algorithms are confronted with two main problems:

- Polysemy:
  The different verb senses are hidden in the representation for one verb. That is, it is not obvious how to filter the uncertain number of senses out of the word-form. The iterative distance clustering completely failed to model verb senses; a polysemous verb was because of its opaque representation either not at all assigned to a cluster, or assigned to one cluster to which one of the verb's senses belongs. The latent class analysis was able to filter the multiple senses and assign them to distinct clusters, but tended to over-interpret.

- Low Frequency:
  Verbs which rarely appear were difficult to cluster, since the necessary background is missing. A latent class analysis suffered from this sparse data, since those verbs were always assigned low probabilities. Distance clustering suffered even more, since – in addition to the sparse data concerning the verb's usage – also the information about the co-occurrence with subcategorisation frames was missing, so the verb's distribution contained mostly zeroes, a difficult mathematical basis.

Having interpreted the results of the clustering approaches I now come to an investigation of the underlying standard for the success: Levin's class definitions. Throughout this section, I have listed the correct part of the clusters resulting from the different approaches, interpreted them and looked for explanations. As standard for the evaluation I chose Levin's classes, as explained in subsection 2.3.2, one possibility to judge about the correctness of the clusters. I still think that this basis is a standard to judge about the applicability and usefulness of the clustering approaches, but I nevertheless want to mention the fact that Levin's classes are a standard whose final decisions about class membership are based on subjective judgement, concerning the number and kind of different senses a verb can have, or the importance of the properties a verb has, when assigning that verb to a certain class. So one should have in mind that Levin's definitions are not the ultimatum; I cite some concrete examples which illustrate the influence of subjectiveness. For a separate – and not my own – opinion I consulted WordNet about hierarchical relationships between verbs.

- The resulting clusters from iterative distance clustering contain the following pair:

```
C(2) :   propose      * subj:obj          0.308364758313336 *
                      * subj              0.185978745286253 *
                      * subj:to           0.113986972917381 *
                      * subj:that         0.0834761741515255 *
                      * subj:obj:pp.in    0.0313678436750086 *

         promise      * subj:to           0.254433185560481 *
                      * subj:obj          0.237333755541482 *
                      * subj              0.213584547181761 *
                      * subj:obj:obj      0.0660227992400253 *
                      * subj:that         0.0391070297656745 *
```

  Both verbs alternate between an intransitive use, partly accompanied by an infinitival phrase or a *that*-phrase, and a transitive use. Consulting WordNet shows that both verbs are sub-ordinated to the verb *declare*. So there should be a connection between the verbs by uniting them, for example in the cluster of *Declaration*.

- The same method created a cluster containing the following four verbs:

```
C(4) :   put          * subj:obj            0.171394485683987 *
                      * subj:obj:pp.on      0.140111346765642 *
                      * subj:obj:pp.in      0.137725344644751 *
                      * subj:obj:adv        0.0962796041003888 *
                      * subj:obj:pp.into    0.067294096854012 *

         throw        * subj:obj:pp.into    0.167777248929081 *
                      * subj:obj            0.163493574488339 *
                      * subj:obj:adv        0.089719181342218 *
                      * subj:obj:pp.at      0.0844835792479772 *
                      * subj:obj:pp.on      0.0702046644455021 *

         situate      * subj:obj:pp.in      0.250445632798574 *
                      * subj:obj:pp.on      0.147058823529412 *
                      * subj:obj            0.13458110516934 *
                      * subj:obj:obj        0.0436720142602496 *
                      * subj:obj:pp.at      0.0436720142602496 *

         place        * subj:obj:pp.on      0.27240599378004 *
                      * subj:obj:pp.in      0.215860899067006 *
                      * subj:obj            0.116906983319197 *
                      * subj:obj:to         0.0453774385072095 *
                      * subj:obj:pp.at      0.0402883799830365 *
```

  put, *situate* and *place* are in the same Levin class *Placing*, but *throw* is not. But should not one sense of *throw* represent a placing act? WordNet classifies the verb as a synonym of *situate* being sub-ordinated to the synset containing the two other verbs.

- The same clustering method with additional information about selectional preferences determined the cluster

```
C(2) :   learn        * subj:to     LifeForm            0.183047223103825 *
                      * subj        LifeForm            0.155498816960474 *
                      * subj        Agent               0.140522495491024 *
                      * subj:to     Agent               0.136619641097671 *
                      * subj:that   LifeForm            0.0635539000945356 *

         get          * subj:obj    Agent:Abstract      0.122452343630752 *
                      * subj:obj    LifeForm:Abstract   0.119996947463958 *
                      * subj        LifeForm            0.0970322805446853 *
                      * subj        Agent               0.0939162924557201 *
                      * subj:ap     Agent               0.0731423432361642 *
```

  acquire is member in the class of *Learning* in the sense of acquiring knowledge. WordNet gives the information that *get* and *acquire* are synonyms. Is that relationship not possible to be transfered to the domain of knowledge acquisition?

- Clustering into latent classes resulted in a cluster with the following two verbs:

Cluster PROB 0.0183		0.1413	0.0636	0.0568	0.0567
		subj::Abstract	subj:that::Abstract	subj:obj:obj::State:Agent:Location	subj::Psycho
0.5001	guarantee	•	•	•	•
0.4030	suggest	•	•		•

  WordNet defines both verbs as sub-ordinated to the verb *tell*, but so far only *suggest* is member of the *Telling* class.

These were some examples of verb senses assigned to certain semantic verb classes by WordNet, but not by Levin. As said before, I do not doubt Levin's classification system; the background of the examples was to point out the possibility that the evaluation basis is influenced subjectively.

I conclude this chapter with an illustration of the previously described phenomena by comparing the clustering approaches concerning a concrete example.

The semantic class *Admiration* contains the four verbs *admire, envy* (a low-frequent verb), *hate* and *like*. Following Levin, they alternate between using the frame types `subj:obj`, `subj:obj:as`, `subj:obj:pp.for`, `subj:obj:in` and `subj:that`, and they demand a living entity as subject. The polysemous verb *like* is also member of the class *Desire* which varies between `subj:obj`, `subj:obj:as` and `subj:obj:pp.for`, also with a living entity as subject. Following I investigate what happened to them in the clustering processes.

- Clustering according to subcategorisation frames only:

  When clustering only on the basis of the syntactic alternation behaviour, the verbs' pointer to the closest verb in distance were as follows:

  ```
  admire   ->   invent
  envy     ->   admire
  hate     ->   love
  like     ->   need
  ```

  As described before, without information about the selectional preferences, *admire* chose *invent* as most similar verb, since the syntactic frames are similar. *like* agrees most with *need*, a verb from the semantic class *Desire* the another sense of *like* belongs to. *envy* and *hate* correctly chose another verb from the same class.

  Applying *One-Step Distance Clustering* assigned the verbs to the clusters

  ```
  C(22) :   accumulate
            acquire
            gain

            provide
            supply

            arrange
            construct
            create
            develop
            invent
            pour
            produce

            break
            cut
            split
            tear
  ```

```
            analyse
            study

            kick
            smash

            admire
            envy

C(2) :      hate
            love

C(4) :      desire
            like
            need
            want
```

Because of the missing limit on the number of members within a cluster and wrong pointers like that of *admire*, one cluster contains 22 verbs, uniting verbs from six different clusters (marked by the ordering of the verbs). *hate* was correctly clustered together with *love*, and *like* were correctly clustered in the class *Desire*.

