uOttawa

L'Université canadienne
Canada's university

FACULTÉ DES ÉTUDES SUPÉRIEURES ET POSTOCTORALES

uOttawa

L'Université canadienne
Canada's university

FACULTY OF GRADUATE AND POSDOCTORAL STUDIES

Maria Fernanda Caropreso

AUTEUR DE LA THÈSE / AUTHOR OF THESIS

Ph.D. (Computer Science)

GRADE / DEGREE

School of Information Technology and Engineering

FACULTÉ, ÉCOLE, DÉPARTEMENT / FACULTY, SCHOOL, DEPARTMENT

Syntactically-Informed Representation for Sentence Selection

TITRE DE LA THÈSE / TITLE OF THESIS

Stan Matwin

DIRECTEUR (DIRECTRICE) DE LA THÈSE / THESIS SUPERVISOR

CO-DIRECTEUR (CO-DIRECTRICE) DE LA THÈSE / THESIS CO-SUPERVISOR

EXAMINATEURS (EXAMINATRICES) DE LA THÈSE / THESIS EXAMINERS

Igor Jurisica

Nathalie Japkowicz

Diane Inkpen

Franz Oppacher

Gary W. Slater

Le Doyen de la Faculté des études supérieures et postdoctorales / Dean of the Faculty of Graduate and Postdoctoral Studies

Syntactically-Informed Representation for Sentence Selection

Maria Fernanda Caropreso

Thesis submitted to the

Faculty of Graduate and Postdoctoral Studies

In partial fulfillment of the requirements

For the PhD degree in Computer Science

School of Information Technology and Engineering

Faculty of Engineering

University of Ottawa

Your file Votre référence
ISBN: 978-0-494-50720-9
Our file Notre référence
ISBN: 978-0-494-50720-9

Acknowledgements

My greatest thanks go to my supervisor Stan Matwin, who believed in me from the beginning and guided me through my research. His enlightenment and advice enriched not only this dissertation but also my development as a researcher. I would also like to thank the members of my dissertation committee. Their suggestions at different stages of my research contributed to a more complete work.

Other faculty, researchers, students and staff at the School of Information Technology and Engineering, and in general at the University of Ottawa, helped me during this process. And my family and friends supported and encouraged me all these years. My thanks go to all of them as well.

This research was supported by the Natural Sciences and Engineering Research Council of Canada (NSERC) and by Ontario Centres of Excellence (OCE). The University of Ottawa also financially supported my studies.

Table of Contents

Table of tables

Table of Examples

Table of Figures

Abstract

Sentence Selection consists of identifying the sentences relevant to a particular topic, task, user or linguistic structure. This is a prerequisite step in many document-processing tasks, such as Information Extraction and Text Summarization.

Researchers in these areas have typically borrowed ideas and tools from the Automatic Text Categorization (ATC) domain and applied them to their Sentence Selection problems. This is the case with the standard Bag of Words text representation and the machine learning algorithms used.

Even though Sentence Selection and ATC are related, not all their characteristics are the same. Because of their differences, some variations to the standard representations and techniques usually used for ATC might be beneficial for Sentence Selection.

Consequently, the main contribution of this thesis is the exploration of the benefits of a syntactically and semantically enriched text representation for the Sentence Selection task on technical domains. We further take advantage of the syntactic and semantic relations between words by moving from the propositional learners to a relational model.

In particular, we experiment on three documents datasets, two in the Genetics domain and one in the Legal domain. In the first two domains we incorporate semantic knowledge by means of hierarchical dictionaries, while in the third one we use Named Entity Recognition for the same purpose. The syntactic knowledge is obtained from automatic parsing.

Bags of words, enriched with the syntactic and semantic features, are given as input to different classifiers induction algorithms. Sentences to be selected constitute the positive class, while the remaining sentences of a document constitute the negative class.

We present results with the state of the art algorithms Naive Bayes and Support Vector Machine, as well as the relational learner Aleph. We evaluate the learning performance by comparing on several runs of N-fold cross-validation and time based training/testing splits, and we study the implications of the classification threshold when appropriate.

We show the gains of enriching the representation in a syntactic or semantic way, we analyze the cases in which each one is more beneficial, and we explain the particular contributions of each of them.

1. Introduction

With the increase of information availability, it becomes more important every day to develop good methods to deal with text, such as obtaining the desired texts and filtering the unwanted ones, classifying and/or summarizing documents, and so on.

One particular task in dealing with text is Automatic Text Categorization (ATC), which consists of automatically building programs capable of labeling natural language texts with categories from a predefined set. It is performed using standard Machine Learning methods in a supervised learning task, which usually requires large amounts of previously classified data. Because of its nature, it combines knowledge and tools from the Information Retrieval, Artificial Intelligence, Linguistics and Statistics areas among others.

A main concern of the text classification research area is the lack of syntactic and semantic information in the Bag of Words (BOW) text representation used in many cases. Given that by syntactic information we refer to the grammatical structure of a sentence and by semantic information we refer to the meaning of the words and of the sentence as a whole[1], it is straight forward to notice that by keeping only the words of a sentence all that information is lost. This problem has been addressed in different ways, like stemming words, clustering similar terms together, and using background knowledge to name a few on the semantic side. Less has been done on the syntactic side. In this respect, using position-related predicates in an ILP system, and incorporating the order of noun phrases into the representation have been tried.

---

[1] As discussed in semiotics, the theory of signs, by the Vienna Circle, particularly in their International Encyclopedia of Unified Science, the field breaks out into three branches:

- Semantics: Relation between signs and the things they refer to, their denotata.
- Syntactics: Relation of signs to each other in formal structures.
- Pragmatics: Relation of signs to their impacts on those who use them.

http://en.wikipedia.org/wiki/Semantic

A particular case of ATC is Sentence Selection, which consists of identifying the sentences relevant to a particular topic, task, user or linguistic structure. This is a necessary step in many document-processing tasks, such as Information Extraction (IE) and Text Summarization (TS). The proportion of sentences in a given document considered relevant for the mentioned tasks is usually low, making some pre-filtering a prerequisite.

Even though Sentence Selection and ATC are related, not all their characteristics are the same. One of the differences is that the sentences are short in length, with few words from the vocabulary occurring in each of them. This results in an even more sparse representation than in the ATC case. Another difference is that ATC is usually used to recognize the general topic of a document, while Sentence Selection concentrates on more specific details. In table 1 we summarize these and other similarities and differences among Sentence Selection and other related tasks.

Table 1: Comparison among Information Retrieval, Automatic Text Categorization and Sentence Selection.

	Information Retrieval	**Automatic Text Categorization**	**Sentence Selection**
Problem	Find documents about	Decide on the general topic of a document	Decide whether the sentence is useful for a specific task
characteristics	Long documents Few words given	Long documents Sparse representation	Short sentences Very sparse
Algorithms	Distance in vector space representation	SVM - Naïve Bayes Decision Trees - kNN	?
Representation	Bag of Words	Bag of Words Extensive research on other representation	?

We address the task of Sentence Selection working on several technical corpora. We believe that because of the particular characteristic of Sentence Selection mentioned before, some variations to the standard representations and techniques usually used for ATC might be beneficial for Sentence Selection. In particular, because of those differences and because the vocabulary of the corpora we use is highly specific, the use of syntactic and semantic knowledge could be even more beneficial than in a collection of a more general nature. Our work consists of exploring the benefits of a syntactically and semantically enriched text representation for the Sentence Selection task on technical domains.

To further motivate the need for syntactic and semantic information in our specific problem, we briefly introduce an example in table 2. In it we observe a positive and a negative sentence from our Bacillus Subtilis dataset, the positive sentence expressing an interaction between two genes or proteins. Taking into consideration only the words that appear in those sentences, it would be very difficult to decide which one is the positive sentence. As it can be seen in the example, several gene or proteins names appear in both sentences, and also the words `activated` and `transcription` appear in both of them. It would therefore be useful to have extra information available, such as some semantic knowledge about words in those sentences that are gene or protein names, and whether they are syntactically related to one another. Table 2 also shows some syntactic information for both sentences in the form of syntactically related pair of words. From the syntactic information for the positive sentence, it can be seen that relevant information can be retrieved. In particular, some of the pairs can be arranged in the following way: `transcription_cotB - transcription_is - is_activated - activated_protein - protein_called - called_gerE`. From these pairs it is easier to decide that the sentence

expresses an interaction between genes or proteins, by reading it as "transcription of cotB is activated by a protein called gerE". Semantic knowledge, as a generalization providing information about cotB and gerE being genes or protein names, would also simplify the learning task.

Table 2: Example of Sentence Selection with the BOW representation and the syntactically and semantically enriched text representation.

positive sentence	Transcription of the cotB, cotC, and cotX genes by final sigma(K) RNA polymerase is activated by a small, DNA-binding protein called gerE.
words in sentence	activated - called - cotB - cotC - cotD - cotX - DNAbinding - final - genes - gerE - is - polymerase - protein - RNA - sigmaK - small - transcription
syntactic information	activated_protein - DNAbinding_protein - called_gerE - final_polymerase - is_activated - protein_called - RNA_polymerase - sigmaK_polymerase - small_protein - transcription_cotB - transcription_is - transcription_polymerase
negative sentence	HMM were used to identify promoters likely to be activated by SpoOA, final sigma(F), or a third sporulation transcription factor, final sigma(E).
words in sentence	activated - be - factor - final - HMM - identify - likely - promoters - sigmaE - sigmaF - spo0a - sporulation - third - transcription - used - were
syntactic information	activated_factor - be_activated - factor_sigmaE - HMM_were - identify_promoters - sporulation_factor - transcripton_factor - were_used

In the next two sections we first introduce the text categorization task and the current research in different text representations. We then describe our approach together with the datasets and tools we used in section 4, and the experiments in section 5 so far.

The main contribution of our work is that this is the first study that we are aware of that concentrates in the text representation for Sentence Selection. It includes extensive experiments with different representations and classifier learning algorithms, as well as in different datasets and domains.

Several findings regarding the contributions of the syntactic and semantic enriched representation are presented and analyzed in section 6, and recommendations are made on when to use each one.

We finalize this thesis with some conclusions and future work in section 7. Some clarifications and related preliminary experiments are presented in the Appendices.

2. Text Categorization Review

Fabrizio Sebastiani [S02] presents a complete survey of the Text Categorization area. He defines Automatic Text Categorization (ATC) as the task of automatically assigning documents to a set of predefined categories, differentiating it from the automatic definition of such set of categories (clustering), and the automatic assignment of documents to a set of categories which is not predefined (indexing). We hereby summarize his survey.

2.1. Formal definition of ATC

Formally, ATC can be expressed as an approximation to a binary function $f: CxD \rightarrow \{0,1\}$ that given a category from $C=\{c_1,\ldots, c_m\}$ and a document from $D=\{d_1,\ldots,d_n\}$ determines whether the document belongs to that category or not.

For this definition the categories are symbolic labels and the attribution of documents to categories is based on the documents content (and not on metadata such as document type, publication source, etc.)

Depending on the application, each document might be associated with only one category or with several categories, which are respectively referred to as the single-label case or the multi-label case. It might also be the case that given an integer k, either $\leq k$ or exactly k or $\geq k$ categories must be assigned to each document, or even that each category must be assigned to either $\leq k$ or exactly k or $\geq k$ documents.

Most of the techniques discussed here apply for either case. In the reminder of this section we will assume always the multi-label case with up to m categories per document (where m is the total number of categories in C.) We will also assume that categories are stochastically independent of each other, and therefore we will be able to see the

categorization problem as m independent problems consisting of approximating the function f_i: D $\rightarrow$ {0,1} that determines which documents (from D=$\{d_1,\ldots,d_n\}$) belong to c_i for each category $c_1,\ldots, c_m$. For each of these problems, the documents of the category c_i being determined are referred as belonging to the positive class, while the rest are referred as belonging to the negative class.

2.2. Machine Learning approach to ATC

In the '80s, the ATC systems were manually built expert systems consisting of sets of disjunctive normal form (DNF) rules per each category that if satisfied by a document denoted that it belonged to the specified class. Even when these systems generally reached high performance, the drawbacks of this approach are the need of both knowledge engineers and domain experts to build and modify the system, and its inadaptability from one domain to another.

Since the early '90s, the machine learning approach to ATC has been developed. In this approach the classifier for each class is automatically built by a general inductive process by observing the characteristics of a set of documents previously classified. The knowledge engineers can now concentrate on this general process that adapts to different domains, while the domain experts only need to provide the classified examples. The performance reached with these systems is comparable to manual classification.

For the purpose of automatically learning a classifier for a class and evaluating its performance, the already classified documents are divided in two sets, not necessarily of equal size:

- the Training Set (Tr) are the documents from which the characteristics are observed and the classifier for the class is automatically induced,
- the Testing Set (Te) are the documents used to evaluate the classifier performance by comparing the class predicted by the classifier with its actual label.

In the cases where some parameters of the inductive process need to be tuned according to the present task in order to optimize the resulting classifier, the training set is further divided saving some documents as a Validation set (Va) on which repeated tests are performed until the parameter has been correctly adjusted.

2.3. Indexing and dimensionality reduction

Indexing is one of the several aspects in which ATC borrows concepts from the Information Retrieval (IR) area. To represent each document, both in IR and ATC, a vector of n weighted index terms is usually used, with the weights ranging from 0 to 1 (and sometimes taking only the 0 or 1 values to denote either the absence or the presence of the term in a binary mode.) When each index term is a word from the collection of documents, the representation is referred as the Bag of Words (BOW). Many other index terms approaches, mainly considering the order of the words in the document and the syntactic role they play (like statistical and syntactical phrases), have been considered in both IR and

ATC research but the results are still controversial. Several papers on this respect are discussed in the next chapter.

In order to determine the weight of term *k* for the representation of document *j*, the *term frequency inverted document frequency* (*tfidf*) is often used. This function is defined as:

$$tfidf(t_k,d_j) = \#(t_k,d_j) * \log (|Tr| / \#(t_k))$$

where Tr is the training set, $\#(t_k,d_j)$ is the number of times t_k occurs in d_j, and $\#(t_k)$ is the number of documents in Tr in which t_k occurs at least once (the document frequency of t_k.) This function captures the idea that the more times a word occurs in a document, the more relevant it is, together with the concept that the more documents from the collection a word occurs in, the less discriminating it is.

To make the weights fall in the [0,1] interval, they are usually normalized by dividing them by the l_2 norm or Euclidean length, that is also known as cosine normalization. The weight of term *k* for the representation of document *j* is therefore denoted by:

$$w_{k,j} = tfidf(t_k,d_j) / sqr(\sum_{s=1..r} (tfidf(t_s,d_j))^2)$$

where *r* is the number of terms that occur at least once in Tr.

As presented by Garcia's tutorial[2], several steps may be performed before indexing, like the removal of function or stop words (e.g. articles), selection of the parts of the texts to be used (in structured documents), stemming (removal of inflection) [P80], and lemmatization (generalization of words to its root form) [S96]. The last two are not widely used since they might hurt effectiveness.

