INSTITUTO POLITECNICO NACIONAL
CENTRO DE INVESTIGACION EN COMPUTACION
Laboratorio de Lenguaje Natural y Procesamiento de Texto

Automatic Extraction of Lexical Functions

TESIS

QUE PARA OBTENER EL GRADO DE
DOCTOR EN CIENCIAS DE LA COMPUTACION

PRESENTA

M. en C. OLGA KOLESNIKOVA

Director:
Dr. Alexander Gelbukh

MEXICO, D.F.   2011

Acknowledgements

Resumen

Función léxica es un concepto que formaliza relaciones semánticas y sintácticas entre palabras. Estas relaciones son muy importantes para cualquier sistema de procesamiento de lenguaje natural.

Una relación colocacional es un tipo de relación léxica entre la base de una colocación y su colocado. Una colocación es una combinación de palabras en la cual las palabras no tienen sus significados típicos. Ejemplos de colocaciones son *to give a lecture*, impartir una conferencia, *to make a decision*, tomar una decisión. En esas colocaciones las bases son *lecture*, conferencia, *decision*, decisión, *support*, apoyo, y los colocados son *give*, impartir, *make*, tomar, *lend*, dar. Sin embargo, cuando se utiliza el significado típico, las combinaciones de palabras se conocen como combinaciones libres. Ejemplos de combinaciones libres son: *to give a book*, dar un libro, *to make a dress*, hacer un vestido, *to lend money*, prestar dinero.

Existen muchos métodos de extracción automática de colocaciones. El resultado más común de estos métodos son las listas de colocaciones. Las listas de colocaciones contienen solamente las palabras que forman la colocación y la frecuencia de éstas. En este estudio nos enfocamos en agregar información semántica y gramatical a las listas de colocaciones. Esta información es extraída en forma de funciones léxicas. Dicha información hace que las colocaciones sean más útiles para aplicaciones computacionales tales como: analizadores morfológicos, traducción automática de alta calidad, sistemas de paráfrasis y aprendizaje de idiomas extranjeros asistido por computadora. Entonces nuestra meta es extraer funciones léxicas de tipo verbo-sustantivo en español de un corpus. Para lograr eso, proponemos representar el significado léxico de una palabra con el conjunto de sus hiperónimos del diccionario Spanish WordNet y usar métodos de aprendizaje de maquina supervisados para predecir funciones léxicas de colocaciones desconocidas. Evaluamos varios algoritmos de aprendizaje de máquina utilizando el conjunto de entrenamiento y el conjunto independiente de prueba. Los resultados obtenidos (el valor de la medida F-measure de 75%) muestran que es posible detectar funciones léxicas automáticamente.

Contents

List of Tables

List of Figures

Resumen

Función léxica es un concepto que formaliza relaciones semánticas y sintácticas entre palabras. Estas relaciones son muy importantes para cualquier sistema de procesamiento de lenguaje natural. Por ejemplo, para seleccionar el significado de una palabra se necesita identificar sus relaciones con otras palabras en el contexto.

Una relación colocacional es un tipo de relación léxica entre la base de una colocación y su colocado. Una colocación es una combinación de palabras en la cual las palabras no tienen sus significados típicos. Ejemplos de colocaciones son *to give a lecture*, impartir una conferencia, *to make a decision*, tomar una decisión, to *lend support,* dar apoyo. En esas colocaciones las bases son *lecture*, conferencia, *decision*, decisión, *support,* apoyo, y los colocados son *give*, impartir, *make*, tomar, *lend*, dar. Sin embargo, cuando se utiliza el significado típico, las combinaciones de palabras se conocen como combinaciones libres. Ejemplos de combinaciones libres son: *to give a book*, dar un libro, *to make a dress*, hacer un vestido, *to lend money,* prestar dinero.

Existen muchos métodos de extracción automática de colocaciones. El resultado más común de estos métodos son las listas de colocaciones. Las listas de colocaciones contienen solamente las palabras que forman la colocación y la frecuencia de éstas. En este estudio nos enfocamos en agregar información semántica y gramatical a las listas de colocaciones. Esta información es extraída en forma de funciones léxicas. Dicha información hace que las colocaciones sean más útiles para aplicaciones computacionales tales como: analizadores morfológicos, traducción automática de alta calidad, sistemas de paráfrasis y aprendizaje de idiomas extranjeros asistido por computadora. Entonces nuestra meta es extraer funciones léxicas de tipo verbo-sustantivo en español de un corpus. Para lograr eso, proponemos representar el significado léxico de una palabra con el conjunto de sus hiperónimos del diccionario Spanish WordNet y usar métodos de aprendizaje de maquina supervisados para predecir funciones léxicas de colocaciones desconocidas. Evaluamos varios algoritmos de aprendizaje de máquina utilizando el conjunto de entrenamiento y el conjunto independiente de prueba. Los resultados obtenidos (el valor de la medida F-measure de 75%) muestran que es posible detectar funciones léxicas automáticamente.

Abstract

Lexical function is a concept which formalizes semantic and syntactic relations between lexical units. Relations between words are a vital part of any natural language system. Meaning of an individual word largely depends on various relations connecting it to other words in context. Collocational relation is a type of institutionalized lexical relations which holds between the base and its partner in a collocation (examples of collocations: *gives a lecture*, *make a decision*, *lend support* where the bases are *lecture*, *decision*, *support* and the partners, termed collocates, are *give*, *make*, *lend*). Collocations are opposed to free word combination where both words are used in their typical meaning (for example, *give a book*, *make a dress*, *lend money*). Knowledge of collocation is important for natural language processing because collocation comprises the restrictions on how words can be used together. There are many methods to extract collocations automatically but their result is a plain list of collocations. Such lists are more valuable if collocations are tagged with semantic and grammatical information. The formalism of lexical functions is a means of representing such information. If collocations are annotated with lexical functions in a computer readable dictionary, it will allow effective use of collocations in natural language applications including parsers, high quality machine translation, periphrasis system and computer-aided learning of lexica. In order to create such applications, we need to extract lexical functions from corpora automatically.

It is our intent to extract Spanish verb-noun collocations belonging to a given lexical function from corpora. To achieve this task, it has been proposed to represent the lexical meaning of a given word with a set of all its hyperonyms and to use machine learning techniques for predicting lexical functions as values of the class variable for unseen collocations. Hyperonyms are extracted from the Spanish WordNet. We evaluate many machine learning algorithms on the training set and on an independent test set. The obtained results show that machine learning is feasible to achieve the task of automatic detection of lexical functions.

Chapter 1. Introduction

1.1. Lexical Functions: Preliminaries

In this section, we introduce the concept of lexical functions with simple illustration. The definition and more profound explanation will be given in Chapter 3, Section 3.3.

It does not surprise us that *a bank* can be a financial institution as well as a piece of land. It is quite often that one word is used with different meanings. But sometimes the opposite happens: we choose different words to express the same idea. For example, *to give a smile* means *to smile*, and *to lend support* means *to support*. These two combinations convey the same idea: *to smile* is to *"perform"*, or *"do" a smile*, and *to support* is to *"do" support*, so that both verb-noun combinations share the same semantics: to *do* what is denoted by the noun. Likewise we find that *to acquire popularity* and *to sink into despair* both mean 'to *begin to experience* the <noun>', and *to plant a garden* and *to write a letter* mean 'to *create* the <noun>'. Such semantic patterns or classes are called **lexical functions**.

The notion of lexical functions is closely related to another linguistic concept, i.e., collocations. Again, collocations are presented in a simple, illustrative and informal way here. More details are given in Chapter 3, Section 3.2.2 and in the same chapter, Section 3.4.

The meaning of word combination such as *give a book* or *lend money* can be obtained by mechanically combining the meaning of the two constituting words: to *give* is to hand over, a *book* is a pack of pages, then to *give a book* is to hand over a pack of pages. However, the meaning of such word combinations as *give a lecture* or *lend support* is not obtained in this way: to *give a lecture* is not to hand it over. Such word pairs are called collocations. Collocations are difficult for a computer system to analyze or generate because their meaning cannot be derived automatically from the meaning of their constituents. However, this difficulty could be resolved, if each collocation is assigned its respective lexical function, since the latter represents the collocational meaning.

More than 70 lexical functions have been identified. Each lexical function represents a group of word combinations possessing the same semantic content. If we construct a database of word combinations annotated with semantic data in the form of lexical functions, natural language processing systems will have more knowledge at their disposal to fulfill such an urgent and challenging task as word sense disambiguation. Lexical functions can also be helpful in text generation, automatic translation, text paraphrase, among others.

1.2. Objective

Our objective is to construct a database of collocations annotated with lexical functions. This objective can be achieved by completing the following tasks:

1. Create a lexical resource of Spanish verb-noun collocations annotated with lexical functions and senses of the Spanish WordNet [Vossen 1998, SpWN]. Collocations are to be extracted automatically from the Spanish WebCorpus [SpWC] by the Sketch Engine [SE, Kilgarriff *et al.* 2004], natural language processing software.

2. Compile a training set for machine learning experiments using the lexical resource of collocations.

3. Compile a test set of collocations. Collocations for the test set are to be extracted automatically from a corpus other than the Spanish Web Corpus.

4. Apply the machine learning methods as implemented in WEKA learning toolkit [WEKA] and evaluate them in fulfilling the task of predicting lexical functions for unseen Spanish verb-noun combinations using the training set and the test set. Find what methods are the best for predicting lexical functions chosen for the experiments.

5. For each lexical function chosen for the experiments, use the learning algorithm that predicts this function in the most effective way in order to detect what collocations in the test set belong to the lexical function in question; annotate collocations with this lexical function.

6. Save collocations annotated with lexical functions in a database.

1.3. Methodology

To achieve the objective given in Section 1.2, we use the following methods:

1. To extract collocations from the Spanish Web Corpus, we use the Sketch Engine, a program designed to process corpora.

7. To compile a lexical resource of collocations, we rely on our intuition to assign lexical functions to collocations and to tag each word in collocations with an appropriate sense of the Spanish WordNet.

8. To construct the training set and the test set accessible by machine learning methods, we use the ARFF format. The training set and the test set are built automatically with the help of a Perl program written by us for this purpose.

9. To experiment with machine learning methods, we use WEKA 3-6-2 learning toolkit.

10. To evaluate the performance of machine learning algorithms on the training set, we use 10-fold cross-validation technique.

11. To evaluate the performance of machine learning algorithms on the test set, we rely on our intuition to verify if the lexical functions predicted by classifiers for particular collocations in the test set are the same as we would choose for these collocations. In this manner, we estimate the values of precision, recall and F-measure for the classifiers. To make the output of the machine learning algorithms human readable, we process it by a Perl program written by us for this purpose whose code in given in Appendix 4.

1.4. Organization of the Document

This document is organized in seven chapters. Chapter 1 is introductory, in Chapter 2 contributions of this work are birefly outlined. Chapter 3 describes state-of-the-art theoretical framework, the basis of which is Meaning-Text Theory. It also explains a number of basic concepts among which the concept of lexical function is fundamental for our work. Chapter 4 presents data and Chapter 5 explains how our data is represented in a new way for machine learning experiments. Chapter 6 gives a detailed description of methods used to compile the training set and the test set for machine learning experiments and learning algorithms applied to fulfill the task of automatic detection of lexical functions. In Chapter 7, we present and consider the experimental results. Chapter 8 speaks of some computational applications of lexical functions, and in Chapter 9, a result analysis with respect to our contribution to linguistic research is given. Chapter 10 presents conclusions, and in Chapter 11 future work is outlined. The document concludes with a list of author's publications, awards, appendices and references.

Chapter 2. Contributions

2.1. Scientific contributions

1. A new method of lexical meaning representation has been proposed in this work. Lexical meaning is represented using the hyperonym hierarchy (in the case of our research the hyperonym hierarchy of the electronic dictionary Spanish WordNet was used). This representation has its potential use in other natural language processing tasks such as word sense disambiguation, textual entailment among others. This meaning representation has been proved effective in our case study, i.e. automatic extraction of lexical functions, realized on Spanish material.

2. A new method of textual meaning representation has been proposed in this work. Textual meaning is represented as a union of hyperonyms of all words used in a particular text. One of advantages of such representation is its simplicity, since the meaning of a set of words is represented in an additive manner, i.e. as a simple union of hyperonyms of each word in the set. Therefore, this representation makes it possible to make a sum or subtract of meanings which benefits its use in a variety of areas like information retrieval, making text summaries, text classification, etc. In the case of our research, we used this method of semantic representation for pairs of Spanish words, and it was demonstrated to be efficient.

3. In this work, the task of recognition of lexical functions in text has been achieved with a high precision. Given two works, the system can predict the corresponding lexical function. At the same time, since each lexical function has its proper semantics, the meaning of lexical function disambiguates the words presented to the system. This makes it possible to predict new meanings in texts although such new meanings do not appear in sense inventories. Therefore, lexical function recognition contributes to a better text understanding.

4. It has been demonstrated in this work that lexical functions characterize words which show a strong mutual association in texts. Such word combinations are called collocations or instutionalized combinations. Therefore, given a word and a corresponding lexical function, we can predict the other word to be used in combination with the given word. Also, the information on lexical functions makes it possible to predict new meanings and helps to generate texts in natural language.

5. A new meaning representation opens new ways for research. The metrics based on hyperonym information can be used in data structures organized hierarchically and can be applied within the Word Sense Model which is the future work.

6. The task of automatic detection of lexical functions has been formulated in a way that permits a thorough investigation of new data representations and methodologies. In other words, automatic extraction of lexical functions can not only help to resolve other tasks of computational linguistics and natural language processing, it is also an independent area of research.

2.2. Technical contributions

1. A database of hyperonyms of Spanish verb-noun combinations has been created automatically using Perl programs whose code is given in Appendix 4.

2. A database of Spanish verb-noun lexical functions has been created. This lexical resource includes 900 verb-noun pairs annotated with lexical functions and word senses of the Spanish WordNet. The database is open to the public and is located at www.Gelbukh.com/lexical-functions.

3. Best algorithms for recognition of lexical functions has been determined, see Chapter 7, Section 7.1.2.

In general, machine learning techniques have been demonstrated to be feasible to fulfill the task of automatic extraction of lexical functions. A wide-range comparison of many machine learning techniques has been made for the task of automatic detection of lexical functions, and methods have been identified that are able to give high quality predictions of lexical functions for unseen verb-noun collocations.

Chapter 3.   State of the Art

3.1. Lexical Functions: Theoretical Framework

As a concept, Lexical function (LF) was introduced in the frame of Meaning-Text Theory described in [Mel'čuk 1974, 1996]. Later on, the LF formalism was used for various purposes: word sense disambiguation, automatic translation, text paraphrasing, second language teaching, etc.

Meaning-Text Theory (MTT) was created by I.A. Mel'čuk in the 1960s in Moscow, Russia. Nowadays, the ideas that gave rise to MTT are being developed by Moscow semantic school [Apresjan 2008]. MTT was proposed as a universal theory applicable to any natural language; so far it has been elaborated on the basis of Russian, English and French linguistic data.

MTT views natural language as a system of rules which enables its speakers to transfer meaning into text (speaking, or text construction) and text into meaning (understanding, or text interpretation). However, for research purposes, the priority is given to the meaning-text transfer, since it is believed that the process of text interpretation can be explained by patterns we use to generate speech. MTT sets up a multilevel language model stating that to express meaning, humans do not produce text immediately and directly, but the meaning-text transfer is realized in a series of transformations fulfilled consecutively on various levels. Thus, starting from the level of meaning, or semantic representation, we first execute some operations to express the intended meaning on the level of deep syntax, then we go to the surface syntactic level, afterwards proceeding to the deep morphological level, then to the surface morphological level, and we finally arrive to the phonological level where text can be spoken and heard (oral text or speech). Another option is a written text which is actually speech represented by means of an orthography system created to facilitate human communication. Each transformational level possesses its own "alphabet", or units, and rules of arranging units together as well as rules of transfer from this level to the next one in the series. So at each level we obtain a particular text representation – e.g. deep syntactic representation, surface morphological representation, etc.

Semantic representation is an interconnected system of semantic elements, or network; syntactic representations are described by dependency trees; morphological and phonological representations are linear.

The most significant aspects of MTT are its syntactic theory, theory of lexical functions and the semantic component – explanatory combinatorial dictionary. The syntactic part is most fully

presented in [Mel'čuk 1993-2000], lexical functions are explained further in this work, and the best achievement in lexicography is explanatory combinatorial dictionaries for Russian and French [Mel'čuk and Zholkovskij 1984, Mel'čuk *et al.* 1984].

3.2. Lexical Functions: Relations in Word Combinations

Theory of lexical functions is presented with the purpose to apply it to classification of collocations. Therefore, we successively explain the concepts of institutionalized lexical relations, collocational relations, lexical functions and syntagmatic lexical functions.

3.2.1. Institutionalized Lexical Relations

[Wanner 2004] states, that lexical function (LF) is a concept which can be used to systematically describe "institutionalized" lexical relations. We will consider the notion of institutionalized lexical relations and show its relevance to collocation. Wanner clarifies that "a lexical relation is institutionalized if it holds between two lexical units L_1 and L_2 and has the following characteristics: if L_1 is chosen to express a particular meaning *M*, its choice is predetermined by the relation of M to L_2 to such an extent that in case M and L_2 is given, the choice of L_1 is a language-specific automatism." Institutionalized lexical relations can be of two types – paradigmatic and syntagmatic. Paradigmatic relations are those between a lexical unit and all the other lexical units within a language system (as between synonyms, hyperonyms and hyponyms, etc.) and syntagmatic relations are those between a lexical unit and other lexical units that surround it within a text. These are some examples of words between which there exist institutionalized lexical relations: feeling – emotion, move – movement, snow – snowflake, acceptance – general, argument – reasonable, make – bed, advice – accept, etc. In the first three pairs of words we observe paradigmatic institutionalized lexical relations and the syntagmatic ones in the rest of the examples.

In the above definition the term lexical unit is used to define a form-meaning composite that represents a lexical form and single meaning of a lexeme [Loos 1997]. The phrase "institutionalized lexical relation" does not have the meaning of the relation between lexical units which build a cliché as in [Lewis 1994]. Speaking of multi-word entities as a whole, Lewis divides them into two groups: institutionalized expressions and collocations. He defines collocations as made up of habitually co-occurring individual words. Then he adds that collocations tell more about the content of what a language user expresses as opposed to institutionalized expressions which tell more about what the language user is doing, e.g.

agreeing, greeting, inviting, asking, etc. Institutionalized lexical relations in [Wanner 2004] possess the quality of association present between habitually co-occurring words, or collocations, if we consider the syntagmatic type of institutionalized lexical relations. Indeed, it can be seen from definitions of collocation (see Appendix 2) that the relations between collocation components, i.e. between the base and the collocate, are characterized by high probability of occurrence of the collocate given the base and by arbitrariness of the collocate choice. The latter is emphasized by [Lewis 1997] who insists that collocations do not result from logic but are decided only by linguist convention.

We can notice that the concept of institutionalized lexical relations is wider than the concept of relations between collocation elements. As it was mentioned in the beginning, institutionalized lexical relations can be paradigmatic and syntagmatic. Paradigmatic relations connect lexical units along the vertical axis of a language system while syntagmatic relations connect words positioned along the horizontal axis of the language; they tie together lexical units within a linear sequence of oral utterance or a written text as, for example, they link the base and its collocate in a collocation. So the relations within a collocation are institutionalized syntagmatic lexical relations which can be also termed collocational relations.

3.2.2. Collocational Relations

[Wanner 1996] gives the following definition of collocational relation. "A collocational relation holds between two lexemes L_1 and L_2 if the choice of L_1 for the expression of a given meaning is contingent on L_2, to which this meaning is applied. Thus, between the following pairs of lexical units collocational relations hold: *to do*: *a favor*, *to make*: *a mistake*, *close*: *shave*, *narrow*: *escape*, *at*: *a university*, *in*: *a hospital*." The term **lexeme** used in the above definition is the minimal unit of language which has a semantic interpretation and embodies a distinct cultural concept, it is made up of one or more form-meaning composites called lexical units [Loos 1997].

As it is seen from the definition, a collocational relation holds between components of non-free word combinations, i.e. such combinations whose semantics is not fully compositional and has to be partially or entirely derived from the phrase as a whole. Non-free combinations are opposed to free word combinations where syntagmatic relations hold between words in a phrase with purely compositional semantics. Examples of free word combinations are: *a black cable, a different number, to put a chair* [*in the corner*], *to write a story, to run quickly, to decide to do something*. Examples of non-free word combinations are: *a black box, as different again, to put into practice, to write home about, to run the risk of* [*being fired*], *to decide on* [*a hat*]. The

distinction between free and non-free combinations is a general distinction usually made in linguistic research with respect to syntagmatic relations.

Collocational relations can be classified according to lexical, structural and semantic criteria. The most fine-grained taxonomy of collocations based on semantic and structural principle was given by [Mel'čuk 1996]. This taxonomy uses the concept of lexical function which we will consider now.

3.3. Lexical Functions: Definition and Examples

The concept of lexical function was first introduced in [Mel'čuk 1974] within the frame of a Meaning-Text linguistic model. It is a way to organize collocational data with the original purpose of text generation. Here we will supply the definition of lexical function from [Mel'čuk 1996].

"The term **function** is used in the mathematical sense: $f(X) = Y$. …Formally, a Lexical Function f is a function that associates with a given lexical expression L, which is the argument, or keyword, of f, a set $\{L_i\}$ of lexical expressions – the value of f – that express, contingent on L, a specific meaning associated with f:

$$f(\text{L}) = \{\text{L}_i\}.$$

Substantively, a Lexical Function is, roughly speaking, a special meaning (or semantico-syntactic role) such that its expression is not independent (in contrast to all "normal" meanings), but depends on the lexical unit to which this meaning applies. The core idea of Lexical Functions is thus lexically bound lexical expression of some meanings."

To explain the concept of lexical function in a simple and illustrative way, let us consider two sentences where the Spanish verb *dar*, give, is used.

(1) *Te quiero dar un regalo*, lit., I want to give you a gift.
(2) *Quero dar un paseo*, lit., I want to give a walk, the correct translation: I want to take a walk.

In the first sentence the meaning of the phrase *dar un regalo*, give a gift, is composed of the meaning of *dar* and the meaning of *regalo*. That is, the meaning of this phrase is a simple sum of the meaning of its components. This is true for free word combinations to which *dar un regalo* certainly belongs. If we observe the phrase *dar un paseo* in sentence (2), we notice that its meaning can not be represented as a sum of the meanings of its components. The noun *paseo* is used in its common or most frequent meaning. However, this is not the case of the verb *dar*. It

adds such a semantic component to the meaning of *dar paseo* that makes the whole phrase convey the idea of *pasear*, to walk, or "*efectuar*" *un paseo*, "carry out" a walk. It can also be mentioned that in order to express the meaning *efectuar* with *paseo*, only the verb *dar* is chosen, and in fact this is the only choice because no other verb would be used in such a case by a native Spanish speaker. This restrictive word co-occurence is characteristic of collocations to which *dar un paseo* belongs. The same phenomenon is observed in the sentences

(1) *Toma la mano*, lit, Take the hand.
(2) *Toma la decisión*, lit. Take the decision, the correct translation: Make the decision.

In the first sentence we see a free word combination, and in the second sentence – a collocation *tomar la decisión*, make the decision, which has the meaning "efectuar" la decisión, "carry out" the decision. Now, we have two collocations with the same semantic component **efectuar**: *dar un paseo* and *tomar una decisión*. Designating the semantic element **efectuar** by **Oper** (abbreviated Latin verb *operor,* carry out) and using mathematical notation, we write the following:

Oper(paseo) = dar,
Oper(decisión) = tomar.

Oper is the name of a lexical function, its value for the argument, or keyword *paseo* is *dar*, for the keyword *decisión* – *tomar*. We can say that the generalized meaning or gloss of the lexical function **Oper** is **efectuar**. Here are other examples of **Oper**:

Oper(*conferencia*, conference) = *impartir*, give,
Oper(*anuncio*, announcement) = *hacer*, make.

Oper values make up collocations with their respective keywords: *impartir una conderencia*, *hacer un anuncio*, where the keyword is the base and the LF value is the collocate.

Thus, lexical function **Oper** denotes the collocational relation with the gloss 'efectuar', 'perform', 'experience', 'carry out'. Other glosses, and therefore, lexical functions can be distinguished among collocational relations. Consider some LFs in Table 1.

Table 1. Examples of lexical functions

LF	Name description	Gloss	Keyword	Value	Collocation
Fact	from Latin *factum*, fact	accomplish itself	*sueño,* dream	*cumplirse,* come true	*el sueño se cumplió*, the dream came true

Real	from Latin *realibus* , real	fulfill the requirement contained in the argument	*invitación*, invitation	*aceptar*, accept	*aceptar una invitación*, accept an invitation
Caus	from Latin *causa*, cause	cause to exist	*asociación*, association	*fundar*, found	*fundar una asociación*, found an association
Magn	from Latin *magnus*, big	intense, intensely, very (intensifier)	*temperatura*, temperature	*alta*, high	*la temperatura alta*, high temperature

Latin abbreviations are used for the names of lexical functions. The names are accompanied by numeric subscripts. They signify how the LF argument and the LF semantic structure are projected onto the syntactic structure of the LF value. For example, used as a subscript for **Oper,** 1 means that the Agent (the first participant) in the situation denoted by the argument is the verb's subject and the argument itself is its object. For example:

Oper$_1$(*demanda*, demand) = *presentar*, present (*presentar una demanda*, present a demand).

Remember that the gloss of Oper$_1$ is **efectuar**. As a subscript for the same LF, 2 means that the verb's subject is the Undergoer (the second participant) in the situation denoted by the argument. For example:

Oper$_2$(*culpa*, guilt) = *tener*, have (*tener la culpa*, have the guilt, be guilty).

The gloss of Oper$_2$ is **experimentar**. Zero (0) as a subscript for the lexical function **Func** (Latin *fungor*, realizar) shows that the LF argument is the subject of an intransitive verbal value. For example:

Func$_0$(*viento*, wind) = *soplar*, blow (*el viento sopla*, the wind blows).

3.4. Lexical Functions and Collocations

As it was mentioned in the beginning, lexical function is a concept used to systematically describe "institutionalized" lexical relations, both paradigmatic and syntagmatic. Since we are interested in collocational relations, i.e. syntagmatic institutionalized lexical relations, only syntagmatic lexical functions will be considered in this work.

3.4.1. Syntagmatic Lexical Functions

Now we provide a definition of syntagmatic lexical function [Wanner 2004]. A syntagmatic LF is a (directed) standard abstract relation that holds between the base A and the collocate B of the collocation A$^\oplus$B and that denotes 'C' $\in$ 'A$^\oplus$C' with 'A$^\oplus$C' being expressed by A$^\oplus$B. 'Directed' means that the relation is not symmetric. 'Standard' means that the relation apples to a

large number of collocations. 'Abstract' means that the meaning of this relation is sufficiently general and can therefore be exploited for purposes of classification.

Alongside with formalizing semantic information, lexical functions specify syntactic patterns of collocations. For this purpose, subscripts are used with the names of LFs as explained above. Subscripts identify syntactic functions of words denoting basic thematic roles associated with LF argument. We will not take semantic roles into account in our research and will treat subscripts as a part of LF name. LF names serve as tags with which collocations are to be annotated as a result of classification.