Iterative clustering avoids some noise introduced by wrong pointers and results in the clusters

```
C(1) :   admire    * subj:obj          0.515470297029703 *
                   * subj              0.172648514851485 *
                   * subj:obj:pp.for   0.0513613861386139 *
                   * subj:obj:obj      0.041460396039604 *
                   * subj:obj:adv      0.041460396039604 *


C(1) :   envy      * subj:obj          0.419724770642202 *
                   * subj:obj:obj      0.185779816513761 *
                   * subj              0.121559633027523 *
                   * subj:obj:pp.for   0.0252293577981651 *
                   * subj:s            0.0160550458715596 *


C(2) :   hate      * subj:obj          0.641761612620508 *
                   * subj:to           0.0996932515337423 *
                   * subj              0.0479842243645925 *
                   * subj:vger         0.0453549517966696 *
                   * subj:obj:obj      0.0322085889570552 *

         love      * subj:obj          0.603842412451362 *
                   * subj:to           0.0962224383916991 *
                   * subj              0.0947632944228275 *
                   * subj:obj:adv      0.0623378728923476 *
                   * subj:obj:obj      0.0396400778210117 *
```

```
C(4) :   need      * subj:to           0.382847629835582 *
                   * subj:obj          0.318590601723132 *
                   * subj              0.0962654034943192 *
                   * subj:obj:to       0.0536333367658669 *
                   * subj:obj:pp.for   0.0189647478804105 *

         like      * subj:to           0.344067278287462 *
                   * subj:obj          0.34302752293578 *
                   * subj              0.142110091743119 *
                   * subj:obj:adv      0.0364220183486239 *
                   * subj:obj:obj      0.0262691131498471 *

         want      * subj:to           0.533195075557434 *
                   * subj:obj          0.149146676529642 *
                   * subj              0.110892423121632 *
                   * subj:obj:to       0.102729049984149 *
                   * subj:to:adv       0.0163663742999049 *

         desire    * subj:obj          0.25 *
                   * subj              0.244535519125683 *
                   * subj:to           0.203551912568306 *
                   * subj:obj:to       0.069672131147541 *
                   * subj:s            0.0204918032786885 *
```

admire and *envy* were not clustered, the other clusters look the same as after one step. *hate* and *love* agree in four subcategorisation frames (concerning the noise compare section 3.2.2), and both have strong preferences for a transitive frame. *like* was clustered in the class *Desire*, because it agrees with all other verbs in the same cluster in the three subcategorisation frames describing an alternation between a transitive and a (linguistically wrong) intransitive use, the latter preferably accompanied by an infinitival phrase. Comparing the distribution with that of *need* – the most similar verb – in addition the probabilities for `subj:to` and `subj:obj` strongly agree.

The latent class analysis did not cluster *admire* and *envy*, but *hate* was clustered with at least *love* and *like*, and the algorithm recognised five senses of *like* according to five different kinds of alternation behaviour, once within most verbs from the *Desire* class, once within most verbs from the *Admiration* class, and three times with no semantic background according to our defined classes; that is, the subcategorisation frames were over-interpreted for the verb's senses:

Cluster PROB 0.0271		0.4972	0.2408	0.0835	0.0669
		subj:to	subj:obj	subj:obj:to	subj
0.5992	want	•	•	•	•
0.1983	need	•	•	•	•
0.1381	get	•	•	•	•
0.0330	like	•	•	•	•

Cluster PROB 0.0173		0.7064	0.2477	0.0170	0.0136
		subj:obj	subj:obj:adv	subj:obj:obj	subj:obj:pp.with
0.2726	tell	•	•	•	•
0.1143	like	•	•	•	•
0.0893	love	•	•	•	•
0.0340	hit	•	•	•	•

Cluster PROB 0.0151		0.5950	0.1405	0.1134	0.0550
		subj	subj:obj	subj:s	subj:obj:adv
0.5170	say	•	•	•	•
0.1443	think	•	•	•	•
0.0584	leave	•	•	•	•
0.0450	like	•	•	•	•

Cluster PROB 0.0112		0.3166	0.3072	0.2642	0.0221
		subj:vger	subj	subj:obj	subj:adv
0.3312	stop	•	•	•	•
0.1061	like	•	•	•	•
0.0903	need	•	•	•	•
0.0900	finish	•	•	•	•

Cluster PROB 0.0035		0.6517	0.1398	0.0539	0.0285
		subj:obj	subj:to	subj:obj:obj	subj:pp-about
0.7632	like	•	•	•	•
0.0768	love	•	•	•	•
0.0486	need	•	•	•	•
0.0464	hate	•	•	•	•

- Clustering according to subcategorisation frames and selectional preferences:

  Including information about the selectional preferences allows to identify additional common tendencies, but also introduces more noise into the clustering process.

  The verbs' pointers to the closest verb in distance changed as follows:

  ```
  admire   ->   envy
  envy     ->   admire
  hate     ->   love
  like     ->   promise
  ```

  The most similar verb to *hate* is still *love* which is not surprising, since both agree in most of the used subcategorisation frames. Interestingly, *hate* was itself chosen by five verbs as most similar verb, which was caused by the common usage of the transitive frame with a living entity as subject and object.
  With the additional information, *admire* now chose *envy* as most similar verb. The choice of *like* changed to the verb *promise*, however. They show overlap in their distributions, especially for the subcategorisation frame `subj:to` with a living subject.

  Applying *One-Step Distance Clustering* assigned the verbs to the clusters

  ```
  C(6) :    admire
            envy
            hear
            see
            leave
            warn
  ```

```
C(16) :   beat
          kick
          hate
          love
          execute
          kill
          murder
          kiss
          meet
          visit
          put
          rub
          tell
          dismiss
          send
          find

C(3) :    like
          want
          promise
```

As the clusters show, the four verbs of interest were not united. The clusters include more noise than those in version A; in addition to classes included in the larger clusters (*execute, kill, murder*, for example), several verbs are the only representatives of their classes (*tell, dismiss, find*, for example).

Applying *Iterative Distance Clustering* resulted in the clusters

```
C(2) :   envy      * subj:obj:obj   Agent:LifeForm:State   0.202422062386625 *
                   * subj:obj       LifeForm:Agent         0.113387120464967 *
                   * subj:obj       LifeForm:LifeForm      0.112901377570876 *
                   * subj:obj       Agent:Agent            0.112565961049389 *
                   * subj:obj       Agent:LifeForm         0.112083735947702 *

         admire    * subj:obj       Agent:Agent            0.152353351495087 *
                   * subj:obj       Agent:LifeForm         0.149670971445564 *
                   * subj:obj       LifeForm:Agent         0.146368312962172 *
                   * subj:obj       LifeForm:LifeForm      0.143791307345169 *
                   * subj           LifeForm               0.111411846548935 *


C(4) :   hate      * subj:obj       Agent:LifeForm         0.236281502698908 *
                   * subj:obj       LifeForm:LifeForm      0.22890913316933 *
                   * subj:obj       Agent:Agent            0.191105125209101 *
                   * subj:obj       LifeForm:Agent         0.185142332582744 *
                   * subj:to        LifeForm               0.0664037165252146 *

         love      * subj:obj       Agent:LifeForm         0.181210762214099 *
                   * subj:obj       LifeForm:LifeForm      0.176967988618079 *
                   * subj:obj       Agent:Agent            0.171084728246242 *
                   * subj:obj       LifeForm:Agent         0.167079040290314 *
                   * subj:to        Agent                  0.0563829837622933 *
```

```
promise   * subj:to        Agent                0.153087674596593 *
          * subj           Agent                0.138905050088486 *
          * subj:to        LifeForm             0.136593521233072 *
          * subj           LifeForm             0.13173467656184 *
          * subj:obj       Agent:Agent          0.0750377634797114 *

like      * subj:to        Agent                0.208995545583442 *
          * subj:to        LifeForm             0.20381019496847 *
          * subj:obj       Agent:LifeForm       0.127170462376897 *
          * subj:obj       LifeForm:LifeForm    0.124687606109041 *
          * subj           Agent                0.0847730105005142 *
```