A definitely needed step before indexing is dimensionality reduction (DR). Unlike in IR, in ATC the high dimensionality of the term space is problematic because the learning

[2] http://www.miislita.com/information-retrieval-tutorial/indexing.html

algorithms used for classifier induction do not scale well to high values of r (the number of terms in Tr) and therefore r has to be reduced to r' << r. Another problem where DR comes to help is that of overfitting, when a classifier is tuned also to the contingent (rather than only constitutive) characteristics of the training data. In order to avoid overfitting, a number of examples proportional to the number of index terms used is needed, and by reducing the terms dimensionality less examples are then required.

DR can be performed locally (for each category) or globally (for all categories together) by using the same techniques. Depending on the set of indexing terms (or features) resulting from the DR process, it is denominated feature selection or feature extraction.

2.3.1. Feature Selection

It is also called term space reduction (TSR) and it consists of selecting, from the original set of r features, the subset of r' features that yields the best performance in classification. This could be achieved, at a high cost, by trying different subsets in an iterative learning and evaluating process known as *wrapper*.

Another approach to feature selection, computationally cheaper and more commonly used is *filtering*. It keeps the r' features that score highest according to a function that measures the importance of the term for the categorization task.

Document frequency ($\#(t_k)$) is a simple and effective global feature selection commonly used. An empirical way consisting of removing all terms that occur in x training documents at most ($x <= 3$) is adopted by many authors, either by itself or before applying more sophisticated functions.

Some functions that are used for feature selection by filtering are shown in Table 3. Most of them try to capture the intuition that the most valuable terms are those most differently distributed in the positive and negative examples of a given category. Even some functions that contradict this intuition, as it is the case of Chi square, have been successfully used in DR for ATC. When the probability is needed in the functions, it is approximated by the frequency.

Table 3: Some commonly used filtering feature selection functions.

Function	Denoted by	Mathematical Form
Frequency	**F(t)**	$\Pr(t)$
Information Gain	**IG(t)**	$-\sum_{c_i}[\Pr(c_i)\log(\Pr(c_i))]+\Pr(t)\sum_{c_i}[\Pr(c_i/t)\log(\Pr(c_i/t))]+\Pr(-t)\sum_{c_i}[\Pr(c_i/-t)\log(\Pr(c_i/-t))]\equiv\sum_{c_i}\left[\Pr(t,c_i)\times\log\left(\frac{\Pr(t,c_i)}{\Pr(t)\times\Pr(c_i)}\right)\right]+\sum_{c_i}\Pr(-t,c_i)\times\left[\log\left(\frac{\Pr(-t,c_i)}{\Pr(-t)\times\Pr(c_i)}\right)\right]$
Mutual Information	**MI(t,c)**	$\log\left(\frac{\Pr(t,c)}{\Pr(t)\times\Pr(c)}\right)$
Chi Square	**CHI(t,c)**	$\#doc\times\frac{[(\Pr(t,c)\times\Pr(-t,-c))-(\Pr(-t,c)\times\Pr(t,-c))]^2}{\Pr(t)\times\Pr(-t)\times\Pr(c)\times\Pr(-c)}$
Odd Ratio	**OR(t,c)**	$\log\left(\frac{\Pr(t,c)\times\Pr(-t,-c)}{\Pr(t,-c)\times\Pr(-t,c)}\right)$

In this table, t ($-t$) represents the presence (absence) of a term in a document d, and c ($-c$) represents the fact that the document d belongs (does not belong) to the class c.

2.3.2. Feature Extraction

It consists of synthesizing from the original set of r features, a set of r' << r new features that yields the best performance in classification. The synthetic features would avoid the problems of polysemy, homonymy and synonymy that words suffer from. Two approaches that have been used for feature extraction are clustering terms by means of semantic relatedness and Latent Semantic Indexing (LSI).

LSI infers the dependency among the original terms from a collection of documents (corpus) and creates a new set of independent features (each one representing one of the previously related subsets of terms.) The function mapping the original features into the new features is obtained by applying singular value decomposition (SVD) to the matrix formed by the original document vectors. Because of how they are built, the new features are not intuitively interpretable. Another disadvantage of LSI is that if one term was particularly good at discriminating the class, that power might be lost in the new set of features. However, if there is a great number of terms that contribute a small amount of critical information, LSI might keep that information while other DR techniques might lose it. In Appendix C we describe LSI in more details and present experiments we performed using it.

2.4. Machine Learning algorithms

The construction of a classifier for a category $c_i \in C$ usually consists of two steps:

1. the definition of a categorization status value function (CSV) that given a document d returns a number representing the evidence that d should be classified under c_i.

$CSV_i : D \rightarrow [0, 1]$.

2. the definition of a threshold τ_i such that $CSV_i \geq \tau_i$ is interpreted as a decision to categorize document d under category c_i and otherwise when $CSV_i < \tau_i$. The value of τ_i is defined by using a validation set.

The threshold τ_i is a value of the CSV_i function. It is defined by testing different possible values on the validation set and choosing the one that yields the best effectiveness. The threshold τ_i is not needed in the cases when the classifier already provides a binary judgment ($CSV_i : D \rightarrow \{0, 1\}$). And depending on the application, sometimes this step is not desirable as the real value of CSV_i is preferred (e.g.: for ranking different categories, for proportionally assigning categories to a test set.)

The function CSV_i could be defined in different ways (e.g. in terms of a probability or as a distance between vectors) according to the machine learning algorithm chosen to construct the classifier. Some of these algorithms are introduced in the following subsections.

2.4.1. Probabilistic Classifiers - Naïve Bayes (NB)

In the probabilistic classifiers [L98] the function CSV_i is defined in terms of the probability $P(c_i|d_j)$ that document d_j (represented as the vector of either weighted or binary features $d_j = \langle w_{j,1},\ldots,w_{j,r}\rangle$) falls within category c_i). They attempt to compute this probability using Bayes' theorem: $P(c_i|d_j)=P(c_i)P(d_j|c_i)/P(d_j)$.

In this equation, $P(c_i)$ is the probability that a randomly picked document d falls into class c_i (which can be estimated from the training set) and $P(d_j)$ is the probability that a randomly picked document has vector d_j as its representation (which can be disregarded

since we assume all documents have the same probability and the fact that we are not concerned with the real probability but rather with the comparison among the different classes.)

The estimation of $P(d_j| c_i) = P(w_{j1},\ldots, w_{jr} | c_i)$ is the problematic part of the previous formula. Since the number of possible vectors d_j is too high, their approximation cannot be calculated by counting occurrences. To alleviate this problem it is common to make the naïve assumption that any two features are statistically independent (when viewed as random variables.) Under this independence assumption $P(d_j| c_i) = \prod_{k=1..r} P(w_{kj}| c_i)$. The probabilities $P(w_{kj}| c_i)$ can then be easily estimated by counting occurrences in the training set. It must be noticed though that the use of smoothing techniques is required to avoid values of 0 for any $P(w_{kj}| c_i)$.

Probabilistically classifiers that make use of this assumption are called Naïve Bayes and have been widely used for text categorization. Even when it is clear that words are not independent of each other within a document, and therefore the independence assumption is violated, the Naïve Bayes algorithm is still able to provide highly accurate classification results.

2.4.2. Decision Trees (DT)

Decision trees [M96 chapter 3] are a very popular classification method. A decision tree provides a structure that when followed from the root to the leaves determines the class of a particular item in a classification problem.

When used for text classification, each internal node of the tree corresponds to a feature and the branches departing from it correspond to the possible values for that feature.

When the feature values are binary it results in a binary tree (one with two branches departing from each node). The leaves of the tree are the categories for the classifier, and many leaves could have the same category. Given a document d_j, a decision tree determines the category in which the document falls by recursively following the branch with the value of the document for the feature on each node.

A possible procedure for the induction of a decision tree from a set of training examples is the "divide and conquer" strategy. It consists of recursively selecting the feature with the higher information gain, partitioning the training set according to that feature, and creating a sub-tree for each possible value of the feature. The iteration stops when all the examples in a branch belong to the same class, creating a leaf with that class as the label. This process may result in over-specific trees that will not generalize well in the testing examples, and for this reason the branches are usually pruned some levels up from the leaves, assigning the majority class to the whole branch.

2.4.3. Inductive rule learning - Ripper and Flipper

The classifiers built by an inductive rule learning system [C95a, C95c] consist of a set of disjunctive normal form (DNF) rules (or equivalently if-then-else rules) similar to the ones of the expert systems of the 80's. The resulting rules set covers (i.e. correctly classifies) all the positive examples from the training set.

As it was the case with Decision Trees, DNF rules can encode any Boolean function. An advantage over Decision Trees is that DNF rules learners usually generate more compact classifiers. While the divide-and-conquer strategy builds DTs in a top-down way, DNF rules are built in a bottom up fashion. Initially every example is considered as a rule (one that

classifies instances into the example's class), and later on that set of rules is generalized to decrease the over fitting. The generalization is performed by removing conditions from each rule and merging rules together while still covering all the examples. A pruning phase similar to the one in DTs can also be applied at the end of the learning process.

For ATC learners of propositional logic rules are mainly used, but learning first order logic (FOL) rules have also been tried. Examples of such systems are Cohen's creations Ripper [C95a] and its FOL version Flipper [C95c].

2.4.4. Instance-based classifiers – k Nearest Neighbors (k-NN)

These classifiers are considered lazy learners since, instead of performing an early generalization of the training data, they use the classified examples directly when they have to classify a new instance [M96]. In this way, all the computational cost is transferred from the learning to the testing phase. Still the time needed to perform one particular classification remains linear to the size of the training set.

One implementation of these classifiers that has been used for ATC is the k Nearest Neighbors algorithm [Y94]. Given a new document d_j, the algorithm assigns the class of the majority among the k training documents most similar to it, the parameter k being optimized on a validation set. The decision could be weighted according to the similarity of the examples (instead of simple majority) and the similarity measure used could be of different types (e.g. probabilistic, distance between vectors, etc.) One advantage of this method is that it does not separate the space in only two subspaces, but instead in several smaller subspaces, allowing a more local classification.

2.4.5. Neural Networks (NN)

A neural network applied to ATC is a network of units, where the input units are the indexing terms and the output unit represents the category (several output units are used for the multi-label case.) The units are connected and the connections are weighted. For classifying a test document, its term weights are loaded into the input units, the activation of these units is propagated forward through the network, and the value of the output unit(s) determines the categorization decision(s).

A typical way of training NNs is back-propagation, that is: the term weights of a training document are loaded into the input units, and a classification value obtained; if a misclassification occurs the error is "back-propagated" so as to change the weights of the network connections and eliminate or minimize the error.

The simplest type of NN, the perceptron [NGL97], is a linear classifier with input units connected directly to the output unit(s). A nonlinear NN is instead a network with one or more additional "layers" of units, which in ATC represents relations between terms for which the network is able to learn its strengths (the weights of the connections). The performance of both types seems to be similar for ATC.

2.4.6. Support Vector Machine (SVM)

The Support Vector Machine method [J98] finds the (r-1) dimensional surface that separates examples from two classes with the widest possible margin. For example, in a linearly separable 2 dimensional space, out of all the lines that separate the positive of the negative examples, SVM chooses the middle line from the set of lines in which the maximum distance between two training examples is highest.

Some advantages of this method are that it is applicable even when the data is not linearly separable (by transforming the examples into a dimension where they are linearly separable), that the decision surface is determined by only a small set of training examples, and that non dimensionality reduction is required since it does not suffer from over fitting.

2.5. Evaluation

The evaluation of the classifiers is typically done experimentally, and it concentrates in the effectiveness of the classifier, i.e. its capability of taking the right categorization decision. In this section we discuss some of the metrics that are commonly used.

Accuracy and error are metrics widely used in machine learning. Accuracy is the proportion of examples that are correctly classified, while the proportion of the examples that are not correctly classified is the error.

Given the following contingency table

	predicted		
actual		+	-
	+	TP	FN
	-	FP	TN

where:

- TP (True Positives) is the number of positive examples that were correctly classified,
- FP (False Positives) is the number of negative examples that were incorrectly classified,
- FN (False Negatives) is the number of positive examples that were incorrectly classified,
- TN (True Negatives) is the number of negative examples that were correctly classified,

the previous measures are defined as:

- Accuracy = (TP + TN) / (TP + TN+ FP +FN)
- Error = (FP + FN) / (TP + TN+ FP +FN) = 1 - Accuracy

In ATC it is often the case that the dataset is highly imbalanced (usually many more negative than positive examples.) In this case the accuracy and error measures are not very sensitive to variations in the number of correct decisions. In addition, the trivial rejecter (a classifier that determines that every example belongs to the negative class) will score well according to these measures.

Because of the previous reasons, the precision and recall measures from Information Retrieval are more commonly used for ATC (with the necessary modifications). Precision is the proportion of documents that actually belong to the class out of the ones that were classified as belonging to it. Recall is the proportion of the documents that were classified as belonging to the class out of the total that actually belongs to it.

Using the values from the contingency table, these metrics are defined as:

- Precision (Pr) = TP / (TP + FP)
- Recall (Re) = TP / (TP + FN)

Precision and recall are related metrics that better express the effectiveness of the classifier when used together. By changing the classifier threshold value we can improve one to the detriment of the other. Because of that, the following measures that combine Pr and Re are used among others.

- 11 point average Pr: the threshold is adjusted to obtain recall values of 0.0, 0.1,..., 0.9. and 1 and the 11 different precision values are averaged.

- Breakeven point (BEP): the threshold is set to the value that makes Pr and Re (almost) equal.
- F1 measure: is a function that gives equal importance to Pr and Re for a fixed threshold. It has been shown that given a classifier, its breakeven value is always less or equal than its F1 value. The function formula is: $F1 = 2*Pr*Re / (Pr + Re)$

Once an effectiveness measure has been chosen, the classifier can be tuned as to obtain the best possible effectiveness (according to that metric).

In the cases where several classifiers for different classes of the same dataset have been learned, as *n* binary classifiers for a *n* classes problem, it might be desirable to obtain a global measure of performance. Any of the previously introduced measures could be used to obtain an average according to one of the following criteria:

- Micro-averaging: it first calculates the chosen metric (e.g.: precision) values for each classifier, using its respective contingency table, and then averages those values.
- Macro-averaging: it adds up together the corresponding values from the contingency tables of all the classifiers, and then it calculates the chosen metric using the global table.

The use of one or the other criteria is based in whether the importance of the classification results is related to the frequency of each class (macro-averaging) or every class is given similar importance (micro-averaging).

3. Recent bibliography review

3.1. Text Classification Results

We present here several methods that have been applied to ATC and the results obtained with these methods. Several researchers have directly compared different approaches, and we also present here some of their conclusions.

Dumais et al. [DPHS98] test various classification techniques and feature selection methods on the Reuters-21578 data set using the ModApte split (which is a particular division into training and testing examples). The best results are obtained for SVM on a representation using a feature set selected using Mutual Information. They achieve a 92.0% BEP on the 10 most frequent categories of Reuters.

Yang and Liu evaluate and compare several approaches to ATC. Yang [Y99] performed experiments using kNN, a Linear Least Squares Fit mapping method (LLSF) and a baseline consisting of the presence of the class word (WORD). She ran the experiments on 5 different subsets of the Reuters collection and compared these results with other published ones. She concluded that, on the datasets where all the examples were labeled, kNN, LLSF and Neural Networks performed the best. Yang and Liu [YL99] experiments on Reuters concluded that SVM, kNN and LLSF outperform Neural Networks and Naïve Bayes for the categories with less than 10 positive training examples, and that all methods perform similar for common enough categories (over 300 examples).