LFs can be grouped according to parts of speech to which collocational components belong. The following classes of collocations are distinguished:

1. Verb-noun: *to make a decision, to take a walk*
2. Noun-noun: *heart of the desert, prime of life*
3. Adjective-noun: *infinite patience, strong tea*
4. Adverb-verb: *to laugh heartily, to walk steadily*
5. Noun-preposition: *on the typewriter, by mail*
6. Verb-preposition: *to fly by* [*plane*], *to go to* [*the park*]

About 20 of standard simple LFs capture verb-noun collocations. Simple LFs can further combine to form complex LFs. Complex LFs reflect compositional semantics of collocates. For example,

Magn + Oper$_1$(*laughter*) = *to roar* [*with laughter*],

where *roar* means **do** [= Oper$_1$] **big** [= Magn] **laughter**. In complex LFs, or in a configuration of LFs, the syntactically central LF which determines the part of speech of the configuration and the value is written rightmost. In the above example the value is a verb, so **Oper** is written rightmost in the configuration:

MagnOper$_1$(*laughter*) = *roar* [*with ~*].

3.4.2. Syntagmatic Lexical Functions as a Collocation Typology

Collocation definition has been a controversial issue for a number of years. Various criteria have been suggested to distinguish collocations from free word combinations (see Appendix 2). Definitions based on statistical distribution of lexical items in context cover frequently encountered collocations but such collocations as feeble imagination are overlooked since they occur rarely in corpus and thus are not considered collocations in the statistical sense. On the

other hand, collocation definition which suggest arbitrariness of lexical choice of the collocate depending on the base does not encompass such phrases as strong man and powerful neighbor which are considered recurrent free combinations.

We are interested in a collocation classification that can give an insight in the collocational meaning. Syntagmatic lexical functions have glosses which represent a semantic component added to the collocational base to form the meaning of a collocation. A gloss is an element of meaning found common in a group of collocations. Such groups of collocations form classes, each of the classes is associated with a particular lexical function. For verb-noun collocations, 20 lexical functions are identified. Compared to the only other existing semantic and syntactic typology, the one proposed by [Benson *et al*. 1997], which includes 8 types of grammatical collocations and 7 types of lexical collocations, the LF typology is very fine-grained. Verb-noun lexical functions are listed and exemplified in Appendix 3.

3.5. Automatic Extraction of Lexical Functions

Not much research has been done on automatic detection of lexical functions. In fact, there are only two papers that report results on performance of a few machine learning algorithms on classifying collocations according to the typology of lexical functions, [Wanner 2004], [Wanner *et al*. 2006]. In this section, we will summarize the work done in [Wanner 2004], [Wanner *et al*. 2006], then we will also mention another research on automatic extraction of lexical functions [Alonso Ramos *et al*. 2008] based on an approach different from the work in [Wanner 2004] and [Wanner *et al*. 2006]. We conclude this section with three statements, or hypotheses, made in [Wanner *et al*. 2006].

3.5.1. Research in [Wanner 2004] and [Wanner et al. 2006]

In 2004, L. Wanner proposed to view the task of LF detection as automatic classification of collocations according to LF typology. To fulfill this task, the nearest neighbor machine learning technique was used. Datasets included Spanish verb-noun pairs annotated with nine LFs: $CausFunc_0$, $Caus_2Func_1$, $IncepFunc_1$, $FinFunc_0$, $Oper_1$, $ContOper_1$, $Oper_2$, $Real_1$, $Real_2$. Verb-noun pairs were divided in two groups. In the first group, nouns belonged to the semantic field of emotions; in the second groups nouns were field-independent. As a source of information for building the training and test sets, hyperonymy hierarchy of the Spanish part of EuroWordNet was used. The words in the training set were represented by their hyperonyms, Basic Concepts and Top Concepts. The average *F-measure* of about 70% was achieved in these experiments.

The best result for field-independent nouns was F-measure of 76.58 for $CausFunc_0$ with the meaning 'cause the existence of the situation, state, etc.' The Causer is the subject of utterances with $CausFunc_0$.

In [Wanner *et al*. 2006], four machine learning methods were applied to classify Spanish verb-noun collocations according to LFs, namely Nearest Neighbor technique, Naïve Bayesian network, Tree-Augmented Network Classification technique and a decision tree classification technique based on the ID3-algorithm. As in [Wanner 2004], experiments were carried out for two groups of verb-noun collocations: nouns of the first group belonged to the semantic field of emotions; nouns of the second group were field-independent. Lexical functions were also identical with [Wanner 2004] as well as data representation. The best results for field-independent nouns were shown by ID3 algorithm (F-measure of 0.76) for $Caus_2Func_1$ with the meaning 'cause something to be experienced / carried out / performed', and by the Nearest Neighbor technique (F-measure of 0.74) for $Oper_1$ with the meaning 'perform / experience / carry out something'. The Causer is the subject of utterances with $Caus_2Func_1$, and the Agent is the direct object of the verb which is the value of $Cuas_2Func_1$. In utterances with $Oper_1$, the Agent is the subject.

As we are interested in experimented with verb-noun collocations where the nouns are have various semantics, i.e., the nouns are field-independent, Tables 2—5 summarizes the results for field-independent nouns only in [Wanner 2004] and [Wanner *et al*. 2006]. Table 2 gives the meaning of lexical functions used in experiments only with field-independent nouns [Wanner 2004], examples in Spanish with literal translation in English; #Tr stands for the number of examples of a given LF in the training set; #T stands for the number of examples of a given LF in the test set; #Tt stands for the total number of examples of a given LF in the training set and in the test set. Table 3 lists LFs with respective number of examples in [Wanner *et al*. 2006] for verb-noun combinations with field-independent nouns. Table 4 presents the results reported in the referenced paper. Table 5 shows the results in [Wanner *et al*. 2006]; the values of precision, recall and F-measure are given in the following format: <precision> | <recall> | <F-measure>. Not all four machine learning methods in Table 4 were applied to all LFs; if experiments were not made for a particular method and LF, N/A is put instead of precision, recall, and F-measure.

Table 2. Data in [Wanner 2004]

Name	Meaning	Examples in Spanish	Lit. translation in English	#Tr	#T	#Tt
$Oper_1$	experience, perform, carry out something	dar golpe presentar una demanda hacer campaña	give a blow present a demand make a campaign	35	15	50

		dictar la sentencia	dictate a sentence			
Oper$_2$	undergo, be source of	someterse a un análisis afrontar un desafío hacer examen tener la culpa	submit oneself to analysis face a challenge make exam have guilt	33	15	48
CausFunc$_0$	cause the existence of the situation, state, etc.	dar la alarma celebrar elecciones provocar una crisis publicar una revista	give the alarm celebrate elections provoke a crisis publish a magazine	38	15	53
Real$_1$	act accordingly to the situation, use as forseen	ejercer la autoridad utilizar el teléfono hablar la lengua cumplir la promesa	exercise authority use a telephone speak a language keep a promise	37	15	52
Real$_2$	react accordingly to the situation	responder a objeción satisfacer un requisito atender la solicitud rendirse a persuasión	respond to an objection satisfy a requirement attend an application surrender to persuasion	38	15	53

Table 3. Data in [Wanner *et al.* 2006]

LF	Number of Examples
CausFunc$_0$	53
Oper$_1$	87
Oper$_2$	48
Real$_1$	52
Real$_2$	53

Table 4. Results in [Wanner 2004]

F-measure/LF	CausFunc$_0$	Oper$_1$	Oper$_2$	Real$_1$	Real$_2$
field-independent nouns	76.58	60.93	75.85	74.06	58.32

Table 5. Results in [Wanner *et al.* 2006]

LF	Machine learning technique			
	NN	NB	TAN	ID3
CausFunc$_0$	0.59 | 0.79 | 0.68	0.44 | 0.89 | 0.59	0.45 | 0.57 | 0.50	N/A
Caus$_2$Func$_1$	N/A	N/A	N/A	0.53 | 0.65 | 0.50
FinFunc$_0$	N/A	N/A	N/A	0.53 | 0.40 | 0.40
IncepFunc$_1$	N/A	N/A	N/A	0.40 | 0.48 | 0.40
Oper$_1$	0.65 | 0.55 | 0.60	0.87 | 0.64 | 0.74	0.75 | 0.49 | 0.59	0.52 | 0.51 | 0.50
Oper$_2$	0.62 | 0.71 | 0.66	0.55 | 0.21 | 0.30	0.55 | 0.56 | 0.55	N/A
ContOper$_1$	N/A	N/A	N/A	0.84 | 0.57 | 0.68
Real$_1$	0.58 | 0.44 | 0.50	0.58 | 0.37 | 0.45	0.78 | 0.36 | 0.49	N/A
Real$_2$	0.56 | 0.55 | 0.55	0.73 | 0.35 | 0.47	0.34 | 0.67 | 0.45	N/A

3.5.2. Research in [Alonso Ramos et al. 2008]

[Alonso Ramos et al. 2008] propose an algorithm for extracting collocations following the pattern "support verb + object" from FrameNet corpus of examples [Ruppenhofer *et al.* 2006] and checking if they are of the type Oper$_n$. This work takes advantage of syntactic, semantic and collocation annotations in the FrameNet corpus, since some annotations can serve as indicators of a particular LF. The authors tested the proposed algorithm on a set of 208 instances. The algorithm showed accuracy of 76%. The researchers conclude that extraction and semantic classification of collocations is feasible with semantically annotated corpora. This statement sounds logical because the formalism of lexical function captures the correspondence between the semantic valency of the keyword and the syntactic structure of utterances where the keyword is used in a collocation together with the value of the respective LF.

3.5.3. Three Hypothesis Stated in [Wanner et al. 2006]

[Wanner *et al.* 2006] experiment with the same type of lexical data as in [Wanner 2004], i.e. verb-noun pairs. The task is to answer the question: what kind of collocational features are fundamental for human distinguishing among collocational types. The authors view collocational types as LFs, i.e. a particular LF represents a certain type of collocations. Three hypotheses are put forward as possible solutions, and to model every solution, an appropriate machine learning technique is selected. Below we list the three hypotheses and the selected machine learning techniques.

1. Collocations can be recognized by their similarity to the prototypical sample of each collocational type; this strategy is modeled by the Nearest Neighbor technique.

2. Collocations can be recognized by similarity of semantic features of their elements (i.e., base and collocate) to semantic features of elements of the collocations known to belong to a specific LF; this method is modeled by Naïve Bayesian network and a decision tree classification technique based on the ID3-algorithm.

3. Collocations can be recognized by correlation between semantic features of collocational elements; this approach is modeled by Tree-Augmented Network Classification technique.

It should be mentioned, that having proposed three hypotheses, the authors have not yet demonstrated their validity by comparing the performance of many machine learning techniques known today, but applied only four learning algorithms to illustrate that three human strategies mentioned above are practical. This will be considered in more detail in Chapter 7.

3.6. Automatic Detection of Semantic Relations

There has been some research done on semantic relations in word combinations, for example, one that deals with automatic assignment of semantic relations to English noun-modifier pairs in [Nastase and Szpakowicz 2003, Nastase *et al*. 2006]. Though in our work, verb-noun combinations are treated, we believe that the principles of choosing data representation and machine learning techniques for detection of semantic relations between a noun and a modifier can also be are used to detect semantic relations in verb-noun pairs. The underlying idea is the same: learning the meaning of word combinations. In [Nastase and Szpakowicz 2003, Nastase *et al*. 2006], the researchers examined the following relations: causal, temporal, spatial, conjunctive, participant, and quality. They used two different data representations: the first is based on WordNet relations, the second, on contextual information extracted from corpora. They applied memory-based learning, decision tree induction and Support Vector Machine. The highest F-measure of 0.847 was achieved by C5.0 decision tree to detect temporal relation based on WordNet representation.

Chapter 4.   Data

We have created a unique lexical resource of Spanish lexical functions in order to compile training sets for machine learning experiments.

4.1. Data for the Training Sets

4.1.1. Lexical Resources

Lexical resources are widely used in natural language processing and their role is difficult to overestimate. Lexical resources vary significantly in language coverage and linguistic information they include, and have many forms: word lists, dictionaries, thesauri, ontologies, glossaries, concordances, etc. For Spanish, this diversity of forms can be illustrated with the following lexicographic works: A Medieval Spanish Word List [Oelschläger 1940], Diccionario de la Lengua Española (Dictionary of the Spanish Language) [RAE 2001], Streetwise Spanish Dictionary/Thesaurus [McVey and Wegmann 2001], Spanish part of EuroWordNet [Vossen 1998], an electronic lexical onthology, Glosario de voces comentadas en ediciones de textos clásicos [Fontecha 1941], Concordancia electrónica de la Biblia online (for Reina Valera version, 1960) [CEB]. Machine readable resources are of special interest, since they comprise an integral part of computer systems aimed at automatic language treatment and language generation.

Though computerized lexicography has achieved a significant progress over last years, compilation of high quality dictionaries still requires a lot of manual work. In such a multi-faceted area as computational linguistics, it is difficult sometimes to find an adequate lexical resource (and for the language you need) for a specific research task or application. One way to solve this problem is to develop computational procedures which can adjust existing resources to the demands of a researcher. However, this is not always effective. Certainly, the best solution of this problem is to compile a new lexical resource, but this is not always feasible in view of its cost.

We present a list of most frequent Spanish verb-noun pairs which contains semantically annotated collocations and free word combinations. It is a machine readable lexical resource where each verb-noun pair is associated with the following linguistic data:

1. whether a pair is a free word combination or a collocation;

2. if a verb-noun pair is a collocation, it is marked with lexical functions;

3. word senses of the Spanish WordNet [Vossen 1998, SpWN] are assigned to both elements of the verb-noun pair.

4.1.2. Work on Lexical Resources

In this Section, we give a number of lexical resources that contain lexical functions. Almost all of them are not specialized dictionaries of lexical functions, but include lexical functions together with other linguistic information.

The concept of lexical function was originally proposed by researchers of the Russian semantic school. Lexical functions have been applied there for description of lexica and machine translation. A dictionary in Russian compiled by Apresjan [referenced in Apresjan 2004] for the machine translation system ETAP includes more that 100 lexical functions with definitions and examples. For instance, for the verbal lexical function $Oper_1$, the dictionary contains several hundreds of samples.

Lexical functions are used to describe the word's combinatory power in Explanatory Combinatorial Dictionaries compiled for Russian [Mel'čuk and Zholkovskij 1984] and for French [Mel'čuk et al. 1984, 1988]. For every word, its lexical entry includes a list of lexical functions applicable to it with their respective values. For French, an on-line dictionary, the DiCo, is referenced in [Wanner 2004] but we could not access it on the web.

For Spanish, there exists a dictionary of collocations, Diccionario de colocaciones del Español [DiCE] [Alonso Ramos 2003] annotated with lexical functions, but the DiCE is limited only to nouns belonging to the semantic field of emotions. [Sanromán 1998, 2003] compiled collections of Spanish collocations also for emotion nouns classified in terms of lexical functions. [Wanner 2004], [Wanner et al. 2006] used Sanromán's collections for machine learning experiments, and for the same purpose, compiled additional lists of Spanish verb-noun collocations annotated with lexical functions. In the additional lists nouns were semantically field independent. The overall number of LF instances in the latter lists were 256 [Wanner 2004] and 293 [Wanner et al. 2006]. Unfortunately, these lists are no longer available in full.

4.1.3. Description of the Lexical Resource

Compilation

Firstly, the Spanish Web Corpus [SpWC] was chosen as a source of verb-noun pairs with the pattern verb + direct object. All such verb-noun pairs used in the Spanish Web Corpus five or more times, were extracted automatically from the said corpus by the Sketch Engine [Kilgarriff

et al. 2004], a web-based program for corpus processing. Fig. 1 displays the interface of the Sketch Engine where several corpora are listed including the Spanish Web Corpus. The obtained list contained 83,982 verb-noun pairs, and it was ranked by frequency.

Fig 1. Sketch Engine with the Spanish Web Corpus.

The software is on the Sketch Engine website: http://www.sketchengine.co.uk/

Follow the links from this page to either set up an account, or log in. The "home" screen looks like this:

Secondly, one thousand pairs were taken from the upper part of the list, i.e. most frequent verb-noun pairs.

Thirdly, in the list of one thousand pairs, erroneous combinations were marked with the label ERROR. Erroneous pairs included, for instance, past participle or infinitive instead of noun, or contained symbols like --, « , © instead of words. How did errors emerge? The automatic extraction procedure was set to search for combinations with the pattern verb + direct object in the corpus. This procedure needs part of speech (POS) and lemma information, and such data is supplied by TreeTagger, software used to annotate the Spanish Web Corpus with POS and lemmas. The TreeTagger is a leading tool applied for POS tagging and lemmatisation, it achieves high accuracy but still is error-prone. Due to errors made by the TreeTagger, the set of extracted verb-noun pairs contained fallacious combinations. For the sake of preserving the

original design of automatically extracted set, these incorrect combinations were not removed from the list but identified as wrong. The total number of erroneous pairs was 61, so after their removal the list contained 939 pairs.

Fourthly, collocational verb-noun pairs were annotated with lexical functions. The rest of the pairs were annotated as free word combinations using the label FWC.

Lastly, all verbs and nouns in the list were disambiguated with word senses from the Spanish WordNet, an electronic lexicon structured the same way as WordNet for English. For some verb-noun pairs, relevant senses were not found in the above mentioned dictionary, and the number of such pairs was 39. For example, in the combinaiton *dar cuenta*, *give account*, the noun *cuenta* means razón, *satisfacción de algo* (*reason, satisfaction of something*). This sense of *cuenta* is taken from Diccionario de la Lengua Española (Dictionary of the Spanish Language) [RAE 2001]. Unfortunately, this sense is absent in the Spanish WordNet so the expression *dar cuenta* was left without sense annotation. All such words were annotation N/A, i.e. not available.

The annotated list was formatted as a table and saved in an MS Excel file. Fig. 2 shows the process of the compilation of the lexical resource schematically.

Fig. 2. The process of lexical resource compilation.
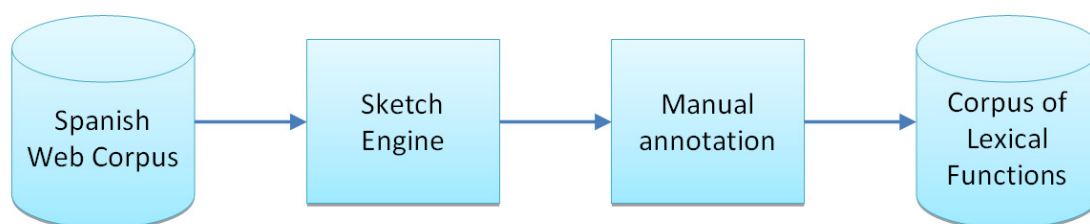
Contents of the lexical resource

A partial representation of the list is given in Table 6; Table 7 lists all lexical functions found in the list of 1000 most frequent verb-noun pairs, their frequencies in the Spanish Web Corpus, and the number of examples for each of them.

Table 6. Partial representation of the lexical resource

LF/ FWC/ ERROR	VERB	Verb Sense Number	NOUN	Noun Sense Number	FREQ
Oper_1	dar	2	cuenta	N/A	9236
CausFunc_0	formar	2	parte	1	7454
Oper_1	tener	1	lugar	4	6680
Oper_1	tener	1	derecho	1	5255
CausFunc_1	hacer	2	falta	N/A	4827
CausFunc_1	dar	9	lugar	4	4180
Oper_1	hacer	15	referencia	2	3252
Func_0	hacer	N/A	año	2	3211
Oper_1	tener	1	problema	7	3075
Func_0	hacer	N/A	tiempo	1	3059
IncepOper_1	tomar	4	decisión	2	2781
Oper_1	tener	1	acceso	3	2773
Oper_1	tener	1	razón	2	2768
$\text{Caus}_2\text{Func}_1$	llamar	8	atención	1	2698
Oper_1	tener	1	sentido	1	2563
ERROR	haber	ERROR	estado	ERROR	2430
FWC	hacer	6	cosa	3	2374
Oper_1	tener	3	miedo	1	2226
ERROR	haber	ERROR	hecho	ERROR	2168

Table 7. Lexical functions with their respective frequency in corpus and the number of instances in the list of verb-noun pairs

LF	Freq	#	LF	Freq	#
Oper_1	165319	280	PerfFunc_0	1293	1
FWC	70211	202	$\text{Caus}_1\text{Oper}_1$	1280	2
CausFunc_1	45688	90	$\text{Caus}_1\text{Func}_1$	1085	3
CausFunc_0	40717	112	IncepFunc_0	1052	3
ERROR	26316	61	PermOper_1	910	3
Real_1	19191	61	CausManifFunc_0	788	2
Func_0	17393	25	CausMinusFunc_0	746	3
IncepOper_1	11805	25	Oper_3	520	1
Oper_2	8967	30	LiquFunc_0	514	2
$\text{Caus}_2\text{Func}_1$	8242	16	IncepReal_1	437	2
ContOper_1	5354	16	Real_3	381	1
Manif	3339	13	PlusOper_1	370	1
Copul	2345	9	CausPerfFunc_0	290	1
CausPlusFunc_0	2203	7	AntiReal_3	284	1
Func_1	1848	4	MinusReal_1	265	1
PerfOper_1	1736	4	AntiPermOper_1	258	1
CausPlusFunc_1	1548	5	ManifFunc_0	240	1
Real_2	1547	3	CausMinusFunc_1	229	1
FinOper_1	1476	6	FinFunc_0	178	1

4.2. Data for the Test Sets

To build the test set, we extracted all verb-noun pairs from a corpus other than the corpus used to construct the training sets. So the data for test sets was mined from the Spanish Treebank Cast3LB [Civit and Martí 2004]. The number of all verb-noun pairs extracted from Cast3LB was 5181. We constructed four test sets, including, respectively, 100%, 75%, 50%, and 25% of all verb-noun pairs taken from Treebank Cast3LB.

Fig. 3. A partial representation of the list of verb-noun pairs used for building the test set.

```
v tener 1
n aire 1
v tener 1
n aire 2
...
v tener 9
n aire 10
v tener 9
n aire 11
v tener 9
n aire 12
v salir 1
n error 1
v salir 1
n error 2
v salir 1
n error 3
v salir 1
n error 4
v salir 1
n error 5
v salir 1
n error 6
...
```

We did not disambiguate verb-noun pairs for the test sets manually. Instead, for each verb-noun, we built all possible verb-noun combinations of all senses in the Spanish WordNet. As an example, let us consider the pair *representar papel*, lit. *represent role*. The verb *representar* has 12 senses in the Spanish WordNet, and the noun *papel*, 5. This gives totally 60 combinations of *representar* and *papel* (12 multiplied by 5). The initial list for the test set included 5,181 verb-noun pairs which resulted in totally 96,079 instances in the test set. A partial representation of the list is given in Fig. 3.

Chapter 5. A New Method of Meaning Representation

5.1. Data Representation

Each verb-noun pair in the training set and in the test set is represented as a set of all hyperonyms of the noun and all hyperonyms of the verb. The noun and the verb of the verb-noun pair were considered as zero-level hyperonyms and thus were included in the set of hyperonyms.

5.1.1. Hyperonyms and Hyponyms

In linguistics, a hyponym is a word or phrase whose meaning is included within the meaning of another word, its hyperonym (spelled *hypernym* in natural language processing literature). To put it simpler, a hyponym shares a type-of relationship with its hyperonym. For example, *restaurant*, *rest house*, *planetarium*, *observatory*, *packinghouse*, *outbuilding*, *Pentagon* are all hyponyms of *building* (their hyperonym), which is, in turn, a hyponym of *construction*.

In computer science, the relationship of hyperonymy is often termed an "is-a" relationship. For example, the phrase *Restaurant is a building* can be used to describe the hyponymic relationship between *restaurant* and *building*.

Thus, hyperonymy is the semantic relation in which one word is the hyperonym of another one.

5.1.2. Spanish WordNet as a Source of Hyperonyms

The Spanish Wordnet follows the EuroWordNet [Vossen 1998] framework and is structured in the same way as the American WordNet for English [Miller 1998] in terms of synsets (sets of synonymous words) with basic semantic relations between them.

Fig. 4. The Spanish WordNet, hyperonyms for *gato*, cat.
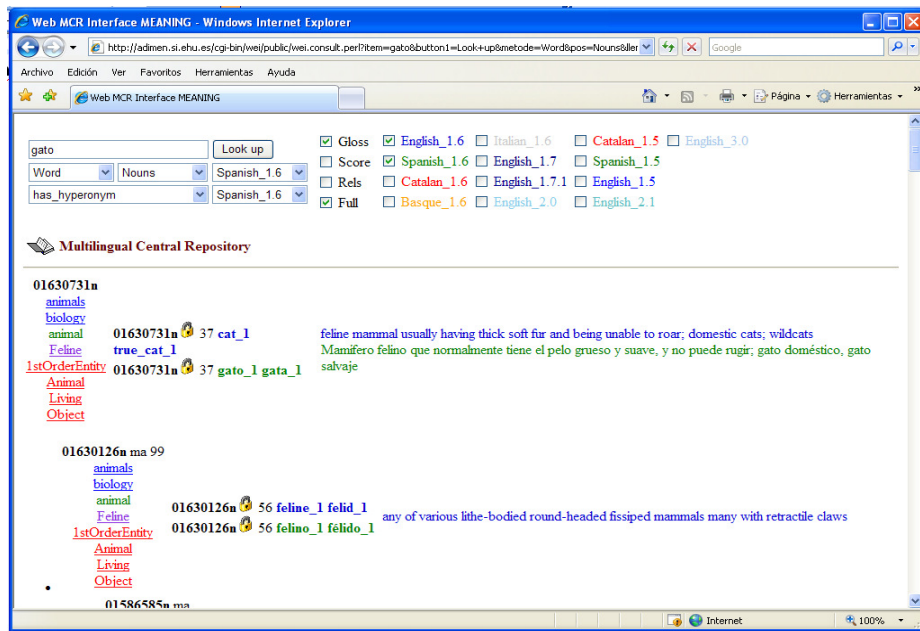
Fig. 5. The Spanish WordNet, hyperonyms for *gato*, cat (continued).
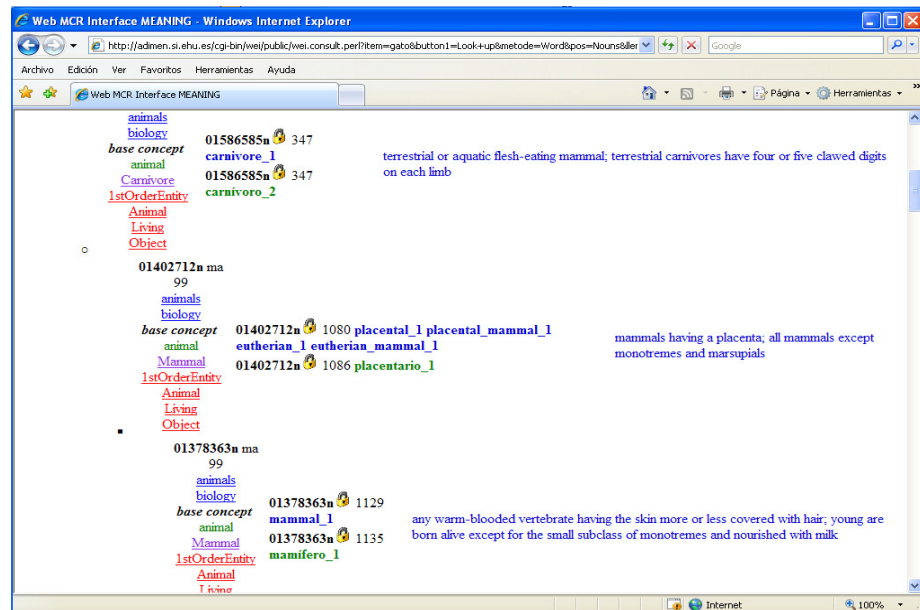
Spanish nouns and verbs are organized into synonym sets, each representing one underlying lexical concept. Different relations, for example, hyperonym relations, link the synonym sets.