admire and *envy* were again recognised as belonging into the same class. They share their preference for a transitive frame with a living subject and a living object. *hate* and *love* do so, too, but with their own characteristic strength. *like* was actually assigned to the same cluster because of a partial overlap with the transitive frame and the overlap of the `subj:to` frame with a living subject.

The latent class analysis assigned *admire* to two different clusters, none of them representing a semantic class based on our definitions, though the overlap in subcategorisation frames was filtered in a right way. The low-frequent verb *envy* was assigned to one cluster only, caused by the possible use of the ditransitive frame. *hate* and *like* were actually assigned to the same cluster, but together with *want*, a verb within the semantic class of the second sense of *like*, and *promise*, also a verb similar in use as *like*:

Cluster PROB 0.0428		0.5105	0.4895	0.0000	0.0000
		subj::LifeForm	subj::Agent	subj:obj::LifeForm:LifeForm	subj:obj::Agent:LifeForm
0.0431	write	•	•		
0.0431	admire	•	•	•	•
0.0430	storm	•	•		
0.0428	promise	•	•	•	•

Cluster PROB 0.0255		0.2366	0.2333	0.2311	0.2267
		subj:obj::Agent:LifeForm	subj:obj::Agent:Agent	subj:obj::LifeForm:LifeForm	subj:obj::LifeForm:Agent
0.1380	admire	•	•	•	•
0.0720	tell	•	•	•	•
0.0712	warn	•	•	•	•
0.0694	murder	•	•	•	•

Cluster PROB 0.0186		0.2485	0.2398	0.1065	0.1022
		subj:to::Agent	subj:to::LifeForm	subj:obj::Agent:LifeForm	subj:obj::LifeForm:LifeForm
0.2929	promise	•	•	•	•
0.1974	want	•	•	•	•
0.1739	like	•	•	•	•
0.1693	hate	•	•	•	•

Cluster PROB 0.0056		0.3292	0.1295	0.1191	0.0821
		subj:obj:obj::Agent:LifeForm:State	subj::Agent	subj::LifeForm	subj::PhysObject
0.9909	envy	•	•	•	•
0.0091	promise	•	•	•	
0.0000	jump		•	•	
0.0000	want		•	•	

Chapter 4

Conclusions

The goal of this thesis was to automatically classify verbs semantically, based on their alternation behaviour.
I chose 153 verbs from Levin's already provided classification; the verbs represent 226 verb senses from 30 semantic classes. The representation of the verbs was realised in two versions: `A` – the alternation behaviour of the verbs was defined by the syntactic use of subcategorisation frame types, and `B` – the alternation behaviour of the verbs was defined by the use of subcategorisation frame types refined with their selectional preferences for the arguments within the frames. For the syntactic information I utilised a statistical head-entity parser, for the semantic information about the arguments I queried the WordNet hierarchy.

For the classification of the verbs I developed two clustering approaches. First, an algorithm iteratively clustering the verbs according to their distances. For this approach, both versions for the verbs' representation were considered as probability distributions over the different types of subcategorisation frames. The distances were calculated as the geometrical distances according to euclidean distance and cosine, and as the difference according to the relative entropy between the distributions. Secondly, an algorithm utilising a latent class analysis based on the joint frequencies of verbs and frame types in version `A` and the association between the verbs and their subcategorisation frames in version `B` was applied.
The main difference between the concepts of the two algorithms concerns the central clustering question to which extent the entities to cluster (in our case the verbs) have to be similar to belong together. The distance clustering determines the extent by the distance/difference between the verbs' representations and therefore takes all subcategorisation frames a verb goes with

into account. The latent class analysis searches for shared components in the verbs' representations which allows to distinguish between the different verb senses.

The distance clustering succeeded for 61% of the verbs in version `A` and 38% in version `B`. That is, the respective percentage of verbs was clustered together with verbs from the same semantic class. The latent class analysis succeeded for 54% and 31%, respectively.

An investigation of the resulting clusters showed that the assignment of the verbs was actually based on their shared linguistic properties: the verbs in a cluster presented a common alternation behaviour. The common properties within one cluster were refined when adding information about the selectional preferences to the syntactic information of the subcategorisation frames.

The interpretation demonstrated that some problems in the classification process have to be solved:

- The definition of the verbs' representations includes noise concerning the choice of subcategorisation frames, the choice of conceptual classes for the arguments, and the formulation of their preferences. The causes can be attributed to parsing mistakes, extraction mistakes, or mistakes in the definition of the representation. The degree of noise is not exceptional, though.

- An obvious problem in the clustering is the fact that the results due to version `B` are always worse than those in version `A`. As filtered in the general interpretation, the representation of the subcategorisation frames including information about their selectional preferences should be improved.

- The polysemy of verbs presents a problem, especially for the distance clustering, which cannot distinguish between the multiple senses. To exclude this problem, the verbs should be disambiguated before being clustered. An approach like [Yarowsky, 1995] which considers the context of a word it appears with could be applied to first disambiguate the verbs before sending them into the clustering process.
  Metaphorical uses of verbs might be excluded by querying a dictionary (on-line resources provide information about that use) before applying the clustering process to the verbs.

- Both approaches have difficulties in clustering low-frequency verbs, since the data cannot be delimited in the clustering process.

- It is difficult to find an optimal evaluation basis, since most already available classification systems are subjectively influenced. A possibility would be to create a questionnaire about the specific classifications of verbs which could then represent a reliable basis.

The different issues show that there are possibilities to improve the classification process in some promising ways. Most important are – in my opinion – the incorporation of context into the representation of the verbs and their alternation behaviour, and an improvement of the representation concerning the selectional preferences.

Considering the overall desire of this thesis, a successful step into the direction of presenting the connection between the verbs' alternation behaviour and their semantics by automatic means is done. Nevertheless, there are possibilities to improve the process.

Acknowledgements

I thank Marc Light, Mats Rooth, Glenn Carroll, Helmut Schmid, Detlef Prescher, Martin Emele and Anke Lüdeling for their support in various parts of the thesis.