Forman and Cohen [FC04] also compared several algorithms on different datasets for ATC. They evaluated SVM, Logistic Regression, Naïve Bayes and Multinomial Naïve

Bayes on 19 subsets of Reuters, OSHMED and TREK. The percentage of positive examples in the training set was on average 5%. They concluded that "different models excel in different regions of the learning surface, leading to meta-knowledge about which to apply in different situations."

Using the structure of the categorized texts – document title, headings – has also provided features that are often more informative than features extracted from the body of a document [F99]. A comparison between Naïve Bayes and Decision Trees using a mixed of structured and textual features in the task of E-mail filtering is presented by Diao et al. [DLW00]. They report the importance of the different features and conclude that "while the Decision Tree based classifier outperforms the Bayesian classifier when features and training size are selected optimally for both, a carefully designed naïve Bayesian classifier is more robust."

Another approach has been that of boosting the power of rather simple or weak classifiers by combining the decision of several of them. Shapire and Singer [SS98] and Weiss et al. [WADJOGH99] use boosting algorithms with decision stumps and decision trees respectively to achieve 86% BEP on all Reuters categories, and 87.8% respectively, on the 95 largest categories of Reuters (ModApte split).

Considerable research has also been done in the feature selection research area when applied for ATC. The following two works present approaches to feature selection that adjust well to their particular problem. Mladenic and Grobelnik [MG99] work with a probabilistic classifier on imbalanced datasets, while Xu [X03] uses decision trees for learning from small datasets. They showed that considering the characteristics of the

problem and the classifier when performing the feature selection can improve the results of the final classification.

3.2. Background Knowledge in Automatic Text Classification

A major concern of the text classification research area is the lack of semantic information in the text representation used in many cases. Siolas [S03] approaches this fact trying to introduce semantic knowledge into the kernel function of kernel based classifiers. The author first tried that by building a kernel that takes into consideration the semantic distance between words using information from WordNet. However this suffers from the limited amount of words expressed in this thesaurus. He then tries a new kernel based on Fisher metrics, which by itself catches some semantic knowledge grouping concepts alike (in a way similar to LSI.) Using supervised classification he empirically proved that this new representation is more efficient than using the tf-idf one.

Cohen and Hirsh [CH98] present the system WHIRL as an extension of relational databases able to perform similarity joins on texts. They evaluated it on several ATC tasks, showing that it achieves accuracies comparable to those of C4.5, Ripper and Nearest Neighbors algorithms. They also show that the system can be used to select among many unlabeled examples those that could be classified with high confidence. Using the same system, Zelikovitz and Hirsh [ZH00] describe a method for improving the classification of short texts by adding longer related texts that are unlabeled. These extra texts serve as bridges between the labeled ones. They show that the use of background knowledge in this way decreases error rates mainly if the training sets are small (either short texts or few examples.)

Zelikovitz and Hirsh [ZH01] also explored the inclusion of background knowledge into LSI to help text classification. The background knowledge they considered in these experiments is of different types, sometimes being unlabeled examples from the same domain and other times coming from other domains. Since the matrix SVD performed by LSI does not take into consideration the label of the documents, it is simple to add these new documents to enrich the representation. The empirical results show that "background knowledge is most useful when the training set is small."

In Nigam's Doctoral Dissertation [N01], he introduces an iterative Expectation Maximization based algorithm using a naïve Bayes classifier for text classification. The method works with few classified examples and a pool of many unclassified extra examples. It basically first learns a classifier based on the labeled examples, and classifies every other example with this classifier. It then re-learns the classifier and iterates until convergence. In his work with other authors [NMTM00], two extensions are introduced for the cases in which the "generative assumptions of the model" are violated. In this way they reduce the classification error up to 30% in real data applications.

Joachims [J99] presents a variation of SVM to which he refers to as Transductive SVM. In this algorithm the decision of where to divide the examples space into positives and negatives is helped by an iterative process of labeling the testing examples and re-learning. The resulting separation maximizes a margin where neither training not testing examples occur. He experimentally shows, in three different text classification collections, that the Transductive SVM works better than SVM mainly when the amount of training data is small.

An interesting work related to the use of background knowledge, even when the experiments are not in the ATC research area, is the one by Latinne et al. [LSD01]. For classification methods that provide estimates of the posteriori probabilities of the classes, these depend on their a priori probabilities. Therefore applying the learned classifier to data with different a priori probabilities than the training data could affect its accuracy. If the new class distribution is known, then the outputs could be corrected. However, when introducing unlabeled examples as extra background knowledge, the a priori probabilities are often unknown. In this paper the authors introduce an iterative method (based on Expectation Maximization) to approximate the class proportions while correcting the classification decisions. They try this procedure in a real world classification problem, "the automatic labeling of geographical maps based on remote sensing information" using a logistic regression model. They show significant classification improvement compared to the same algorithm without the correction procedure and other algorithms.

3.3. Text Representation

A main concern of the text classification research area is the lack of syntactic and semantic information in the Bag of Words (BOW) text representation used in many cases. This problem has been addressed in different ways, like stemming words, clustering similar terms together, and using background knowledge to name a few on the semantic side. Less has been done on the syntactic side, mainly studying the value of including noun or statistical phrases.

In this section we introduce some studies done in relation to the text representation on different areas of research. It must be noticed that none of the related work in this section addresses specifically sentence selection.

The usefulness of syntactic and statistical phrases compared to the BOW was first studied by Fagan [F87] in the Information Retrieval (IR) context. Statistical phrases are words that happen together in the collection often enough. In these experiments it was shown that statistical phrases were not only easier to obtain but they also improved performance more than syntactical phrases.

Lewis [L92a] [L92b] compared different representations using either words or syntactic phrases (but not a combination of both) for IR and ATC. One of the representations clustered the phrases in order to make them more general. The results with the phrases representation showed no significant improvement with respect to the representation using the BOW.

Mitra et al. [MBSC97] study the usefulness of linguistic knowledge for an IR system. The results indicate that the noun phrases are useful for lowly ranked answers but not much for the highly ranked answers where the words alone perform well. Similar results were obtained in ATC by Furnkranz et al. [FMR98] when building syntactical phrases following some particular syntactic patterns learned from the data by an extraction system.

Dumais et al. [DPHS98] studied the use of syntactic phrases with a variety of text classifiers on the Reuters-21578 collection showing no benefit at all from the use of this representation. Scott and Matwin [SM99] also noted no significant improvement of the

performance by adding noun phrases to the representation of the same corpus but using a different Machine Learning algorithm.

In another study in the top 10 categories of the Reuters collection, Nastase et al.. [NSC07] showed that better accuracy and precision is obtained for most of the classes when adding syntactically enriched features in the representation. An analysis of the classes where the use of the syntactic information negatively affected the performance showed the lack of syntactic information in its documents, as for example a class where most of the documents consist of only tables. Furthermore a correlation was found between the sparseness of the syntactically enriched representation and the low performance obtained by it.

Furnkranz et al. [FMR98], Mladenic and Grobelnik [MG98] and Caropreso et al. [CMS01] studied the usefulness of statistical phrases in ATC. The most discriminating phrases were added to the BOW. The experiments showed that the use of these phrases could in some cases improve the classification.

The GURU system, by Maarek et al. [MBK94], used lexical affinities for indexing purposes in an IR task. Linguistically, lexical affinities are words that are involved in a modifier-modified relationship and that appear often together in the language. This work, however, only takes into consideration the closeness of the chosen words.

Cohen and Singer [CS99] study the importance of introducing the order of the words in the text representation by defining position related predicates in an ILP system. This has been extended by Goadrich et al. [GOS04] in recent research in the IE area, incorporating the order of noun phrases into the representation.

In another work in IE, Ray and Craven [RC01] incorporate syntactical phrases in a Hidden Markov Model (HMM) that recognizes the grammatical structure of sentences expressing biomedical relations. The results show that this approach learns more accurate models than simpler HMMs that do not use phrases in the representation.

One more approach to IE that uses syntactic information is Temkin and Gilder [TG03] work. In this work a Context Free Grammar (CFG) was defined to recognize protein, gene and small molecule interactions. The results show that efficient parsers can be constructed for extracting these relations. From this work we borrowed the list of words commonly used in the genetic bibliography to denote interactions.

Several studies have introduced semantic knowledge in a text classification task. Siolas [S03] did that by building a kernel that takes into account the semantic distance: first between the different words based on WordNet, and then using Fisher metrics in a way similar to Latent Semantic Indexing (LSI). Zelikovitz and Hirsh [ZH01] [ZH02] showed that the text classification accuracy can be improved by adding extra semantic knowledge into the LSI representation coming from unclassified extra documents.

3.4. Sentence Selection

As previously mentioned, sentence selection is a necessary step in many document-processing tasks, such as Information Extraction (IE), Rhetorical Analysis (RA) and Text Summarization (TS). The context in which the sentence selection is performed determines the purpose of the selection. For example, in the context of IE, the sentences selected would be the ones believed to have the information to be extracted. And in the context of TS, the

sentences selected would be those considered to contain information of relevance for a summary of the whole document.

In this section we briefly introduce the IE and TS research areas, and we present some recent works that make use of sentence selection.

3.4.1. Information Extraction and Biological Sentence Selection

The task of Information Extraction consists of automatically filling in structured templates with information found in documents written in natural language. This requires knowledge of the context and of the characteristics of the specific information that is desired. It is usually performed through several steps involved in the linguistic analysis of the documents and the application of extraction patterns or rules. These patterns or rules explain how to identify in the text the values to fill in each field of the templates.

In the biological context the classical Information Extraction systems are based on hand-written patterns proposed by domain experts, which usually obtain high precision but at low recall levels. As an alternative, statistical methods based on co-occurrences of technical terms have been used, increasing the recall values in decrement of the precision.

More recently several approaches involving Machine Learning have been tried. Ray and Craven [RC01] study the application of Hidden Markov Models (HMMs) to the biological IE problem. A HMM is an automaton whose states are linked by transitions with probabilistic weights learned from the training examples. Ray and Craven learn two models that represent the grammatical structure of sentences as HMM states, one model for the positive sentences examples and the other one for the negative examples. In that way they

perform sentence selection while at the same time extracting from the states of the model the values to complete the template.

Goadrich et all. [GOS04] experimented with Inductive Logic Programming (ILP) applied to IE from documents in the biomedical domain. The predicates used for the text representation include knowledge of the sentences structure, statistical word frequency, lexical properties and biomedical dictionaries. They learned ensembles of rules and they proposed a way of combining them in order to obtain comparable testing precision-recall curves in a fraction of the training time (with respect to the same system without this variation.)

During the CADERIGE project (introduced in the next chapter) a multidisciplinary approach was proposed. Abstracts were first filtered by selecting the sentences that probably describe interactions between genes, according to co-occurrence of gene names and automatically learned classifiers. Biology and morpho-syntactic knowledge is then combined to obtain a semantic representation of these sentences, and semantic relations learned by means of an Inductive Logic Programming system. From the semantic relations the values of interest could then be extracted and provided to the templates. As part of this project, Ould [O05] experimented with different algorithms for the sentence selection module of the system using the classical BOW text representation. The best results were obtained when using a modification of the Naïve Bayes algorithm specially adapted to the case of short texts.

3.4.2. Text Summarization and Legal Sentence Selection

Text Summarization consists of producing a summary or abstract for a given document. This can be done automatically by either generating it from a semantic representation of the text or by extracting relevant document fragments. The latter can be achieved by dividing the document into semantically coherent groups of sentences and then identifying in them the phrases that characterize the text.

One of such systems is CALLISTO [CJS02] which allows to define many different configurations for the text segmentation and the key extraction modules. During his master thesis, Rigouste [R03] studied how to, given a particular summarization problem, automatically find the best set of configuration parameters using machine learning.

Many of the existent automatic Text Summarizers follow the text extraction approach, in particular performing some kind of sentence selection. The chosen sentences are usually reordered and lightly modified to increase the coherence between them.

Teufel and Moens [TM02] went beyond simple sentence selection and classified source sentences according to their rhetorical status in order to be able to generate different kinds of summaries. They performed their experiments on a scientific corpus.

Following Teufel and Moens approach, Hachey and Gover [HG05] experimented in determining the rhetorical status of sentences for the legal domain. This was done as an initial step to the sentence selection component of their text summarization system. For this purpose they compiled and annotated a corpus of judgments of the UK House of Lords which they made available and we are using it for our research (an introduction to this corpus is presented in the next section.)

4. Syntactically and semantically enriching the Bag of Words.

Our approach consists of incorporating both syntactic and semantic information into the text representation for the sentence selection task. For this purpose we incorporate semantic background knowledge (such as technical terms or entities names) as well as noun phrases and other syntactic relations between words (such as the subject performing the action and the object of the action.) It is understood in linguistics that syntactically related words express semantic concepts [F68]. By using syntactically related words (to which we will refer as syntactic bi-grams) we are then further incorporating semantics into the representation. We obtained the syntactic bi-grams using the Link Parser [ST91] which specifically provides the relation between words in the sentence by establishing a link between them.

Using the syntactic bi-grams together with their single words, we represented the sentences, we automatically learned classifiers that categorize them and we evaluated the classification performance of this representation compared to the BOW. Our Experiments include state of the art machine learning algorithms Decision Trees, Naïve Bayes and Support Vector Machine from the Weka package, as well as the relational learners Aleph for which a particular relational representation was created. In the cross-validation experiments, the statistical significance of the results was calculated using the default method on the Weka experimenter, which is the corrected resample two tailed T-test with 0.05 confidence.

In the remainder of this section we introduce the 3 datasets that we have used, as well as the Link Parser and the relational learner Aleph.

4.1. Datasets

4.1.1. The Bacillus Subtillis dataset

Most functional genomics knowledge is available to the scientific community in natural language in scientific articles over the Internet. But the overflow of information sets a challenge for today's genomic researchers to identify genes and/or proteins interactions from these documents. Therefore automatic access to the information is of interest to the scientific community.

With this in mind, the CADERIGE project[3] [NOB01] studied the possibility of automatically extracting the genes and/or proteins interaction from the documents using Information Extraction (IE) tools. They conducted the studies in several collections of sentences relevant for the task.

Most of our experiments were done on a corpus created by the Caderige project and referred to as BS. The examples are sentences automatically selected from the MedLine collection of paper abstracts by a query with the keywords *Bacillus subtilis transcription.* The sentences were then pre-filtered to keep only those 932 ones that contain at least two names of either genes or proteins. These sentences were manually categorized by biologists as positive or negative according to whether they describe or they do not describe a genomic interaction. It resulted on a balanced dataset with 470 positive and 462 negative examples. Table 4 summarizes the features of this dataset.

[3] CADERIGE Project, http://caderige.imag.fr/

Table 4. Bacillus Subtilis dataset

Corpora	BS_links
Origin	MedLine
# examples	932
# positive examples	470
# negative examples	462

Some earlier work done on this corpus is presented in [OCMNM03]. It reports the recall and precision result obtained by the C4.5 algorithm and a variation of Naïve Bayes algorithm (that specializes it for the case of short documents). The attributes were all words after stemming, stop word removal and some filtering using Information Gain. The best results were 84.12% recall and 87.89% precision with the variation of Naïve Bayes.

In his work on the same dataset, Ould [O05] noted that this collection is extremely sparse, with 60% of the 2340 attributes happening in only 1 or 2 sentences.