Since all verbs and nouns have been disambiguated, hyperonyms can be found for each word that has been annotated with its sense of the Spanish WordNet [SpWN]. Hyperonyms were extracted automatically from the database of the dictionary referenced above. Fig. 4 and Fig. 5 display the interface of the Spanish WordNet as it is seen on the web. In the interface, we see hyperonyms of *gato*, cat.

5.1.3. Hyperonyms as a Meaning Representation

A difference between data representation in our experiments and data sets used in [Wanner *et al.* 2006] should be noted here. In the paper just referenced, every word in the training set was accompanied by its synonyms and hyperonyms, its own Base Concepts (BC) and the BCs of its hyperonyms, its own Top Concepts (TC) and the TCs of its hyperonyms taken from the Spanish part of the EuroWordNet [Vossen 1998]. We included only hyperonyms in our training sets. Though in this case the data is annotated with less information, i.e. only with hyperonyms, or in other words, only hyperonyms are used to represent the meaning of verb-noun pairs, we hope that hyperonyms would be sufficient to distinguish between lexical functions. Up to now, there has not been any research done that compares different data representations for the task of predicting lexical functions of verb-noun pairs. Here we can remember the original intent of WordNet compilers [Miller 1998] who claimed that the meaning of any word can be described sufficiently well by semantic relations only, like "is-a-kind-of" semantic relation of hyperonym hierarchy. Later, WordNet authors admitted that their previous assumption had been wrong and glosses were added to distinguish synonym sets. Though practical significance of glosses is generally accepted, we intent to study how well the meaning of lexical functions can be distinguished if only hyperonym information is taken into account. Further research is needed to investigate how information other that hyperonym taxonomy, for example, that of semantic ontologies, changes the performance of machine learning algorithms.

5.2. Linguistic Description of Training Sets and Test Sets

5.2.1. Lexical Functions Chosen for Experiments

Our choice of lexical functions depends on the number of examples that each lexical function has in the lexical resource of Spanish lexical functions created by us and described in Section 4.1. We have selected LFs that have the number of examples sufficient for machine learning experiments. [Wanner 2004] and [Wanner *et al.*2006] experimented with the following number of LF examples: the biggest number of examples that this researcher had in the training set was 87 for $Oper_1$ and the least number of examples was 33 for $Oper_2$. This data can be seen in Section 3.5.1.

Table 8. Lexical functions chosen for the experiments

LF and # of examples	Meaning	Collocation: LF value + keyword	
		Spanish	English translation
$Oper_1$ 280	Lat. *operare* – 'to do, perform'. Experience (if K is an emotion), carry out K.	*alcanzar un objetivo* *aplicar una medida* *corregir un error* *satisfacer una necesidad*	*achieve a goal* *apply a measure* *correct a mistake* *satisfy a necessity*
$CausFunc_0$ 112	Lat. *causare* – 'to cause'. Do something so that K begins occurring.	*encontrar respuesta* *establecer un sistema* *hacer campaña* *producir un efecto*	*find an answer* *establish a system* *conduct a campaign* *produce an effect*
$CausFunc_1$ 90	A person/object, different from the agent of K, does something so that K occurs and has effect on the agent of K.	*abrir camino* *causar daño* *dar respuesta* *producir un cambio*	*open the way* *cause damage* *give an answer* *produce a change*
$Real_1$ 61	Lat. *realis* – 'real'. To fulfill the requirement of K, to act according to K.	*contestar una pregunta* *cumplir el requisito* *solucionar un problema* *utilizar la tecnología*	*answer a question* *fulfill the requirement* *solve a problem* *use technology*
$Func_0$ 25	Lat. *functionare* – 'to function'. K exists, takes place, occurs.	*el tiempo pasa* *hace un mes* *una posibilidad cabe* *la razón existe*	*time flies* *a month ago* *there is a possibility* *the reason exists*
$Oper_2$ 30	Undergo K, be source of K	*aprender una lección* *obtener una respuesta* *recibir ayuda* *sufrir un cambio*	*learn a lesson* *get an answer* *receive help* *suffer a change*
$IncepOper_1$ 25	Lat. *incipere* – 'to begin'. Begin to do, perform, experience, carry out K.	*adoptar una actitud* *cobrar importancia* *iniciar una sesión* *tomar posición*	*take an attitude* *acquire importance* *start a session* *obtain a position*
$ContOper_1$ 16	Lat. *continuare* – 'to continue'. Continue to do, perform, experience, carry out K.	*guardar silencio* *mantener el equilibrio* *seguir un modelo* *llevar una vida (ocupada)*	*keep silence* *keep one's balance* *follow an example* *lead a (busy) life*

Table 8 presents LFs that we have chosen for our experiments. For each LF, we give the number of examples, its meaning, and sample verb-noun combinations.

In the lexical resource, we have annotated free word combinations with the tag FWC. The number of FWC is 261. We considered free word combinations as a lexical function FWC in its own right and experimented how machine learning algorithms can predict this class of word combinations. Therefore, the total number of LFs we experimented with is 9.

Remember, that in the training set and test set, each verb-noun combination is represented as a set of all hyperonyms of the noun and all hyperonyms of the verb. To construct this representation, the number of sense for every verb and noun must be identified. But sometimes, an appropriate sense was absent in the Spanish WordNet. Such words were tagged with abbreviation N/A (not available) instead of the number of word sense. In the training set, we included only verb-noun combinations that are disambiguated with word senses of the Spanish

WordNet. In Table 8, the numbers of examples include only these verb-noun pairs in which all the words are disambiguated with the Spanish WordNet.

The total number of examples for all 9 lexical functions is 900.

5.2.2. Training Sets

For each of 9 LF chosen for experiments, we built a training set, so we had 9 training sets. All training sets included the same list of 900 verb-noun combinations. The only difference between training sets was the annotation of examples as positive and negative. As an example, let us consider the training set for $Oper_1$. In the list of 900 verb-noun pairs, there are 266 examples of $Oper_1$, so these examples are marked as positive in the training set, and all the rest of verb-noun combinations whose number is 634 ($900 - 266 = 634$) were marked as negative examples. This procedure was applied to each training set.

5.2.3. Test Sets

The test sets were built independently of the training set. 5181 verb-noun combinations for the test set were extracted from the Spanish Treebank Cast3LB [Civit and Martí 2004]. Four test sets were constructed, including, respectively, 100%, 75%, 50%, and 25% of all verb-noun pairs taken from Treebank Cast3LB. Words in the test set were not annotated with lexical functions. Table 9 gives the number of verb-noun pairs in all four test sets.

Table 9. Number of verb-noun combination in the test sets

Test set	Number of verb-noun combinations
100%	5181
75%	3886
50%	2590
25%	1295

Chapter 6. Methodology

6.1. Machine Learning Algorithms

Our approach is based on supervised machine learning algorithms as implemented in the WEKA version 3-6-2 toolset [WEKA], [Hall *et al*. 2009], [Witten and Frank 2005]. We performed two groups of experiments. In the first group of experiments, we evaluated the prediction of LFs meanings on the training sets using 10-fold cross-validation technique. In the second group of experiments, the same meanings were predicted for the instances of an independent test set. Table 10 lists all 68 machine learning algorithms we experimented with.

Table 10. Machine learning algorithms used in the experiments

Algorithm	Algorithm	Algorithm
AODE	ClassificationViaClustering	VFI
AODEsr	ClassificationViaRegression	ConjunctiveRule
BayesianLogisticRegression	CVParameterSelection	DecisionTable
BayesNet	Dagging	JRip
HNB	Decorate	NNge
NaiveBayes	END	OneR
NaiveBayesSimple	EnsembleSelection	PART
NaiveBayesUpdateable	FilteredClassifier	Prism
WAODE	Grading	Ridor
LibSVM	LogitBoost	ZeroR
Logistic	MultiBoostAB	ADTree
RBFNetwork	MultiClassClassifier	BFTree
SimpleLogistic	MultiScheme	DecisionStump
SMO	OrdinalClassClassifier	FT
VotedPerceptron	RacedIncrementalLogitBoost	Id3
Winnow	RandomCommittee	J48
IB1	RandomSubSpace	J48graft
IBk	RotationForest	LADTree
KStar	Stacking	RandomForest
LWL	StackingC	RandomTree
AdaBoostM1	ThresholdSelector	REPTree
AttributeSelectedClassifier	Vote	SimpleCart
Bagging	HyperPipes	

We evaluated the performance of the selected algorithms by comparing precision, recall, and *F-measure* (values for predicting the positive class). The precision is the proportion of the examples which truly have class x among all those which were classified as class x. The recall is the proportion of examples which were classified as class x, among all examples which truly have class x.

The *F-measure* is the harmonic mean of precision and recall:

$$F = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}.$$

6.1.1. WEKA Data Mining Toolkit

WEKA (Waikato Environment for Knowledge Analysis) is well-known software for machine learning and data mining developed at the University of Waikato. This program is written in Java. WEKA is an open-source workbench distributed under the GNU-GPL license. For machine learning experiments, we have used WEKA version 3-6-2 [WEKA, WM]. WEKA workbench has a graphical user interface that leads the user through data mining tasks and has data visualization tools that help understand the models.

6.1.2. Stratified 10-Fold Cross-validation Technique

For evaluating the prediction of machine learning algorithms on the training set, we have used stratified 10-fold cross-validation technique. The simplest form of evaluating the performance of classifiers is using a training set and a test set which are mutually independent. This is referred to as hold-out estimate.

We have chosen a more elaborate evaluation method, i.e. cross-validation. Here, a number of folds *n* is specified. The dataset is randomly reordered and then split into n folds of equal size. In each iteration, one fold is used for testing and the other *n-1* folds are used for training the classifier. The test results are collected and averaged over all folds. This gives the cross-validation estimate of the accuracy. The folds can be purely random or slightly modified to create the same class distributions in each fold as in the complete dataset. In the latter case the cross-validation is called stratified. We have applied the stratified option of 10-fold cross-validation method.

6.2. Representing Data for Machine Learning Techniques

6.2.1. Training Sets

In Section 5.4, we have given a linguistic description of the training set and the test set. We have mentioned that each verb-noun pairs was represented as a set of all hyperonyms of the noun and all hyperonyms of the verb.

However, the machine learning techniques can access data only if it is represented in the Attribute-Relation File Format (ARFF).

A dataset in ARFF format is a collection of examples, each one of class `weka.core.Instance`. Remember, that WEKA is written in Java. Each `Instance` consists of a

number of attributes, any of which can be nominal (one of a predefined list of values), numeric (a real or integer number) or a string (an arbitrary long list of characters, enclosed in "double quotes"). In our case, all attributes are nominal.

For each verb-noun pair, we used binary feature representation. Every hyperonym is represented as a nominal attribute which can take one of two values: "1" if it is a hyperonym of any word in a given verb-noun pair, and "0" if it is not. The fact that a verb-noun combination belongs or does not belong to a particular LF is identified by the class attribute with two possible values: "yes" for positive examples of a particular LF and "no" for negative ones.

All training sets include 900 verb-noun pairs represented by hyperonyms. This gives 798 features to represent the nouns, and 311 features to represent the verbs. Total number of features that are hyperonyms is 1109. There is also one class attribute; therefore, each training set includes 1110 attributes. Verb-noun pairs in the training set are represented as vectors of length 1110:

$$v_1, v_2, ..., v_{798}, n_1, n_2, ..., n_{311}, LF,$$

where v_n, n_k can be 0 or 1, and *LF* is a class attribute having the value *yes* for positive instances of LF for which classification is done, and *no* for negative instances. A partial representation of the training set in the ARFF format is given in Fig. 6.

Fig. 6. A partial representation of the training set in the ARFF format.

```
@relation Oper1
@attribute n00001740 {0,1} % entidad_1
@attribute n00002086 {0,1} % ser_vivo_1
@attribute n00003731 {0,1} % agente_causal_1 causa_4
...
@attribute v01128460 {0,1} % causar_5 producir_4 ocasionar_1
@attribute v01130277 {0,1} % hacer_2
@attribute v01131926 {0,1} % dar_9
...
@attribute category {yes,no}

@data
1,0,0,0,1,0,0,1,0,0,0,0,0,0,...,0,0,0,yes % v_hacer_15 n_mención_1
...
0,0,0,0,0,0,0,0,0,0,0,0,0,1,...,0,0,0,no % v_abrir_5 n_camino_5
```

In Fig. 6, after the tag `@relation`, we define the internal name of the dataset. Nominal attributes are defined with the tag `@attribute`. The record `@data` marks the beginning of the data section. The data section consists of comma-separated values for the attributes – one line per example. In Fig. 6 we show only two examples: one positive and one negative. All records beginning with `@attribute` as well as all strings in the data section are accompanied by a comment starting with symbol `%` in ARFF. For attribute entries, comments include words of the synset specified by its number after the record `@attribute`. For data strings, comment includes

the verb-noun pair represented by this string. The pair is annotated with POS and word senses. Comments were added for human viewing of data.

The process of training set compilation explained above can be represented as fulfilled according to the algorithm shown in Fig. 7. The training sets (as well as the test sets) were compiled automatically used the programs whose code is given in Appendix 4.

Fig. 7. Algorithm of compiling the training sets.

```
Algorithm: constructing data sets
Input: a list of 900 Spanish verb-noun collocations annotated with 8 lexical functions
Output: 8 data sets – one for each lexical function
For each lexical function
Create an empty data set and assign it the name of the lexical function.
        For each collocation in the list of verb-noun collocations
                Retrieve all hyperonyms of the noun.
                Retrieve all hyperonyms of the verb.
                Make a set of hyperonyms:
                {noun, all hyperonyms of the noun, verb, all hyperonyms of the verb}.
                If a given collocation belongs to this lexical function
                        assign '1' to the set of hyperonyms,
                Else   assign '0' to the set of hyperonyms.
                Add the set of hyperonyms to the data set.
        Return the data set.
```

6.2.2. Test Sets

Section 5.4.2 describes characteristics of the training sets used to evaluate the selected machine learning algorithms by 10-fold cross-validation method. That is, the training set is used to build a model on the learning stage and then the model is tested 10 times on parts of the same training set.

When the performance of the machine learning techniques is evaluated on an independent test set, the test set format must be compatible with the training test format. It means that the training set must contain all the attributes which are included in the test set. Since the test set has a lot of words other than the training set, the number of attributes is much bigger than in the case of 10-fold cross-validation.

Therefore, the list of attributes is the same in the training set and in the test set, when the performance of the algorithms is validated on an independent test set. Table 11 gives characteristics of the training set and the test sets. In Table 11, verb-noun pairs are actual verb-noun combinaitons extracted from Spanish Treebank Cast3LB, and instances are combinations of all senses of the verbs and all senses of the nouns that are found in the Spanish WordNet. For more details, see Section 3.2.

Table 11. Characteristics of data sets

Test set	# of verb-noun pairs from Spanish Treebank Cast3LB	# of instances in the test set	# of attributes in the training set and the test set
100%	5181	96079	10544
75%	3886	73021	9495
50%	2590	48904	8032
25%	1295	22254	5857

6.3. Classification Procedure

Fig. 8 presents the classification procedure schematically.

Fig. 8. The classification procedure.
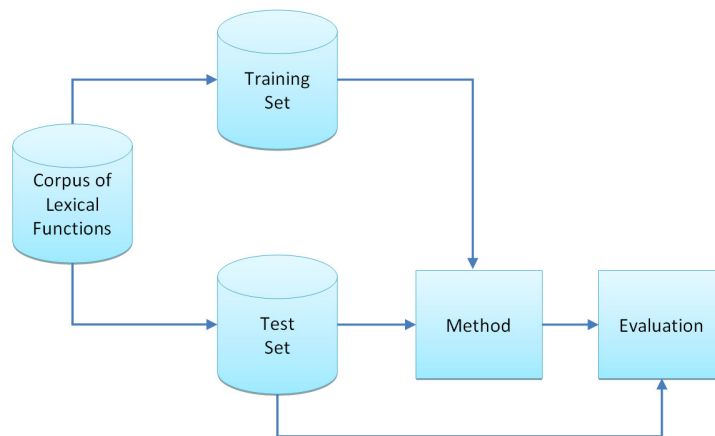
Now, we will explain the classification procedure using the WEKA graphical user interface. First, the training set is downloaded and WEKA shows various characteristics of the data in Explorer, a part of the interface developed for the purpose of analyzing data in the data set. This stage is demonstrated in Fig.9.

Fig. 9. Characteristics of data viewed in WEKA Explorer.
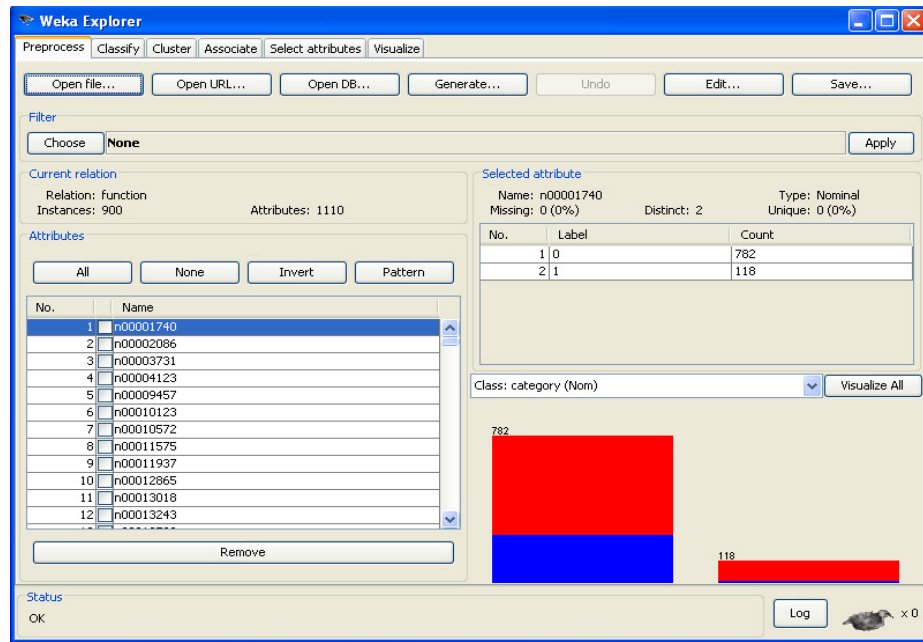
Secondly, a classifier is chosen for data classification. This step can be vied in Fig. 10.

Fig. 10. Selection of the classifier.
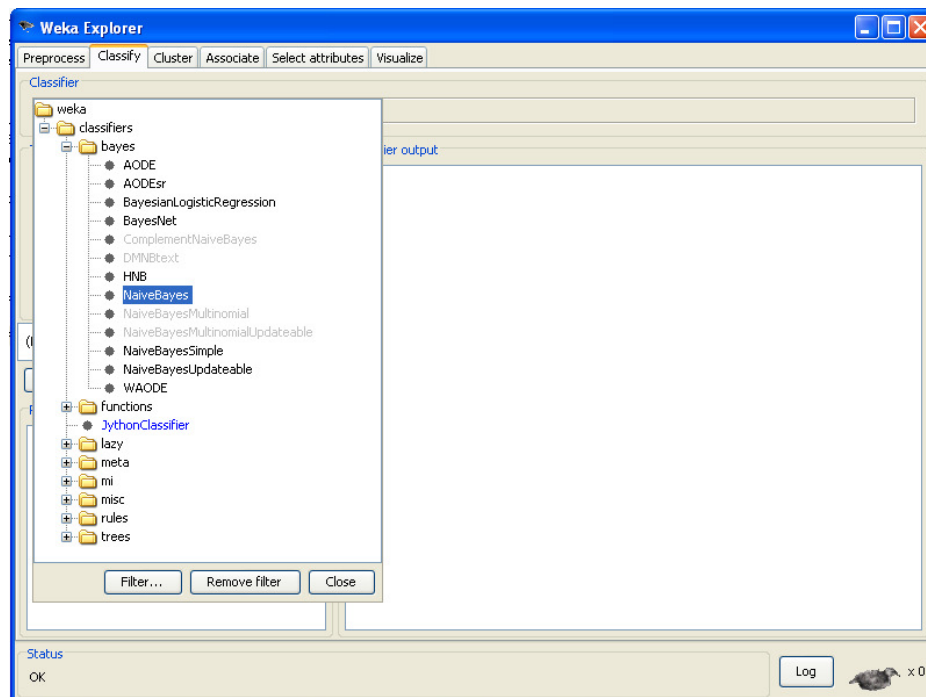
Thirdly, the chosen classifier starts working and after finishing the learning and the test stages, it outputs various estimates of its performance. In our experiments, we used the values of precision, recall and F-measure for „yes" class, or the positive class, to evaluate the performance of the classifiers. A classifier's output is demonstrated in Fig. 11.

Fig. 11. Output of the selected classifier.
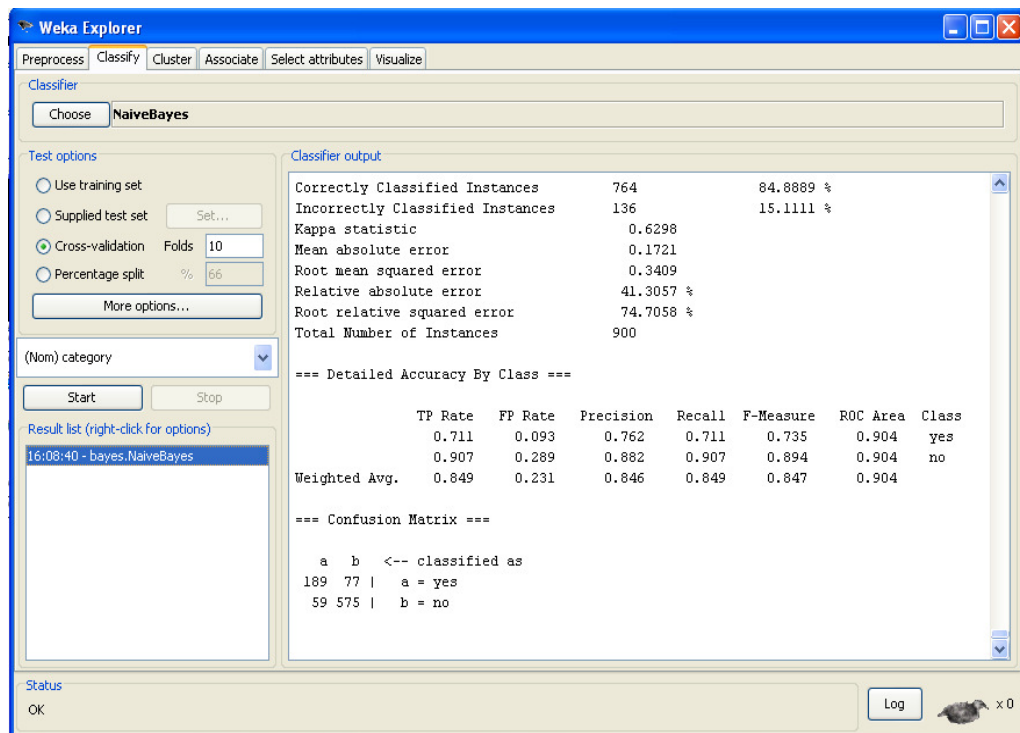
While for initial experiments the included graphical user interface is quite sufficient, for in-depth usage the command line interface is recommended, because it offers some functionality which is not available via the graphical user interface (GUI). When data is classified with the help of GUI, the default heap memory size is 16—64 MB. For our data sets, this is too little memory. The heap size of java engine can be increased via

−Xmx1024m for 1GB in the command line. Also, via the command line using the

−cp option we set CLASSPATH so that includes weka.jar.

Taking the above advantages into account, we operated WEKA through the command line using the following options for the algorithm PART of class rules taken as an example:

```
set classpath=%classpath%;C:\Archivos de programa\Weka-3-6\weka.jar
set classpath=%classpath%;C:\Archivos de programa\Weka-3-6\rules.jar
set classpath=%classpath%;C:\Archivos de programa\Weka-3-6\PART.jar
java -Xmx1024m weka.classifiers.rules.PART -t training_set.arff -d part.model

java -Xmx1024m weka.classifiers.rules.PART -l part.model -T test_set.arff -p
0 >> result1.txt 2> $$1.tmp
```

The first three lines of the above example show how CLASSPATH is set for WEKA and the chosen classifier in particular. The fourth line tells the classifier to learn on the training set and to save the model created for the training data in a file with the extension model. The last line commands the classifier to test the learnt model on the test set and save the results of prediction

in a file. To preserve error messages of the classifier for further debug, the standard error stream (2>) can be directed to a file and saved in it.

Chapter 7. Experimental Results

7.1. Algorithm Performance Evaluation on the Training Set

In Table 12, we present the main results obtained in our experiments. For each LF, we list three machine learning algorithms that have shown the best performance. The symbol # stands for the number of examples in the respective training set; P stands for precision, R stands for recall, F stands for *F-measure*. The baseline is explained in Section 7.1.1.

Table 12. Best results showed by algorithms on the training set of lexical functions

LF	#	Algorithm	P	R	F	Baseline
$Oper_1$	280	BayesianLogisticRegression	0.879	0.866	**0.873**	
		Id3	0.879	0.861	0.870	0.311
		SMO	0.862	0.866	0.864	
$CausFunc_0$	112	JRip	0.747	0.705	**0.725**	
		EnsembleSelection	0.744	0.659	0.699	0.124
		REPTree	0.750	0.648	0.695	
$CausFunc_1$	90	J48	0.842	0.696	**0.762**	
		FilteredClassifier	0.842	0.696	0.762	0.100
		END	0.842	0.696	0.762	
$Real_1$	61	Prism	0.735	0.832	**0.781**	
		BayesianLogisticRegression	0.788	0.553	0.650	0.068
		SMO	0.722	0.553	0.627	
$Func_0$	25	BFTree	0.667	0.727	**0.696**	
		Id3	0.571	0.727	0.640	0.028
		AttributeSelectedClassifier	0.636	0.636	0.636	
$Oper_2$	30	PART	0.923	0.571	**0.706**	
		AttributeSelectedClassifier	0.923	0.571	0.706	0.033
		END	0.923	0.571	0.706	
$IncepOper_1$	25	Prism	0.750	0.800	**0.774**	
		NNge	0.923	0.600	0.727	0.028
		SMO	0.813	0.650	0.722	
$ContOper_1$	16	SimpleLogistic	0.909	0.769	**0.833**	
		DecisionTable	0.909	0.769	0.833	0.018
		AttributeSelectedClassifier	0.833	0.769	0.800	
FWC	261	Prism	0.639	0.702	**0.669**	
		BayesianLogisticRegression	0.658	0.629	0.643	0.290
		SMO	0.656	0.623	0.639	
Total:	900			*Average best:*	0.758	

7.1.1. Baseline

Often, in classification experiments, the baseline is the performance of ZeroR classifier. ZeroR is a trivial algorithm that always predicts the majority class. It happens that the majority class in our training sets is always the class of negative instances. Even in the case of the LF which has the largest number of positive instances in the training set (280 positive examples of $Oper_1$), the number of negative instances is still larger (900 − 280 = 620 negative examples of $Oper_1$).