Appendix A

Subcategorisation Frames

Following is a list of the 88 subcategorisation frames (without information about the selectional preferences) I utilised as attributes for the verbs' distributions in version `A`. The frames are numbered from 0 to 87.

```
0       subj
1       subj:adv
2       subj:ap
3       subj:obj
4       subj:obj:adv
5       subj:obj:ap
6       subj:obj:as
7       subj:obj:obj
8       subj:obj:obj:adv
9       subj:obj:obj:pp.at
10      subj:obj:obj:pp.for
11      subj:obj:obj:pp.in
12      subj:obj:obj:pp.on
13      subj:obj:obj:pp.to
14      subj:obj:obj:pp.with
15      subj:obj:pp.about
16      subj:obj:pp.after
17      subj:obj:pp.against
18      subj:obj:pp.as
19      subj:obj:pp.at
20      subj:obj:pp.before
21      subj:obj:pp.between
22      subj:obj:pp.by
23      subj:obj:pp.during
24      subj:obj:pp.for
25      subj:obj:pp.from
26      subj:obj:pp.in
27      subj:obj:pp.in:adv
```

```
28      subj:obj:pp.in:pp.in
29      subj:obj:pp.into
30      subj:obj:pp.like
31      subj:obj:pp.of
32      subj:obj:pp.on
33      subj:obj:pp.out_of
34      subj:obj:pp.over
35      subj:obj:pp.through
36      subj:obj:pp.to
37      subj:obj:pp.under
38      subj:obj:pp.with
39      subj:obj:pp.within
40      subj:obj:pp.without
41      subj:obj:ppart
42      subj:obj:s
43      subj:obj:sub
44      subj:obj:that
45      subj:obj:to
46      subj:obj:vbase
47      subj:obj:vger
48      subj:pp.about
49      subj:pp.across
50      subj:pp.after
51      subj:pp.against
52      subj:pp.as
53      subj:pp.at
54      subj:pp.at:adv
55      subj:pp.between
56      subj:pp.by
57      subj:pp.for
58      subj:pp.for:adv
59      subj:pp.from
60      subj:pp.from:pp.to
61      subj:pp.in
62      subj:pp.in:adv
63      subj:pp.into
64      subj:pp.like
65      subj:pp.of
66      subj:pp.on
67      subj:pp.on:adv
68      subj:pp.out_of
69      subj:pp.over
70      subj:pp.through
71      subj:pp.to
72      subj:pp.to:adv
73      subj:pp.towards
74      subj:pp.under
75      subj:pp.up_to
76      subj:pp.upon
```

```
77      subj:pp.with
78      subj:pp.with:adv
79      subj:ppart
80      subj:s
81      subj:sub
82      subj:that
83      subj:to
84      subj:to:adv
85      subj:vbase
86      subj:vbase:adv
87      subj:vger
```

Appendix B

WordNet Concepts

B.1 File Numbers

There are 25 files (actually 26 because of the top file `Tops`), numbered from `03` to `28`. The range of the numbers follows from the fact that the files for other parts of speech are preceding/following; the noun files are only a part of the overall definition.
Each file is identified by the file number and a corresponding file name:

```
03    Tops
04    act
05    animal
06    artifact
07    attribute
08    body
09    cognition
10    communication
11    event
12    feeling
13    food
14    group
15    location
16    motive
17    object
18    person
19    phenomenon
20    plant
21    possession
22    process
23    quantity
24    relation
```

```
25      shape
26      state
27      substance
28      time
```

B.2 (Top) Synset Numbers

There are 11 top level nodes of 11 hierarchies in WordNet. Since the concept of `Entity` seemed too general as conceptual class, I replaced it by the next lower levels (13 different synsets). Each WordNet synset number is followed by an identifying abbreviation for the synset and the nouns member of the synset:

```
00002403   Entity: entity
           => 00002728  LifeForm: life form, organism, being, living thing
           => 00003711  Cell: cell
           => 00004473  Agent: causal agent, cause, causal agency
           => 00009469  PhysObject: object, inanimate object, physical object
           => 01958400  Thing: thing
           => 01959683  Whole: whole, whole thing, unit
           => 02985352  Content: subject, content, depicted object
           => 05650230  Unit: unit, building block
           => 05650477  Part: part, piece
           => 05763289  Essential: necessity, essential, requirement,
                        requisite, necessary, need
           => 05763845  Inessential: inessential
           => 05764087  Variable: variable
           => 05764262  Anticipation: anticipation
00012517   Psycho: psychological_feature
00012670   Abstract: abstraction
00014314   Location: location
00014558   Shape: shape, form
00015437   State: state
00016459   Event: event
00016649   Action: act, human_action, human_activity
00017008   Group: group, grouping
00017394   Possession: possession
00019295   Phenomenon: phenomenon
```

B.3 Additionally Defined Nouns

Following are the nouns not appearing in WordNet which I provided with WordNet synset nodes. `00002403` defines an entity, `00004865` a person.

pn	00004865
i	00004865
me	00004865
you	00004865
he	00004865
him	00004865
she	00004865
her	00004865
it	00002403
we	00004865
us	00004865
they	00002403
them	00002403
myself	00004865
yourself	00004865
himself	00004865
herself	00004865
itself	00002403
ourselves	00004865
yourselves	00004865
themselves	00002403
this	00002403
that	00002403
these	00002403
those	00002403
everyone	00004865
everybody	00004865
someone	00004865
somebody	00004865
anyone	00004865
anybody	00004865

Appendix C

Distances between Verbs

The table represents the distances between the 7 verbs *break, eat, envy, like, load, move* and *place* to all of the 153 verbs I worked with.