4.1.2. The Yeast Protein Dataset

Our next dataset is a corpus created by Ray and Craven at the University of Wisconsin. Each example consists of a single sentence. The sentences were obtained by selecting location tuples from the Yeast Protein Database (YPD) and accessing the Medline abstracts they refer to. The sentences were manually categorized as positive or negative

according to whether they contain or they do not contain (respectively) the words in a protein location tuple. It resulted on an imbalanced dataset with 780 positive and 6349 negative examples. Table 5 summarizes the features of this dataset.

Table 5. YPD dataset

Corpora	YPD
Origin	MedLine
# examples	7129
# positive examples	780
# negative examples	6349

This dataset[4] was used by Ray and Craven [RC01] for Information Extraction of protein-location tuples using Hidden Markov Models (HMM). Two models were built and combined, one for the positive and ones for the negative examples. They present the results as precision-recall curves from which the breakeven point can be estimated to be around 0.3 or a little higher when further adjusting the HMM parameters.

4.1.3. The HOLJ Dataset

The HOLJ dataset was built at the University of Edinburgh as part of a Text Summarization of legal judgments project. The original documents came from a corpus of

[4] The dataset used by Ray and Craven [RC01] is a previous version of the current available one. In the new version of the dataset several aspects of it have been cleaned up. In particular, the class 'other' has been eliminated and bulletized items are considered as independent sentences.

judgments of the UK House of Lords. Details of the original corpus and of the process of building the dataset we are using can be found in the work of Hachey and Grover [HG05].

The HOLJ dataset consist of 11,429 sentences from different Lords' judgments, which have been manually annotated with their rhetorical status. The rhetorical roles of sentences reflect the argumentative structure of the texts. For this legal domain, Hachey and Grover say "the author's primary communicative goal is to convince his peers that his position is legally sound, having considered the case with regard to all relevant points of law." Following this they derived seven rhetorical role categories, shown in table 6. The category "other" is not present in the current version of the dataset.

Table 6. Rhetorical Roles Labels of the HOLJ Dataset (from [**HG05**])

Label	Freq.	Description
FACT	862 (8.5%)	The sentence recounts the events or circumstances which gave rise to legal proceedings. E.g. *On analysis the package was found to contain 152 milligrams of heroin at 100% purity.*
PROCEEDINGS	2434 (24%)	The sentence describes legal proceedings taken in the lower courts. E.g. *After hearing much evidence, Her Honour Judge Sander, sitting at Plymouth County Court, made findings of fact on 1 November 2000.*
BACKGROUND	2813 (27.5%)	The sentence is a direct quotation or citation of source of law material. E.g. *Article 5 provides in paragraph 1 that a group of producers may apply for registration ...*
FRAMING	2309 (23%)	The sentence is part of the law lord's argumentation. E.g. *In my opinion, however, the present case cannot be brought within the principle applied by the majority in the Wells case.*
DISPOSAL	935 (9%)	A sentence which either credits or discredits a claim or previous ruling. E.g. *I would allow the appeal and restore the order of the Divisional Court.*
TEXTUAL	768 (7.5%)	A sentence which has to do with the structure of the document or with things unrelated to a case. E.g. *First, I should refer to the facts that have given rise to this litigation.*
OTHER	48 (0.5%)	A sentence which does not fi t any of the above categories. E.g. *Here, as a matter of legal policy, the position seems to me straightforward.*

The HOLJ datasets has been automatically annotated with several linguistic features as Part of Speech (POS) labels, lemmas, Verb and Subject features and recognized Named

Entities. These features combined were already used by Hachey and Grover for sentence classification purpose[5], obtaining a micro-averaged F-score of 51.8 with Naïve Bayes and of 60.6 with SVM (using the Weka package). They also experimented with Decision Trees and Winnow (from Weka) and with a maximum-entropy model, obtaining promising results only with the latter.

It is not mentioned in Hachey and Grover paper in which subset they performed the testing. In a personal communication with Hachey, he informed us that they used 10-fold cross-validation. Given the fact that the collection is clearly separated by the year the original documents were produced, we decided to create separated training and testing sets, the first one containing the sentences from documents from 2001 and 2002, and the latter containing the sentences from documents from 2003. In this way, the training set resulted in 7395 sentences while the testing set contains 4034 sentences. Table 7 summarizes the features of the HOLJ dataset.

Table 7: HOLJ dataset

Corpora	HOLJ
Origin	HOLJ
# examples	11,429

[5] The dataset used for the experiments reported in Hachey and Grover's paper [HG05] is a preliminary version of the one currently available, and it is slightly smaller, containing 10,169 sentences.

4.2. Link Parser

The Link Parser (LP) [ST91] is an implementation of a dependency-based computational grammar. After analyzing a sentence, the LP returns what is called a linkage, a graph where the nodes are the sentence's words, the edges (called links) connect syntactically related words, and the edge's labels express the grammatical relationships between the related words (e.g.: the A link denotes an adjective modifying the head noun on a noun phrase.) The links must not cross and the labels of the links must satisfy the linking constraints specified for each word in the grammar.

A linkage is complete when the graph is connected. In the cases when the LP cannot construct a complete linkage for a sentence, it relaxes the connectedness requirement in order to still provide a partial analysis. In these cases some of the words of the sentence may not belong to the linkage.

Because of the ambiguity of natural language, more than one linkage can often be constructed for a given sentence. These linkages are ordered following heuristic goodness criteria based on the number of words not used, the length of the links, etc. The first given linkage is consider the best one according to the established criteria. An example of two complete linkages for the sentence "The big dog chased the black cat in the park." is shown in Appendix A.

Among other parameters, the LP permits to set the maximum number of linkages allowed and the maximum time in which to construct the full set of linkages (after which it will enter in panic mode and produce a restricted analysis.)

4.3. Aleph Relational Learner

Aleph [S93] is an inductive rule learner system that operates in a FOL representation which gives it the advantage of being able to express relations between the elements of the domain. FOL is used to represent not only the examples but also background knowledge and the learned rules.

Aleph uses a top-down covering algorithm, taking as input background information in the form of predicates, a list of modes declaring how these predicates can be chained together, and a designation of one predicate as the head predicate to be learned. Aleph is able to use a variety of search methods to find good clauses, such as the standard methods of breadth-first search and depth-first search, as well as heuristic methods requiring an evaluation function. We use the default evaluation function *coverage* (the number of positive and negative examples covered by the clause) in our work.

5. Experiments

In this chapter we present the experiments that we have performed together with the results that we have obtained and an analysis of them. Preliminary partial versions of these experiments have been published as either part of the Caderige project publications [OCMNM03] or on their own at CAI 06 [CM06] and RANLP 07 [CM07].

5.1. Preliminary Study of the Usefulness of Statistical Phrases and Noun Phrases for Sentence Selection in the Bacillus Subtilis Dataset

5.1.1. Introduction

In the last ten years a major question in the area of text classification has been whether including statistical or syntactical phrases in the set of attributes could improve or not the performance of the classifiers and query answering systems [CMS01], [L92a], [MBSC97], [CNPB00].

Intuitively, noun phrases would have a more specific meaning than simple words, and therefore would provide a better discriminating power for the classification. However, in categorization problems in large collections of general domain, their use could result in incorrect classifiers due to the sparseness of the text in the attribute space and too specific classifiers that do not generalize for new examples.

Statistical phrases are consecutives words that happen often enough in the collection to be considered relevant to the classification. Because of this statistical property, they cover more documents than rarely used noun phrases, therefore relaxing the sparseness problem. However, they could lack semantic coherence introducing noise in the representation.

Here we analyze the usefulness of noun phrases and statistical phrases for the categorization of sentences from highly technical genomic texts, the Bacillus Subtilis Dataset. Our work consists of identifying sentences from scientific abstracts written in natural language, which can be searched via the Internet using keyword queries. However, as expressed by Ould et al. [OCMNM03], the queries would retrieve a large superset of relevant papers from which we would like to identify the sentences that express an interaction between genes and/or proteins. We automatically learn classifiers that categorize the sentences from the previously mentioned abstracts into two classes: those that describe an interaction between genes and/or proteins and those that do not.

This is a very specific context, in which occurrence (or not) of specialized terms is expected to discriminate between sentences that contain information about a genetic interaction, and those that do not contain that information. For that reason, we intuitively expect noun phrases that represent technical terms to perform well in the categorization task at hand. We hypothesize that in the highly technical text noun phrases will do better than in texts of general interest, e.g. the Reuters corpus [L92a]. We use the statistical phrases and the standard Bag of Words representations as baselines for our comparison.

5.1.2. The Noun Phrases Extractor and the Statistical Phrases Extractor.

The Noun Phrases Extractor system was written in Perl. It assumes that every word in the text from which the noun phrases will be extracted has already been tagged with its part of speech tag. The Penn TreeBank tags returned by the Brill tagger [B93] were used. The noun phrase extractor looks for sequences of tagged words that fulfill functions of determiners (articles, ordinals, etc.), modifiers (adjective, present participle, past participle, noun) and head noun, and returns them indicating for each word the role it fulfills. The sequence of determiners was excluded from the phrases in the present work.

Some of the phrases of length 2 recognized by this noun phrases extractor in this collection are: "sigma factors", "in vivo", "bacillus subtilis", "acid sequence", "binding protein", and "escherichia coli". Not many phrases of length 3 or greater were recognized[6].

The Statistical Phrases Extractor system was also written in Perl. It identified every pair of consecutive words and afterwards filtered out those that did not appear at least 3 times in the dataset. The phrases containing determiners or other stop words were not removed, but it is expected that they will be filtered out by the feature selection process for not being discriminative enough.

Some examples of Statistical Phrases are: "genes in", "proteins we", "sigmab mutant", "sporulation transcription", "of these".

[6] Even when there are some noun phrases of length 3 or greater in the collection, most of them include a prepositional phrase as a modifier, which we are not taking into consideration in this work.

5.1.3. The Experiments and Results

The experiments were done using five different learning algorithms from the Weka package: Naïve Bayes Simple, Support Vector Machine, Decision Tree, Neural Networks, and Nearest Neighbor. The first two are considered the state of the art algorithms for ATC, and they performed better than the others in our experiments. We present here only their results, and since then they have been the algorithms of choice for all our fore coming experiments.

The data was represented in 3 different feature formats (*total number of features indicated in parentheses*):

1. **Words**: all the words after stop words removal (*2329*)
2. **Words + Statistical Phrases**: occurring at least three times (*1908*)
3. **Words + Noun Phrases**: occurring at least twice (*1389*).

The number of features was reduced using Information Gain as a filter and following the method used in [CMS01]: this measure was calculated for each of the features, they were then ranked and a fix number N of features (the N ones with the higher Information Gain) were used on the representation. To set the value of N a preliminary experiment was run on a 66% training and 33% testing random partition, and different number of features were selected in intervals of 100. N was set to 500, as this was the number of features producing the best classification results on this preliminary experiment. It must be noticed that the

feature selection was performed separately for each partition when using cross-validation.[7] This was all done using the Weka system [WF05].

The following table shows the performance values obtained when evaluated using 10 runs of 10-fold cross-validation. The cases where Statistical Phrases or Noun Phrases perform significantly better than Words are shown in bold. The equivalent when the results are significantly worse is shown in italics.

Table 8: Performance of Statistical Phrases and Noun Phrases

Learning Algorithm	Measure	Words	Statistical Phrases	Noun Phrases
Naïve Bayes Simple	Accuracy	0.71	0.71	0.72
	Precision	0.70	**0.76**	0.71
	Recall	0.74	*0.63*	0.75
	F1	0.72	0.70	0.73
Support Vector Machine	Accuracy	0.70	0.67	0.71
	Precision	0.70	0.68	0.71
	Recall	0.69	0.67	0.71
	F1	0.70	0.68	0.71

We can see from the results that the use of noun phrases in this particular context of a highly specialized technical genomic corpus seems to be worthwhile. Even when the differences are not statistically significant, the use of the syntactic information in the representation shows an increase of the performance with respect to using only words. This increase is observed for all the measures that we evaluated, and when using either of the two

[7] This is an important practice as the Information Gain measure takes into account the class of the examples. If the feature selection would have instead been performed in the whole collection and only after that the dataset partitioned for the N-fold cross-validation, the class information from the testing partitions would have been taken into account.

algorithms. This seems to support our theory that features with a stronger semantic meaning (brought in this case by means of the noun phrase syntactic relation) would be helpful in our particular task.

On the other hand, the use of statistical phrases showed a statistically significant increase only in the precision when using the Naïve Bayes algorithm. This increase in normally expected when using any type of phrase, as they are essentially more specific than single words. Since this increase was in detriment of the recall, the accuracy and F1-measure were not significantly affected. According to the SVM algorithm all the values when using the statistical phrases representation were lower than when using the simpler BOW representation.

5.2. Syntactic Text Representation for Biological Sentence Classification

5.2.1. Introduction

According to our preliminary results, we would like to study whether further enrichment of the text representation would be more beneficial for the classification. We study the usefulness of including syntactic and semantic knowledge in the text representation by adding pairs of related words obtained from a syntactical parser (to which we will refer as syntactic bi-grams) together with technically related dictionaries. The noun phrases with which we experimented in the previous section are a particular case of syntactic bigrams.

We expect syntactic bi-grams to provide detailed information on whether two genes and/or proteins are interacting with each other. Such phrases could be formed for example

by the main noun in the subject or object role of a sentence together with its verb, or the main noun in a prepositional phrase together with either the noun or verb it modifies. Examples of these are the sub-sentences "the expression of spo0h encodes sigmah", "both sigk and gere were essential for ykvp expression" and "the addition of clpx to in vitro transcription reactions resulted in the stimulation of rnap holoenzyme activity".

Using the syntactic bi-grams together with their single words, we represented the sentences and we evaluated the classification performance of this representation compared to the bag of words. Once again, our Experiments include state of the art machine learning algorithms Naïve Bayes and Support Vector Machine, and they were performed using Weka.

We further enrich the representation by introducing some more semantic knowledge to help with the specific vocabulary. A list of proteins and genes was extracted from the SwissProt Protein Knowledgebase[8]. The words found on this list were replaced in our representation by a lexical marker (the word "`geneprot`"). The list of interactions used in Temkin and Gilder [TG03] work was included as facts in one of the experiments with the relational representation.

[8] Swiss-Prot is an annotated protein sequence database available on-line at http://ca.expasy.org/sprot/. Among all the information provided is a "Short description of entries in Swiss-Prot" from which we extracted the names of proteins and genes.

5.2.2. The Bigrams Representation

In this section we present an example of the analysis performed by the Link Parser, the links it recognized in our collections and how they are used in the text representation.