Therefore, the ZeroR does not classify any test instances as positives, which gives always recall of 0 and undefined precision. Thus ZeroR is too bad a baseline to be considered.

However, the baseline can be a random choice of a positive or a negative answer to the question "Is this collocation of this particular lexical function?" In such a case we deal with the probability of a positive and negative response. Since we are interested in only assigning the positive answer to a collocation, we calculate the probability of "yes" class for eight lexical functions in the experiments according to the formula: probability of "yes" = 1 / (the number of all examples / the number of positive examples of a given lexical function). These probabilities will be results of a classifier that assigns the class "yes" to collocations at random. Since we will compare the probabilities of the random choice with the results obtained in our experiments, we present the former as numbers within the range from 0 to 1 in Table 13 as well as in Table 12.

Table 13. Probability of selecting "yes" class at random

Lexical function	Number of examples	Probability of the class "yes"
$Oper_1$	280	0.311
$CausFunc_0$	112	0.124
$CausFunc_1$	90	0.100
$Real_1$	61	0.068
$Func_0$	25	0.028
$Oper_2$	30	0.033
$IncepOper_1$	25	0.028
$ContOper_1$	16	0.018
FWC	261	0.290

7.1.2. Three Best Machine Learning Algorithms for Each Lexical Function

As it is seen from Table 12, no single classifier is the best one for detecting all LFs. For each LF, the highest result is achieved by a different classifier. However, Prism reaches the highest F-score for both $IncepOper_1$ and FWC, though recall that FWC (free word combinations) is not a lexical function but is considered as an independent class along with LFs. The maximum F-measure of 0.873 is achieved by BayesianLogisticRegression classifier for $Oper_1$. The lowest best F-measure of 0.669 is shown by Prism for FWC. The average F-measure (calculated over only the nine best results, one for each LF) is 0.758.

We observed no correlation between the number of instances in the training set and the results obtained from the classifiers. For example, a low result is shown for the class FWC which has the largest number of positive examples. On the contrary, the second top result is achieved for LF $ContOper_1$, with the smallest number of positive examples. The minimum F-measure is obtained for FWC whose number of positive examples (261) is a little less than the largest number of positive examples ($Oper_1$ with 280 examples) but the detection of $Oper_1$ was the best.

For comparison, Table 14 gives the state of the art results reported in [Wanner *et al*. 2006] for LF classification using machine learning techniques. Out of nine LFs mentioned in [Wanner *et al*. 2006] we give in Table 14 only those five that we used in our experiments, i.e., that are represented in Table 12. The numbers of our results are rounded to include two figures after the point, since the results of [Wanner *et al*. 2006] are represented in this manner. Also, as we have explained in Section 3.5.1 that [Wanner *et al*. 2006] reports the results for two different datasets: one for a narrow semantic field (that of emotions) and another for a field-independent (general) dataset. Since our dataset is also general, comparing them with a narrow-field dataset would not be fair, so in Table 14 we only give the field-independent figures from [Wanner *et al*. 2006].

Table 14. State of the art results for some LFs taken from [Wanner *et al*. 2006]

LF	NN			NB			ID3			TAN			Our
	P	R	F	P	R	F	P	R	F	P	R	F	F
$Oper_1$	0.65	0.55	0.60	0.87	0.64	**0.74**	0.52	0.51	0.51	0.75	0.49	0.59	**0.87**
$Oper_2$	0.62	0.71	**0.66**	0.55	0.21	0.30	N/A			0.55	0.56	0.55	**0.71**
$ContOper_1$	N/A			N/A			0.84	0.57	**0.70**	N/A			**0.83**
$CausFunc_0$	0.59	0.79	**0.68**	0.44	0.89	0.59	N/A			0.45	0.57	0.50	**0.73**
$Real_1$	0.58	0.44	**0.50**	0.58	0.37	0.45	N/A			0.78	0.36	0.49	**0.78**
Best average: 0.66												*Average:*	0.78

Not all methods have been applied in [Wanner *et al*. 2006] for all LFs; if a method was not applied for a particular LF, the corresponding cells are marked as N/A. In this table, NN stands for the Nearest Neighbor technique, NB for Naïve Bayesian network, ID3 is a decision tree classification technique based on the ID3-algorithm, and TAN for the Tree-Augmented Network Classification technique; P, R, and F are as in Table 12. In fact [Wanner *et al*. 2006] did not give the value of F-measure, so we calculated it using the formula in Section 6.1. The last column repeats the best F-measure results from Table 12, for convenience of the reader. For each LF, the best result from [Wanner *et al*. 2006], as well as the overall best result (including our experiments), are marked in boldface.

As seen from Table 14, for all LFs our experiments gave significantly higher figures than those reported in [Wanner *et al*. 2006]. The best average F-measure from [Wanner *et al*. 2006] is 0.66, while our experiments demonstrate the best average F-measure of 0.75 (calculated from Table 12) and the average F-measure is 0.78. However, the comparison is not fair because

different datasets have been used: the exact dataset used in [Wanner *et al*. 2006] is unfortunately not available anymore[1], ours is available from [LFs].

7.1.3. Algorithm Performance on the Training Set

In Tables 15—23, we present the results of performance of 68 machine learning algorithms on 9 training sets, i.e., one training set for each of 9 LFs chosen for the experiments. As in previous tables, P stands for precision, R stands for recall and F stand for F-measure. All algorithms are ranked by F-measure.

Table 15. Algorithm performance ranked by F-measure on the training set for $Oper_1$

Algorithm	Precision	Recall	F-measure
bayes.BayesianLogisticRegression	0.879	0.866	0.873
trees.Id3	0.879	0.861	0.870
functions.SMO	0.862	0.866	0.864
trees.FT	0.858	0.866	0.862
trees.LADTree	0.873	0.851	0.862
trees.SimpleCart	0.873	0.851	0.862
functions.SimpleLogistic	0.872	0.847	0.859
meta.ThresholdSelector	0.835	0.876	0.855
meta.EnsembleSelection	0.871	0.837	0.854
trees.BFTree	0.871	0.837	0.854
trees.ADTree	0.859	0.847	0.853
rules.JRip	0.859	0.842	0.850
meta.AttributeSelectedClassifier	0.851	0.847	0.849
meta.LogitBoost	0.862	0.837	0.849
meta.Bagging	0.854	0.842	0.848
functions.Logistic	0.787	0.916	0.847
meta.MultiClassClassifier	0.787	0.916	0.847
meta.END	0.842	0.847	0.844
meta.FilteredClassifier	0.842	0.847	0.844
meta.OrdinalClassClassifier	0.842	0.847	0.844
rules.PART	0.857	0.832	0.844
trees.J48	0.842	0.847	0.844
trees.J48graft	0.841	0.837	0.839
rules.DecisionTable	0.854	0.812	0.832
trees.REPTree	0.832	0.832	0.832
meta.RotationForest	0.850	0.812	0.830
meta.ClassificationViaRegression	0.804	0.832	0.818
rules.NNge	0.794	0.842	0.817
rules.Ridor	0.827	0.807	0.817
meta.Decorate	0.781	0.847	0.812
functions.VotedPerceptron	0.856	0.767	0.809
meta.Dagging	0.801	0.817	0.809
meta.RandomCommittee	0.742	0.827	0.782
rules.Prism	0.735	0.832	0.781
trees.RandomForest	0.735	0.822	0.776
meta.RandomSubSpace	0.913	0.624	0.741
misc.VFI	0.764	0.688	0.724
bayes.HNB	0.841	0.629	0.720
bayes.BayesNet	0.694	0.743	0.718
bayes.WAODE	0.734	0.698	0.716

---

[1] Personal communication with L. Wanner.

bayes.AODE	0.757	0.678	0.715
bayes.NaiveBayes	0.742	0.683	0.711
bayes.NaiveBayesSimple	0.742	0.683	0.711
bayes.NaiveBayesUpdateable	0.742	0.683	0.711
trees.RandomTree	0.662	0.718	0.689
functions.RBFNetwork	0.758	0.619	0.681
lazy.LWL	0.811	0.574	0.672
bayes.AODEsr	0.691	0.599	0.642
misc.HyperPipes	0.583	0.693	0.633
lazy.IB1	0.540	0.728	0.620
lazy.IBk	0.519	0.757	0.616
lazy.KStar	0.556	0.683	0.613
meta.AdaBoostM1	0.914	0.366	0.523
functions.Winnow	0.450	0.401	0.424
meta.MultiBoostAB	0.976	0.203	0.336
rules.OneR	0.976	0.203	0.336
trees.DecisionStump	0.976	0.203	0.336
rules.ConjunctiveRule	0.857	0.208	0.335
meta.ClassificationViaClustering	0.314	0.134	0.188
functions.LibSVM	0	0	0
meta.CVParameterSelection	0	0	0
meta.Grading	0	0	0
meta.MultiScheme	0	0	0
meta.RacedIncrementalLogitBoost	0	0	0
meta.Stacking	0	0	0
meta.StackingC	0	0	0
meta.Vote	0	0	0
rules.ZeroR	0	0	0

Table 16. Algorithm performance ranked by F-measure on the training set for $CausFunc_0$

Algorithm	Precision	Recall	F-measure
rules.JRip	0.747	0.705	0.725
meta.EnsembleSelection	0.744	0.659	0.699
trees.REPTree	0.750	0.648	0.695
meta.ClassificationViaRegression	0.693	0.693	0.693
trees.SimpleCart	0.727	0.636	0.679
trees.LADTree	0.674	0.682	0.678
trees.BFTree	0.733	0.625	0.675
meta.Bagging	0.726	0.602	0.658
functions.SMO	0.675	0.614	0.643
trees.ADTree	0.718	0.580	0.642
trees.FT	0.655	0.625	0.640
rules.Ridor	0.694	0.568	0.625
meta.AttributeSelectedClassifier	0.746	0.534	0.623
trees.Id3	0.618	0.625	0.621
meta.RotationForest	0.712	0.534	0.610
meta.END	0.730	0.523	0.609
meta.FilteredClassifier	0.730	0.523	0.609
meta.OrdinalClassClassifier	0.730	0.523	0.609
trees.J48	0.730	0.523	0.609
functions.Logistic	0.552	0.659	0.601
meta.MultiClassClassifier	0.552	0.659	0.601
rules.DecisionTable	0.653	0.557	0.601
rules.Prism	0.577	0.612	0.594
bayes.BayesianLogisticRegression	0.662	0.534	0.591
meta.Decorate	0.602	0.568	0.585
functions.SimpleLogistic	0.672	0.511	0.581
rules.PART	0.605	0.557	0.580
meta.RandomCommittee	0.630	0.523	0.571
trees.RandomForest	0.677	0.477	0.560

meta.LogitBoost	0.702	0.455	0.552
rules.NNge	0.595	0.500	0.543
meta.ThresholdSelector	0.597	0.489	0.538
trees.J48graft	0.745	0.398	0.519
bayes.BayesNet	0.537	0.500	0.518
misc.VFI	0.443	0.580	0.502
functions.VotedPerceptron	0.614	0.398	0.483
meta.Dagging	0.784	0.330	0.464
misc.HyperPipes	0.395	0.557	0.462
bayes.WAODE	0.690	0.330	0.446
trees.RandomTree	0.507	0.398	0.446
lazy.KStar	0.485	0.364	0.416
bayes.AODEsr	0.365	0.477	0.414
lazy.IBk	0.452	0.375	0.410
meta.RandomSubSpace	0.735	0.284	0.410
lazy.IB1	0.468	0.330	0.387
rules.OneR	0.697	0.261	0.380
bayes.NaiveBayes	0.621	0.205	0.308
bayes.NaiveBayesSimple	0.621	0.205	0.308
bayes.NaiveBayesUpdateable	0.621	0.205	0.308
bayes.AODE	0.824	0.159	0.267
functions.Winnow	0.217	0.341	0.265
functions.RBFNetwork	0.625	0.114	0.192
bayes.HNB	0.571	0.091	0.157
meta.ClassificationViaClustering	0.092	0.091	0.091
meta.AdaBoostM1	0.500	0.011	0.022
functions.LibSVM	0	0	0
lazy.LWL	0	0	0
meta.CVParameterSelection	0	0	0
meta.Grading	0	0	0
meta.MultiBoostAB	0	0	0
meta.MultiScheme	0	0	0
meta.RacedIncrementalLogitBoost	0	0	0
meta.Stacking	0	0	0
meta.StackingC	0	0	0
meta.Vote	0	0	0
rules.ConjunctiveRule	0	0	0
rules.ZeroR	0	0	0
trees.DecisionStump	0	0	0

Table 17. Algorithm performance ranked by F-measure on the training set for CausFunc$_1$

Algorithm	Precision	Recall	F-measure
trees.J48	0.842	0.696	0.762
meta.FilteredClassifier	0.842	0.696	0.762
meta.END	0.842	0.696	0.762
meta.OrdinalClassClassifier	0.842	0.696	0.762
functions.SimpleLogistic	0.828	0.696	0.756
meta.LogitBoost	0.828	0.696	0.756
meta.AttributeSelectedClassifier	0.774	0.696	0.733
rules.DecisionTable	0.833	0.652	0.732
rules.JRip	0.783	0.681	0.729
trees.FT	0.742	0.710	0.726
trees.LADTree	0.818	0.652	0.726
trees.SimpleCart	0.804	0.652	0.72
trees.BFTree	0.811	0.623	0.705
trees.Id3	0.700	0.710	0.705
rules.NNge	0.742	0.667	0.702
rules.PART	0.696	0.696	0.696
trees.REPTree	0.808	0.609	0.694
trees.J48graft	0.851	0.580	0.690
meta.Bagging	0.792	0.609	0.689

bayes.BayesianLogisticRegression	0.726	0.652	0.687
meta.RotationForest	0.833	0.58	0.684
rules.Ridor	0.804	0.594	0.683
functions.SMO	0.697	0.667	0.681
meta.Dagging	0.800	0.580	0.672
functions.VotedPerceptron	0.769	0.580	0.661
lazy.LWL	0.796	0.565	0.661
meta.AdaBoostM1	0.796	0.565	0.661
meta.MultiBoostAB	0.796	0.565	0.661
rules.ConjunctiveRule	0.796	0.565	0.661
rules.OneR	0.796	0.565	0.661
trees.ADTree	0.796	0.565	0.661
trees.DecisionStump	0.796	0.565	0.661
meta.EnsembleSelection	0.78	0.565	0.655
rules.Prism	0.612	0.695	0.651
meta.ClassificationViaRegression	0.741	0.580	0.650
meta.Decorate	0.571	0.696	0.627
trees.RandomForest	0.720	0.522	0.605
meta.RandomCommittee	0.615	0.580	0.597
lazy.IBk	0.500	0.493	0.496
trees.RandomTree	0.452	0.478	0.465
lazy.IB1	0.492	0.435	0.462
bayes.WAODE	0.632	0.348	0.449
lazy.KStar	0.483	0.406	0.441
meta.ThresholdSelector	0.329	0.667	0.440
meta.RandomSubSpace	0.778	0.304	0.438
functions.Logistic	0.306	0.696	0.425
meta.MultiClassClassifier	0.306	0.696	0.425
bayes.BayesNet	0.438	0.406	0.421
bayes.AODEsr	0.375	0.478	0.42
misc.VFI	0.343	0.507	0.409
misc.HyperPipes	0.286	0.522	0.369
functions.RBFNetwork	0.462	0.174	0.253
functions.Winnow	0.163	0.348	0.222
meta.ClassificationViaClustering	0.081	0.087	0.084
bayes.AODE	0.429	0.043	0.079
bayes.HNB	0.429	0.043	0.079
bayes.NaiveBayes	0.333	0.043	0.077
bayes.NaiveBayesSimple	0.333	0.043	0.077
bayes.NaiveBayesUpdateable	0.333	0.043	0.077
functions.LibSVM	0	0	0
meta.CVParameterSelection	0	0	0
meta.Grading	0	0	0
meta.MultiScheme	0	0	0
meta.RacedIncrementalLogitBoost	0	0	0
meta.Stacking	0	0	0
meta.StackingC	0	0	0
meta.Vote	0	0	0
rules.ZeroR	0	0	0

Table 18. Algorithm performance ranked by F-measure on the training set for $Real_1$

Algorithm	Precision	Recall	F-measure
rules.Prism	0.735	0.832	0.781
bayes.BayesianLogisticRegression	0.788	0.553	0.650
functions.SMO	0.722	0.553	0.627
trees.FT	0.650	0.553	0.598
rules.NNge	0.614	0.574	0.593
trees.Id3	0.600	0.574	0.587
trees.LADTree	0.733	0.468	0.571
meta.LogitBoost	0.833	0.426	0.563

rules.DecisionTable	0.750	0.447	0.560
rules.JRip	0.750	0.447	0.560
functions.SimpleLogistic	0.864	0.404	0.551
trees.J48graft	0.864	0.404	0.551
trees.BFTree	0.769	0.426	0.548
functions.Logistic	0.440	0.702	0.541
meta.MultiClassClassifier	0.440	0.702	0.541
rules.PART	0.677	0.447	0.538
meta.END	0.714	0.426	0.533
meta.FilteredClassifier	0.714	0.426	0.533
meta.OrdinalClassClassifier	0.714	0.426	0.533
trees.J48	0.714	0.426	0.533
trees.SimpleCart	0.714	0.426	0.533
meta.Bagging	0.857	0.383	0.529
misc.VFI	0.473	0.553	0.510
trees.ADTree	0.679	0.404	0.507
meta.AttributeSelectedClassifier	0.588	0.426	0.494
meta.RandomCommittee	0.613	0.404	0.487
meta.Decorate	0.453	0.511	0.480
meta.ThresholdSelector	0.409	0.000	0.478
trees.RandomForest	0.600	0.383	0.468
misc.HyperPipes	0.488	0.447	0.467
meta.EnsembleSelection	0.727	0.340	0.464
trees.REPTree	0.667	0.340	0.451
meta.RotationForest	0.778	0.298	0.431
meta.ClassificationViaRegression	0.486	0.362	0.415
rules.Ridor	0.813	0.277	0.413
lazy.LWL	0.636	0.298	0.406
meta.AdaBoostM1	0.636	0.298	0.406
meta.MultiBoostAB	0.636	0.298	0.406
rules.ConjunctiveRule	0.636	0.298	0.406
trees.DecisionStump	0.636	0.298	0.406
lazy.KStar	0.436	0.362	0.395
lazy.IBk	0.367	0.383	0.375
trees.RandomTree	0.367	0.383	0.375
lazy.IB1	0.390	0.340	0.364
meta.RandomSubSpace	0.889	0.170	0.286
bayes.BayesNet	0.357	0.213	0.267
functions.Winnow	0.211	0.340	0.260
bayes.AODEsr	0.211	0.319	0.254
functions.VotedPerceptron	0.467	0.149	0.226
rules.OneR	0.500	0.128	0.203
bayes.WAODE	0.571	0.085	0.148
meta.ClassificationViaClustering	0.071	0.106	0.085
bayes.NaiveBayes	0.333	0.021	0.040
bayes.NaiveBayesSimple	0.333	0.021	0.040
bayes.NaiveBayesUpdateable	0.333	0.021	0.040
bayes.AODE	0	0	0
bayes.HNB	0	0	0
functions.LibSVM	0	0	0
functions.RBFNetwork	0	0	0
meta.CVParameterSelection	0	0	0
meta.Dagging	0	0	0
meta.Grading	0	0	0
meta.MultiScheme	0	0	0
meta.RacedIncrementalLogitBoost	0	0	0
meta.Stacking	0	0	0
meta.StackingC	0	0	0
meta.Vote	0	0	0
rules.ZeroR	0	0	0

Table 19. Algorithm performance ranked by F-measure on the training set for $Func_0$

Algorithm	Precision	Recall	F-measure
trees.BFTree	0.667	0.727	0.696
trees.Id3	0.571	0.727	0.640
meta.AttributeSelectedClassifier	0.636	0.636	0.636
meta.END	0.636	0.636	0.636
meta.FilteredClassifier	0.636	0.636	0.636
meta.OrdinalClassClassifier	0.636	0.636	0.636
misc.HyperPipes	0.636	0.636	0.636
trees.J48	0.636	0.636	0.636
lazy.LWL	0.750	0.545	0.632
meta.AdaBoostM1	0.750	0.545	0.632
meta.LogitBoost	0.750	0.545	0.632
rules.OneR	0.750	0.545	0.632
trees.DecisionStump	0.750	0.545	0.632
misc.VFI	0.583	0.636	0.609
bayes.BayesianLogisticRegression	0.667	0.545	0.600
meta.ClassificationViaRegression	0.667	0.545	0.600
rules.JRip	0.667	0.545	0.600
trees.ADTree	0.667	0.545	0.600
meta.MultiBoostAB	0.833	0.455	0.588
functions.SMO	0.538	0.636	0.583
trees.LADTree	0.538	0.636	0.583
rules.DecisionTable	0.600	0.545	0.571
rules.PART	0.600	0.545	0.571
meta.RotationForest	0.714	0.455	0.556
rules.Ridor	0.545	0.545	0.545
functions.SimpleLogistic	0.625	0.455	0.526
trees.SimpleCart	0.625	0.455	0.526
rules.NNge	0.500	0.545	0.522
trees.FT	0.667	0.364	0.471
trees.J48graft	0.667	0.364	0.471
functions.Logistic	0.368	0.636	0.467
meta.MultiClassClassifier	0.368	0.636	0.467
rules.Prism	0.364	0.571	0.444
trees.RandomTree	0.375	0.545	0.444
lazy.IBk	0.333	0.455	0.385
meta.Bagging	0.600	0.273	0.375
meta.EnsembleSelection	0.600	0.273	0.375
trees.REPTree	0.600	0.273	0.375
meta.RandomCommittee	0.500	0.273	0.353
lazy.IB1	0.333	0.364	0.348
meta.ThresholdSelector	0.308	0.364	0.333
trees.RandomForest	0.429	0.273	0.333
lazy.KStar	0.300	0.273	0.286
functions.RBFNetwork	0.500	0.182	0.267
bayes.AODEsr	0.120	0.545	0.197
functions.VotedPerceptron	1.000	0.091	0.167
meta.RandomSubSpace	1.000	0.091	0.167
rules.ConjunctiveRule	1.000	0.091	0.167
meta.Decorate	0.103	0.273	0.150
functions.Winnow	0.048	0.091	0.063
bayes.AODE	0	0	0
bayes.BayesNet	0	0	0
bayes.HNB	0	0	0
bayes.NaiveBayes	0	0	0
bayes.NaiveBayesSimple	0	0	0
bayes.NaiveBayesUpdateable	0	0	0
bayes.WAODE	0	0	0
functions.LibSVM	0	0	0

meta.ClassificationViaClustering	0	0	0
meta.CVParameterSelection	0	0	0
meta.Dagging	0	0	0
meta.Grading	0	0	0
meta.MultiScheme	0	0	0
meta.RacedIncrementalLogitBoost	0	0	0
meta.Stacking	0	0	0
meta.StackingC	0	0	0
meta.Vote	0	0	0
rules.ZeroR	0	0	0

Table 20. Algorithm performance ranked by F-measure on the training set for $Oper_2$

Algorithm	Precision	Recall	F-measure
rules.PART	0.923	0.571	0.706
meta.AttributeSelectedClassifier	0.923	0.571	0.706
meta.END	0.923	0.571	0.706
meta.FilteredClassifier	0.923	0.571	0.706
meta.OrdinalClassClassifier	0.923	0.571	0.706
trees.J48	0.923	0.571	0.706
meta.LogitBoost	0.857	0.571	0.686
meta.RandomCommittee	0.722	0.619	0.667
trees.LADTree	0.667	0.667	0.667
rules.JRip	0.786	0.524	0.629
trees.SimpleCart	0.909	0.476	0.625
trees.FT	0.667	0.571	0.615
trees.BFTree	0.733	0.524	0.611
bayes.BayesianLogisticRegression	0.632	0.571	0.600
rules.NNge	0.632	0.571	0.600
functions.SMO	0.688	0.524	0.595
rules.Prism	0.545	0.632	0.585
functions.SimpleLogistic	0.900	0.429	0.581
meta.EnsembleSelection	0.900	0.429	0.581
rules.DecisionTable	0.900	0.429	0.581
rules.OneR	0.900	0.429	0.581
trees.J48graft	0.900	0.429	0.581
trees.REPTree	0.900	0.429	0.581
trees.Id3	0.542	0.619	0.578
meta.ClassificationViaRegression	0.714	0.476	0.571
trees.ADTree	0.714	0.476	0.571
meta.Bagging	0.818	0.429	0.563
functions.VotedPerceptron	0.889	0.381	0.533
rules.Ridor	0.692	0.429	0.529
meta.RotationForest	0.727	0.381	0.500
lazy.KStar	0.476	0.476	0.476
trees.RandomTree	0.407	0.524	0.458
meta.RandomSubSpace	0.857	0.286	0.429
lazy.IBk	0.370	0.476	0.417
trees.RandomForest	0.538	0.333	0.412
meta.ThresholdSelector	0.333	0.524	0.407
functions.Logistic	0.302	0.619	0.406
meta.MultiClassClassifier	0.302	0.619	0.406
misc.HyperPipes	0.323	0.476	0.385
misc.VFI	0.303	0.476	0.370
bayes.AODEsr	0.245	0.571	0.343
meta.Decorate	0.256	0.476	0.333
lazy.IB1	0.286	0.381	0.327
meta.Dagging	1.000	0.190	0.320
meta.MultiBoostAB	0.667	0.095	0.167
functions.Winnow	0.067	0.143	0.091
rules.ConjunctiveRule	1.000	0.048	0.091

meta.AdaBoostM1	0.500	0.048	0.087
trees.DecisionStump	0.500	0.048	0.087
lazy.LWL	0.333	0.048	0.083
meta.ClassificationViaClustering	0.050	0.095	0.066
bayes.AODE	0	0	0
bayes.BayesNet	0	0	0
bayes.HNB	0	0	0
bayes.NaiveBayes	0	0	0
bayes.NaiveBayesSimple	0	0	0
bayes.NaiveBayesUpdateable	0	0	0
bayes.WAODE	0	0	0
functions.LibSVM	0	0	0
functions.RBFNetwork	0	0	0
meta.CVParameterSelection	0	0	0
meta.Grading	0	0	0
meta.MultiScheme	0	0	0
meta.RacedIncrementalLogitBoost	0	0	0
meta.Stacking	0	0	0
meta.StackingC	0	0	0
meta.Vote	0	0	0
rules.ZeroR	0	0	0

Table 21. Algorithm performance ranked by F-measure on the training set for $IncepOper_1$