	Version A							Version B						
	break	eat	envy	like	load	move	place	break	eat	envy	like	load	move	place
accumulate	0.58	0.37	0.56	1.59	0.79	0.82	1.31	6.11	8.45	8.09	9.23	6.62	5.28	7.20
acquire	0.39	0.46	0.41	1.12	0.52	1.33	1.07	6.51	9.89	8.92	9.68	8.33	6.57	7.81
admire	0.35	0.23	0.24	0.72	0.54	1.14	1.37	7.65	4.03	1.03	1.42	8.23	4.10	6.12
advise	1.22	1.00	1.12	1.94	1.93	2.03	1.65	8.71	6.07	3.18	3.97	7.77	6.99	7.44
allocate	1.13	1.05	0.69	1.58	1.24	1.26	1.46	9.84	10.9	10.3	10.4	9.13	7.44	8.59
analyse	0.43	0.42	0.50	1.12	0.48	1.28	0.91	6.06	11.3	9.48	10.2	8.52	6.65	11.5
announce	0.99	0.59	0.81	2.00	1.26	1.94	1.02	8.52	6.69	7.30	7.87	8.48	7.10	8.17
argue	1.69	1.14	1.74	2.47	2.30	2.46	3.33	7.39	4.20	5.00	6.27	6.83	7.34	8.41
arrange	0.78	0.69	0.76	0.67	0.64	1.48	1.01	9.16	8.05	6.94	4.77	7.99	7.88	6.87
assess	0.60	0.67	0.72	1.38	0.50	1.60	0.74	7.47	11.4	9.87	10.3	8.91	7.73	11.1
beat	0.49	0.43	0.41	1.44	0.54	1.23	0.91	8.70	6.62	2.37	4.25	8.36	5.27	6.65
begin	1.58	1.20	1.80	0.57	1.73	1.74	2.35	6.80	7.50	6.19	6.62	6.35	8.00	8.18
believe	1.85	1.18	1.69	1.72	2.52	2.64	3.04	8.17	5.41	3.92	5.01	6.90	7.15	7.78
bounce	0.72	0.71	1.14	2.20	1.04	0.54	1.71	2.50	5.26	5.16	7.23	3.58	2.99	6.43
break		0.29	0.53	1.28	0.51	0.80	1.43		4.77	7.30	8.52	4.56	4.99	6.29
brush	0.64	0.58	0.81	1.69	0.59	1.11	1.50	5.96	7.92	7.93	10.8	6.27	5.13	7.81
build	0.51	0.64	0.66	1.63	0.48	1.32	0.50	6.00	4.29	7.44	8.41	4.99	6.46	5.52
buy	0.49	0.37	0.40	1.20	0.53	1.30	1.06	5.40	2.56	6.45	7.42	3.74	5.73	5.23
characterize	0.78	0.94	0.80	1.61	1.07	2.21	1.80	7.92	11.7	9.86	10.5	8.38	7.49	8.34
classify	0.98	1.07	1.10	2.11	1.11	2.09	1.61	7.16	8.92	7.55	10.2	8.65	6.97	8.95
climb	0.35	0.46	0.64	1.49	0.89	0.33	1.64	3.62	3.35	6.36	7.66	2.94	4.19	5.11
collect	0.49	0.39	0.54	1.46	0.57	1.14	1.07	4.77	4.19	6.69	7.44	4.97	5.83	5.67
communicate	0.74	0.61	0.96	1.54	0.82	0.82	1.83	6.95	8.98	8.16	10.1	7.68	7.11	9.21
confess	1.38	0.94	1.58	1.80	1.92	1.05	3.08	7.73	4.29	3.44	4.84	7.24	6.46	8.08
construct	0.63	0.63	0.48	1.38	0.55	1.31	0.89	5.79	6.71	9.11	9.73	5.83	6.96	5.91
continue	1.42	1.11	1.78	0.53	1.82	1.48	2.85	6.06	7.22	6.35	6.70	6.42	7.67	8.19
cook	0.37	0.17	0.33	1.08	0.43	0.88	1.17	3.48	1.79	5.79	6.85	2.93	3.62	4.16
correspond	2.70	2.41	3.04	2.85	2.43	1.20	3.69	8.16	9.12	6.92	8.55	7.20	7.71	9.04
create	0.47	0.45	0.42	1.06	0.51	1.47	1.01	6.89	10.2	10.1	10.8	8.16	6.89	8.65
crush	0.39	0.51	0.61	1.56	0.39	1.08	0.88	5.02	7.71	6.73	7.76	5.07	4.08	6.29
cut	0.36	0.56	0.65	1.51	0.55	0.95	1.18	5.22	7.61	9.70	10.3	6.02	7.27	8.47
declare	1.06	0.54	0.86	1.68	1.56	1.71	1.53	7.55	5.57	4.86	6.38	7.71	7.41	7.25
delete	0.53	0.56	0.53	1.36	0.78	1.22	1.31	4.67	6.33	7.26	8.25	4.35	5.05	7.22
demolish	0.43	0.57	0.55	1.37	0.54	1.42	1.00	4.12	4.75	6.53	7.74	3.17	4.64	5.14
depart	1.16	0.83	1.23	1.56	1.29	0.82	2.14	4.38	4.04	3.92	5.14	3.67	4.19	7.78
describe	0.98	0.64	0.99	2.35	1.61	2.06	1.82	7.13	6.75	7.13	7.83	7.64	7.77	9.63
desire	0.78	0.54	0.87	0.29	1.24	0.93	1.89	7.04	4.82	2.58	2.57	6.59	5.63	6.52
destroy	0.31	0.39	0.44	0.96	0.55	1.52	1.20	4.38	8.09	9.67	10.6	5.02	5.43	6.37
develop	0.21	0.20	0.43	1.07	0.49	0.80	1.16	5.62	6.74	8.61	9.66	6.85	6.10	7.66