In order to create a syntactic representation we performed the following steps:

1. We ran the Link parser on each sentence of the data collection[9].
2. We identified some syntactic links, such as the object of a verb.
3. We built syntactic bi-grams with the linked words

Out of the many links identified by the parser, we only took into consideration those ones that we believed could help enrich our representation. These are:

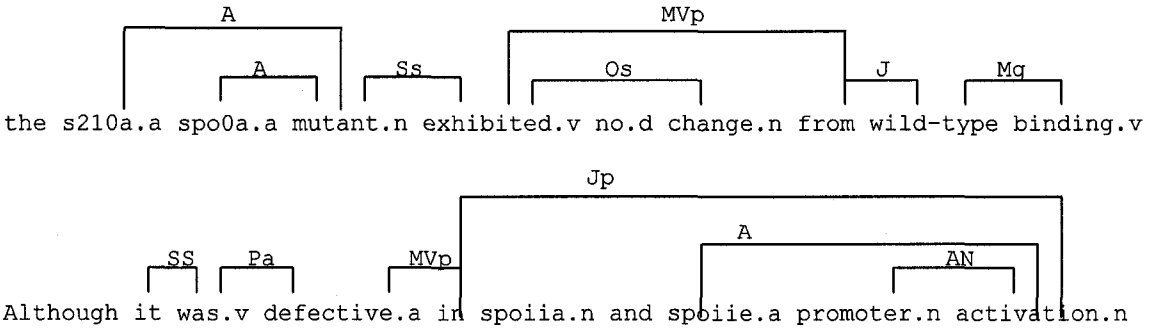
- A and AN: link an adjective or a noun (respectively) to the noun it modifies.
- Ss: links the head of a noun phrase to the verb to which the phrase is the subject.
- Os: links the head of a noun phrase to the verb to which the phrase is the object.
- Pa: links forms of the verb "have" to a participle verb.
- Mg: links nouns with present participles
- MVp and J (or Jp): MVp links the verb to a preposition at the beginning of a propositional phrase, and J (or Jp) links that preposition to the noun that is head of the noun phrase inside the prepositional phrase. We established the M relation, which links the verb in a MVp to the noun in a corresponding Js.

[9] The Link parser returns for each given phrase several different links sets, and a cost vector value associated to each of them. Only the highest ranked links set was used in these experiments.

Example 1 shows all the links we identified among the set of links returned by the Link Parser for the first sentence of our collection.

Example 1: Links identified for the first sentence of our collection

The s210a spo0a mutant exhibited no change from wild-type binding, although it was defective in #spoiia# and #spoiie# promoter activation.



From the analysis shown in example 1, the following syntactic bi-grams could be built:

- Noun phrases (A and AN links): spo0a_mutant, s210a_mutant, spoiie_activation, promoter_activation
- Subject (Ss link): mutant_exhibited, it_was
- Object (Os link): exhibited_change
- Prepositional phrases (MVp-J/Js links): exhibited_wild-type, defective_activation
- Others: wild-type_binding, was_defective

These are all the syntactic bi-grams we built. In the following experiments only some of them were used at a time, according to the kind of link we were permitting (e.g. when representing noun phrases only the A and AN links were permitted). The type of the link and the morphological information (as which words are nouns, adjectives and verbs, which is also provided by the Link Parser) were not included in the representation. We are planning to include this distinction in our future work.

Some of the previous syntactic bi-grams were modified because they contained a gene or protein name from the SwissProt list. Thus `s210a_mutant` was replaced by `geneprot_mutant`. Unfortunately `spo0a` and `spoiie` were not found in the list and therefore not replaced by our lexical marker.

It must be noticed that for few of the examples[10] the parser either ran out of time (in which case a time extension was given) or was not able to produce a parse tree of the sentence for some other reason. In those cases there are no bi-grams associated with the example and its representation is always the same as the basic BOW representation.

5.2.3. Experiments

We introduce here the experiments that we performed using the previously described syntactic bi-grams and present the results we obtained.

[10] Around 5% in the BS dataset, 1% in the YPD and 10% in the HOLJ.

As a baseline we use the bag of words representation. We compare its performance with the one obtained when using the words together with some or all of the recognized syntactic bi-grams. We differentiate the Noun Phrases representation (that including only the A and AN links from the Link Parser in order to recognize a syntactic phrase), the Subject and Object representation (only Ss and Os links), the Prepositional Phrases representation (only the M links) and the representation using all the links together.

As in our preliminary results, here as well 500 out of the total of features were selected using the information gain metric. This filter kept some of the syntactic bi-grams among the most discriminating features, confirming the usefulness of them for the classification task, as discussed by Caropreso et al. [CMS01]. Among the selected syntactic bi-grams, several include our lexical marker, even when by itself it was not selected as discriminating when using the BOW representation. Some examples of these syntactic bi-grams are `promoter-geneprot`, `geneprot-protein`, `mutant-geneprot`, `binds-geneprot`, `encodes-geneprot`. Some examples of other syntactic bi-grams are `subtilis-bacillus`, `indicated-analysis`, `protein-encodes`, `sequence-amino`, `transcription-stimulates`.

Table 9 shows the Accuracy, Precision, Recall and F1-measure obtained by the Naïve Bayes Simple and the Support Vector Machine learners of the Weka Package. The experiments were performed using the default parameters for each learner and the number of features that resulted in the best accuracy in preliminary experiments.

Table 9: Performance of different types of syntactic bigrams.

Learning Algorithm	Performance measure	Words	Noun Phrases Bi-grams	Subject and Object Bi-grams	Prepositional Phrases Bi-grams	All the syntactic Bi-grams
Naïve Bayes 500 features	Accuracy	0.66	0.66	0.67	0.66	0.68
	Precision	0.61	0.61	0.62	0.61	0.62
	Recall	0.90	0.90	0.88	0.91	0.89
	F1-measure	0.73	0.73	0.73	0.73	0.73
SVM 500 features	Accuracy	0.68	0.68	0.68	0.67	0.69
	Precision	0.68	0.68	0.67	0.67	0.70
	Recall	0.67	0.69	0.68	0.67	0.67
	F1-measure	0.67	0.67	0.68	0.67	0.69

The results do not show statistically significant difference with respect to the BOW when using noun phrases bi-grams, subject/object bi-grams or prepositional phrases bi-grams to enrich the BOW representation. However, the best results for both classifiers were obtained when using all the syntactic bi-grams together. That indicates that at least two kinds of the used syntactic bi-grams are relevant to the classification when combined.

We also performed preliminary experiments using a decision tree learner from the Weka Package. The pattern of increased accuracy when using all the syntactic bi-grams holds, but the values were only in the 60% range. This low performance (considering that we are working with a balanced dataset) was already noticed by Ould [O05] when also using

a decision tree learner, and who argued that it might be a direct result of the sparseness of the collection.

Once again, morphological information and the information on what kind of link relates the involved words were not included in the representation. This is a limitation of our present work, and we are planning to include this distinction in our future work.

5.2.5. Discussion

We have presented the problem of sentence selection of a genetic corpus and how we envisioned the contribution of semantic and syntactic knowledge in this task.

We introduced basic semantic knowledge in the representation by replacing the words found in a list of genes/proteins. Syntactic knowledge was also incorporated to the representation by mean of syntactic relations. This was accomplished extending the set of features with bi-grams obtained from a syntactic parser. We have empirically shown that this knowledge is useful for sentence selection of this genetic corpus when using several different machine learning methods.

5.3. Semantic Text Representation for Biological Sentence Classification

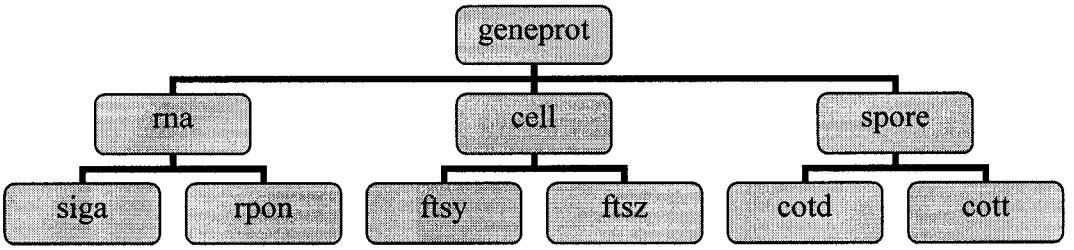
In this section we study different ways of incorporating hierarchical semantic information into the sentences representation. We start from the basic BOW representation and we then add the syntactic features as presented in the previous section. In these two representations we then try different alternatives for incorporating the hierarchical knowledge, as presented in the second sub-section.

5.3.1. Hierarchical representation

While the syntactic representation goes some way towards producing a richer task representation, it lacks additional semantic knowledge. For this we turn to one of the several hierarchical knowledge bases available for our domain (eg. GeneOntology, SwissProt.) In this way, our enriched representation on one hand generalizes with respect to the BOW representation, and on the other hand enriches the representation semantically, which to some extent should alleviate the sparseness problem.

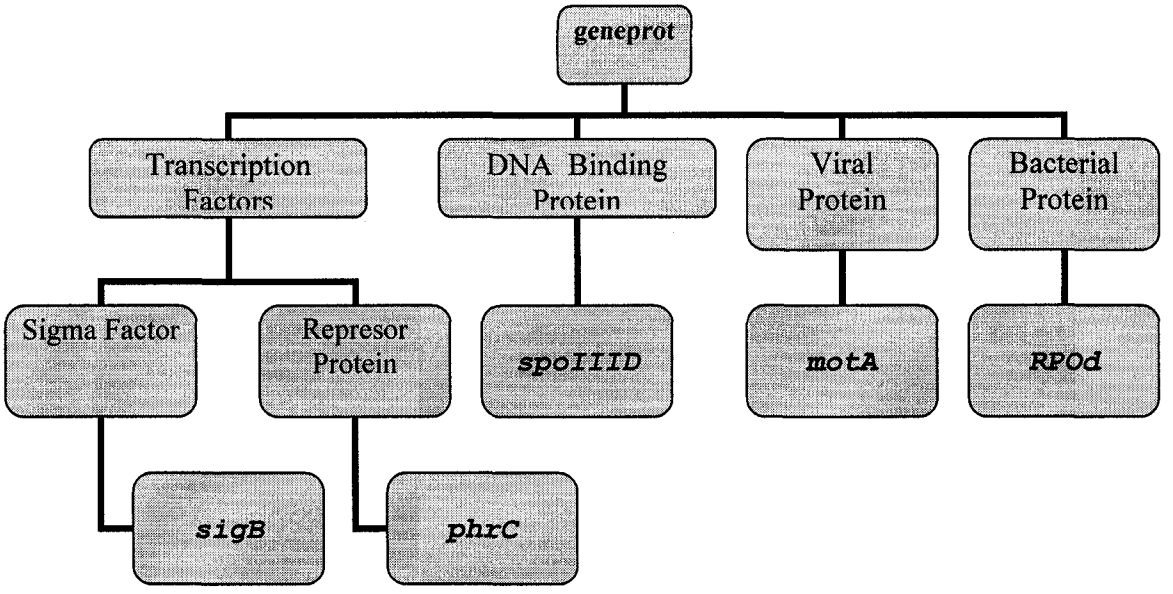
The first hierarchy we used was created from information contained in the Swiss Prot KnowledgeBase. It consists of a three-level tree, the leaves being the gene or protein names and the root the generic term "`geneprot`". The intermediate level of the hierarchy is a classification of the gene or protein presented in the database.

Example 2: Extract from the hierarchy created from SwissProt.



The second hierarchy we used was created from information extracted from Mesh[11]. This time it consists of a 3 or 4 levels tree (depending on the case), again the leaves being the gene or protein names and the root the generic term "geneprot". The intermediate level or levels of the hierarchy is a classification of the gene or protein presented in the database, and described in example 3.

Example 3: Extract from the hierarchy created from Mesh.



For our experiments we generated different representations using these hierarchies, replacing a word in the BOW with the root of the hierarchy as in the previous experiments, or adding words from the hierarchy into the representation. We then compared their performance with the ones obtained when using the BOW (the basic representation using the gene or protein names) with and without the syntactic information. The new representations are: a) the representation created by replacing the gene or protein names with the root of the

[11] http://www.nlm.nih.gov/mesh/meshhome.html

hierarchy, the generic term "geneprot", b) the representation created by adding (instead of replacing) the root of the hierarchy ("geneprot") for each gene or protein name, c) the representation created by adding the first ancestors of the gene or protein name, and d) the representation created by adding all the ancestors of the gene or protein name and the root ("geneprot").

We now show the differences in all the representations when using the hierarchy from Swiss Prot for the following sub-sentence: "we isolated a temperature-sensitive sporulation defective mutant of the siga gene..." Considering that according to our hierarchy the gene/protein name "siga" has as ancestor "rna", and that the root of the hierarchy is the generic word "geneprot", the following representations will result. (In parenthesis we display the names given to each representation and which will be used later when we present the results.)

Basic representation using the gene or protein names (names):

words: we, isolated, temperaturesensitive, sporulation, defective, mutant, *siga*, gene.

links: we_isolated, mutant_isolated, mutant_temperaturesensitive, mutant_sporulation, mutant_defective, gene_mutant, gene_*siga*.

Representation replacing the gene or protein names by the top of the hierarchy (repl_root):

words: we, isolated, temperaturesensitive, sporulation, defective, mutant, *geneprot*, gene.

links: we_isolated, mutant_isolated, mutant_temperaturesensitive, mutant_sporulation, mutant_defective, gene_mutant, gene_*geneprot*.

Representation adding the generic term (add_root):

words: we, isolated, temperaturesensitive, sporulation, defective, mutant, *siga*, *geneprot*, gene.

links: we_isolated, mutant_isolated, mutant_temperaturesensitive, mutant_sporulation, mutant_defective, gene_mutant, gene_*siga*, gene_*geneprot*.

Representation adding the gene or protein name ancestor (add_anc):

words: we, isolated, temperaturesensitive, sporulation, defective, mutant, *siga*, rna, gene.

links: we_isolated, mutant_isolated, mutant_temperaturesensitive, mutant_sporulation, mutant_defective, gene_mutant, gene_*siga*, gene_*rna*

Representation adding the ancestors of the gene or protein names and the root of the hierarchy (add_both):

words: we, isolated, temperaturesensitive, sporulation, defective, mutant, *siga*, *rna*, *geneprot*, gene.

links: we_isolated, mutant_isolated, mutant_temperaturesensitive, mutant_sporulation, mutant_defective, gene_mutant, gene_*siga*, gene_*rna*, gene_*geneprot*

5.3.2. Experiments and results

In this section we present the experiments we performed using the machine learning algorithms Naive Bayes (NB) and Support Vector Machine (SVM) from the Weka package and both hierarchies.

As a baseline we use the basic representation, the BOW. We compare its performance with all our alternative representations, first considering only information from the hierarchical dictionary with its different variations in the representation, and then all the same variations including the recognized syntactic bi-grams.

For the classification, a subset of the total set of features was selected using the information gain metric. This filter kept some of the syntactic bi-grams among the most discriminating features, giving a first confirmation of their usefulness for the classification task [2]. Among the selected terms, we also find the root of our hierarchy "`geneprot`", both by itself and as part of syntactic bi-grams, as well as some of the first level ancestors as "`cg`", "`atp`", and "`arsenical`". Some examples of these syntactic bi-grams are `geneprot-protein`, `mutant-geneprot`, `encodes-geneprot`, `arsenical-arsr`, `gene-hypothetical`. Some examples of other syntactic bi-grams also kept after filtering are `bacillus-subtilis`, `indicated-analysis`, `protein-encodes`.

Tables 10 and 11 respectively show the Accuracy, Precision, Recall and F1-measure obtained by the Naïve Bayes Simple and the Support Vector Machine learners of the Weka Package. The experiments were performed using the default parameters for each learner and the number of features that resulted in the best accuracy in preliminary experiments. The values shown are the average of 10 runs of 10-fold cross-validation. The results in bold denote a statistically significant increase over the basic BOW representation

(first column in the 'words' section), while the use of italic denotes a statistically significant decrease.