Algorithm	Precision	Recall	F-measure
rules.Prism	0.750	0.800	0.774
rules.NNge	0.923	0.600	0.727
functions.SMO	0.813	0.650	0.722
functions.SimpleLogistic	0.857	0.600	0.706
bayes.BayesianLogisticRegression	0.917	0.550	0.687
trees.LADTree	0.917	0.550	0.687
trees.Id3	0.846	0.550	0.667
trees.FT	0.786	0.550	0.647
misc.VFI	0.733	0.550	0.629
meta.LogitBoost	0.769	0.500	0.606
meta.RandomCommittee	0.818	0.450	0.581
meta.AttributeSelectedClassifier	0.667	0.500	0.571
meta.END	0.667	0.500	0.571
meta.FilteredClassifier	0.667	0.500	0.571
meta.OrdinalClassClassifier	0.667	0.500	0.571
rules.DecisionTable	0.667	0.500	0.571
trees.BFTree	0.667	0.500	0.571
trees.J48	0.667	0.500	0.571
rules.JRip	0.625	0.500	0.556
trees.RandomForest	0.889	0.400	0.552
trees.ADTree	0.692	0.450	0.545
rules.PART	0.556	0.500	0.526
lazy.LWL	0.727	0.400	0.516
trees.SimpleCart	0.727	0.400	0.516
trees.RandomTree	0.600	0.450	0.514
meta.RandomSubSpace	0.778	0.350	0.483
trees.REPTree	0.700	0.350	0.467
misc.HyperPipes	0.636	0.350	0.452
meta.Decorate	0.343	0.600	0.436
lazy.KStar	0.750	0.300	0.429
functions.Logistic	0.324	0.600	0.421
meta.MultiClassClassifier	0.324	0.600	0.421
lazy.IBk	0.600	0.300	0.400
meta.RotationForest	0.714	0.250	0.370
functions.VotedPerceptron	0.625	0.250	0.357
lazy.IB1	0.625	0.250	0.357

trees.J48graft	0.625	0.250	0.357
meta.ThresholdSelector	0.296	0.400	0.340
rules.Ridor	0.500	0.250	0.333
bayes.AODEsr	0.233	0.500	0.317
meta.ClassificationViaRegression	0.571	0.200	0.296
meta.Bagging	0.600	0.150	0.240
meta.AdaBoostM1	0.500	0.100	0.167
meta.MultiBoostAB	0.500	0.100	0.167
rules.OneR	0.500	0.100	0.167
trees.DecisionStump	0.500	0.100	0.167
meta.EnsembleSelection	0.400	0.100	0.160
rules.ConjunctiveRule	0.500	0.050	0.091
meta.ClassificationViaClustering	0.016	0.050	0.024
bayes.AODE	0	0	0
bayes.BayesNet	0	0	0
bayes.HNB	0	0	0
bayes.NaiveBayes	0	0	0
bayes.NaiveBayesSimple	0	0	0
bayes.NaiveBayesUpdateable	0	0	0
bayes.WAODE	0	0	0
functions.LibSVM	0	0	0
functions.RBFNetwork	0	0	0
functions.Winnow	0	0	0
meta.CVParameterSelection	0	0	0
meta.Dagging	0	0	0
meta.Grading	0	0	0
meta.MultiScheme	0	0	0
meta.RacedIncrementalLogitBoost	0	0	0
meta.Stacking	0	0	0
meta.StackingC	0	0	0
meta.Vote	0	0	0
rules.ZeroR	0	0	0

Table 22. Algorithm performance ranked by F-measure on the training set for ContOper$_1$

Algorithm	Precision	Recall	F-measure
functions.SimpleLogistic	0.909	0.769	0.833
rules.DecisionTable	0.909	0.769	0.833
meta.AttributeSelectedClassifier	0.833	0.769	0.800
meta.END	0.833	0.769	0.800
meta.FilteredClassifier	0.833	0.769	0.800
meta.OrdinalClassClassifier	0.833	0.769	0.800
rules.JRip	0.833	0.769	0.800
rules.PART	0.833	0.769	0.800
trees.BFTree	0.833	0.769	0.800
trees.J48	0.833	0.769	0.800
trees.SimpleCart	0.833	0.769	0.800
functions.Logistic	0.733	0.846	0.786
meta.MultiClassClassifier	0.733	0.846	0.786
meta.Bagging	0.900	0.692	0.783
rules.Prism	0.750	0.818	0.783
rules.Ridor	0.90	0.692	0.783
functions.SMO	0.818	0.692	0.750
meta.LogitBoost	0.818	0.692	0.750
rules.NNge	0.818	0.692	0.750
trees.FT	0.818	0.692	0.750
trees.Id3	0.818	0.692	0.750
trees.LADTree	0.818	0.692	0.750
meta.EnsembleSelection	0.889	0.615	0.727
meta.RandomSubSpace	0.889	0.615	0.727
lazy.LWL	0.800	0.615	0.696

trees.ADTree	0.800	0.615	0.696
trees.REPTree	0.800	0.615	0.696
bayes.BayesianLogisticRegression	0.778	0.538	0.636
meta.ClassificationViaRegression	0.857	0.462	0.600
meta.MultiBoostAB	0.857	0.462	0.600
rules.OneR	0.857	0.462	0.600
trees.DecisionStump	0.857	0.462	0.600
trees.RandomForest	0.857	0.462	0.600
functions.VotedPerceptron	0.750	0.462	0.571
meta.RandomCommittee	0.750	0.462	0.571
meta.AdaBoostM1	0.833	0.385	0.526
meta.RotationForest	0.833	0.385	0.526
meta.ThresholdSelector	0.714	0.385	0.500
trees.J48graft	0.714	0.385	0.500
lazy.IBk	0.625	0.385	0.476
meta.Decorate	0.345	0.769	0.476
trees.RandomTree	0.556	0.385	0.455
lazy.KStar	0.667	0.308	0.421
rules.ConjunctiveRule	1.000	0.231	0.375
lazy.IB1	0.600	0.231	0.333
misc.VFI	0.227	0.385	0.286
bayes.AODEsr	0.143	0.538	0.226
misc.HyperPipes	0.150	0.231	0.182
meta.Dagging	0.500	0.077	0.133
functions.RBFNetwork	0.333	0.077	0.125
functions.Winnow	0.095	0.154	0.118
bayes.AODE	0	0	0
bayes.BayesNet	0	0	0
bayes.HNB	0	0	0
bayes.NaiveBayes	0	0	0
bayes.NaiveBayesSimple	0	0	0
bayes.NaiveBayesUpdateable	0	0	0
bayes.WAODE	0	0	0
functions.LibSVM	0	0	0
meta.ClassificationViaClustering	0	0	0
meta.CVParameterSelection	0	0	0
meta.Grading	0	0	0
meta.MultiScheme	0	0	0
meta.RacedIncrementalLogitBoost	0	0	0
meta.Stacking	0	0	0
meta.StackingC	0	0	0
meta.Vote	0	0	0
rules.ZeroR	0	0	0

Table 23. Algorithm performance ranked by F-measure on the training set for FWC

Algorithm	Precision	Recall	F-measure
rules.Prism	0.639	0.702	0.669
bayes.BayesianLogisticRegression	0.658	0.629	0.643
functions.SMO	0.656	0.623	0.639
bayes.BayesNet	0.609	0.667	0.637
meta.RandomCommittee	0.639	0.635	0.637
trees.Id3	0.627	0.635	0.631
trees.FT	0.642	0.610	0.626
meta.Decorate	0.627	0.591	0.608
misc.VFI	0.542	0.692	0.608
rules.NNge	0.628	0.585	0.606
bayes.WAODE	0.634	0.579	0.605
lazy.IBk	0.577	0.635	0.605
meta.RotationForest	0.701	0.516	0.594
trees.RandomForest	0.625	0.566	0.594

trees.RandomTree	0.611	0.572	0.591
trees.LADTree	0.608	0.566	0.586
lazy.KStar	0.599	0.572	0.585
lazy.IB1	0.571	0.585	0.578
meta.EnsembleSelection	0.731	0.478	0.578
trees.ADTree	0.636	0.528	0.577
meta.Bagging	0.700	0.484	0.572
trees.REPTree	0.607	0.535	0.569
functions.SimpleLogistic	0.718	0.465	0.565
meta.ClassificationViaRegression	0.612	0.516	0.560
trees.BFTree	0.688	0.472	0.560
bayes.AODEsr	0.525	0.597	0.559
rules.PART	0.609	0.509	0.555
functions.VotedPerceptron	0.627	0.497	0.554
trees.SimpleCart	0.622	0.497	0.552
bayes.NaiveBayes	0.626	0.484	0.546
bayes.NaiveBayesSimple	0.626	0.484	0.546
bayes.NaiveBayesUpdateable	0.626	0.484	0.546
rules.DecisionTable	0.626	0.484	0.546
bayes.AODE	0.647	0.472	0.545
misc.HyperPipes	0.440	0.711	0.543
meta.Dagging	0.719	0.434	0.541
rules.JRip	0.610	0.472	0.532
functions.Logistic	0.449	0.560	0.499
meta.MultiClassClassifier	0.449	0.560	0.499
meta.ThresholdSelector	0.435	0.585	0.499
functions.RBFNetwork	0.625	0.409	0.494
meta.END	0.714	0.377	0.494
meta.FilteredClassifier	0.714	0.377	0.494
meta.OrdinalClassClassifier	0.714	0.377	0.494
meta.RandomSubSpace	0.714	0.377	0.494
trees.J48	0.714	0.377	0.494
meta.AttributeSelectedClassifier	0.687	0.358	0.471
trees.J48graft	0.746	0.333	0.461
lazy.LWL	0.557	0.371	0.445
bayes.HNB	0.718	0.321	0.443
rules.Ridor	0.654	0.321	0.430
meta.AdaBoostM1	0.581	0.340	0.429
trees.DecisionStump	0.550	0.346	0.425
meta.MultiBoostAB	0.552	0.333	0.416
functions.Winnow	0.374	0.447	0.407
meta.LogitBoost	0.638	0.277	0.386
rules.ConjunctiveRule	0.551	0.239	0.333
rules.OneR	0.419	0.164	0.235
meta.ClassificationViaClustering	0.188	0.132	0.155
functions.LibSVM	0	0	0
meta.CVParameterSelection	0	0	0
meta.Grading	0	0	0
meta.MultiScheme	0	0	0
meta.RacedIncrementalLogitBoost	0	0	0
meta.Stacking	0	0	0
meta.StackingC	0	0	0
meta.Vote	0	0	0
rules.ZeroR	0	0	0

7.1.4. Analysis of the Results

> ### Three Hypothesis in [Wanner et al. 2006] and Our Results

In Section 3.5.3, we have mentioned three hypotheses expressed in [Wanner *et al*. 2006]. These hypotheses are three possible methods of how humans recognize and learn collocations. Now we formulate these three hypothetic methods and comment them using our results.

Method 1. Collocations can be recognized by their similarity to the prototypical sample of each collocational type; this was modeled by Nearest Neighbor technique. WEKA implements the nearest neighbor method in the following classifiers: NNge, IB1, IBk and KStar [Witten and Frank, 2005]. NNge belongs to the class `rules`; classifiers of this type are effective on large data sets, and our training sets include many examples (900) and possess high dimensionality. NNge is an algorithm that generalizes examples without nesting or overlap. NNge is an extension of Nge, which performs generalization by merging examples exemplars, forming hyper-rectangles in feature space that represent conjunctive rules with internal disjunction. NNge forms a generalization each time a new example is added to the database, by joining it to its nearest neighbor of the same class. Unlike Nge, it does not allow hyper-rectangles to nest or overlap. This is prevented by testing each prospective new generalization to ensure that it does not cover any negative examples, and by modifying any generalizations that are later found to do so. NNge adopts a heuristic that performs this post-processing in a uniform fashion. The more details about this algorithm can be found in [Roy 2002].

In spite of its advantages, NNge does no perform very well on predicting LFs. The only LF which NNge predicts well is $IncepOper_1$ with the meaning 'to begin doing something' (F-measure of 0.727). For the rest 8 LFs NNge does not show high results: for $Oper_1$ the value of F-measure is 0.817, for $ContOper_1$ it is 0.750, for $CausFunc_1$ the value of F-measure is 0.702, for FWC it is 0.606, for $Oper_2$ it is 0.600, for $Real_1$ it is 0.593, for $CausFunc_0$ it is 0.543, for $Func_0$ it is 0.522. The average F-measure for all nine LFs is 0.651.

Classifiers IB1, IBk and KStar, also based on the nearest neighbor algorithm, belong to the class lazy. It means that their learning time is very short, in fact learning in its full sense does not take place in these algorithms. An unseen example is compared with all instances annotated with LFs, and the LF whose instance is closer to the unseen example, is assigned to the latter. On our training sets, IB1, IBk and KStar show worse performance than NNge. Here we give average values of F-measure for IB1, IBk and KStar. The average value of F-measure for $Oper_1$ is 0.616, for FWC it is 0.589, for $CausFunc_1$ it is 0.466, for $ContOper_1$ it is 0.410, for $Oper_2$ it is 0.407, for $CausFunc_0$ it is 0.404, for $IncepOper_1$ it is 0.395, for $Real_1$ it is 0.378, for $Func_0$ it is 0.340. The

average F-measure for all nine LFs is 0.445. It means that it is difficult to find a very distinctive prototypical LF instance that could indeed distinguish the meaning of a particular LF. $IncepOper_1$ is an exception here, and the distance between the examples of $IncepOper_1$ and the examples of all the rest of LF is significantly bigger that the distance between the examples of $IncepOper_1$. But for 8 LFs, our results demonstrate that Method 1 does not produce high quality results.

Method 2. Collocations can be recognized by similarity of semantic features of collocational elements to semantic features of elements of collocations known to belong to a specific LF; this was modeled by Naïve Bayesian network and a decision tree classification technique based on the ID3-algorithm. We tested three WEKA Naïve Bayesian classifiers – NaiveBayes, NaiveBayesSimple, and NaiveBayesUpdateable [Witten and Frank, 2005]. All three classifiers show low results for $Oper_1$ (F-measure of 0.711), FWC (F-measure of 0.546), $CausFunc_0$ (F-measure of 0.308), $CausFunc_1$ (F-measure of 0.077), $Real_1$ (F-measure of 0.040). These three Bayesian classifiers failed to predict $Func_0$, $Oper_2$, $IncepOper_1$, $ContOper_1$; their result for these LFs is 0. Bayesian classifiers are based on the assumption that attributes used to represent data are independent. As the results show, this is not the case with our training sets. Indeed, the features used to represent verb-noun combinations are hyperonyms of the verb and hyperonyms of the noun. Hyperonyms are organized in a hierarchy, and are placed in a fixed order in the branches of their taxonomy, so that the position of each hyperonym depends on the location of other hyperonyms in the same branch. Moreover, it seems that in verb-noun pairs, the verb depends on the noun since they both form a collocation but not a free word combination where there is no lexical dependency between the constituents. These are the reasons why Bayesian algorithms perform poorly on LFs. Since Bayesian methods intent to model human recognition of collocational relations, it can be concluded, that our results for Bayesian classifiers do not support the hypothesis of learning collocations by similarity of semantic features of collocational elements to semantic features of elements of collocations known to belong to a specific LF.

Now let is consider another machine learning algorithm that model the same method of collocation recognition, ID3 algorithm. This algorithm is implemented in WEKA as Id3 classifier of the class `trees`. It shows the top result for $CausFunc_1$ as compared with the performance of the other classifiers for predicting the same LF, (F-measure of 0.762), the second best result for predicting $Oper_1$ (F-measure of 0.870), another second best result for $ContOper_1$ (F-measure of 0.800). For $Oper_2$ the value of F-measure is 0.706, for $IncepOper_1$ it is 0.667, for $Func_1$ the value of F-measure is 0.636, for FWC it is 0.631, for $CausFunc_0$ the value of F-measure is 0.621, for $Real_1$ it is 0.587. The average F-measure for all nine LFs is 0.698. This

classifier gives average results close to the results of NNge of Method 1 but which still are rather low. However, for Oper1 and ContOper1 the results are quite satisfactory. It means that the semantic features of these two lexical function which in our case are hyperonyms, distinguish sufficiently well these two lexical functions from the rest seven LFs.

Method 3. The third method was modeled by Tree-Augmented Network (TAN) Classification technique. As it is seen from results in [Wanner *et al*. 2006] demonstrated in Table 5, the nearest neighbor algorithm gives better results in terms of recall than TAN. So it can be concluded that there are more evidence in favor of Method 2 than Method 3. We did not apply TAN method in our experiments.

➢ ***Best Machine Learning Algorithms***

Now we consider the best algorithm for each LF chosen for our experiments.

Oper$_1$: the Best Algorithm is BayesianLogisticRegression

For Oper$_1$, BayesianLogisticRegression has shown precision of 0.879, recall of 0.866, and F-measure of 0.873.

Logistic Regression is an approach to learning functions of the form $f : X \rightarrow Y$, or $P(Y|X)$ in the case where Y is discrete-valued, and $X = \{X_1 ...X_n\}$ is any vector containing discrete or continuous variables. In the case of our training data, the variables are discrete. X is a variable for the attributes, and Y is a Boolean variable which corresponds to the class attribute in the training set. Logistic Regression assumes a parametric form for the distribution $P(Y|X)$, then directly estimates its parameters from the training data. Logistic Regression is trained to choose parameter values that maximize the conditional data likelihood. The conditional data likelihood is the probability of the observed Y values in the training data, conditioned on their corresponding X values.

CausFunc$_0$: the Best Algorithm is JRip

For CausFunc$_0$, JRip has shown precision of 0.747, recall of 0.705, and F-measure of 0.725.

JRip (Extended Repeated Incremental Pruning) implements a propositional rule learner, "Repeated Incremental Pruning to Produce Error Reduction" (RIPPER), as proposed in [Cohen 1995]. JRip is a rule learner alike in principle to the commercial rule learner RIPPER.

CausFunc$_1$: the Best Algorithm is J48

For CausFunc$_1$, J48 has shown precision of 0.842, recall of 0.696, and F-measure of 0.762.

J48 is a rule base classifier algorithm that generates C4.5 decision trees. J48 is the C4.5 clone implemented in the WEKA data mining library. In its turn, C4.5 implements ID3 algorithm. The basic ideas behind ID3 are the following. Firstly, in the decision tree each node corresponds to a non-categorical attribute and each arc to a possible value of that attribute. A leaf of the tree specifies the expected value of the categorical attribute for the records described by the path from the root to that leaf. Secondly, in the decision tree at each node should be associated the non-categorical attribute which is most informative among the attributes not yet considered in the path from the root. Thirdly, entropy is used to measure how informative is a node. C4.5 is an extension of ID3 that accounts for unavailable values, continuous attribute value ranges, pruning of decision trees, rule derivation, etc.

$Real_1$: the Best Algorithm is Prism

For $Real_1$, Prism has shown precision of 0.735, recall of 0.832, and F-measure of 0.781.

Prism is a rule based classification algorithms It is based on the inductive rule learning and uses separate-and-conquer strategy. It means, that a rule that works for many instances in the class is identified first, then the instances covered by this rule are excluded from the training set and the learning continues on the rest of the instances. These learners are efficient on large, noisy datasets. Our training sets included 900 instances represented as vectors of the size 1109 attributes, and rule induction algorithms performed very well.

$Func_0$: the Best Algorithm is BFTree

For $Func_0$, BFTree has shown precision of 0.667, recall of 0.727, and F-measure of 0.696.

BFTree is a best-first decision tree learner and it is a learning algorithm for supervised classification learning [19]. Best-first decision trees represent an alternative approach to standard decision tree techniques such as C4.5 algorithm since they expand nodes in best-first order instead of a fixed depth-first order.

$Oper_2$: the Best Algorithm is PART

For $Oper_2$, PART has shown precision of 0.923, recall of 0.571, and F-measure of 0.706.

PART is a rule base classifier that generates partial decision trees rather that forming one whole decision tree in order to achieve the classification.

$IncepOper_1$: the Best Algorithm is Prism

For $IncepOper_1$, Prism has shown precision of 0.750, recall of 0.800, and F-measure of 0.774.

Since Prism is also the best classifier for $Real_1$, it has been described above. We remind, that for $Real_1$, Prism has shown precision of 0.735, recall of 0.832, and F-measure of 0.781. $IncepOper_1$ has the meaning 'to begin doing what is designated by the noun, and $Real_1$ means 'to act according to the situation designated by the noun'. The meaning of $IncepOper_1$ is distinguished better because the semantic element 'begin' is very different from 'act' which is more general and has more resemblance with the meaning 'perform' ($Oper_1$) or 'cause to exist' ($CausFunc_0$, $CausFunc_1$).

ContOper$_1$: the Best Algorithm is SimpleLogistic

For $ContOper_1$, SimpleLogistic has shown precision of 0.909, recall of 0.769, and F-measure of 0.833.

Simple logistic regression finds the equation that best predicts the value of the Y variable (in our case, the class attribute) for each value of the X variable (in our training data, values of attributes that represent hyperonyms). What makes logistic regression different from linear regression is that the Y variable is not directly measured; it is instead the probability of obtaining a particular value of a nominal variable. The algorithm calculates the probability of Y applying the likelihood ratio method. This method uses the difference between the probability of obtaining the observed results under the logistic model and the probability of obtaining the observed results in a model with no relationship between the independent and dependent variables.

FWC: the Best Algorithm is Prism

For FWC, Prism has shown precision of 0.639, recall of 0.702, and F-measure of 0.669.

Since Prism is also the best classifier for $Real_1$, it has been described above. Prism has the top result for $IncepOper_1$ as well. Still, for free word combinations, Prism show weaker results that for predicting $IncepOper_1$ and $Real_1$. We remind that for $IncepOper_1$, Prism has shown precision of 0.750, recall of 0.800, and F-measure of 0.774; and for $Real_1$, this algorithm has shown precision of 0.735, recall of 0.832, and F-measure of 0.781.

The meaning of free word combinations is less distinguishable since their semantic content is very diverse in comparison with the meaning of lexical functions.

7.2. Algorithm Performance Evaluation on the Test Set

Some of the algorithms that showed best results for predicting LFs were evaluated on an independent test set built as described in Section 4.2, Section 5.4.3, and Section 6.2.2. Tables 24, 25 present the results for these algorithms. We listed the values of precision, recall and *f-*

measure for each classifier in this way: <precision>|<recall>|<f-measure>; BLR in the column *Algorithm* stands for BayesianLogisticRegression. As we explain below, the test sets had such a big size that some classifiers failed to make predictions within a reasonable time period. For such classifiers, we put N/A instead of metrics as for the other classifiers.

Table 24. Algorithm performance on the test set

Meaning	Algorithm	Test set size	
		100%	75%
	PART	0.261|0.864|0.400	0.304|0.864|0.382
$Oper_1$	SimpleCart	N/A	N/A
	BLR	0.178|0.830|0.293	0.212|0.818|0.337
	JRip	0.231|0.662|0.342	0.189|0.662|0.294
$CausFunc_0$	SimpleCart	N/A	0.168|0.662|0.268
	LADTree	0.285|0.676|0.401	0.168|0.676|0.269
	FT	N/A	N/A
$IncepOper_1$	SMO	0.331|0.793|0.467	0.567|0.793|0.661
	NNge	0.302|0.724|0.426	0.567|0.724|0.636
	Ridor	**0.799**|0.480|0.600	**0.993**|0.480|0.647
$ContOper_1$	REPTree	0.581|0.480|0.526	**0.820**|0.480|0.606
	LWL	N/A	N/A

It was mentioned in Section 4.2, that since we did not disambiguate verb-noun pairs in the test sets, for each pair we build the number of instances equal to the number of senses for the verb multiplied by the number of senses for the noun. Remember, that this has given us 96079 instances and 10544 attributes in 100% test set, 73021 instances and 9495 attributes in 75% test set, 48904 instances and 8032 attributes in 50% test set, and 22254 instances and 5857 attributes in 25% test set. SimpleCart, FT, LWL had difficulties in predicting the value of the class variable on test sets of sizes more than 25%. Among these three algorithms, SimpleCart was better because this algorithm was effective enough to process a 75% and 50% set. SimpleCart and FT are decision tree algorithms, and LWL is a nearest-neighbor instance-based learner. Note, that almost all the best classifiers that could process a full-size test set, belong to the class rules. BayesianLogisticRegression also performs well and the only algorithm of the class trees that did not experience time problems was LADTree.

Table 25. Algorithm performance on the test set

Meaning	Algorithm	Test set size	
		50%	25%
	PART	0.245|0.864|0.382	0.162|0.852|0.272
$Oper_1$	SimpleCart	0.405|0.864|0.551	0.281|0.852|0.423
	BLR	0.205|0.818|0.328	0.145|0.807|0.246
	JRip	0.189|0.662|0.294	0.174|0.662|0.276
$CausFunc_0$	SimpleCart	0.203|0.662|0.311	0.177|0.662|0.279
	LADTree	0.144|0.676|0.237	0.177|0.676|0.281
$IncepOper_1$	FT	N/A	0.409|0.724|0.523
	SMO	0.464|0.793|0.585	0.404|0.793|0.535

	NNge	0.603|0.724|**0.658**	0.451|0.724|0.556
	Ridor	**0.958**|0.480|0.640	**1.000**|0.480|**0.649**
ContOper$_1$	REPTree	0.694|0.480|0.567	0.667|0.480|0.558
	LWL	N/A	**1.000**|0.480|**0.649**

As it is seen from Tables 24, 25, the best precision was shown by Ridor. This method (Ridor = RIpple-DOwn Rule learner) have been developed for knowledge acquisition where it is hard to add a new rule and be sure that it would not cause the inconsistency of the rules generated before. Ridor algorithm is different from covering algorithms for constructing the rule set; instead it generates exceptions for the existing rules that work within the confines of these rules thus not affecting other rules. Then it iterates on the exceptions for the best solution. This scheme allowed the classifier to reach 100% precision. Unfortunately, it can not boast the best recall which is only 0.649 for ContOper$_1$ on a 25% test set. Still, it is the second best recall in our experiments on test sets. The top recall is 0.658 shown by NNge for the meaning BEGIN on a 50% test set.

Another algorithm that gives the best precision of 100% is LWL when performing predictions for the meaning ContOper$_1$ on a 25% test set. But, like Ridor, it shows the same low recall of 0.658. However, a high precision of Ridor and LWL makes them appropriate for fulfilling the tasks where precision is of special importance, for example, for automatic construction of dictionaries.

Chapter 8. Computational Applications of Lexical Functions

8.1. Important Properties of Lexical Functions with Respect to Applications

1. LFs are universal. It means that a significantly little number of LFs (about 70) represent the fundamental semantic relations between words in the vocabulary of any natural language (paradigmatic relations) and the basic semantic relations which syntactically connected word forms can obtain in the text (syntagmatic relations).

2. LFs are idiomatic. LFs are characteristic for idioms in many natural languages. An example is the lexical function Magn which means 'a great degree of what is denoted by the key word'. In English, it is said *to sleep soundly* and *to know firmly,* not the other way round: *to sleep firmly* or *to know soundly.* But in Russian the combination *krepko spat'* (literally *to sleep firmly*) is quite acceptable although it is not natural in English.