	break	eat	envy	like	load	move	place	break	eat	envy	like	load	move	place
disconnect	0.68	0.82	0.96	1.34	1.00	1.17	1.93	5.50	6.24	7.17	9.62	4.20	5.87	6.37
dismiss	0.76	0.76	0.68	1.64	0.84	2.08	1.48	7.26	7.18	3.82	5.06	8.18	6.48	6.29
distinguish	1.13	1.10	1.02	2.74	1.55	2.08	1.84	8.63	10.3	8.58	10.4	8.02	7.70	9.21
drink	0.29	0.07	0.44	0.85	0.53	0.66	1.47	4.92	0.11	3.66	4.24	4.47	4.00	4.88
eat	0.29		0.43	0.83	0.55	0.73	1.47	4.77		5.07	4.92	4.43	4.18	4.96
eliminate	0.53	0.64	0.58	1.24	0.60	1.45	1.18	6.94	11.2	9.64	10.1	8.50	7.29	9.36
end	0.79	0.42	1.10	1.76	0.96	0.88	1.80	5.20	7.96	7.67	9.28	7.06	6.48	9.05
entrust	1.96	2.08	2.16	2.59	1.16	1.61	2.42	6.97	7.93	6.25	7.35	9.53	5.68	7.85
envy	0.53	0.43		0.94	0.63	1.30	1.50	7.30	5.07		2.86	6.87	4.59	6.14
evaluate	0.40	0.53	0.40	1.23	0.41	1.37	0.84	6.91	10.9	9.25	9.72	9.45	7.39	10.3
execute	0.42	0.49	0.52	1.36	0.47	1.41	0.78	8.11	7.94	3.47	4.96	7.72	5.27	5.60
exist	1.48	0.95	1.72	1.87	2.04	1.04	3.15	3.80	6.08	6.33	7.64	4.27	5.56	8.83
exit	0.86	0.60	0.75	1.52	0.98	0.80	1.83	5.85	2.00	1.50	3.31	5.36	4.91	6.41
explain	1.00	0.56	1.04	1.28	1.36	1.05	2.38	8.30	4.70	5.20	5.14	9.92	7.86	9.84
extract	0.81	0.89	0.87	1.55	0.89	1.59	2.04	4.26	4.56	6.67	7.87	4.09	4.52	6.79
feel	1.32	0.82	1.20	1.95	1.93	1.91	2.59	7.29	5.90	4.70	6.99	6.87	7.84	7.82
fight	0.66	0.42	0.84	0.73	0.79	0.96	1.37	7.77	7.23	6.71	4.59	6.66	6.89	7.79
find	0.86	0.71	0.84	1.65	1.26	1.94	1.26	7.24	2.50	3.18	4.97	6.41	5.35	5.58
finish	0.54	0.24	0.60	0.78	0.69	1.20	1.71	3.38	2.86	6.35	6.88	4.62	4.80	5.45
flee	0.87	0.65	1.28	1.86	1.32	0.39	2.12	7.75	4.66	4.70	5.68	5.30	3.18	8.03
float	0.81	0.70	1.20	1.90	1.14	0.45	1.34	4.19	6.59	5.17	8.50	4.05	3.18	7.92
fly	0.72	0.62	1.11	1.66	1.20	0.22	2.09	4.17	3.77	4.97	6.51	3.74	0.71	7.59
gain	0.48	0.47	0.43	1.42	0.61	1.22	1.07	7.26	11.5	9.96	10.6	9.94	7.29	9.10
get	0.37	0.34	0.61	0.54	0.82	0.70	1.57	5.01	2.40	5.12	2.94	6.89	6.93	6.55
give	1.26	1.12	0.57	1.76	1.35	2.03	1.87	8.44	11.2	9.31	11.9	9.06	10.4	10.9
guarantee	1.00	0.81	0.42	0.82	1.04	1.65	1.57	7.33	9.96	9.56	10.3	8.82	7.61	8.59
hate	0.75	0.68	0.59	0.36	0.86	1.58	1.68	8.53	5.55	1.22	0.51	8.76	3.79	5.74
hear	0.91	0.54	0.70	1.48	1.23	1.74	1.94	6.46	5.47	4.42	5.23	6.46	6.48	7.19
hit	0.35	0.42	0.57	1.04	0.38	1.45	0.98	5.66	8.07	7.47	8.41	5.69	4.66	7.68
identify	0.76	0.56	0.69	1.51	0.64	1.82	1.44	6.62	9.19	9.70	10.5	9.53	6.32	9.16
instruct	1.10	1.01	1.04	1.52	1.68	2.11	1.44	8.55	6.31	2.98	3.75	7.23	7.26	7.48
invent	0.48	0.41	0.26	0.87	0.56	1.42	1.16	3.72	3.17	6.88	7.07	3.13	4.69	4.40
jump	0.76	0.69	1.11	1.89	1.34	0.31	1.87	6.11	2.63	5.62	6.67	4.21	5.08	7.74
kick	0.43	0.33	0.58	1.50	0.47	0.95	0.97	4.74	3.67	3.10	4.50	4.92	3.72	4.45
kill	0.36	0.42	0.57	1.10	0.56	1.59	1.05	7.70	7.79	4.05	5.26	7.59	4.04	6.42
kiss	0.38	0.30	0.50	0.95	0.50	1.07	1.07	6.91	4.98	2.10	3.44	7.85	4.41	6.30
learn	1.06	0.68	1.14	0.52	1.66	1.25	2.43	7.17	5.36	4.73	2.60	7.55	7.84	8.29
leave	0.68	0.30	0.63	1.13	0.52	1.09	0.97	6.92	1.80	3.21	4.01	6.23	4.42	5.18
like	1.28	0.83	0.94		1.73	1.45	2.30	8.52	4.92	2.86		7.98	5.99	7.21
live	1.35	0.89	1.71	1.73	1.75	0.91	2.28	9.08	6.34	5.79	8.66	7.17	6.97	9.21
load	0.51	0.55	0.63	1.73		1.35	1.13	4.56	4.43	6.87	7.98		4.83	6.21
love	0.60	0.52	0.56	0.27	0.80	1.35	1.65	8.38	5.08	1.33	0.41	8.49	4.23	6.17
meet	0.29	0.16	0.51	0.97	0.44	0.95	1.17	7.11	5.42	2.26	2.52	7.58	2.54	6.19
moan	1.19	0.76	1.19	1.64	1.40	0.70	2.56	7.58	1.14	1.98	3.05	7.67	5.82	7.89
move	0.80	0.73	1.30	1.45	1.35		2.47	4.99	4.18	4.59	5.99	4.83		7.56
murder	0.32	0.41	0.43	0.94	0.50	1.47	1.14	8.24	5.47	1.25	1.94	8.40	3.54	5.58
need	1.23	0.85	0.99	0.10	1.88	1.57	2.27	6.32	2.83	5.72	5.04	6.41	7.54	7.05
notice	0.83	0.43	0.77	1.52	1.37	1.67	1.82	7.35	4.20	3.59	4.49	7.61	6.86	7.28
offer	0.87	0.62	0.39	0.80	0.76	1.42	1.31	6.47	10.6	9.30	7.39	8.02	6.96	9.29
pack	0.38	0.41	0.74	1.82	0.21	1.23	1.13	4.59	3.87	6.62	7.77	3.73	4.64	4.60
part	0.68	0.34	0.89	0.97	0.77	0.67	1.84	3.59	5.00	4.15	7.43	4.71	2.97	6.58
pass	0.34	0.35	0.60	1.82	0.85	0.59	1.44	6.23	7.21	9.03	9.77	6.47	7.55	7.35
pay	0.65	0.57	0.45	1.09	0.75	1.27	1.27	9.60	8.69	7.86	8.94	9.12	8.46	7.12
persist	1.36	1.02	1.79	2.10	1.98	0.98	3.15	4.36	6.52	6.95	8.11	5.50	6.97	9.04
place	1.43	1.47	1.50	2.30	1.13	2.47		6.29	4.96	6.14	7.21	6.21	7.56	
play	0.29	0.22	0.49	1.18	0.43	0.94	1.03	6.64	8.26	9.77	10.4	6.96	6.14	11.4
position	0.69	0.87	0.86	1.38	0.71	1.45	0.59	5.02	3.16	5.41	6.16	3.31	5.34	4.02
pour	0.59	0.77	0.49	2.55	0.72	1.12	1.63	6.12	3.18	5.69	6.46	3.16	5.39	6.32
produce	0.33	0.36	0.37	0.89	0.44	1.39	1.06	4.72	7.96	10.3	11.1	6.03	6.28	9.11
promise	1.17	0.65	0.68	0.35	1.48	1.29	2.19	8.22	4.84	1.91	0.23	7.65	6.16	7.42
propose	0.92	0.56	0.67	0.42	1.07	1.28	1.73	7.48	8.63	7.86	5.32	8.39	7.45	7.99
provide	0.91	0.75	0.69	1.43	0.63	1.85	1.36	6.15	11.4	10.9	11.6	10.3	7.81	10.7
purchase	0.69	0.57	0.41	1.54	0.55	1.44	1.04	5.58	4.19	8.71	7.78	4.53	6.35	5.33
put	0.91	1.01	1.26	1.98	0.82	1.48	0.36	7.24	7.14	5.85	7.73	8.15	7.89	6.43
qualify	0.99	0.83	1.06	1.29	1.79	1.66	2.31	7.33	8.31	6.74	8.36	6.58	6.20	8.51
rain	1.24	0.64	1.48	1.67	1.72	0.87	2.87	4.16	3.85	4.37	5.63	4.92	3.04	7.19
read	0.39	0.16	0.39	0.84	0.54	0.85	1.38	2.26	3.90	6.76	7.29	5.65	7.19	7.95
receive	0.70	0.52	0.51	1.43	0.61	1.36	1.17	7.54	7.89	7.67	8.51	9.39	7.96	7.39
remove	0.84	1.03	0.88	1.86	0.98	1.70	1.63	5.27	7.41	10.1	10.8	5.43	6.59	7.57
return	1.24	0.87	1.45	1.27	1.65	0.42	2.88	6.42	6.45	6.99	8.66	6.50	4.58	8.58
roll	0.36	0.34	0.66	1.31	0.67	0.31	1.46	3.01	4.10	4.25	6.48	3.88	2.05	5.81
rub	0.54	0.60	0.82	1.61	0.37	1.38	0.97	4.55	4.10	4.96	7.49	5.76	5.24	5.03
ruin	0.37	0.46	0.31	0.92	0.53	1.53	1.36	6.23	8.41	8.62	9.25	5.80	6.47	8.61
run	0.31	0.40	0.64	1.51	0.81	0.43	1.51	6.72	5.74	8.26	9.21	5.50	5.47	5.56
say	1.32	0.75	1.42	1.41	1.79	1.06	3.12	9.01	1.43	2.16	2.45	9.20	6.57	9.19
scream	1.14	0.66	1.36	1.61	1.60	0.83	2.80	8.63	1.81	2.05	2.37	8.81	6.51	8.94
see	0.86	0.44	0.48	1.29	1.02	1.60	1.63	8.92	4.84	6.07	4.17	8.61	6.36	7.91
sell	0.62	0.44	0.50	1.52	0.64	1.06	1.09	4.35	3.50	6.57	7.66	3.68	4.63	5.02
send	1.22	1.20	0.85	1.74	1.17	1.37	1.70	9.15	8.56	3.99	5.83	7.96	8.20	8.13
separate	0.68	0.68	1.13	1.74	1.11	1.39	2.15	7.28	8.50	8.14	10.7	5.20	5.02	8.07
shout	1.08	0.70	1.25	1.62	1.53	0.74	2.70	8.23	1.84	1.94	2.08	9.15	6.57	8.97