Table 10: Performance of the syntactic features and the hierarchical information (Naïve Bayes)

WORDS					
	names	**repl_root**	**add_root**	**add_anc.**	**add_both**
Percent correct	0.66	0.66	0.68	0.69	**0.70**
Precision	0.61	0.61	0.62	0.64	**0.65**
Recall	0.90	0.90	0.90	*0.87*	0.88
F measure	0.73	0.73	0.73	0.74	**0.75**
LINKS					
	names	**repl_root**	**add_root**	**add_anc**	**add_both**
Percent correct	**0.70**	0.68	0.70	0.69	**0.70**
Precision	0.63	0.62	0.64	0.64	0.64
Recall	**0.93**	0.89	*0.87*	*0.87*	0.90
F measure	**0.75**	0.73	0.74	0.74	**0.75**

Table 11: Performance of the syntactic features and the hierarchical information (SVM)

WORDS					
	names	**repl_root**	**add_root**	**add_anc**	**add_both**
Percent correct	0.70	0.68	0.70	0.69	0.68
Precision	0.70	0.68	0.71	0.69	0.68
Recall	0.69	0.67	0.68	0.68	0.66
F measure	0.70	0.67	0.69	0.68	0.67
LINKS					
	names	**repl_root**	**add_root**	**add_anc**	**add_both**
Percent correct	0.72	0.69	0.73	0.70	0.70
Precision	0.72	0.70	0.73	0.71	0.71
Recall	0.71	0.67	0.71	0.69	0.68
F measure	0.71	0.69	0.72	0.70	0.69

Our first observation is that the replacement of the gene or protein name by the generic word "geneprot" hurts the performance in most cases and according to all the used measures. The exception is when using Naive Bayes without the syntactic information, in which case all the values remained the same as those obtained with the BOW representation.

The addition (instead of replacement) of the hierarchical information to the representation improves the performance of the classifiers with respect to the BOW.

The best accuracy and F1 measure values with the NB classifier are obtained with either of the following representation:

1. the representation including both levels of the hierarchical dictionary extracted from Swiss Prot (WORDS / add_both),

2. the representation including all the syntactic relations without any hierarchical information (LINKS / names),
3. the representation including both the syntactic relations and both levels of the hierarchical dictionary extracted from Swiss Prot, (LINKS / add_both).

The accuracy and F1 measure values in all the previous cases is statistically significant better than those of the BOW representation.

With SVM the best results were obtained with the following two representations:

1. the representation including all the syntactic relations without any hierarchical information (LINKS / names),
2. the representation including both the syntactic relations and the highest level of the hierarchical dictionary extracted from Swiss Prot, (LINKS / add_root).

We also compared the performance when using the hierarchy obtained from Mesh. In tables 12 and 13 we repeat the values for the basic BOW representation and the ones obtained when adding both levels of the SwissProt based hierarchy showed in the previous tables. The new values in these tables are found in the last column and they correspond to the values obtained when adding in the representation the information coming from all levels of the Mesh based hierarchy.

Table 12: Performance of the hierarchical information from Mesh (Naïve Bayes)

WORDS			
	names	**add_both Swiss Prot**	**add_all MESH**
Percent correct	0.66	0.70	0.67
Precision	0.61	0.65	0.61
Recall	0.90	0.88	0.90
F measure	0.73	0.75	0.73
LINKS			
	names	**add_both Swiss Prot**	**add_all MESH**
Percent correct	0.70	0.70	0.66
Precision	0.63	0.64	0.61
Recall	0.93	0.90	0.90
F measure	0.75	0.75	0.73

Table 13: Performance of the hierarchical information from Mesh (SVM)

WORDS			
	names	**add_both Swiss Prot**	**add_all MESH**
Percent correct	0.70	0.68	0.69
Precision	0.70	0.68	0.70
Recall	0.69	0.66	0.68
F measure	0.70	0.67	0.69
LINKS			
	names	**add_both Swiss Prot**	**add_all MESH**
Percent correct	0.72	0.70	0.67
Precision	0.72	0.71	0.68
Recall	0.71	0.68	0.66
F measure	0.71	0.69	0.67

It can be observed in these tables that the use of the Mesh-based hierarchy in the representation did not bring any improvement in the performance of either classifier, and therefore further experiments with this hierarchy were discontinued.

5.4. Study of the performance on a time split and of the effects of the Threshold

5.4.1. Introduction

Using a time split is a protocol which more realistically evaluates the real use of the system when a classifier will be learned on instances available prior to the time of training and used to predict the class of new examples as they become available.

It is known that, over time, there is often a concept drift in the documents of a collection. In these cases, if N-fold cross-validation is used, it will not be sensitive to the effects of concept drift, because training and testing instances are spread over the entire time axis. It is therefore expected to obtain over optimistic results.

In this section, we use this approach to evaluate our representations, taking advantage of the knowledge that the sentences are ordered according to the date when the abstracts they belong to were submitted to PubMed[12].

5.4.2. Experiments

Tables 14 and 15 show the Accuracy, Precision, Recall and F1-measure obtained respectively by the Naïve Bayes and SVM algorithms in a time related training/test split.

[12] PubMed abstracts are identified by a unique number given in ascending order at the time they are submitted.

The 60% of the sentences, which originally were part of the earlier abstracts, are used as training, while the remaining ones from the latest abstracts are used as testing.

We first observe that the use of the syntactic features importantly improved (by around 5%) the accuracy of both classifiers, Naïve Bayes and SVM (tables 14 and 15 respectively), while only slightly improving the F1 measure in most cases.

The addition or replacement of the hierarchical information to the representation does not consistently affect the performance of the classifiers. However, the best accuracy and F1 measure, respectevely 0.72 and 0.70, are obtained with the representation including the links and the top level of the hierarchical dictionary (the representation referred as add_root in the tables) when learning a Naïve Bayes classifier (table 14).

Table 14: Time Split performance of the syntactic features and the hierarchical information (Naïve Bayes)

WORDS (BOW + hierarchical information)					
	names	repl_root	add_root	add_anc	add_both
Accuracy	0.65	0.64	0.64	0.65	0.66
Precision	0.57	0.56	0.56	0.58	0.57
Recall	0.86	0.84	0.86	0.87	0.87
F1	0.68	0.67	0.67	0.69	0.69
LINKS (BOW + syntactic + hierarchical information)					
	names	repl_root	add_root	add_anc	add_both
Accuracy	0.71	0.69	0.72	0.71	0.70
Precision	0.66	0.63	0.66	0.64	0.63
Recall	0.72	0.73	0.74	0.73	0.75
F1	0.69	0.68	0.70	0.68	0.69

Table 15: Time Split performance of the syntactic features and the hierarchical information (SVM)

WORDS (BOW + hierarchical information)					
	names	repl_root	add_root	add_anc	add_both
Accuracy	0.59	0.62	0.62	0.60	0.60
Precision	0.53	0.55	0.55	0.53	0.53
Recall	0.70	0.69	0.71	0.69	0.69
F1	0.60	0.61	0.62	0.60	0.60
LINKS (BOW + syntactic + hierarchical information)					
	names	repl_root	add_root	add_anc	add_both
Accuracy	0.65	0.62	0.65	0.64	0.64
Precision	0.59	0.55	0.59	0.58	0.58
Recall	0.69	0.69	0.69	0.64	0.64
F1	0.64	0.61	0.63	0.61	0.61

The correlation of the precision and recall measures is evident in Table 14. While the precision increases in the "Links" part of the table, the recall decreases in a similar proportion. This is why usually the F-measure, which is a weighted average of both Precision and Recall, is presented together with them. In our case we presented the F1 measure, which gives equal weight to both. In order to present another measure that relates the Precision and Recall values, we calculated the breakeven point, which is the value where precision and recall are equal, and it can be obtained by changing the classifier threshold. We did that with the NB algorithm and we found the BOW representation to have the lowest breakeven point at 0.64, while by adding the syntactic information, with or without the

hierarchical information, it was 0.67, and the higher (and best) breakeven point of 0.69 was reached when adding only the hierarchical information.

To better understand the effects of the variations of the threshold in the Naïve Bayes algorithm, we show in Fig. 1 the different values of the F1 measure for a simplified 11 points threshold curve in the training/test split. We only show the representations on the first and last columns (names and add_both) of table 14, with and without the syntactic information, called here Words, Links, Words_Hier and Links_Hier respectively.

We can observe that while the F1 measure is similar for the four representations at the 0.5 threshold (the default threshold used by the algorithm), it presents considerable variations for other values of the threshold. In particular, for thresholds lower than 0.5 (which implies higher recall, and is shown on the left side of the graph in Fig. 1,) the two representations that make use of the syntactical information (Links and Links_Hier) yield higher F1 measure values than the two representations that do not consider the syntactic information, and vice-versa.

In Fig. 2 we present the Precision/Recall curves for the same four representations as in figure 1. These curves confirm the observation that at high levels of recall, and in particular for recall values over 0.80, the representations that consider the syntactic information perform better than the ones that do not.

Fig. 2 also shows the differences in the breakeven point values mentioned before (observe the graph around the point 0.7 for both precision and recall). It is clear from the graph that the representation that takes into consideration the hierarchical semantic information (Hier) results in a higher precision not only at the breakeven point but on the whole interval of recall between the values of 0.4 and 0.8. At levels of recall lower than 0.4,

the representations containing the semantic information obtain lower precision than the ones that do not (the Words and Links on their own.)

Figure 1

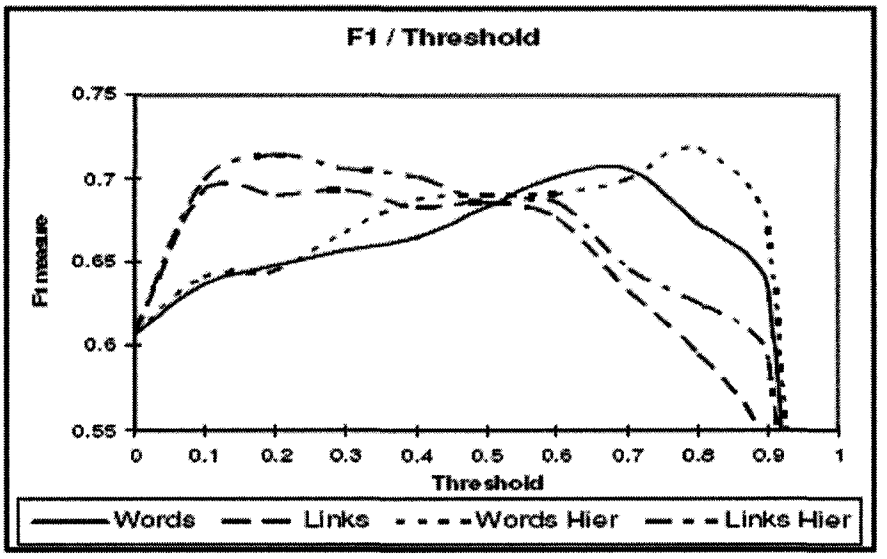
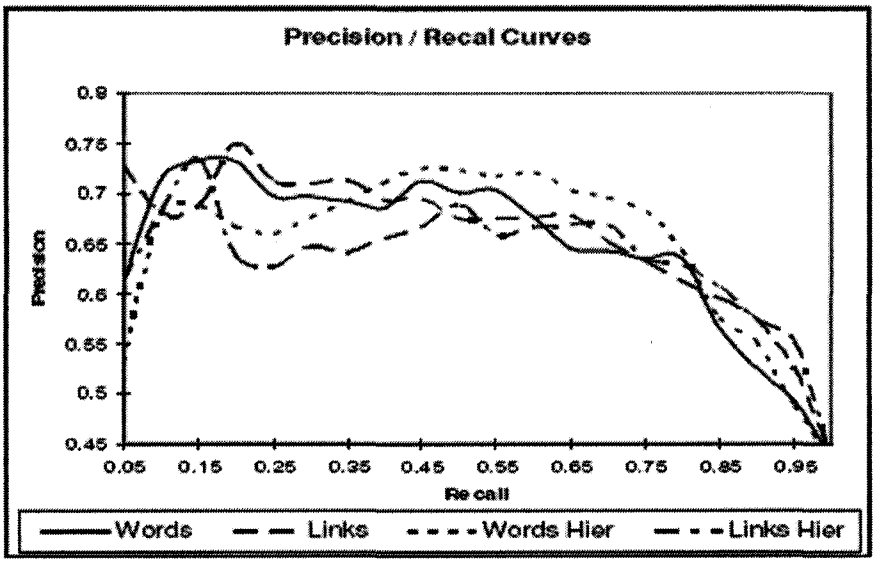


Figure 2



5.4.3. Discussion

In a time based train/test split, we found the basic representation with only words to work well enough when relatively low values of recall are accepted. Adding the hierarchical semantic information brought the performance up for medium to high values of recall. When the highest values of recall are required, the representations that add the syntactic information to either of the previous ones perform the best.

5.5. Exploring the Relational Nature of the Data

5.5.1. Relational representation

As noted in the previous experiments, the syntactic relation between some words in a sentence seems to be relevant to the sentence selection task we are performing. It is natural then to think of a relational representation that can capture these relations. This new representation can then be used by a relational learner system, providing the advantages of this kind of systems as discussed by Furnkranz [F94]. Among others, predicates to help the classification can be easily defined, and relations among three words could be discovered (as two bi-grams with a transitive relation).

In order to obtain this relational representation, the same links obtained for the syntactical representation were used and relations were built between the linked words. The information contained in the hierarchical dictionary is also included in this representation.

Given the same sentence as in 3.1., the following relations could be built:

```
links(d5,isolated,we),
links(d5,mutant,isolated),
link(d5,mutant,temperaturesensitive),
link(d5,mutant,sporulation),
link(d5,mutant,defective),
link(d5,gene,mutant),
link(d5,gene,siga),
link(d5,gene,rna),
link(d5,gene,geneprot).
```

The last 3 relations are only used when appropriate: only `link(d5,gene,siga)` in the basic representation with the gene or proteins names, only `link(d5,gene,geneprot)` in the representation replacing the gene or protein names, and the three of them in the representation adding all the ancestors (we did not try in the relational experiments the representation that adds only one level of the hierarchy.)

5.5.2. Relational Experiments

The relational representation's performance was evaluated using the relational learner Aleph [S93]. In this section we first introduce the experiments performed and we then show an example of the rules learned and analyze them. Finally we present and analyze the classification results.

The relational representation we presented before introduced the syntactic relations of the words in the sentences. This representation was compared in our experiments with a

baseline using propositional logic denoting whether a word occurs or not in a sentence but missing the information of the relations between words, which is equivalent to the BOW.