3. LFs can be paraphrased. For example, the LFs Oper and Func can form combinations with their arguments which are synonymous to the basic verb like in the following utterances: *The government controls prices – The government has control of prices –* The *government keeps prices under control – The prices are under the government's control.* Most paradigmatic lexical functions (synonyms, antonyms, converse terms, various types of syntactic derivatives) can also substitute for the keyword to form synonymous sentences.

4. LFs are diverse semantically. Sometimes the values of the same LF from the same key word are not synonymous. This is especially characteristic of the LF Magn. We can describe a great degree of *knowledge* in the following three ways: a) as *deep* or *profound*; b) as *firm*; c) as *broad* or *extensive.* Although all these adjectives are valid values of the Magn, the three groups should somehow be distinguished from each other because the respective adjectives have very different scopes in the semantic representation of the keyword. *Deep* and *profound* characterize *knowledge* with regard to the depth of understanding; *firm* specifies the degree of its assimilation; *broad* and *extensive* refer to the amount of acquired knowledge. It was proposed in [Apresjan *et al*. 2003] that in order to keep such distinctions between different values of the same LFs in the computerized algebra of LFs it is sufficient to ascribe to the standard name of an LF the symbol NS (non-standardness) plus a numerical index and maintain the correspondences between the

two working languages by ascribing the same names to the respective LFs in the other language.

8.2. Lexical Functions in Word Sense Disambiguation

Syntagmatic LFs can be used to resolve syntactic and lexical ambiguity. Both types of ambiguity can be resolved with the help of LFs Oper, Func and Real. LFs like Magn (with the meaning 'very', intensifier), Bon (with the meaning 'good'), Figur (with the meaning 'metaphoric expression typical of the key word') among others, can be used to resolve lexical ambiguity.

8.2.1. Syntactic Ambiguity

Syntactic ambiguity and its resolution with the help of LFs is explained in this section by means of an example. Let us consider such English phrases as *support of the parliament* or *support of the president.* The word *support* is object in the first phrase, but it is subject (agent) in the second phrase. Syntactically, both phrases are identical: *support* + the preposition *of* + a noun, this is the sourse of syntactic ambiguity, and for that reason both phrases may mean both: 'support given by the parliament (by the president)', which syntactically is the subject interpretation with the agentive syntactic relation between *support* and the subordinated noun, and 'support given to the parliament (to the president)' which syntactically is the object interpretation with the first completive syntactic relation between *support* and the subordinated noun. This type of ambiguity is often extremely difficult to resolve, even within a broad context. LF support verbs can be successfully used to disambiguate such phrases because they impose strong limitations on the syntactic behaviour of their keywords in texts.

Now let us view the same phrases in a broader context. The first example is *The president spoke in support of the parliament,* where the verb *to speak in* is Oper$_1$ the noun *support.* Verbs of the Oper$_1$ type may form collocations with their keyword only on condition that the keyword does not subordinate directly its first actant. The limitation is quite natural: Oper$_1$ is by definition a verb whose grammatical subject represents the first actant of the keyword. Since the first actant is already represented in the sentence in the form of the grammatical subject of Oper$_1$, there is no need to express it once again. This is as much as to say that the phrase *The president spoke in support of the parliament* can only be interpreted as describing the support given to the parliament, with *parliament* fulfilling the syntactic function of the complement of the noun *support*.

On the other hand, verbs of the Oper$_2$ type may form such collocations only on condition that the keyword does not subordinate directly its second actant. Again, the limitation is quite natural: Oper$_2$ is by definition a verb whose grammatical subject represents the second actant of the keyword. Since the second actant is already represented in the sentence in the form of the grammatical subject of Oper$_2$, there is no need to express it once again. So in the second example we consider in this section, *The president enjoyed* (Oper$_2$) *the support of the parliament,* the phrase *the support of the parliament* implies the support given to the president by the parliament, with *parliament* fulfilling the syntactic function of the agentive dependent of the noun *support*.

In cases of syntactic ambiguity, syntactically identical phrases are characterized by different lexical functions which in this case serve as a tool of disambiguation.

8.2.2. Lexical Ambiguity

LFs are also useful in resolving lexical ambiguity. For the sake of brevity, we shall give only one illustrative example. The Russian expression *provodit' razlichie* and its direct English equivalent *to draw a distinction* can be analyzed as composed of OPER1 + its keyword. Taken in isolation, the Russian and the English verbs are extremely polysemous, and choosing the right sense for the given sentence becomes a formidable problem. *Provodit'*, for example, has half a dozen senses ranging from 'spend' via 'perform' to 'see off', while *draw* is a polysemic verb for which dictionaries list 50 senses or more. However, in both expressions the mutual lexical attraction between the argument of the LF and its value is so strong that, once the fact of their co-occurrence is established by the parser, we can safely ignore all other meanings and keep for further processing only the one relevant here.

8.3. Lexical Functions in Computer-Assisted Language Learning

One of the purposes in developing software for language learning is to acquire lexicon. It has been proposed (for example, in [Diachenko 2006]) to organize learning in the form of linguistic games. There are games that operate on a word dictionary, but in order to learn collocations, a lexical function dictionary can be used whose advantage is that it includes the linguistic material on word combinations which is absent in word dictionaries. Below an example of a game oriented to lexical functions is given.

Game "Lexical Function"

In the game "LF", the user needs to supply values of a concrete LF for each given argument. The user chooses the LF she is going to play with and enters the number of questions. The

system gathers the material for the game by random selection of arguments from the dictionary. The system also shows the user the definition of the LF and two examples.

According to the difficulty of learning, all LFs were divided into 3 levels.

While some LF values have a compound format and may include the argument itself or pronouns, the system generates hints for such values. For example, CausFact$_0$ (with the meaning 'to cause something to function according to its destination) for *clock* (in the sense of 'time-measuring instrument) is *wind up* (*a watch*) / *start* (*a watch*). For this value the hint will look like this:

"— (*a watch*)"

CausFact$_0$ of *imagination* is *fire* (somebody's *imagination*). The hint will look like this

"— (somebody) (*imagination*)".

If the user cannot supply an answer, the system shows him the list of correct answers.

8.4. Lexical Functions in Machine Translation

Two important properties of LFs mentioned in Section 8.1., i.e. their semantic universality and cross-linguistic idiomaticity, make them an ideal tool for selecting idiomatic translations of set expressions in a MT system. The way it can be done is explained by an example of prepositions in English and Spanish.

As is well known, locative prepositions used to form prepositional phrases denoting places, sites, directions, time points, periods, intervals etc. reveal great versatility within one language and incredibly fanciful matching across languages. If we were to account properly for the discrepancies existing between the uses of these prepositions, say, in English and Spanish, we would have to write too detailed translation rules involving complicated semantic and pragmatic data. However, a large part of the task may be achieved with the help of LFs.

Consider the following correspondences between English and Russian that may be easily found with the help of the LFs Dir (preposition denoting a movement toward the location expressed by the key word):

Dir (city) = to (the city), Dir (cuidad) = a (la cuidad)
Dir (friend) = to (my friend), Dir (amiga) = con (mi amiga)

In order to ensure the production of these equivalents in machine translation, we must only identify the arguments and the value of the LF during parsing and substitute the correct value from the target language dictionary during generation.

Implementation Example

A module that annotates word combinations with lexical functions (such word combinations will be collocations, see Section 3.4) represented in Fig. 12, can be included in any machine translation system based on interlingua like UNL. UNL is a project of multilingual personal networking communication initiated by the University of United Nations based in Tokyo [Uchida et al. 2006].

Fig. 12. Lexical Function Module of a Machine Translation System.
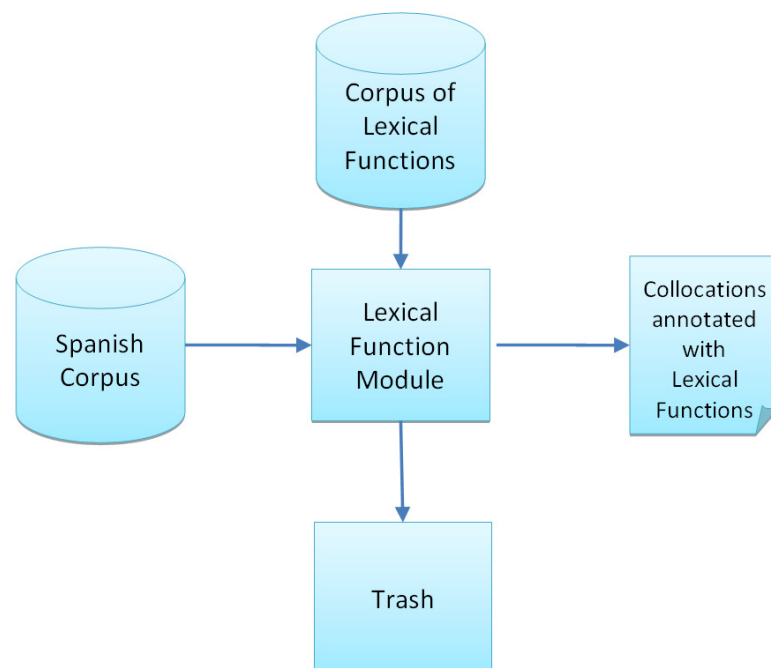
Chapter 9. Result Analysis with Respect to our Contribution to Linguistic Research

Linguistics as a scientific study of human language intends to describe and explain it. However, validity of a linguistic theory is difficult to prove due to volatile nature of language as a human convention and impossibility to cover all real-life linguistic data. In spite of these problems, computational techniques and modeling can provide evidence to verify or falsify linguistic theories. As a case study, we conducted a series of computer experiments on a corpus of Spanish verb-noun collocations using machine learning methods, in order to test a linguistic point that collocations in the language do not form an unstructured collection but are language items related via what we call collocational isomorphism, represented by lexical functions of the Meaning-Text Theory. Our experiments allowed us to verify this linguistic statement. Moreover, they suggested that semantic considerations are more important in the definition of the notion of collocation than statistical ones.

9.1. Computer Experiments in Linguistics

Computer experiments play a very important role in science today. Simulations on computers have not only increased the demand for accuracy of scientific models, but have helped the researcher to study regions which can not be accessed in experiments or would demand very costly experiments.

Our computational experiments made on the material of Spanish verb-noun collocations like *seguir el ejemplo*, follow the example, *satisfacer la demanda*, meet the demand, *tomar una decisión,* make a decision, can contribute to verify if the linguistic statement specified in Section 9.2 is true. Out results also make it possible to derive another important inference on the nature of collocation presented in Section 9.4.

It should be added here that testing a linguistic hypothesis on computer models not only demonstrates validity or rejection of the hypothesis, but also motivates the researcher to search for more profound explanations or to explore new approaches in order to improve computational operation. Thus, starting from one linguistic model, the researcher can evaluate it and then go further, sometimes into neighboring spheres of linguistic reality, in her quest of new solutions, arriving at interesting conclusions. One of the original intents of our research was to test one

linguistic model experimentally. If we develop a computer program on the premise of a certain linguistic model and this program accomplishes its task successfully, then the linguistic model being the basis of the program is thus verified. The results we obtained not only produced evidence for verifying a linguistic model or statement we make in the next section, but they also made it possible to get more insight into the nature of collocation which has been a controversial issue in linguistics for many years.

9.2. Linguistic Statement: our Hypothesis

Collocations are not a stock or a "bag" of word combinations, where each combination exists as a separate unit with no connection to the others, but they are related via collocational isomorphism represented as lexical functions.

9.2.1. Collocational Isomorphism

Considering collocations of a given natural language, it can be observed that collocations are not just a "bag" of word combinations, as a collection of unrelated items where no association could be found, but there are lexical relations among collocations, and in particular, we study the lexical relation which may be called 'collocational isomorphism'. It has some resemblance to synonymy among words which is the relation of semantic identity or similarity. Collocational isomorphism is not a complete equality of the meaning of two or more collocations, but rather a semantic and structural similarity between collocations.

What do we mean by semantic and structural similarity between collocations? For convenience of explanation, we will comment on the structural similarity of collocations first. The latter is not a novelty, and a detailed structural classification of collocations (for English) was elaborated and used to store collocational material in the well-known dictionary of word combinations *The BBI Combinatory Dictionary of English* (Benson *et al.* 1997). However, we will exemplify collocational structures with Spanish data, listing some typical collocates of the noun *alegría*, joy:

> verb + noun: *sentir alegría*, to feel joy
>
> adjective + noun: *gran alegría*, great joy
>
> preposition + noun: *con alegría*, with joy
>
> noun + preposition: *la alegría de* (esa muchacha), the joy of (this girl).

The above examples are borrowed from the dictionary of Spanish collocations entitled *Diccionario de colocaciones del Español* [Alonso Ramos 2003], a collection of collocations in

which the bases are nouns belonging to the semantic field of emotions. So collocations have structural similarity when they share a common syntactic structure.

We say that two or more collocations are similar semantically if they possess a common semantic content. In Table 26, we present collocations with the same syntactic structure, namely, 'verb + noun'. For these collocations, the meaning is given for us to see what semantic element can be found that is common to all of them.

Table 26. Verb-noun collocations and their meaning

Spanish collocation	English literal translation	Translation into natural English	Meaning of collocation	English translation
hacer uso	make use	make use	*usar*	use
dar un abrazo	give a hug	give a hug	*abrazar*	hug
prestar atención	lend attention	pay attention	*fijarse*	pay attention
tener interés	have interest	take interest	*interesarse*	be interested
tomar la medida	take measure	take action	*actuar*	act

Table 27. Verb-noun collocations grouped according to their common semantic pattern

Semantic pattern	Spanish collocations	English literal translation	Translation into natural English
create an entity or process	*escribir un libro* *elaborar un plan* *construir la sociedad* *dar vida*	write a book elaborate a plan construct a society give life	write a book develop a plan build a society give life
intensify a property or attribute	*aumentar el riesgo* *elevar el nivel* *desarrollar la capacidad* *mejorar la condición*	increase the risk lift the level develop a capacity improve a condition	increase the risk raise the level develop a capacity improve a condition
reduce a property or attribute	*disminuir la probabilidad* *reducir el consumo* *bajar el precio* *limitar los derechos*	lessen the probability reduce consumption lower the price limit rights	lower chances reduce consumption bring down the price restrict rights
begin to realize an action or begin to manifest an attribute	*iniciar la sesión* *tomarla palabra* *asumir el papel* *adoptar una actitud*	initiate a session take the word assume a role adopt the attitude	start a session take the floor assume a role take the attitude
preserve a property or process	*mantener el equilibrio* *guardar silencio* *seguir el modelo* *llevar una vida*	maintain the balance keep silence follow a model carry a life	keep the balance keep quiet follow an example lead a life

It may be noted that the meaning of all collocations in Table 26 is generalized as 'do, carry out or realize what is denoted by the noun', in other words, that these collocations are built according to the semantic pattern 'do the noun'. In turn, observing the meaning of the nouns, we see that their semantics can be expressed in general terms as 'action' (*uso*, *abrazo*, *medida*) or 'psychological attribute' (*atención*, *interés*), so the resulting semantic pattern of the collocations in Table 26 is 'do an action / manifest a psychological attribute'. Since these collocations share

common semantics and structure, we may say that they are isomorphic, or that they are tied to one another by the relation we termed above as 'collocational isomorphism'. Table 27 gives more examples of isomorphic collocations.

9.2.2. Collocational Isomorphism Represented as Lexical Functions

Several attempts to conceptualize and formalize semantic similarity of collocations have been made. As far back as in 1934, the German linguist W. Porzig claimed that on the syntagmatic level, the choice of words is governed not only by grammatical rules, but by lexical compatibility, and observed semantic similarity between such word pairs as *dog – bark*, *hand – grasp*, *food – eat*, *cloths – wear* [Porzig 1934]. The common semantic content in these pairs is 'typical action of an object'. Research of J. R. Firth [Firth 1957] drew linguists' attention to the issue of collocation and since then collocational relation has been studied systematically. In the article of J. H. Flavell and E. R. Flavell [Flavell and Flavell 1959] and in the paper by Weinreich [Weinreich 1969], there were identified the following meanings underlying collocational isomorphism: an object and its typical attribute (*lemon – sour*), an action and its performer (*dog – bark*), an action and its object (*floor – clean*), an action and its instrument (*axe – chop*), an action and its location (*sit – chair*, *lie – bed*), an action and its causation (*have – give*, *see – show*), etc. Examples from the above mentioned writings of Porzig, Flavell and Flavell, Weinreich are borrowed from [Apresjan 1995: 44].

The next step in developing a formalism representing semantic relations between the base and the collocate as well as semantic and structural similarity between collocations was done by I. Mel'čuk. Up to now, his endeavor has remained the most fundamental and theoretically well-grounded attempt to systematize collocational knowledge. This scholar proposed a linguistic theory called the Meaning-Text Theory, which explained how meaning, or semantic representation, is encoded and transformed into spoken or written texts [Mel'čuk 1974]. His theory postulates that collocations are produced by a mechanism called lexical function. Lexical function is a mapping from the base to the collocate; it is a semantically marked correspondence that governs the choice of the collocate for a particular base. About 70 lexical functions have been identified in [Mel'čuk 1996]; each is associated with a particular meaning according to which it receives its name. Table 28 demonstrates a correspondence between semantic patterns of collocations and lexical functions.

Table 28. Semantic patterns represented as lexical functions

Semantic pattern and examples	Complex lexical function representation	Complex lexical function description
create an entity or process *escribir un libro* *dar vida*	$\text{CausFunc}_0(libro) = excribir$ $\text{CausFunc}_0(vida) = dar$	CausFunc_0 = cause an entity or process to function.
intensify a property or attribute *aumentar el riesgo* *elevar el nivel*	$\text{CausPlusFunc}_1(riesgo) = aumentar$ $\text{CausPlusFunc}_1(nivel) = elevar$	CausPlusFunc_1 = cause that a property or attribute manifest itself to a larger degree.
reduce a property or attribute *disminuir la probabilidad* *reducir el consumo*	$\text{CausMinusFunc}_1(probabilidad) = disminuir$ $\text{CausMinusFunc}_1(consumo) = reducir$	CausMinusFunc_1 = cause that a property or attribute manifest itself to a lesser degree.
begin to realize an action or begin to manifest an attribute *iniciar la sesión* *adoptar una actitud*	$\text{IncepOper}_1(sesión) = iniciar$ $\text{IncepOper}_1(actitud) = adoptar$	IncepOper_1 = cause that an action begin to be realized or an attribute begin to manifest itself.
preserve a property or process *mantener el equilibrio* *guardar silencio*	$\text{ContOper}_1(probabilidad) = disminuir$ $\text{ContOper}_1(probabilidad) = disminuir$	ContOper_1 = cause that an action continue to be realized or an attribute continue to manifest itself.

Now we are going to see if computer experiments can supply evidence to the existence of collocational isomorphism as defined by lexical functions. The idea is to submit a list of collocations to the computer and see if it is able to distinguish collocations belonging to different lexical functions. If a machine can recognize lexical functions, then it is a strong testimony to their existence.

9.3. Discussion of Our Results: Testing the Linguistic Statement

One of the purposes of this work is to provide evidence for the linguistic statement made in the beginning of Section 9.2. Now let us review it in the light of our experimental results. The statement affirms that collocations are not a stock, or a "bag" of word combinations, where each combination exists as a separate unit with no connection to others, but they are related via collocational isomorphism represented as lexical functions.

What evidence have we obtained concerning lexical functions? We presented a sufficient number of collocations annotated with lexical functions to the computer that learned characteristic features of each function. It was demonstrated that the computer was able to assign lexical functions to unseen collocations with a significant average accuracy of 0.759. Is it satisfactory? We can compare our result with computer performance on another task of natural language processing: word sense disambiguation, i.e., identifying the intended meanings of words in context. Today, automated disambiguating systems reach the accuracy of about 0.700 and this is considered a substantial achievement. As an example of such works see [Zhong and Tou Ng 2010]. Therefore, our result is weighty enough to be a trustworthy evidence for the linguistic statement under discussion.

In Section 9.1 we stated, that if we develop a computer program on the premise of a certain linguistic model and this program accomplishes its task successfully, then the linguistic model being the basis of the program is thus verified. In our experiments, we have observed that machine learning methods are able to detect lexical functions of collocations. Thus lexical functions as a linguistic concept get evidence received in computational experiments which can be repeated on the same data as well as on new data. It means that the formalism of lexical functions is a legitimate model of collocational isomorphism described in Section 9.2.1.

9.4. Discussion of Our Results Concerning the Nature of Collocation: Statistical vs. Semantic Approach

What knowledge is necessary and sufficient for the computer to analyze and generate texts in natural language? And what type of knowledge should it be? Up to now, the two foremost approaches in natural language processing have been the statistical approach and the symbolic one. Our results demonstrated that rule-based methods outperformed statistical methods in detecting lexical functions. It means that collocations are analyzed better by rules than by frequency counts; that rules tell us more of what collocations are than frequency counts do; that collocations can be recognized better semantically than statistically.

The fact that the semantic aspect of collocation outweighs the statistical one has an important effect on the definition of collocations. Definition of a concept must contain necessary and sufficient criteria for distinguishing this concept from other concepts. The debate over the most relevant criterion for defining collocations has already lasted over a long period. Should this criterion be statistical or semantic? [Wanner 2004] gives a good concise overview of this debate. The statistical definition of collocation, i.e. based on probabilistic knowledge, says that collocation is the syntagmatic association of lexical items, quantifiable, textually, as the probability that there will occur, at *n* removes (a distance of *n* lexical items) from an item *x*, the items *a*, *b*, *c* ... [Halliday 1961:276]. The semantic definition of collocation explains how the collocational meaning is formed: a collocation is a combination of two words in which the semantics of the base is autonomous from the combination it appears in, and where the collocate adds semantic features to the semantics of the base [Mel'čuk 1995]. For example, in the phrase *She fell to the floor*, all the words are used in their typical sense and the verb *to fall* means *to drop oneself to a lower position*, but when it is said *She fell in love*, we understand that the same verb is not used in its typical, full meaning, but attains a different sense 'begin to experience something'. *WordReference* Online Dictionary[5] gives a description of this sense: pass suddenly

and passively into a state of body or mind. To illustrate the definition, the dictionary provides the following examples: *to fall into a trap*, *She fell ill*, *They fell out of favor*, *to fall in love*, *to fall asleep*, *to fall prey to an imposter*, *fall into a strange way of thinking*. This meaning of *fall* is more abstract as compared with its typical meaning given in [SpWN] 'descend in free fall under the influence of gravity', e.g., *The branch fell from the tree*. *Fall* reveals its characteristic meaning in free word combinations, and its more abstract sense, in collocations. What do we mean by more abstract sense? An abstract sense is not independent, it is not complete, but rather can be called a "semantic particle" whose function is not to express the full semantics, but to add semantic features to the base of collocation.

To explain what is meant by "adding semantic features to the base", let us make an analogy with semantics of grammatical categories which is also very abstract. The verb be in its function as an auxiliary verb does not express any meaning except abstract grammatical categories of time, aspect, and person. In the sentence *This castle was built in the 15th century*, the verb build carries the meaning of an action, and what be does is adding semantic features to the verb, i.e. that this action took place in past, it is passive, not active, and was applied to a single object, because the grammatical number of *be* is singular. Likewise, fall does not express an event, or a state, but to the word denoting an event or state 'adds' the semantic feature 'begin to occur'.

According to the semantic definition of collocation, the latter differs from free word combinations in the way it constructs its semantics. While the semantics of a free word combination is the sum of the meanings of its elements, collocational meaning is formed by adding more abstract semantic features expressed by the collocate to the full meaning of the base.

Our experiments showed that collocations are recognized better using rules, or conceptual knowledge. It means that the basic criterion for distinguishing collocations from free word combinations is semantic, so there is a good evidence and reason to build definition of collocation on the semantic, not statistical, criterion.

Chapter 10. Conclusions

1. Our experiments have shown that verb-noun collocations can be classified according to semantic taxonomy of lexical functions using WEKA learning toolset. We have shown that it is feasible to apply machine learning methods for predicting the meaning of unseen Spanish verb-noun collocations.

2. Verb-noun pairs were represented as sets of hyperonyms for both the verb and the noun. As our experiments have shown, hyperonyms of the Spanish WordNet function sufficiently well as features distinguishing between the meanings we chosen to be predicted by classifiers. Therefore, this representation can be used for the task of automatic extraction of lexical functions. With this we re-confirmed that the set of hyperonyms can be used to describe lexical meaning and discriminate word senses.

3. According to 10-fold cross-validation technique, the best performance was demonstrated by bayes.BayesianLogisticRegression algorithm for detecting the lexical function $Oper_1$ and by SimpleLogistic classifier for detecting the lexical function $ContOper_1$. Both algorithms can be applied for high quality semantic annotation of verb-noun collocations based on the taxonomy of lexical functions.

4. According to evaluation of algorithms on an independent test set, the best performance was shown by Ridor and LWL algorithms for detecting the lexical function $ContOper_1$. These algorithms can be used for high quality annotation of verb0noun collocations with this lexical function.

5. The best f-measure achieved in our experiments is 0.873 using the training set and 10-fold cross-validation technique. This is significantly higher than the previously reported result of 0.740 for F-measure, though the comparison is not fair because we looked for the meaning which is similar to the meaning predicted in [Wanner *et al.* 2006], but not the same one. The highest F-measure achieved in the experiments on an independent test set was only 0.658. This could be explained by the fact that the best ratio between the training set and the test set has not yet been found by us. More experiments on test sets of various sizes are needed.

6. We have tested if the three hypothesis stated in [Wanner *et al.* 2006] were valid and well-grounded. These hypothesis claim that collocations can be recognized: first, by their similarity to the prototypical sample of each collocational type (this strategy is

modeled by the Nearest Neighbor technique); second, by similarity of semantic features of their elements (i.e., base and collocate) to semantic features of elements of the collocations known to belong to a specific LF (this method is modeled by Naïve Bayesian network and a decision tree classification technique based on the ID3-algorithm); and third, by correlation between semantic features of collocational elements (this approach is modeled by Tree-Augmented Network Classification technique). Our research has shown that there machine learning methods other than mentioned in the three hypotheses that can be used for high quality annotation of verb-noun collocations of lexical funciton. To these methods the following algorithms belong: JRip, J48, Prism, PART, SimpleLogistic, Ridor.

References

[Alonso Ramos 2003] Alonso Ramos, M.: Hacia un Diccionario de colocaciones del español y su codificación.' In: M. A. Martí et al. (eds.), *Lexicografía computacional y semántica*. Barcelona: Edicions de l'Universitat de Barcelona, pp. 11—34

[Alonso Ramos *et al*. 2008] Alonso Ramos, M., Rambow O., Wanner L.: Using semantically annotated corpora to build collocation resources. Proceedings of LREC, Marrakesh, Morocco, pp. 1154—1158

[Apresjan 1995] Apresjan, Ju. D.: *Lexical Semantics*. (In Russian). Moscow: Vostochnaya Literatura RAN.

[Apresjan 2004] Apresjan, Ju. D.: About semantic nonemptiness and motivatedness of verbal lexical functions. (In Russian.) *Voprosy jazykoznanija*: pp. 3—18

[Apresjan 2008] Apresjan, Ju. D.: Systematic Lexicography. Translated by Kevin Windle. Oxford University Press US.