	break	eat	envy	like	load	move	place	break	eat	envy	like	load	move	place
show	0.90	0.52	0.68	1.75	1.17	1.87	1.52	5.82	9.34	9.24	10.2	7.49	7.80	9.38
situate	1.25	1.20	1.07	2.38	1.17	1.74	0.34	9.22	8.38	9.61	10.8	6.09	6.86	6.82
slide	0.82	0.96	1.20	2.04	1.06	0.52	2.01	6.00	4.61	4.84	7.96	3.50	3.28	6.55
smash	0.29	0.49	0.54	2.05	0.37	0.83	1.17	3.61	4.80	7.23	8.34	2.39	3.39	5.98
smell	0.67	0.60	0.66	1.74	1.31	1.64	1.64	4.53	3.85	4.99	6.04	4.13	5.10	5.71
snow	1.27	1.18	0.91	2.27	1.10	0.92	2.10	3.40	4.44	2.78	6.13	4.31	4.68	5.62
split	0.46	0.57	0.72	1.72	0.51	1.14	1.45	4.67	7.85	7.80	10.1	6.13	4.76	7.44
spray	0.69	0.75	0.88	1.75	0.31	1.18	0.99	5.68	7.03	6.64	9.82	3.41	5.55	6.87
spread	0.58	0.51	0.96	1.47	1.08	0.53	1.53	2.62	4.26	5.57	7.00	4.32	4.22	5.54
start	1.12	0.79	1.28	0.51	1.36	1.57	2.05	6.15	5.23	5.74	6.94	5.82	6.81	7.53
stay	1.21	0.82	1.55	1.59	1.63	0.86	2.44	5.83	2.59	4.49	7.21	6.57	5.16	8.12
stop	0.89	0.45	1.05	0.88	1.58	1.55	2.23	4.60	5.57	3.94	5.60	5.91	4.22	7.67
storm	0.39	0.40	0.65	2.02	0.95	0.58	1.84	6.83	4.86	6.32	7.42	6.32	6.33	5.63
study	0.40	0.34	0.42	1.15	0.48	1.25	0.85	5.02	2.05	5.59	6.05	6.09	6.49	7.35
suggest	1.63	0.99	1.40	2.59	2.26	2.51	3.00	6.32	8.29	7.53	9.63	6.66	8.35	8.93
supply	0.78	0.58	0.71	1.66	0.36	1.49	1.19	5.62	7.85	9.84	10.6	3.36	5.80	7.61
suppose	2.65	1.85	2.48	2.35	2.83	2.59	2.76	9.02	6.33	5.03	7.91	7.60	8.56	9.12
survive	0.51	0.26	0.79	1.08	1.00	0.64	2.00	5.40	8.24	8.07	9.02	6.06	5.74	9.31
teach	0.87	0.63	0.48	1.20	1.02	1.48	1.27	8.37	7.85	3.57	4.96	8.47	5.91	7.30
tear	0.69	0.86	1.19	1.75	0.95	1.07	1.47	6.87	7.52	6.99	7.92	5.99	6.34	7.44
tell	0.89	0.95	0.76	1.55	0.95	1.94	1.82	8.60	6.14	1.61	2.82	8.58	4.33	6.12
think	2.33	1.59	2.15	2.45	2.70	2.64	3.14	9.16	6.46	5.16	8.03	7.74	8.79	9.26
throw	0.98	0.92	0.93	2.21	0.70	1.53	0.97	6.70	4.13	6.30	7.12	5.31	7.24	5.67
tickle	0.58	0.64	0.56	1.61	0.65	1.22	1.30	3.98	6.14	5.42	8.43	6.10	4.37	5.51
touch	0.29	0.43	0.50	0.91	0.41	1.24	1.15	4.43	5.72	5.95	7.13	5.34	4.42	7.39
transfer	1.45	1.46	1.42	2.33	1.41	1.22	2.26	10.1	10.5	8.77	10.7	8.75	8.62	9.81
transport	0.83	0.94	0.54	1.70	0.82	0.97	1.41	6.98	8.06	7.37	8.05	4.38	5.61	7.02
visit	0.61	0.50	0.52	1.37	0.60	1.54	0.70	7.92	4.40	3.46	4.98	6.54	4.87	7.59
want	1.75	1.26	1.50	0.26	2.43	1.98	2.92	8.82	5.59	4.62	0.36	7.41	8.06	8.28
warn	1.25	0.77	1.06	2.29	1.77	2.14	2.48	7.82	4.74	2.68	3.97	6.91	5.68	7.09
waste	0.52	0.50	0.60	1.44	0.72	1.55	0.75	5.32	7.41	6.77	7.59	6.99	6.16	8.10
whisper	1.26	0.81	1.54	1.66	1.78	0.68	3.14	8.60	1.43	2.64	3.12	8.70	6.44	9.72
write	0.77	0.37	0.69	1.13	0.91	0.72	1.47	6.58	2.20	4.26	3.88	7.20	6.58	8.77

Bibliography

[Abe and Li, 1996] Abe, N. and Li, H. (1996). Learning Word Association Norms Using Tree Cut Pair Models. In *Proceedings of the 13th International Conference on Machine Learning*.

[Abney, 1991] Abney, S. (1991). Parsing by Chunks. In Berwick, R., Abney, S., and Tenny, C., editors, *Principle-Based Parsing*. Kluwer Academic Publishers, Dordrecht.

[Abney and Light, 1998] Abney, S. and Light, M. (1998). Hiding a Semantic Class Hierarchy in a Markow Model. Manuscript.

[Agirre and Rigau, 1996] Agirre, E. and Rigau, G. (1996). Word Sense Disambiguation Using Conceptual Density. In *Proceedings of COLING-96*, pages 16–22.

[Allen, 1995] Allen, J. (1995). *Natural Language Understanding*. The Benjamin/Cummmings Publishing Company, Redwood City, CA, 2nd edition.

[Armstrong, 1994] Armstrong, S., editor (1994). *Using Large Corpora*. MIT Press, Cambridge - London.