Instead of physically creating the propositional representation, we simulated it by defining the predicate *lexexist* that represents the presence of a word in a particular sentence and the fact that the word is involved in a link (some words, as for example the articles, were not included in any of the considered links).

```
lexexist(S,W) :- link(S,W,_).
lexexist(S,W) :- link(S,_,W).
```

We also compared the previous relational representation performance with the one obtained when adding extra background knowledge. For this purpose, we added to the representation the list of words denoting interactions presented by Temkin and Gilder [TG03][13]. Each of the words in the list was given as a parameter of the fact *interaction*. The predicate *interacts* was defined representing that a particular word in a sentence is linked to an interaction word.

```
interaction(initiate).
interaction(stimulate).
interaction(upregulate).
...
interacts(S,W) :- link(S,W,I), interaction(I).
interacts(S,W) :- link(S,I,W), interaction(I).
```

[13] Temkin and Gilder [TG03] present a list of 27 verbs and their conjugation variations. They denote interactions of 15 different types, as for example activate, break bond and react. In the present work this hierarchy was flatten and all of the verbs were considered as denoting an 'interaction'. In the future the hierarchy could be used with its three levels.

In this way we performed three different experiments by providing Aleph with only one file containing the genomic information in a relational representation and by instructing it on what kind of rules could be learned.

In examples 4, 5 and 6 we present and analyze a few rules learned by the experiments we described above.

Example 4: Some rules learned by Aleph when allowing only words

```
[Rule1]pos(S):-lexexist(S,sigmak).[Pos=29 Neg=1]
[Rule2]pos(S):-lexexist(S,_expression),lexexist(S,fusion).[Pos=16 Neg=0]
[Rule3]pos(S):-lexexist(S,geneprot),lexexist(S,vivo).[Pos=15 Neg=1]
```

By allowing only words (with the use of only the predicate 'lexexist') we obtain a representation equivalent to both the BOW and the rules obtained from a propositional learner.

Our first observation is that few training examples are covered by each rule, the first one covering 29 examples and thereafter dropping to around 15 examples (approximately 2% of the training examples.)

The second observation is that "`geneprot`" is already chosen in the third rule, marking the usefulness of having replaced the gene and protein names by this lexical marker. However, the gene/protein *sigmak,* which was not found in the Swissprot list and therefore was not replaced, seems to be very discriminating in this dataset. This gives us the hint that we might want to consider different levels in a hierarchy of genes/proteins instead of a single leveled list. We will consider this alternative in our future work.

Example 5: Some rules learned by Aleph when allowing words and syntactic links

```
[Rule1]pos(S):-lexexist(S,sigmak).[Pos=29 Neg=1]
[Rule2]pos(S):-lexexist(S,_expression),lexexist(S,fusion).[Pos=16 Neg=0]
[Rule3]pos(S):-lexexist(S,geneprot),lexexist(S,vivo).[Pos=15 Neg=1]
[Rule4]pos(S):-link(S,processing,B).[Pos=14 Neg=1]
[Rule5]pos(S):-link(S,B,geneprot),link(S,B,bdependent).[Pos=14 Neg=1]
[Rule6]pos(S):-link(S,geneprot,transcription),link(S,is,B).[Pos=13 Neg=0]
...
[Rule12]pos(S):-link(S,geneprot,protein).[Pos=11 Neg=1]
[Rule13]pos(S):-link(S,geneprot,gene).[Pos=10 Neg=1]
```

We observe here that the first three rules are not affected by the introduction of the links, but starting on rule 4 the links help discovering more discriminating rules. In this way, the first three rules are still equivalent to rules produced by a propositional learner, but starting on rule 4 we really take advantage of the relational power of the learner, the representation and this particular classification problem.

In rule 4 we notice that it is not always a pair of words that is important, but as in this case, the fact that the word *processing* is linked to another one makes it discriminating of the class. In rule 5 there is a word linked to both *geneprot* and *bdependent*, being the double link relevant for the rule to be discriminating. And in rule 6 we find the pair *geneprot_transcription* as a discriminating feature. Rules 12 and 13 show the pairs *geneprot_protein* and *geneprot_gene* (a protein or gene name with its type) as the most discriminating after various other rules have been applied.

Some more rules learned when allowing words and syntactic links together with a list of interactions given as background knowledge and the predicate *interacts* defined:

Example 6: Some rules learned by Aleph when allowing words, syntactic links and interactions

```
[Rule1]pos(S):-lexexist(S,sigmak).[Pos=29 Neg=1]
...
[Rule5]pos(S):-link(S,B,geneprot),link(S,B,bdependent).[Pos=14 Neg=1]
[Rule6]pos(S):-link(S,geneprot,transcription),link(S,is,B).[Pos=13 Neg=0]
[Rule7]pos(S):-link(S,show,B),interacts(S,C).[Pos=12 Neg=1]
...
[Rule17]pos(S):-interacts(S,transcription),interacts(S,geneprot).[Pos=11
Neg=0]
```

The predicate *interacts* is used as early as in rule 7 where it establishes that there is a word linked to an interaction term. In rule 17 it establishes that both *transcription* and *geneprot* are linked to an interaction word (potentially different ones.)

We used the following different representations: the basic representation equivalent to the BOW (names), the representation obtained by replacing each gene or protein name by the root of our hierarchy (repl_root), and the representation that adds both levels of the hierarchy for each gene or protein name (add_both), each of them for either a propositional representation using only words or the relational representation including both words and links.

When running the experiments, an average of 90 rules were learned. They were used without any pruning on the test sets. The average Accuracy, Precision, Recall and F1-measure obtained in the time split and after 10 runs of 10-fold cross-validation for the experiments are shown in table 16 and 17 respectevelly.

Table 16: Performance of Aleph in the time split

WORDS				
	names	**repl_root**	**add_both**	
Percent correct	0.62	0.63	0.62	
Precision	0.58	0.60	0.56	
Recall	0.36	0.37	0.40	
F1 measure	0.44	0.45	0.47	
LINKS				
	names	**repl_root**	**add_both**	**repl_root + interactions**
Percent correct	0.62	0.64	0.61	0.63
Precision	0.56	0.60	0.55	0.58
Recall	0.46	0.46	0.48	0.44
F1 measure	0.50	0.52	0.51	0.50

Table 17: Performance of Aleph using 10-fold cross-validation

WORDS				
	names	**repl_root**	**add_both**	
Percent correct	0.62	0.61	**0.64**	
Precision	0.66	0.66	**0.69**	
Recall	0.47	0.46	**0.52**	
F1 measure	0.55	0.54	**0.59**	
LINKS				
	names	**repl_root**	**add_both**	**repl_root + interactions**
Percent correct	0.63	**0.64**	**0.64**	0.63
Precision	0.67	**0.70**	**0.69**	0.68
Recall	0.48	**0.50**	**0.52**	**0.51**
F1 measure	0.56	**0.58**	**0.59**	**0.58**

We first observe that the accuracy on the time split experiments is comparable to the accuracy obtained with Naïve Bayes and SVM on the same split.

When performing cross validation, Aleph did not seem to overestimate the accuracy, and therefore these results were lower than the ones obtained by the other algorithms when performing cross validation. (Aleph did however overestimate the precision, recall and F1 measures with the cross validation approach, showing that more true positive examples were discovered.)

While the precision was in most cases (both in the time split and using cross validation) comparable with the precision obtained by the other algorithms, the more conservative approach of Aleph obtained very low recall, and therefore lower F1 measure.

In the time split, Aleph showed an increase of the recall when extending the representation with either the syntactic or semantic information, or both of them.

When using cross validation, no improvement was observed by the syntactically enriched representation on its own (this might be shaded by the already overestimated values.)

Importantly, statistically significant improvements on the four considered measures were obtained with cross validation when enriching the representation either with semantic knowledge or with both syntactic and semantic knowledge together.

Finally we also observe an increase in the performance when adding extra semantic background knowledge in the representation by mean of the interactions list (the last column

in tables 16 and 17). In the future we plan to extend this knowledge to include a sub-categorization of the interaction verbs.

We have observed that the use of the syntactically and semantically enriched representation with Aleph helps improve the performance by taking advantage of the flexibility of the relations. The relational representation lets the learner choose what are the important parts of a relation (i.e. whether only one or both words should be fixed), making them more flexible than the pre-fixed phrases used in the syntactic representation for the other learners. The relational representation also gives the opportunity of bridging two non-linked words by the mean of a third one linked to both (see rule 5 in the examples).

5.6. Generalization on other Sentence Selection datasets

In order to generalize the previous results, we run experiments on the other two datasets introduced in chapter 3.

5.6.1. Experiments on the YPD dataset

As introduced in section 4.1, this dataset is similar in principle to the BS dataset. The sentences also belong to abstracts of scientific documents and there is also a relationship denoted in the positive examples. Instead of an interaction between genes, this time the relationship is about the location of a protein inside the cells. Another difference is that this dataset is about 10 times bigger than the previous one, and is not a balanced one having an approximate ratio of 10:1 between negative and positive examples.

The experiments were performed in the same way than the previous ones, and the sentences were also analyzed using the Link Parser. We only performed with this dataset the experiments that showed the best result in the BS dataset. We therefore only ran the Naïve Bayes and SVM algorithms, and we only compared the results from the traditionally BOW representation with the ones obtained when using all the same previously used links without distinctions among them. Experiments introducing semantic knowledge were not performed, but the same hierarchical dictionaries used for our experiments in the BS dataset could be used in this dataset in the future.

For the Naïve Bayes and SVM experiments the Weka package with the default parameters was again used. The features were filtered using Information Gain as it was done

before, 1000 features were kept this time for showing the best results in preliminary experiments. It must be noticed that Information Gain is not well suited for imbalanced datasets, and therefore another feature selection metric could be used. As an example, Forman and Cohen [FC04] proposed a feature selection method that prevents the predictive features from the majority class from hiding all useful features from the minority class.

The results obtained after 10 runs of 10-fold cross-validation are shown in table 18.

Table 18: Performance of the syntactic features in the YPD dataset.

		Words	**Links**
Naïve Bayes	Accuracy	0.82	0.83
	Precision	0.34	0.35
	Recall	0.66	*0.59*
	F1-measure	0.45	0.44
SVM	Accuracy	0.89	0.90
	Precision	0.50	**0.55**
	Recall	0.25	**0.29**
	F1-measure	0.34	**0.38**

As in the experiment in the BS dataset, here as well the pattern of improvement holds, and we observe a statistically significant increase in the F1-measure values of the Links representation over the BOW representation for the experiments using SVM. The variation in F1-measure was not significant when using Naïve Bayes. The accuracy for this dataset does not permit to notice any difference between the behavior of the two representations given the imbalanced proportion of the positive and negative examples.

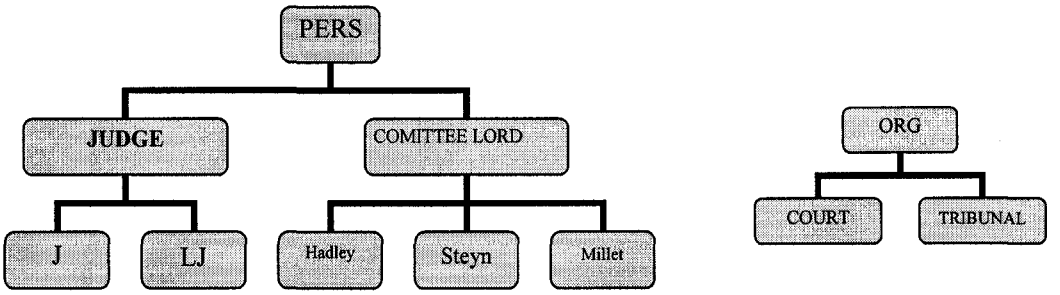
5.6.2. Experiment on the HOLJ dataset

The HOLJ corpus presents a completely different domain, that of legal documents. The sentences are still short, with an average length of 29 words. This and the highly specific legal terms make the representation very sparse.

The task consists of learning the rhetorical role of each sentence in the text. Hachey and Gover [HG05] addressed this by using different sets of features, including linguistic features as well as the sentence location and length. We instead use only the sentence's words and syntactic structure, making our experiments comparable to the results they obtained using only the linguistic features (F1 measure = 0.61 when micro-averaged). We believe that even when there are not clear relations between parts of the sentences (as it was in the biological domain,) there is still a subtle relationship between the words denoted by the sentence's syntactic structure and captured in the syntactic bi-grams and the FOL predicates.

The source of information for our hierarchical generalization is the same corpus, which has been automatically annotated with recognized name entities, as judge's and lawyer's names and court houses. We created two hierarchies, with the original words as leaves and the generic markers "person" and "place" as roots. An extract of these hierarchies is shown in example 7.

Example 7: Example of the two created hierarchies for the HOLJ dataset.



For each original word from the leaves of the hierarchies that was found in any sentence, its ancestors were added into the representation. When a leaf occurring in the sentence is syntactically related to another word, new syntactic features were created for each of the ancestors and these new syntactic features were also added to the representation. In the future, we will consider other hierarchies, as it could be a generalization of the main verbs.

For our experiments, we created four different representations by adding to the basic representation using only words either semantic background knowledge, syntactic features or both. For each of these sentences' representation, we automatically learned classifiers using the Simple Naïve Bayes algorithm from the Weka package. We then evaluated the classification performance with the standard 10-fold cross-validation protocol, using the four classical ATC measures accuracy, precision, recall and F-measure. For the analysis of the results, we compared the performance of the enriched representations compared to the basic one.

The following example shows the syntactic features obtained using the Link parser, which is the first step in creating our representations.

Example 8: Syntactic analysis returned by the Link parser for the sentence
"I would allow the appeal and restore the order of the Divisional Court".

Subject relation: I-would
Object relations: allow-appeal, allow-order
Prepositional relation: order-court
Noun phrase: divisional-court

It can be seen in this example that the analysis returned by the Link Parser is not always correct or complete. In the example, the object relation allow-order is incorrect, while the expected one restore-order is missing. The Link Parser actually returned two different analyses for this sentence, each of them missing some relations. We limited our approach by choosing always the first analysis, which is the one scored higher.

Other limitations of our approach introduced by the use of the Link Parser are time constraints and limited vocabulary. This parser allows us to define how much time to spend analyzing each sentence, and when the time is over, an empty or non-accurate syntactic analysis is provided. This can be solved by increasing the time limit, at least for those difficult to analyze sentences. If this is not an option, the use of other more superficial parsers, as the one presented by Jacquemin [J96], could be considered. Since the HOLJ is annotated with some linguistic and syntactic features, they could also be used to obtain a more accurate syntactic analysis.

The vocabulary the Link Parser recognizes can also be extended, adding to it the technical terms necessary for the particular domain being analyzed. However, even when a particular word is not recognized, if its part of speech can be deduced from the context, the parser can still produce a correct syntactic analysis. Extension to the Link Parser vocabulary in the biomedical domain have been studied by Pyysalo et al. [PGPBJSK04] without showing a significant improvement in the syntactic analysis.

For our experiments we allowed the default time for parsing, and for some of the examples we were not able to obtain a syntactic analysis. Most of those cases were part of sentences expressed as bullets, with an incomplete syntactic structure. Those examples are therefore represented only by words even in the case were syntactic features are used in the representation of the others sentences in the dataset. In the future we plan to use the syntactic features contained in the annotations in order to syntactically enrich the representation of all the sentences.

Our most basic representation (W) is the BOW. For our enriched representations we added, to the representation containing only words, either the recognized entities hierarchies (WH), or the syntactic features (WS), or both (WHS). The following example shows these four different representations for the same sentence as before.

Example 9: Features used in the four different representations for the sentence from ex. 8.