[Apresjan *et al*. 2003] Apresjan, Ju. D., Boguslavsky, I. M., Iomdin, L. L., Tsinman L., L.: Lexical Functions as a Tool of ETAP-3. MTT 2003, Paris, June 16-18, 2003.

[ARFF] The University of Waikato Computer Science Department Machine Learning Group, Attribute-Relation File Format, http://www.cs.waikato.ac.nz/~ml/weka/ arff.html, last viewed June 11, 2010

[Benson *et al*. 1997] Benson, M., Benson, E., Ilson R.: The BBI Combinatory Dictionary of English. John Benjamins, Amsterdam

[Castellón *et al*. 1999] Castellón I., Civit, M., Atserias J.: Syntactic Parsing of Unrestricted Spanish Texts. Proceedings First International Conference on Language Resources and Evaluation (LREC'98), Granada, Spain

[Chiu *et al*. 2007] Chiu A., Poupart P., DiMarco C.: Learning Lexical Semantic Relations using Lexical Analogies, Extended Abstract. Publications of the Institute of Cognitive Science

[Civit and Martí 2004] Civit, M., Martí, M.A.: Building Cast3LB: A Spanish Treebank. In: Research on Language and Computation, vol. 2(4), pp. 549—574. Springer, Netherlands

[CEB] Concordancia electrónica de la Biblia Reina Valera 1960 online, http://www.concordancia.bravefire.com/concordancia.php/, last viewed on June 07, 2010

[Cohen 1995] Cohen, W.: Fast effective rule induction. In: 12th International Conference on Machine Learning, pp.115—123

[Diachenko 2006] Diachenko, P.: Lexical functions in learning the lexicon. In: Current Developments in Technology-Assisted Education, Vol. 1, 2006, pp. 538—542.

[DiCE] Diccionario de colocaciones del Español, http://www.dicesp.com/paginas/, last viewed June 08, 2010

[Duda and Hart 1973] Duda, R. O., Hart, P. E.: Pattern Classification and Scene Analysis. John Wiley and Sons, New York

[Firth 1957] Firth, J. R.: Modes of Meaning. In J. R. Firth, Papers in Linguistics 1934–1951 (pp. 190–215). Oxford: Oxford University Press.

[Flavell and Flavell 1959] Flavell, J. H. & Flavell, E. R.: One Determinant of Judged semantic and associative connection between words. Journal of Experimental Psychology, 58(2), 159–165.

[Fontecha 1941] Fontecha, C.: Glosario de voces comentadas en ediciones de textos clásicos. Madrid: CSIC

[Friedman *et al*. 1997] Friedman, N., Geiger D., Goldszmidt, M.: Bayesian network classifiers. Machine Learning. Vol. 29 pp. 131–163

[Hall *et al*. 2009] Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten I. H.: The WEKA Data Mining Software: An Update. SIGKDD Explorations, Volume 11, Issue 1

[Halliday 1961] Halliday, M. A. K.: Categories of the Theory of Grammar. *Word* 17, 241–292.

[Hanks and Pustejovsky 2005] Hanks, P., Pustejovsky, J.: A Pattern Dictionary for Natural Language Processing. Revue Francaise de Langue Appliquée, 10:2

[Hindle 1983] Hindle, D.: Deterministic Parsing of Syntactic Noun-Fluencies. Proceedings 21st Annual Meeting of the Association for Computational Linguistics

[Kilgarriff *et al*. 2004] Kilgarriff, A., Rychly, P., Smrz, P. and Tugwell, D.: The Sketch Engine. In Proceedings of EURALEX. France, Université de Bretagne Sud: pp. 105—116

[Langley *et al*. 1992] Langley, P., Iba, W., Thompson, K.: An Analysis of Bayesian Classifiers. In Proceedings of Tenth National Conference on Artificial Intelligence. AAAI Press, Mentlo Park, CA

[Leacock and Chodorow 1998] Leacock C., Chodorow, M.: Combining Local Context and WordNet Similarity for Word Sense Identification. In: C. Fellbaum, editor, WordNet. An Electronic Lexical Database, pp. 265—283. MIT Press, Cambridge, MA

[Lewis 1994] Lewis, M.: The Lexical Approach. The State of ELT And A Way Forward, Language Teaching Publications

[Lewis 1997] Lewis, M.: Implementing the Lexical Approach: Putting Theory into Practice, Hove: Language Teaching Publications

[LFs] www.Gelbukh.com/lexical-functions, last viewed on June 10, 2010

[Loos 1997] Loos, E. (general editor), Anderson, S., Day Jr., D. H., Jordan, P. C., Wingate J. D. (editors): Glossary of Linguistic Terms. http://www.sil.org/linguistics/GlossaryOfLinguisticTerms/, last viewed on June 08, 2010

[McVey and Wegmann 2001] McVey Gill, M., Wegmann, B.: Streetwise Spanish Dictionary/Thesaurus. Chicago: McGraw-Hill.

[Mel'čuk 1974] Igor' Mel'čuk . 1974. Opyt teorii lingvističeskix modelej "Smysl ↔ Tekst" ['A Theory of the Meaning-Text Type Linguistic Models']. Moskva: "Nauka".

[Mel'čuk 1996] Igor Mel'čuk. Lexical Functions: A Tool for the Description of Lexical Relations in a Lexicon. In Lexical Functions in Lexicography and Natural Language Processing, pages 37-102. L. Wanner, ed. John Benjamin Publishing Company, 1996.

[Mel'čuk and Zholkovskij 1984] I. A. Mel'čuk, A. K. Zholkovskij. Tolkovo-kombinatornyj slovar' sovremennogo russkogo jazyka. [An Explanatory Combinatorial Dictionary of the Contemporary Russian Language] Wiener Slawistischer Almanach, Sonderband 14, 1984.

[Mel'čuk et al. 1984] Igor Mel'čuk, Nadia Arbatchewsky-Jumarie, Lidija Iordanskaja, Adèle Lessard. Dictionnaire explicatif et combinatoire du français contemporain, Recherches lexico-sémantiques I. Les Presses de l'Université de Montréal, 1984.

[Mel'čuk et al. 1988] Mel'čuk, Igor, Nadia Arbatchewsky-Jumarie, Louise Dagenais, Léo Elnitsky, Lidija Iordanskaja, Marie-Noëlle Lefebvre, Suzanne Mantha. 1988. *Dictionnaire explicatif et combinatoire du français contemporain. Recherches lexico-sémantiques II*. Les Presses de l'Université de Montréal.

[Mel'čuk 1993-2000] Igor Mel'čuk Cours de morphologie générale, vol. 1-5, Montréal: Les Presses de l'Université de Montréal/Paris: CNRS Éditions.

[Mel'čuk 2006] Igor Mel'čuk. Aspects of the Theory of Morphology. Berlin - New York: de Gruyter.

[Mel'čuk and Zholkovskij 1984] Mel'čuk, I. A. and Zholkovskij, A. K. 1984. *An Explanatory Combinatorial Dictionary of the Contemporary Russian Language.* Wiener Slawistischer Almanach, Sonderband 14.

[Miller 1998] Miller, G.A: Foreword. In: Fellbaum, C. (ed.) WordNet. An Electronic Lexical Database, pp. xv–xxii. MIT Press, Cambridge, Mass. (1998)

[Nastase and Szpakowicz 2003] V. Nastase and S. Szpakowicz. Exploring noun-modifier semantic relations. In Fifth International Workshop on Computational Semantics (IWCS-5), Tilburg, The Netherlands, pages 285-301, 2003.

[Nastase et al. 2006] Nastase, V., J. Sayyad-Shiarabad,M. Sokolova, and S. Szpakowicz. 2006. Learning noun-modifier semantic relations with corpus-based and wordnet-based features. In Proceedings of the Twenty-First National Conference on Artificial Intelligence and the Eighteenth Innovative Applications of Artificial Intelligence Conference. AAAI Press.

[Oelschläger 1940] Oelschläger, V.R.B. 1940. *A Medieval Spanish Word-List: A Preliminary Dated Vocabulary of First Appearances Up To Berceo*. Madison, Wisc.: University of Wisconsin Press.

[Ogawa et al. 1991] Y. Ogawa, T. Morita and K. Kobayashi. A Fuzzy Document Retrieval System Using the Keyword Connection Matrix and a Learning Method. Fuzzy Sets and System 39:163-179.

[Patwardhan 2006] Siddharth Patwardhan, Ted Petersen. Using WordNet-based Context Vectors to Estimate the Semantic Relatedness of Concepts. In Proceedings of the EACL 2006 Workshop Making Sense of Sense - Bringing Computational Linguistics and Psycholinguistics Together, pages 1–8.

[Porzig 1934] Porzig, W.: Wesenhafte Bedeutungsbeziehungen. Beträge zur Geschichte der deutsche Sprache und Literatur. No. 58.

[RAE 2001] Real Academia Española 2001. *Diccionario de la Lengua Española*. (Twenty Second Edition.) Madrid: Real Academia Española

[Roy 2002] Roy S. Nearest neighbor with generalization. Christchurch, NZ.

[Ruppenhofer *et al*. 2006] Ruppenhofer, J., Ellsworth, M., Petruck, M., Johnson, C. R. & Scheffczyk, J. *FrameNet II: Extended Theory and Practice*. Available at http://framenet.icsi.berkeley.edu/book/book.pdf. ICSI Berkeley

[Sanromán 1998] B. Sanromán. Contribución lexicográfica al estudio de los nombres de emoción. Master's thesis, Universidad de Coruña.

[Sanromán 2003] B. Sanromán. Semántica, sintaxis y combinatoria léxica de los nombres de emoción en español. PhD thesis, Helsinki: University of Helsinki.

[Sebastián 2000] N. Sebastián. LEXESP, léxico informatizado del español. Edicions de la Universidad de Barcelona, Barcelona.

[Shepard 1968] D. Shepard. A Two Dimensional Interpolation Function for Irregularly Spaced Data. Proceedings of the 23rd National Conference of the ACM, ACM Press.

[SbWC] Spanish Web Corpus. 03 May 2010. http://trac.sketchengine.co.uk/wiki/Corpora/SpanishWebCorpus/

[SpWN] Spanish WordNet, http://www.lsi.upc.edu/~nlp/web/index.php?Itemid=57 &id=31& option=com_content&task=view, last viewed June 02, 2010

[Uchida et al. 2006] Uchida, H., Zhu, M., Senta T. D.: *Universal Networking Language*. UNDL Foundation, 2006

[Vossen 1998] P. Vossen. EuroWordNet: A Multilingual Database with Lexical Semantic Networks. Kluwer Academic, Dordrecht.

[Wanner 1996] L. Wanner. 1996. Lexical Functions in Lexicography and Natural Language Processing. John Benjamin Publishing Company.

[Wanner 2004] Leo Wanner. Towards automatic fine-grained semantic classification of verb-noun collocations. Natural Language Engineering (2004), 10:2:95-143 Cambridge University Press.

[Wanner 2006] Wanner, L., Bohnet, B. and Giereth, M. What is beyond Collocations? Insights from Machine Learning Experiments. EURALEX.

[Webster] Merriam-Webster's Online Dictionary, http://www.merriam-webster.com/dictionary/walk%5B2%5D

[Weinreich 1969] Weinreich, U.: Problems in the Analysis of Idioms. In J. Puhvel (Ed.), Substance and Structure of Language (pp. 23–82). CA, Los Angeles: University of California Press.

[WEKA] The University of Waikato Computer Science Department Machine Learning Group, WEKA download, http://www.cs.waikato.ac.nz/~ml/weka/ index_downloading.html, last viewed June 02, 2010

[WM] WEKA Manual for Version 3-6-2, http://iweb.dl.sourceforge.net/project/weka/documentation/3.6.x/WekaManual-3-6-2.pdf

[Witten and Frank, 2005] Witten, I. H., Frank, E. Data Mining: Practical machine learning tools and techniques, 2nd Edition. Morgan Kaufmann, San Francisco.

[Zhong and Tou Ng 2010] Zhong, Z. and Tou Ng, H.: It Makes Sense: A Wide-Coverage Word Sense Disambiguation System for Free Text. In Proceedings of System Demonstrations, 48th Annual Meeting of the Association for Computational Linguistics (pp. 78–83). Sweden, Uppsala: Uppsala University.

Appendices

Appendix 1. Glossary

Relation	Definition 1 - that feature or attribute of things which is involved in considering them in comparison or contrast with each other; the particular way in which one thing is thought of in connexion with another, any connexion, correspondence, or association, which can be conceived as naturally existing between things. (The Shorter Oxford English Dictionary on Historical Principles, 3rd edition, Oxford, At the Clarendon Press, 1959) Definition 2 - an aspect or quality (as resemblance) that connects two or more things or parts as being or belonging or working together or as being of the same kind, **specif**: a property (as one expressing by *is equal to*, *is less than*, or *is the brother of*) that holds between an ordered pair of objects. (Webster's Ninth New Collegiate Dictionary, Merriam-Webster Inc. Publishers, Springfield, Massachusetts, USA, 1991) Definition 3 - a connection between two or more things. (Longman Dictionary of Contemporary English, 3rd edition, Longman Dictionaries, 1995) Definition 4 - the way in which two or more concepts, objects, or people are connected; a thing's effect on or relevance to another. (The New Oxford Dictionary of English, Clarendon Press, Oxford, 1998) Definition 5 - the way in which two or more things are connected. (Oxford Advanced Learner's Dictionary, A.S.Hornby, 6th edition, Oxford University Press, 2000) Definition 6 - an aspect or quality, e.g. resemblance, that connects two or more things and enables them to be considered together in a meaningful way. (The Penguin English Dictionary, 2nd edition, Penguin Books, 2003) Definition 7 - an abstraction belonging to or characteristic of two entities or parts together. (WordNet 2.1)
Wordform	the phonological or orthographic sound or appearance of a word that can be used to describe or identify something; "the inflected forms of a word can be represented by a stem and a list of inflections to be attached" (WordNet 3.0) http://www.thefreedictionary.com
Lexical form	an abstract unit representing a set of wordforms differing only in inflection and not in core meaning. http://www.sil.org/linguistics/GlossaryOfLinguisticTerms

Meaning	a notion in semantics classically defined as having two components: reference, anything in the referential realm denoted by a word or expression, and sense, the system of paradigmatic and syntagmatic relationships between a lexical unit and other lexical units in a language. http://www.sil.org/linguistics/GlossaryOfLinguisticTerms
Lexical unit	a form-meaning composite that represents a lexical form and single meaning of a lexeme. http://www.sil.org/linguistics/GlossaryOfLinguisticTerms
Lexeme	the minimal unit of language which has a semantic interpretation and embodies a distinct cultural concept, it is made up of one or more form-meaning composites called lexical units. http://www.sil.org/linguistics/GlossaryOfLinguisticTerms
Lexical relation	a culturally recognized pattern of association that exists between lexical units in a language. http://www.sil.org/linguistics/GlossaryOfLinguisticTerms
Paradigmatic lexical relation	a culturally determined pattern of association between lexical units that share one or more core semantic components, belong to the same lexical category, fill the same syntactic position in a syntactic construction, and have the same semantic function. http://www.sil.org/linguistics/GlossaryOfLinguisticTerms "In general, paradigmatic relations subsume all contrast and substitution relations that may hold between lexical units in specific contexts. For example, the lexemes school and student are paradigmatically related in such pairs of phrases as to go to school and to be a student, and so also are the lexemes young and tall in young student and tall student. A paradigmatic relation, in general, does not automatically imply a semantic relation." [Wanner 1996]
Semantic component	a potentially contrastive part of the meaning of a lexical unit. E.g., contrastive semantic component distinguishes one lexical unit from another as "male" is the contrastive semantic component distinguishing *man* from *woman*, and *boy* from *girl*; shared semantic component occurs in each member of a group of lexical units as "human" is a shared component for *man*, *woman*, *boy*, and *girl*. http://www.sil.org/linguistics/GlossaryOfLinguisticTerms
Lexical category	a syntactic category for elements that are part of the lexicon of a language. These elements are at the word level. Also known as part of speech, word class, grammatical category, grammatical class. Lexical categories may be defined in terms of core notions or 'prototypes'. Given forms may or may not fit neatly in one of the categories. The category membership of a form can vary according to how that form is used in discourse. http://www.sil.org/linguistics/GlossaryOfLinguisticTerms
Prototype of any category	the member or set of members of a category that best represents the category as a whole. Not everything fits perfectly in a category. Categories are defined by an intersection of properties that make up their members. Members that have all the properties are the prototype members. Those that contain some, but not all, of the properties are less prototypical. http://www.sil.org/linguistics/GlossaryOfLinguisticTerms

Syntax	(from Ancient Greek συν- *syn-*, "together", and τάξις *táxis*, "arrangement") is the study of the principles and rules for constructing sentences in natural languages. In addition to referring to the discipline, the term *syntax* is also used to refer directly to the rules and principles that govern the sentence structure of any individual language, as in "the syntax of Modern Irish." http://en.wikipedia.org/wiki/Syntax
Syntactic category	either a phrasal category, such as noun phrase or verb phrase, which can be decomposed into smaller syntactic categories, or a lexical category, such as noun or verb, which cannot be further decomposed. The three criteris ised in defining syntactic categories are the type of meaning it expresses, the type of affixes it takes, the structure in which it occurs. http://en.wikipedia.org/wiki/Syntactic_category
Grammatical category	person, number, tense, aspect, mood, gender, case, voice...
Grammatical class	transitive and intransitive verbs; count and mass nouns...
Grammatical relations	subject, direct object, indirect object...
Functional categories	agent, patient, instrument...; topic, comment...; definite NP
Syntagmatic lexical relation	a culturally determined pattern of association between pairs of lexical units (A1-B1, A2-B2, A3-B3...) where the two members of each pairs (A1 and B1) have compatible semantic components, are in a fixed syntactic and semantic relationship to each other, and are typically associated with each other, and corresponding members of each pair (A1, A2, A3...) belong to the same lexical category, fill the same syntactic position in a syntactic construction, and have the same semantic function. http://www.sil.org/linguistics/GlossaryOfLinguisticTerms
Syntagmatic relations	(or *co-occurence* relations) hold between lexical units that can appear together, i.e. that co-occur, in the same phrase or clause. [Wanner 1996]
Token	each running word in the text. Thus a text of length a hundred words contains a hundred tokens. [Sinclair et al. 2004]
Lexical semantic relations	semantic relations between concepts [Chiu *et al.* 2007]

Appendix 2. Definitions of Collocation

Additional information is given as to the source of definition, the criterion used to distinguish collocations from free word combinations, and some comments on definitions.

Author	Definition	Criterion for a word combination to be considered a collocation	Comments
[Firth 1957]	Collocations of a given word are statements of the habitual or customary places of that word.	Lexical criterion: a word is used in a fixed position with respect to a given word. Statistical criterion: frequency of word co-occurrence.	[Firth 1957] first introduced the term 'collocation' from Latin *collocatio* which means 'bringing together, grouping'. Firth believes that speakers make 'typical' common lexical choices in collocational combinations. Collocation is a concept in Firth's theory of meaning: "Meaning by collocation is an abstraction at the syntagmatic level and is not directly concerned with the conceptual or idea approach to the meaning of words. One of the meanings of *night* is its collocability with *dark*, and of *dark*, of course, collocation with *night*."

[Halliday 1961]	Collocation is the syntagmatic association of lexical items, quantifiable, textually, as the probability that there will occur, at n removes (a distance of n lexical items) from an item x, the items a, b, c ...	Lexical criterion: a word is used a fixed position with respect to a given word. Statistical criterion: high co-occurrence frequency.	If a lexical item is used in the text, then it's collocate has the highest probability of occurrence at some distance from the lexical item. Collocations cut across grammar boundaries: e.g. *he argued strongly* and *the strength of his argument* are grammatical transformations of the initial collocation *strong argument*.
[Hausmann 1984, 1985]	Collocations are binary word-combinations, consist of words with limited combinatorial capacity, they are semi-finished products of language, affine combinations of striking habitualness. In a collocation one partner determines, another is determined. In other words: collocations have a basis and a co-occurring collocate.	Lexical criterion: the lexical choice of the collocate depends on the basis.	Word combinations are classified word-combinations according to the features fixed vs. non-fixed, and in this classification collocations are belong to the category of non-fixed affine combinations. Internal structure of collocation: collocation components have functions of a basis and a collocate, and the basis (not the speaker) 'decides' what the collocate will be.

[Benson 1985]	Collocation is a group of words that occurs repeatedly, i. e. recurs, in a language. Recurrent phrases can be divided into grammatical collocations and lexical collocations. *Grammatical collocations* consist of a dominant element and a preposition or a grammatical construction: *fond of*, (*we reached*) *an agreement that...* *Lexical collocations* do not have a dominant word, their components are "equal": *to come to an agreement*, *affect deeply*, *weak tea*.	Functional criterion: collocations are classified according to function of collocational elements. Statistical criterion: high co-occurrence frequency.	Broad understanding of collocation. Classification of collocations according to their compositional structure.
[Benson 1989]	Collocations should be defined not just as 'recurrent word combinations', [but as] 'ARBITRARY recurrent word combinations'	Lexical criterion: arbitrariness and recurrency	'Arbitrary' as opposed to 'regular' means that collocations are not predictable and cannot be translated word by word.

[Van Roey 1990]	Collocation is "that linguistic phenomenon whereby a given vocabulary item prefers the company of another item rather than its 'synonyms' because of constraints which are not on the level of syntax or conceptual meaning but on that of usage."	Statistical criterion: high co-occurrence frequency in corpora.	Van Roey summarizes statistical view stated by Halliday in terms of expression or 'usage'. A collocate can thus simply be seen as any word which co-occurs within an arbitrary determined distance or *span* of a central word or *node* at the frequency level at which the researcher can say that the co-occurrence is not accidental. This approach is also textual in that it relies solely on the ability of the computer program to analyze large amounts of computer-readable texts.
[Cowie 1993]	Collocations are associations of two or more lexemes (or roots) recognized in and defined by their occurrence in a specific range of grammatical constructions.	Structural criterion: collocations are distinguished by patterns	Collocations are classified into types according to their grammatical patterns.

[Howarth 1996]
 	
[Howarth 1996] contd.
	In his lexical continuum model, collocations as composite units are placed on a sliding scale of meaning and form from relatively unrestricted (collocations) to highly fixed (idioms). <u>Restrictive</u> collocations are fully institutionalised phrases, memorized as wholes and used as conventional form-meaning pairings.	Syntactic criterion: commutability – the extent to which the elements in the expression can be replaced or moved (make/reach/take decision vs. shrug one's shoulders).
		
		Semantic criterion: motivation – the extent to which the semantic origin of the expression is identifiable (move the goalposts = to change conditions for success vs. shoot the breeze = to chatter, which is an opaque idiom).
			Classification includes 4 types of expressions with no reference to frequency of occurrence:
			    free collocation: *blow a trumpet* = to play a trumpet,
			    restrictive collocation: *blow a fuse* = to destroy a fuse/to get angry,
			    figurative idiom*: blow your own trumpet* = to sell oneself excessively,
			    pure idiom: *blow the gaff* = to reveal a concealed truth.
			
			The problem with this classification is that is difficult to determine what is meant by 'syntactically fixed', 'unmotivated' or 'opaque'. This is seen in the ambiguous example of *to blow a fuse*.

[Sinclair et al. 2004]	Collocation is the co-occurrence of two items in a text within a specified environment. Significant collocation is regular collocation between two items, such that they co-occur more often than their respective frequencies. Casual collocations are "non-significant" collocations.	Lexical criterion: recurrency of co-occurrence Statistical criterion: high co-occurrence frequency	The degree of significance for an association between items is determined by such statistic tests as Fischer's Exact Test or Poisson Test.

| [Mel'čuk 1995] | Collocation is a combination of two lexical items in which the semantics of one of the lexical items (the *base*) is autonomous from the combination it appears in, and where the other lexical item (the *collocate*) adds semantic features to the semantics of the base. [Gledhill 2000] explains that for Mel'čuk a collocation is a semantic function operating between two or more words in which one of the words keeps its 'normal' meaning. | Semantic criterion: the meaning of a collocation is not inferred from the meaning of the base combined with meaning of the collocate. | Semantics of a collocation is not the meaning of the base + the meaning of the collocate, but rather the meaning of the base + some additional meaning that are included in the meaning of the base. '...the concept of collocation is independent of grammatical categories: the relationship which holds between the verb *argue* and the adverb *strongly* is the same as that holding between the noun *argument* and the adjective *strong*.' [Fontenelle 1994] |

Appendix 3. Syntagmatic Verb-Noun Lexical Functions

Lexical function name and description	Lexical function variant	Examples in English		Examples in Spanish	
		LF Argument	Collocation: LF value + LF argument	LF Argument	Collocation: LF value + LF argument
Oper$_i$ Lat. *operari* 'do, carry out, perform, experience'. The keyword of Oper$_i$ is the name of an action, an activity, a state, a property, a relation, i.e. a noun whose meaning is or includes a predicate in the logical sense of the term this presupposing actants.	**Oper$_1$** 'perform, do, act'	support resistance order	to lend support to put up resistance to give order	apoyo resistencia orden	dar apoyo oponer resistencia dar la orden
	Oper$_2$ 'undergo, meet'	support resistance order	to receive support to meet resistance to receive order	apoyo resistencia orden	recibir apoyo encontrar resistencia recibir la orden
Func$_i$ Lat. *functionare* 'function'. The keyword of Func$_i$ is the name of an action, an activity, a state, a property, a relation, i.e. a noun whose meaning is or includes a predicate in the logical sense of the term this presupposing actants.	**Func$_0$** 'happen, take place'	snow silence smell	falls reigns lingers	vieto silencio accidente	el viento sopla el silencio reina el accidente ocurre
	Func$_1$ 'originate from'	blow proposal support	comes [from N] comes, stems [from N] come [from]	golpe propuesta apoyo	un golpe se produce una propuesta se presenta apoyo se presta

	Func₂ 'concern, apply to'	blow proposal analysis	falls [upon N] concerns [N] concerns	golpe propuesta analisis	el golpe cae la propuesta consiste en N analisis explica
Labor$_{ij(k)}$ Lat. *laborari* 'to work, toil, process' – a verb connecting L and participant(s).	**Labor$_{12}$** a verb connecting the first participant as grammatical subject with the second participant as direct object and with L as indirect object	control respect punishment	to keep N under control to treat N with respect to subject N to punishment	alegría cariño	celebrar algo con alegría tratar algo con cariño
Incep Lat. *incipere* 'begin' – a phrasal verb	**IncepOper₁**	fire [shoot] popularity despair	to open fire on N to acquire popularity to sink into despair	admiración amistad cariño	contagiarse de admiración contraer la amistad cobrar cariño a N
	IncepOper₂	power control	fall under the power of N to get under N's control		
	IncepFunc₁ 'N begins to be experienced'	despair hatred anger	despair creeps over/in N hatred stirs up N anger arises	desesperación odio ira	desesperación entra en N odio se apodera de N ira envade N
Cont Lat. *continuare* 'continue' – a phrasal verb	**ContOper₁** 'continue to experience'	enthusiasm hope anger	maintain enthusiasm hope burns anger boiled over in N	entusiasmo esperanza odio	guardar entusiasmo guardar esperanza conservar odio