[Beale and Jackson, 1990] Beale, R. and Jackson, T. (1990). *Neural Computing: An Introduction*. Institute of Physics Publishing, Bristol - Philadelphia.

[Beckwith et al., 1991] Beckwith, R., Fellbaum, C., Gross, D., and Miller, G. A. (1991). Wordnet: A Lexical Database Organized on Psycholinguistic Principles. In Zernik, U., editor, *Lexical Acquisition – Exploiting On-Line Resources to Build a Lexicon*, chapter 9, pages 211–232. Lawrence Erlbaum Associates, Hillsdale - New Jersey.

[Brent, 1991] Brent, M. R. (1991). Automatic Acquisition of Subcategorization Frames from Untagged Text. In *Proceedings of the 29th Annual Meeting of the Association for Computational Linguistics*, pages 209–214.

[Brent, 1994a] Brent, M. R. (1994a). From Grammar to Lexicon: Unsupervised Learning of Lexical Syntax. In [Armstrong, 1994], pages 203–222.

[Brent, 1994b] Brent, M. R. (1994b). Surface Cues and Robust Inference as a Basis for the early Acquisition of Subcategorization Frames. In Gleitman, L. and Landau, B., editors, *The Acquisition of the Lexicon*, pages 433–470. MIT Press, Cambridge - London, 1st edition.

[Briscoe and Carroll, 1994] Briscoe, T. and Carroll, J. (1994). Generalized Probabilistic LR Parsing of Natural Language (Corpora) with Unification-Based Grammars. In [Armstrong, 1994], pages 25–59.

[Briscoe and Carroll, 1997] Briscoe, T. and Carroll, J. (1997). Automatic Extraction of Subcategorization from Corpora. In *Proceedings of the 5th ACL Conference on Applied Natural Language Processing.*

[Caramazza and Berndt, 1978] Caramazza, A. and Berndt, R. S. (1978). Semantic and Syntactic Processes in Aphasia: A Review of the Literature. *Psychological Bulletin*, 85:898–918.

[Carroll and Rooth, 1998] Carroll, G. and Rooth, M. (1998). Valence Induction with a Head-Lexicalized PCFG. In *Proceedings of the 3rd Conference on Empirical Methods in Natural Language Processing*, Granada, Spain.

[Chomsky, 1965] Chomsky, N. (1965). *Aspects of the Theory of Syntax.* MIT Press, Cambridge, MA.

[Collins and Quillian, 1969] Collins, A. M. and Quillian, M. R. (1969). Retrieval Time from Semantic Memory. *Journal of Verbal Learning and Verbal Behavior*, 8:240–247.

[Gleitman, 1990] Gleitman, L. (1990). The Structural Sources of Verb Meanings. *Language Acquisition*, 1(1):3–56.

[Hindle, 1990] Hindle, D. (1990). Noun Classification from Predicate-Argument Structures. In *Proceedings of the 28th Annual Meeting of the Association for Computational Linguistics*, pages 268–275.

[Hovav and Levin, 1998] Hovav, M. R. and Levin, B. (1998). Building Verb Meanings. In Butt, M. and Geuder, W., editors, *Lexical and Compositional Factors*, pages 97–134. CSLI Publications, Stanford, CA.

[Hughes, 1994] Hughes, J. (1994). *Automatically Acquiring Classification of Words.* PhD thesis, University of Leeds, School of Computer Studies.

[Karp et al., 1992] Karp, D., Schabes, Y., Zaidel, M., and Egedi, D. (1992). A Freely Available Wide Coverage Morpholocal Analyzer for English. In *Proceedings of COLING-92*, Nantes, France.

[Klavans and Kan, 1998] Klavans, J. L. and Kan, M.-Y. (1998). The Role of Verbs in Document Analysis. In *Proceedings of the 17th International Conference on Computational Linguistics (COLING-ACL '98)*, Montreal, Canada.

[Kullback and Leibler, 1951] Kullback, S. and Leibler, R. A. (1951). On Information and Sufficiency. *Annals of Mathematical Statistics*, 22:79–86.

[Lee, 1997] Lee, L. J. (1997). *Similarity-Based Approaches to Natural Language Processing*. PhD thesis, Harvard University, Cambridge, MA.

[Levin, 1993] Levin, B. (1993). *English Verb Classes and Alternations*. The University of Chicago Press, Chicago, 1st edition.

[Levin and Hovav, 1996] Levin, B. and Hovav, M. R. (1996). From Lexical Semantics to Argument Realization. In Borer, H., editor, *Handbook of Morphosyntax and Argument Structure*. Kluwer, Dordrecht.

[Light, 1996] Light, M. (1996). *Morphological Cues for Lexical Semantics*. PhD thesis, University of Rochester, Department of Computer Science, Rochester, NY.

[Luk, 1995] Luk, A. K. (1995). Statistical Sense Disambiguation with Relatively Small Corpora Using Dictionary Definitions. In *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics*, pages 181–188.

[Manning, 1993] Manning, C. D. (1993). Automatic Acquisition of a Large Subcategorization Dictionary from Corpora. In *Proceedings of the 31st Annual Meeting of the Association for Computational Linguistics*, pages 235–242.

[Miller et al., 1990] Miller, G. A., Beckwith, R., Fellbaum, C., Gross, D., and Miller, K. J. (1990). Introduction to Wordnet: An On-line Lexical Database. *International Journal of Lexicography*, 3(4):235–244.

[Pereira et al., 1993] Pereira, F., Tishby, N., and Lee, L. (1993). Distributional Clustering of English Words. In *Proceedings of the 31st Annual Meeting of the Association for Computational Linguistics*, pages 183–190.

[Pinker, 1989] Pinker, S. (1989). *Learnability and Cognition: The Acquisition of Argument Structure.* MIT Press, Cambridge, MA.

[Resnik, 1993] Resnik, P. (1993). *Selection and Information: A Class-Based Approach to Lexical Relationships.* PhD thesis, University of Pennsylvania.

[Resnik, 1997] Resnik, P. (1997). Selectional Preference and Sense Disambiguation. In *Proceedings of the ACL SIGLEX Workshop on Tagging Text with Lexical Semantics: Why, What, and How?*

[Ribas, 1994] Ribas, F. (1994). An Experiment on Learning Appropriate Selectional Restrictions from a Parsed Corpus. In *Proceedings of COLING-94*, pages 769–774.

[Ribas, 1995] Ribas, F. (1995). On Learning More Appropriate Selectional Restrictions. In *Proceedings of the 7th Conference of the European Chapter of the Association for Computational Linguistics*, Dublin, Ireland.

[Rooth, 1996] Rooth, M. (1996). Two-dimensional clusters in grammatical relations. Unpublished.

[Schütze, 1992] Schütze, H. (1992). Dimensions of Meaning. In *Proceedings of Supercomputing*, pages 787–796.

[Yarowsky, 1992] Yarowsky, D. (1992). Word Sense Disambiguation Using Statistical Models of Roget's Categories Trained on Large Corpora. In *Proceedings of COLING-92*, pages 454–460, Nantes, France.

[Yarowsky, 1995] Yarowsky, D. (1995). Unsupervised Word Sense Disambiguation Rivaling Supervised Methods. In *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics*, pages 189–196.

[Zwicky, 1971] Zwicky, A. (1971). In a Manner of Speaking. *Linguistic Inquiry*, 2:223–233.