	ContOper$_2$	attention	to hold attention		
	ContOper$_{12}$	animosity	feel animosity towards/against N	enemistad	mantener el enemistad
	ContFunc$_0$	offer odor	the offer stands the odor lingers		
	ContFact$_0$	luck	her luck holds		
Fin Lat. *finire* 'cease' – a phrasal verb	**FinOper$_1$**	power patience	to lose one's power over N to lose patience	alegría amistad cariño	perder la alegría perder la amistad perder cariño
	FintOper$_2$ 'cease to be the object of somebody's L'	credit	to lose credit with N	admiración	perder la admiración de N
	FinFunc$_0$ 'N ceases to be experienced'	anger hatred enthusiasm	anger defuses hatred ceases enthusiasm wanes	aprención odio entusiasmo	aprención se disipa odio desaparece entusiasmo se desvanece
	FinFunc$_1$	love	his love vanished into thin air	admiración	la admiración ha desaparecido en N
Caus Lat. *causare* 'cause' 'do something so that a situation begins occurring'	**CausOper$_1$**	opinion despair	to lead N to the opinion to throw N into despair	admiración	llenar a N de admiración
	CausOper$_2$	control attention	to put N under X's control to call N to X's attention		
	Caus$_1$Oper$_2$	control	to bring N under one's control		

	CausFunc$_0$ 'cause the existence of N'	crisis difficulty election	to bring about the crisis to create a difficulty to hold elections	alarma elecciones crisis	dar la alarma celebrar elecciones provocar una crisis
	Caus$_1$Func$_2$	suspicion attention	to sow suspicions to show attention to N	cariño sospecha	depositar el cariño en N apuntar la sospecha hacia N
	Caus$_2$Func$_1$ 'cause N to be experienced'	hope surprise anger	to raise the hope in N to give surprise to provoke anger	horror sorpresa odio	causar horror dar sorpresa despertar odio
	Caus$_2$Func$_2$	attention friendship	to grab N's attention to seek frienship	admiración amistad cariño	atraer admiración concitar la amistad atraerse el cariño de N
	CausReal$_1$	suspicion	to fall under suspicion	sospecha	corroborar la sospecha
	Caus$_1$Manif	admiration joy friendship	to produce admiration to show joy to enjoy friendship	admiración alegría amistad	confesar admiración exteriorizar la alegría demostrar amistad
	CausDegrad	joy fire strength	joy was vanishing the fire is dying down my strength is failing	alegría	empañar la alegría
Perm Lat. *permittere* 'permit, allow' 'do nothing which would cause that a situation stops occurring'	**PermFunc$_1$**			gana	dejar a N con las ganas
	Perm$_1$Fact$_0$	anger desire	to let go N's anger to give in to the desire	alegría	dejarse llevar por la alegría
	nonPerm$_1$Fact	laugh impulse tear	to suppress a laugh to check an impulse to hold back a tear	alegría dolor gana	contener la alegría contener el dolor reprimir las ganas

	Perm₁Manif	strength impatience tact	to display N's strength to exhibit impatience to show tact	sospecha	dar rienda suelta a la sospecha
	nonPerm₁Manif	smile hatred laughter	to conceal a smile to hide N's hatred to stifle N's laughter	admiración alegría celos	ocultar la admiración disimular la alegría reprimir los celos
Liqu Lat. *liquidare* 'liquidate' 'do something so that a situation stops occurring'	**LiquOper₂**	liability debt duty	to exempt from liability to release from debts to release N from N's duties	sospecha	alejar a N de toda sospecha
	Liqu₁Func₀ 'put an end to'	support obstacle meeting	withdraw support remove the obstacle end the meeting	alegría amistad celos	apagar la alegría romper la amistad atajar los celos
	LiquFunc₁	custom shyness	the custom is vanishing to get better of N's shyness	alegría celos gana	minar la alegría a N quitar los celos a N quitar las ganas a N
	LiquFunc₂	attention	to divert N's attention from N	sospecha	alejar la sospecha de N
Realᵢ Lat. *realis* 'real'. The gloss is 'to fulfill the requirement of L', 'to do with L what you are supposed to with L', or 'L fulfils its requirement'. The values are fulfillment verbs, differs from **Factᵢ** and **Labrealᵢⱼ** with respect	**Real₁** 'use L according to its destination' 'do with regard to X that which is normally expected of first participant'	duty obligation principle	do one's duty fulfill the obligation follow a principle	amistad cariño celos	conceder amistad a alguien dar cariño consumirse en los celos

	Real₂	challenge	accept a challenge	cariño	recibir cariño de
	Real$_2$ 'do with regard to X that which is normally expected of second participant'	challenge examination insult	accept a challenge pass an examination avenge an insult	cariño examen ofensa	recibir cariño de alguien aprobar el examen vengar la ofensa
Fact$_i$ Lat. *factum* 'fact'. The gloss is 'to fulfill the requirement of L', 'to do with L what you are supposed to with L', or 'L fulfils its requirement'. The values are fulfillment verbs, differs from **Real$_i$** and **Labreal$_{ij}$** with respect to its syntax only.	**Fact$_0$**	hope movie suspicion	his hope comes true the movie is on the suspicion is confirmed		
	Fact$_1$	suspicion hope	arouse suspicion stir up hope	alegría celos dolor	la alegría domina a algien los celos me abrasan el dolor le punzó
	Fact$_2$	suspicion hope	fall under suspicion cherish hope	cariño	el cariño rodea a alguien
Labreal$_{ij}$ It is a hybrid of **Labor** and **Real**. The gloss is 'to fulfill the requirement of L', 'to do with L what you are supposed to with L', or 'L fulfils its requirement'. The values are fulfillment verbs, differs from **Fact$_i$** and **Real$_i$** with respect to its syntax only.	**Labreal$_{12}$**	gallows saw reserve	to string up N on the gallows to cut N with the saw to hold N in reserve	horca sierra cariño	ejecutar en la horca a N la sierra corta N rodear a N de cariño
	Labreal$_{13}$	shame health	to burn with shame to waste N's health	celos	consumir a N de celos
Involv Lat. *involvere* 'drag along' – a verb meaning 'to involve Y', 'to concern Y'; it links L and the name of a non-participant Y which is affected or acted upon by the situation L'		light snowstorm smell	light floods the room the snowstorm caught him in N=place the smell filled the room		

Manif Lat. *manifestare* 'manifest' – verb meaning 'L manifests itself or becomes apparent in Y'		amazement joy scorn	amazement lurks in his eyes joy explodes in them scorn is dripping from every word	dolor enemistad orgullo	el dolor se refleja la enemistad se manifesta el orgullo resplandece
Prox Lat. *proximare* 'approach' – verb meaning 'to be about to do something, to be on the verge of something'	ProxOper_1	despair disaster tears	to be on the edge of despair to be on the brink of disaster to be on the verge of tears		
	ProxFunc_0	thunderstorm	thunderstorm brews		
Prepar Lat. praeparare 'prepare' – verb meaning 'to prepare N for ..., to get N ready for normal use or functioning'	$\text{Prepar}_1\text{Real}_1$	friendship	to propose friendship to N	amistad	ofrecer amistad a N
	$\text{Prepar}_1\text{Real}_2$	plane	to board a plane		
	PreparFact_0	car program table	to fill up the car to load a program into a computer to lay the table		
Degrad Lat. *degradare* 'lower, degrade' – verb meaning 'to degrade, to become permanently worse or bad'		milk clothes teeth	milk goes sour clothes wear off teeth decay	alegría ropa dientes	la alegría se frustra
Son Lat. *sonare* 'sound' – verb meaning 'to emit characteristic sound'		dog banknotes waterfall	the dog barks banknotes rustle the waterfall roars	perro billetes cascada	el perro ladra

Obstr Lat. *obstruere* 'obstruct' – verb meaning 'to function with difficulty'; alphabetical superscripts specify the aspect of obstruction.	**Obstr**	eyes negotiations economy	eyes blur negotiations are stalled economy stagnates		
	Obstr$_2$	breath speech	N is short of breath N stutters, stammers, mumbles		
	Obstr^{stat} 'stat' = 'with respect to vertical position'	body knees	the body crumples his knees give way		
Stop Lat. *stuppare* 'stop up, plug' – verb meaning 'to stop functioning'	**Stop**	voice heart1 heart2	his voice breaks his heart is stopping her heart broke		
	Stop$_2$	breath	N loses his breath		
Excess Lat. *excessus* (past participle of *excédere*) 'exceed' – verb meaning 'to function in an abnormally excessive way'; alphabetical superscripts specify the aspect of excessive functioning.	**Excess**	engine sweat	the engine races sweat rolls down across N's forehead		
	Excess$_2$	heart1	N has palpitations		
	Excess^{motor} 'motor' = 'with respect to movements'	eyes heart1	the eyes pop out on stalks the heart pounds, races		
	Excess$_2^{motor}$	teeth sweat	N grinds his teeth N is bathed in sweat		
	Excess^{color} 'color' = 'with respect to color'	cheeks	cheeks glow		
	Excess$_1^{color}$	cheeks	to be red-cheeked		

	Excess[dim] 'dim' = 'with respect to dimension/size'	eyes	the eyes are like saucers in X's head		
	Excess[fulg] 'fulg' = 'with respect to brightness'	eyes	eyes flash/glitter		
	Excess[trem] 'dim' = 'with respect to dimension/size'	hands	his hand were shaking		
	Excess[t0] 't^0' = 'with respect to temperature'	cheeks	her cheeks burnt		
Sympt Lat. *symptoma* 'a symptom of' – a verbal expression meaning 'symptom of', it denotes a bodily reaction that is a symptom of an emotional or physical state.	**Obstr(1)-Sympt$_{23}$(2)**	1=breath, 2=anger 1=speech, 2=anger	N chokes with anger N sputters with anger	1=respiración, 2=cólera	sofocarse de cólera
	Stop(1)-Sympt$_1$(2)	1=speech, 2=amazement	//N is dumbstruck	1=habla, 2=susto	enmudecer del susto, el susto le hizo enmudecer
	Excess[motor]**(1)-Sympt$_1$(2)**	1=hair, 2=fear 1=eyes, 2=amazement	N's hair stands on end N's eyes start from their sockets	1=pelo, 2=susto 1=ojos, 2=espanto	ponérsele a uno los pelos de punta estar con los ojos fuera de las órbitas
	Excess[motor]**(1)-Sympt$_{213}$(2)**	1=mouth, 2=amazement	N opens N's mouth wide with amazement	1=cuerpo, 2=dolor 1=pecho, 2=orgullo	doblarse del dolor henchir el pecho

	Excess^{motor}(1)-Sympt$_{13}$(2)	1=mouth, 2=astonishment 1=mouth, 2=surprise	N's jaw drops in astonishment the mouth hangs open in surprise	1=corazon, 2=alegría 1=palmas, 2=alegría	el corazon da un vuelco de alegría dar palmas de alegría
Anti Lat. *antonymum* – antonym, i.e. an LF returning for L a lexical unit L´ such that the meanings 'L' and '' differ by a negation of an internal element of one of them.	**AntiReal$_1$**	cancer	to win over N's cancer	dolor sospecha	aguantar el dolor desmentir la sospecha
	AntiReal$_2$	examination piece of advice application	to fail an examination to reject a piece of advice to turn down an application	examen consejo solicitud	reprobar el examen desoír el consejo rechazar la solicitud
Result Lat. *resultare* 'result' – a verb meaning 'to be the expected result of'.	**ResultOper$_3$**	proposal	to have the proposal		

Appendix 4. Program Source Code (Perl)

Step 1. Reading the original data Excel file containing the Corpus of Spanish Verb-Noun Lexical Functions.

```perl
# PROGRAM DESCRIPTION
# Reads the Excel file containing the Corpus of Sanish Verb-Noun
# Lexical Functions and outputs a text file where each line looks
like this:
# v_formar 2 n_parte 1 CausFunc0, where v stands for verb,
# n stands for noun, the numbers which follow the words formar and
parte
# are numbers of their senses in the Spanish WordNet

#!/usr/bin/perl -w
use strict;
use 5.010;
use Win32::OLE qw(in with);
use Win32::OLE::Const 'Microsoft Excel';

$Win32::OLE::Warn = 3; # die on errors...

# open new Excel application
my $Excel = Win32::OLE->GetActiveObject('Excel.Application')
    || Win32::OLE->new('Excel.Application', 'Quit');

# open Excel file
my $Book = $Excel->Workbooks->Open
("C:/Lexical    Resources/Corpus    of    Spanish    Verb-Noun    Lexical
Functions.xls");

# select worksheet number 1 (you can also select a worksheet by
name)
my $Sheet = $Book->Worksheets($ARGV[0]);

warn "Processing $ARGV[0]\n";

my %POS = (
        VERB => "v_",
        NOUN => "n_",
        ADJECTIVE => "a_",
        ADVERB => "r_"
);
my %prefix = (
        2 => ($POS{$Sheet->Cells(1,2)->{'Value'}} or die),
        4 => ($POS{$Sheet->Cells(1,4)->{'Value'}} or die)
);
for my $row (2..1001)
{
        for my $col (2..5, 1)
        {
                # skip empty cells
                die "Empty cell: (row, col) = ($row, $col)" unless
defined $Sheet->Cells($row,$col)->{'Value'};
                # print out the contents of a cell
                print  +($prefix  {$col}  //   ""),   $Sheet-
>Cells($row,$col)->{'Value'}, " ";
        }
  printf "\n";
```

```
    }
    $Book->Close;
```

Step 2. Mark verb-noun combinations as positive or negative examples for the lexical function chosen for the classification procedure.

```perl
    # PROGRAM DESCRIPTION
    # Reads the text file generated at Step 1 and outputs a text file
where
    # each line looks like this:
    # v_formar 2 n_parte 1 no % CausFunc0
    # where v stands for verb, # n stands for noun, the numbers which
follow
    # the words formar and parte are numbers of their senses in the
    Spanish
    # WordNet and no means that this is a negative example for the
    lexical
    # function chosen for the classification procedure (which is
    definitely
    # not CausFunc$_0$).

    #!perl -w
    use strict;
    use 5.010;
    my $f = $ARGV [0];      # the name of the LF chosen
                            # for the classification procedure

    while (<STDIN>)
    {
            chomp;
            next if /^ *$/;
            next if /N\/A/;
            next if /ERROR/;
            die unless my ($line, $func) = /(.*) ([^ ]+) *$/;
            say $line, " ", ($func =~ /^$f/) ? "yes" : "no", " % ",
$func;
    }
```

Step 3. Prepare a wordlist.

```perl
    # PROGRAM DESCIPTION
    # Reads the text file generated at Step 1 and outputs a text file
    # where lines look like these:
    # v formar 2
    # n parte 1
    # and where v stands for verb, n stands for noun and the numbers are
those
    # of the respective senses of words in the Spanish WordNet

    #!perl -w
    use strict;
    use 5.010;

    while (<>)
    {
            chomp;
            next if /^ *$/;
            next if /N\/A/;
```

```perl
        next if /ERROR/;
        die unless my ($pos1, $word1, $sense1, $pos2, $word2,
$sense2) = /^([^ ]+)_([^ ]+) +([0-9]+) +([^ ]+)_([^ ]+) +([0-9]+)/;
        # next unless my ($pos1, $word1, $sense1, $pos2, $word2,
$sense2) = /^([^ ]+)_([^ ]+) +([0-9]+) +([^ ]+)_([^ ]+) +([0-9]+)/;

        say "$pos1 $word1 $sense1";
        say "$pos2 $word2 $sense2";
    }
```

Step 4. Find hyperonyms in the Spanish WordNet.

```perl
    # PROGRAM DESCRIPTION
    # Reads the text file generated at Step 3 and the Spanish WordNet
    # and outputs a text file where each line looks like these:
    # v_formar_2|v01787769|v01788486
    # n_parte_1|n00013018|n00018916|n09945970
    # where v stands for verb, n stands for noun, the numbers which
    follow
    # the words formar and parte are numbers of their senses in the
    Spanish
    # WordNet, codes like v01787769 are hyperonym synset IDs in the
    Spanish
    # WordNet and | is a separator, so for each word all hyperonyms
    # are extracted from the SpanishWordNet

    #!perl -w
    use strict;
    use 5.010;

    my $wndir = $ARGV[0];
    my %hypernym;
    open F, "$wndir/esWN-200611-relation"; # a file of the Spanish
WordNet

    while (<F>)
    {
        # has_hyponym|n|06125829|n|50005897|99
        push @{$hypernym {"$3$4"}}, "$1$2"
        if /^has_hyponym\|(\w)\|(\d+)\|(\w)\|(\d+)\|/;
    }
    close F;
    my %synset;

    open F, "$wndir/esWN-200611-variant"; # a file of the Spanish
WordNet

    while (<F>)
    {
        # v|00005575|parpadear_ligeramente|1|99|-
        /^(\w)\|(\d+)\|([^|]*)\|(\d+)\|/ or die;
        my $word  = "$1_$3_$4";
        my $synset = "$1$2";
        given ("$word|$synset")
        {
            when ("n_pena_1|n10385041")  { $word = "n_pena_13" }
        }
        push @{$synset {$word}}, $synset;
    }
    close F;
```

```
while (<STDIN>)
{
        chomp;
        s/ /_/g;
        my %result;
        if (@{$synset{$_} // []} != 1)
        {
                warn "No synset or multiple synsets for:
                     $_; skipping this word\n";
                next;
        }
                my @ToDo = ($synset{$_} [0]); # only one synset;
                                             # this has been checked
                                      above
        while (my $current = pop @ToDo)
        {
                undef $result {$current};
                push @ToDo, @{$hypernym {$current} // []};
        }
        say "$_|", join '|', sort keys %result;
}
```

Step 5. Compile an attribute list for the ARFF file.

```
# PROGRAM DESCRIPTION
# Reads a particular Spanish WordNet file, the text file generated
at Step 4
# and outputs lines like these (here the beginning of the file is
given):
#
# @relation function
#
# @attribute n00001740 {0,1}
# @attribute n00002086 {0,1}
# @attribute n00003731 {0,1}
# In this file the number of lines is equal to the overall number of
all
# hyperonyms (which are included in the text file generated at Step
4.
# Codes like v01787769 are hyperonym synset IDs in the Spanish
# WordNet and {0,1} are boolean values of the respective attribute

#!perl -w
use strict;
use 5.010;

say "\@relation function";
say "";
my %h;
while (<>)
{
        chomp;
        my @a = split '\|';
        shift @a;
        undef @h {@a};

}

say "\@attribute $_ {0,1}" for sort keys %h;
say "\@attribute category {yes,no}";
say "";
```

```
    say "\@data";
```

Step 6. Add commentaries to the attribute list for the ARFF file.

```perl
  # PROGRAM DESCRIPTION
  # Reads a particular Spanish WordNet file, the text file generated
at
  # Step 5 and outputs lines like these:
  #
  # @attribute n09898220 {0,1} % 10_1 diez_1 decena_1
  # @attribute n09793320 {0,1} % 12.7_Kg_1 aprox._arroba_1
  # @attribute n10882271 {0,1} % 15_de_agosto_de_1945_1 dÝa_VJ_1
  # @attribute n10899730 {0,1} % 1_de_enero_1 a±o_nuevo_1
  #
  # where % is the symbol of a commentary, and commentaries are words
  #  included  in  the  respective  synset  indicated  by  its  ID  like
n09898220

  #!perl -w
  use strict;
  use 5.010;

  my %h;

  while (<>)
  {
         chomp;
         next  unless  /^([nv])\|([0-9]+)\|(.*?)\|([0-9]+)\|/;  #  $1  -
$4
         $h {$1.$2} .= " $3_$4";
  }

  say "\@attribute  $_  {0,1}  \%$h{$_}"  for  sort  {  lc  $h{$a}  cmp  lc
$h{$b} } keys %h;
```

Step 7. Compile the ARFF data file – Stage 1.

```perl
  # PROGRAM DESCRIPTION
  # Reads the text files generated at Step 5 and Step 6
  # Outputs the first part of the ARFF file

  #!perl -w
  use strict;
  use 5.010;

  open FULL,    $ARGV[0]; # the file generated at Step 6
  open INITIAL, $ARGV[1]; # the file generated at Step 5

  my @full = <FULL>;

  while (<INITIAL>)
  {
         chomp;
         my %h;
         unless (/^$/)
         {
                 for my $full (@full)
                 {
                         unless (index $full, $_) # found
                         {
```

```
                                        print $full;
                                        die "Double match for $_" if exists
$h{$_};
                                        undef $h{$_};
                        }
                }
        }

        say unless keys %h;
  }
```

Step 8. Compile the ARFF data file – Stage 2 (final).

```perl
  # PROGRAM DESCRIPTION
  # Reads the text files generated at Step 2, Step 4 and Step 6
  # Outputs the final version of the ARFF file

  #!perl -w
  use strict;
  use 5.010;

  $| = 1;  # do not use buffer for output

  my $cnt_of_attributes;
  my %n_by_attribute;

  open F, "< $ARGV[0]" or die $!; # the file generated at Step 6

  while (<F>)
  {
          next unless /^\@attribute (.*?) {/; # $1
          $n_by_attribute {$1} = $cnt_of_attributes++;

  }

  $cnt_of_attributes--;
  close F;

  warn "$cnt_of_attributes attributes except the category\n";

  my %hypernyms_by_word; # construct a database
  open F, "< $ARGV[1]" or die $!; # the file generated at Step 4

  while (<F>)
  {
          chomp;
          next if /^$/;
          die unless /(.*?)\|(.*)/;
          $hypernyms_by_word {$1} = $2;
  }

  while (<STDIN>) # the file generated at Step 2
  {
          chomp;
          #  my  ($w1,  $w2,  $category)  =  ("v_dar_9",  "v_tomar_4",
"yes");
          die unless /^(\S+) +(\d+) +(\S+) +(\d+) +(\S+)( +% .*)?$/;
          my $w1       = "$1_$2";
          my $w2       = "$3_$4";
          my $category = $5;
          my @a = (0) x $cnt_of_attributes;
```

```perl
            $a [$n_by_attribute {$_}] = 1 for split '\|',
            ($hypernyms_by_word {$w1} . '|' . $hypernyms_by_word {$w2});

            push @a, $category;
            say +(join ',', @a), " % $w1 $w2";
    }
```

Step 8. Read WEKA classifiers output.

```perl
# PROGRAM DESCRIPTION
# search for the line: ----------------- bayes.AODE ----------------
# and print "bayes.AODE" (its the name of one of the classifiers)
# search for the 15th line of numbers after the line
# === Stratified cross-validation ===
# and print this line


#!perl -w
use strict;
use 5.010;

my $line;
my $nextline;
my $counter;



while (my $line = <>)
{
    if ($line =~ /^-----------------\s(\w+\.\w+)/i)
    {
       print $1;
       print "\n";
    }

    if ($line =~ /^=== Stratified cross-validation ===/)

    {
        for ($counter = 1; $counter < 16; $counter++)

        {
            $nextline = <>;
        }
      my $numbers = chomp ($nextline);
      print "$nextline", "\n";
    }
}
```

Chapter 11. Future Work

We plan to do the following:

1. To experiment with different ratios of training and test sets of Spanish verb-non collocations.

1. To evaluate the performance of machine learning algorithms for more lexical functions.

2. To analyze errors of classifiers.

3. To test other classification techniques which were not examined in our experiments.

4. To study the effect of other features, such as WordNet glosses.

5. To experiment with a word space models representing various similarity measures between collocations.

6. To experiment with context-base representation of data. Context can be represented in the form of grammatical relations between words.

7. To experiment with other association measures to estimate distance between verb-noun collocations which belong to different lexical functions.

8. To experiment on English verb-noun collocations.

Author's Publications

1. Olga Kolesnikova, Alexander Gelbukh. Semantic relations between collocations—A Spanish case study. Revista signos (JCR), ISSN 0035-0451, Vol. 45, No. 78, March, 2012, to appear.

2. Alexander Gelbukh and Olga Kolesnikova. Supervised Learning for Semantic Classification of Spanish Collocations. Lecture Notes in Computer Science N 6256, ISSN 0302-9743, Springer, 2010, pp. 362–371

3. Olga Kolesnikova and Alexander Gelbukh. Supervised Machine Learning for Predicting the Meaning of Verb-Noun Combinations in Spanish. Lecture Notes in Artificial Intelligence N 6438, ISSN 0302-9743, Springer, pp. 196-207. **Best paper award (1st place)** at MICAI-2010, the Mexican International Conference on Artificial Intelligence, received over 300 submissions from 34 countries.

4. Olga Kolesnikova and Alexander Gelbukh. Using WEKA for Semantic Classification of Spanish Verb-Noun Collocations. J. Research in Computing Science, ISSN 1870-4069, 2010, to appear.

5. Alexander Gelbukh and Olga Kolesnikova (2011). Multiword Expressions in NLP: General Survey and a Special Case of Verb-Noun Constructions. In: Sivaji Bandyopdahyay, Sudip Kumar Naskar, Asif Ekbal (Eds.). Emerging Applications of Natural Language Processing: Concepts and New Research, IGI-Global Publishing House. In the process of reviewing.

6. Olga Kolesnikova and Alexander Gelbukh (2011). Semantic Annotation of Spanish Verb-Noun Collocations. Computación y Sistemas (indexada por CONACYT). In the process of reviewing.

In Conferences and Workshops

7. Poster "Automatic Extraction of Lexical Functions", the 5th Workshop on Human Language Technologies, October 3, 2008, Sta. María Tonantzintla, Puebla, Instituto Nacional de Astrofísica, Óptica y Electrónica.

8. Talk "Automatic Extraction of Lexical Functions", the 4th Colloqium on Computational Linguistics, UNAM, September 1, 2009.

9. Poster "Automatic Extraction of Lexical Functions", the 6th Workshop on Human Language Technologies, October 30, 2009, Sta. Matía Tonantzintla, Puebla, Instituto Nacional de Astrofísica, Óptica y Electrónica.

10. Talk "Using WEKA for Semantic Classification of Spanish Verb-Noun Collocations", the 11th Conference on Computation CORE-2010, May 26-28, 2010, Centro de Investigación en Computación del Instituto Politécnico Nacional (CIC-IPN).

11. Poster "Corpus de las Funciones Léxicas", the 7th Workshop on Human Language Technologies, October 21-22, 2010, Sta. María Tonantzintla, Puebla, Instituto Nacional de Astrofísica, Óptica y Electrónica.

Revision of Research Articles

1. Additional reviewer, the 9th Conference on Computation CORE-2008, May 28-30, 2008, CIC-IPN.

2. Additional reviewer, the 10th Conference on Computation CORE-2009, May 27-29, 2009, CIC-IPN.

3. Additional reviewer, Journal "Procesamiento del Lenguaje Natural", Revista nº 44, March 2010.

4. Reviewer, the 13th Internation Conference "Text, Speech and Dialogue", Brno, Czech Republic, September 6-10, 2010

5. Additional reviewer, "International Journal of Computational Linguistics and Applications", Vol. 1, No. 1-2, January-December 2010.

Awards

Best paper award (1st place) at MICAI-2010, the Mexican International Conference on Artificial Intelligence, received over 300 submissions from 34 countries.

Published:

Olga Kolesnikova and Alexander Gelbukh. Supervised Machine Learning for Predicting the Meaning of Verb-Noun Combinations in Spanish. Lecture Notes in Artificial Intelligence N 6438, ISSN 0302-9743, Springer, pp. 196-207.