W: `I, would, allow, appeal, order, divisional, court`

WH: `I, would, allow, appeal, order, divisional, court, org`

WS: `I, would, allow, appeal, order, divisional, court, I-would, allow-appeal, allow-order, order-court, divisional-court`

WHS: `I, would, allow, appeal, order, divisional, court, org, I-would, allow-appeal, allow-order, order-court, divisional-court, divisional-org`

An extra filtering was performed afterwards for each of the representations with a feature selection purpose, using the Information Gain measure. The experiments were performed using 2000 features, since that was the number that resulted in the best accuracy in preliminary experiments.

Since there are several classes in the dataset, one class against the rest approach was used. Each of the six main classes were considered, one at a time, as the positive class, and the examples from all the other classes were considered as negative ones in that run. In that way, six binary classifiers were built and evaluated, and their results averaged. In the following table we show the micro-averaged results after 10 runs of 10-fold cross-validation of Simple Naïve Bayes using the default parameters. All the results are compared against the first column, which shows the results for the representation using only words (W). For all the tables, the results in bold denote an increase with respect to the first column, while the results in italic denote a decrease. The results in the shaded cells indicate that the increase or decrease is statistically significant.

Table 19: Micro-averaged performance of the syntactic and hierarchical features in the HOLJ dataset

Dataset	W	WH	WHS	WS
Accuracy	0.89	0.89	0.89	0.89
Precision	0.72	*0.71*	*0.68*	*0.69*
Recall	0.53	0.53	**0.62**	**0.62**
F-measure	0.61	*0.60*	**0.65**	**0.65**

We can observe in this table that when using the syntactic features (WS and WHS), the results show a big and statistically significant increase of the recall and F-measure, while the precision values suffer only a small decrease. Given the imbalanced proportion of the positive and negative examples, the accuracy for this dataset does not permit to notice any difference between the behaviors of the different representations, as it can be seen in the first row of the previous table.

The use of the hierarchical background knowledge did not show any improvement with respect to the representation using only words. In an attempt to discover whether this is a general behavior for all the classes we now present the results per class.

Table 20: Per class accuracy of the syntactic and hierarchical features in the HOLJ dataset

Dataset	**W**	**WH**	**WHS**	**WS**
Background	0.84	0.84	0.84	0.84
Disposal	0.94	0.94	*0.93*	0.94
Fact	0.92	0.92	0.92	0.92
Framing	0.81	0.81	0.81	0.81
Procedings	0.86	0.86	0.86	0.86
Textual	0.97	0.97	0.97	0.97

Table 21: Per class precision of the syntactic and hierarchical features in the HOLJ dataset

Dataset	W	WH	WHS	WS
Background	0.76	0.76	*0.72*	*0.72*
Disposal	0.72	*0.69*	*0.64*	*0.68*
Fact	0.70	*0.69*	*0.64*	*0.64*
Framing	0.70	0.70	*0.64*	*0.65*
Procedings	0.72	*0.71*	*0.68*	*0.70*
Textual	0.72	*0.71*	**0.76**	**0.76**

Table 22: Per class recall of the syntactic and hierarchical features in the HOLJ dataset

Dataset	W	WH	WHS	WS
Background	0.56	0.56	**0.66**	**0.66**
Disposal	0.45	0.45	**0.52**	**0.51**
Fact	0.59	**0.60**	**0.70**	**0.68**
Framing	0.56	**0.57**	**0.69**	**0.68**
Procedings	0.55	*0.54*	**0.60**	**0.62**
Textual	0.49	*0.47*	**0.57**	**0.58**

Table 23: Per class F1-measure of the syntactic and hierarchical features in the HOLJ dataset

Dataset	W	WH	WHS	WS
Background	0.64	0.64	**0.69**	**0.69**
Disposal	0.55	*0.54*	**0.57**	**0.58**
Fact	0.64	0.64	**0.66**	**0.66**
Framing	0.62	**0.63**	**0.67**	**0.66**
Procedings	0.62	0.62	**0.64**	**0.65**
Textual	0.58	*0.56*	**0.64**	**0.66**

From the results per class we can again observe a big and statistically significant increase of the recall and F-measure (for most classes) when using the syntactic features with respect to the basic representation using only words. The decrease in precision is

statistically significant for all of the classes except for Textual, in which we observe a statistically significant increase of both precision and recall.

Since phrases as in our syntactic representation are more specific than the words by themselves, then an increase in the classification precision is to be expected when using them in the representation. Contrary to what was expected, the recall increased consistently. That makes us believe that the use of syntactic relations introduces a semantic knowledge that is general rather than specific. We explain the increase of recall by the good combination of the use of the syntactic features and the feature selection process. While some words are not relevant by themselves and therefore not selected by the Information Gain measure, they become relevant when they happen together. That could be for example the case of the words "allow" and "appeal" and the syntactic feature "allow_appeal" presented in the example in the previous section. Once these syntactic features are included in the representation, they can now be used to classify many more examples as positives, increasing the recall, but also in some cases making more mistakes, decreasing the precision.

In the case of the Textual class, were both precision and recall increased, we explain it with the particular characteristics of the class. This class is formed by sentences that are not related to a particular case, but instead they are sentences that coordinate or arrange the document in a specific way. We believe that, specially in the case of this class, the syntactic features can capture the expected structure of the text. This is confirmed by some of these features been selected as the most relevant only for this class, as for example: "benefit_withdrawal", "appeared_who", "act_protection", "granted_were", "said_lj", "decision_final".

Even in this per class analysis, we could still not see any relevant improvement when using the hierarchical background knowledge with respect to the representation using only words. Even when there are small improvements in the Recall for the Fact and Framing classes when using the hierarchical terms together with the words (WH), and for those two same classes and for the Disposal class when using the hierarchical terms together with the words and the syntactic features (WHS), none of those improvements is statistically significant (respectively compared to the BOW or to the representation including the syntactic relations). We understand that this particular semantic generalization may not be appropriate for the specific sentence classification task at hand, but we believe that a different generalization could still be useful. In the future we plan to study the contribution of a generalization of the main verbs.

5.7. Generalization to Automatic Text Classification.

As mentioned earlier (in sec. 3), the use of syntactic features in the text representation for IR and ATC has not shown a consistent improvement of the performance, depending it in the type of documents, their length and the sparseness of the representation.

In this subsection we study whether the length and other particular characteristic of the sentences could be the reason for the increase in performance observed when the syntactically enriched representation is used.

For this purpose we used the original set of abstracts from where the sentences of the BS dataset were obtained. Each of the abstracts contained an average of three sentences from the dataset, among other sentences that were not included in it. The abstracts were

assigned to the positive class if at least one of the sentences in the abstract belonged to the positive class in the BS dataset and to the negative class otherwise. The resulting imbalanced dataset has a ratio of approximate 2:1 positive to negative examples.

Experiments were run for the Naïve Bayes and SVM algorithms, when using four representation: BOW, BOW + syntactic features, BOW + hierarchical information, and BOW + both syntactic features and hierarchical information. The results are shown in the following table.

Table 24: Performance of the syntactic features and hierarchical information in the abstracts dataset.

	Metric	**BOW**	**BOW + links**	**BOW + hier**	**BOW + links + hierarchy**
Naïve Bayes	Accuracy	0.68	**0.74**	0.68	**0.73**
	Precision	0.68	**0.76**	0.69	**0.76**
	Recall	0.98	*0.92*	0.98	*0.91*
	F1	0.80	0.83	0.81	0.82
SVM	Accuracy	0.68	0.68	0.69	0.68
	Precision	0.73	0.74	0.73	0.74
	Recall	0.86	0.84	0.87	0.84
	F1	0.78	0.78	0.79	0.78

The results with Naïve Bayes show a significant increase of the accuracy and precision when using the syntactic features (either with or without the hierarchical information) with respect to BOW alone. However, the also significant decrease of the recall makes the differences in the F1 values not statistically significant.

The results with SVM show very little variation, with not significantly better performance from any of the four representations.

With this experiment we show once again that the addition of syntactic and semantic knowledge in the text representation for ATC does not bring significant improvement of the performance. As the vocabulary and type of documents are closely related to that of the BS dataset, we believe that it is the nature of Sentence Selection and its characteristics that make the use of syntactic and semantic information useful in that case.

It can also be observed that the performance values in table 24 are in general higher than the corresponding values on tables 10 and 11 (showing the performance on the BS sentences dataset). From this comparison we could infer that the ATC categorization task performed in the abstracts dataset is easier than the sentence selection task performed in the original BS dataset. However, it has to be taken into consideration that the class distribution in both datasets is not the same, and therefore this comparison is not completely fair.

6. Discussion of the Results

In this chapter we present a discussion of the results and explain what we consider are the major contributions of our work.

6.1. Contribution of Term Generalization

By term generalization we mean the addition or replacement of a word for a related one that is more general (e.g.: in the hierarchies we use to generalize the genes or proteins names), or some other way of obtaining a similar effect (as the predicates indicating the interaction verbs in sec. 5.5.2.)

Intuitively, by generalizing terms we expected to obtain features that represent our data in a more general way, and therefore avoiding over-fitting the learned classifier. When comparing results, this should produce an increase of the recall.

In our preliminary experiments with Flipper on the BS dataset (see Apendix B), the increase in the recall is noticed when replacing the genes or proteins names, both when using the equivalent to the BOW representation and when using the links based representation. However the same was not true when the interaction verbs were replaced, resulting in that case in the recall decrease.

Following these preliminary results, only the genes or proteins names were generalized for the BS dataset. For the Aleph experiments the generalization of the interaction verbs was tried again using a different approach, which showed to be beneficial (see sec. 5.5.2).

When replacing the genes or proteins names for our lexical marker "geneprot", its usefulness can be confirmed by the fact that the lexical marker (together with other words) was selected by Aleph to be included among the first rules in the learned rule sets. In these experiments the use of the lexical marker improved the recall as expected.

One disadvantage of the generalization approach by replacing one word by another is the loss of the original word. This is implicitly shown in our experiments by the omission in the genes/proteins replacement list of the word sigmaK. Since this word is used by Aleph among the first rules, had it been replaced by the lexical marker "geneprot", its discriminating power would have been lost.

The loss in the representational power when using the replacement approach is made evident in our experiments with SVM, where the classification results showed some decrease with respect to the BOW. With Naïve Bayes, however, the classification results stayed the same when using the replacement approach or the BOW.

When using the replacement approach there is also the risk of generalizing too much, or the complication to decide to which level on a hierarchy to generalize. This problem, as well as the loss of the original word, can be easily solved by adding (instead of replacing) in the representation several levels of generalization, and letting the feature selection filter or the learning algorithm itself choose the best terms in each case, which could be the original word or a generalized term (or predicate in the case of the experiments with Aleph and the generalization of the interaction verbs). This approach by adding generalization to the representation (instead of simply replacing words) proved to be beneficial in our experiments with Naïve Bayes and Aleph. Table 25 summarizes the successes (+) and defeats (-) of this approach in the BS dataset.

Table 25: Summary of the performance achieved when replacing or adding hierarchical information to the BOW.

	NB	SVM	Aleph
Replacement		-	
Addition	+		+

The added generalization seems even more useful when representing the sentences by noun phrases and/or other syntactically related phrases. Phrases are by nature more specific than single words, and therefore the need of some level of generalization is even stronger. In our experiments several phrases containing generalizing terms were selected among the most discriminating features (over many words and other phrases) by the Information Gain measure. And the generalizing terms were also included in several "link" predicates in the rule sets learned by Aleph.

A drawback of our experiments with the Swiss Prot hierarchy was that the list of genes/proteins that we used was not the same used by the researchers who created the BS dataset, and it does miss some of the terms that they considered of interest. However, using this list allowed us to generalize some more technical terms than only those genes or proteins names. In the future we are planning to combine the Swiss Prot hierarchy with the full list of genes/proteins appearing in the BS dataset.

6.2. Contribution of Syntactic Representation

By Syntactic Representation we mean the representation that makes use of the syntactic relations between words, as it is the case of a noun-modifier relation constituting a noun phrases.

As mentioned in our bibliographic review chapter, the use of noun phrases and other syntactic related phrases for text representation have been tried without much success in ATC. But because of the differences between ATC and Sentence Selection we expected they would be useful in the latter task.

Phrases, and in particular noun phrases, are by nature more specific than single words. Therefore their use would be of interest only if a more specific representation and interpretation is required. This is the case in sentence selection where instead of a general topic for the classification of a document we are interested in finding out something very specific, such as a sentence in a biomedical abstract expressing a relation between genes/proteins, or a sentence in a judgment crediting or discrediting a claim. For example, in our work in the BS dataset, the noun phrase "binding protein" presents us with more specific information than the noun "protein" by itself. In fact, "binding protein" is highly discriminating for our particular classification task.

Because of their specificity characteristic, noun phrases have been considered good disambiguating tools. For example, when the word "bank" appears on a text, the noun phrase "river bank" could help differentiate its use from that of "bank" as a financial institution. In the relatively long texts generally used in ATC this differentiation is not necessary since the presence of many other words would create a context that helps deciding the document topic. For sentence selection, however, the low number of words in the

sentence might not be enough to define a context, and therefore the disambiguating help of noun phrases would be more appreciated.

For machine learning in general, the best features to be used in the examples representation are the more discriminating ones. For ATC, it is believed that the best features for the text representation are those with neither very low nor very high frequency. Noun phrases have been said not to have good statistical properties to become interesting features for ATC, since in general they would not occur in long text enough times compared to individual words. This is another difference between the ATC and Sentence Selection tasks. Because of the short length of the sentences, every term, words and noun phrases, would occur relatively few times, and therefore noun phrases have similar statistical properties to simple words.

In our preliminary experiments in the BS dataset, the representation that used the noun phrases recognized by our noun phrases extractor outperformed the representation using every word according to all the metrics when running both the NB and the SVM classifiers.

If, as described, the noun phrases are more specific than the words by themselves, the obtained increase in the classification precision is to be expected when using them in the representation.

In the case of the BS dataset, we also compared the results obtained when using the representation with all the words involved in the links with the results obtained when adding to it phrases formed by different kinds of syntactic relations: noun phrases, subject-object relations, prepositional phrases, and all of them together without making a distinction. The representation using all the relations together was the one that showed the most interesting

increase both when using Naïve Bayes and SVM. As expected, the use of all the syntactic links produced an increase in precision that in most cases translated to an increase in accuracy and F1 measure as well.

Contrary to what was expected, the recall increased consistently when using the representation with all the relations over the one that includes none of the relations, in both the BS dataset when using the SVM and Aleph classifiers, and the YPD dataset when using SVM. That was also observed in preliminary experiments with the NB algorithm when the threshold was not optimized but left to its default value of 0.5. That is explained by the fact that when either the threshold in NB or the border in SVM are fixed, and a more specific representation is given to the same examples, all the examples that were previously considered as positives will keep the same classification. At the same time, with a more specific representation, some of the false negative examples can now be correctly classified as positive, resulting in a better recall. This does not seem to introduce many new false positives, and precision still remains high. When we allow the threshold to move in order to optimize the F1 measure, however, a decrease of the recall might be allowed if that helps to significantly increase the precision and therefore helps optimizing the F1 measure.

As previously expressed, the generalization of technical terms does also help in the performance of the classifiers when the more specific representation including the syntactic phrases is used. In the BS dataset, the best results were always obtained when using the representation enriched both syntactically and semantically. In some cases the same performance was achieved when using the representations enriched either only syntactically or only semantically. Table 26 summarizes the cases where the best performance was obtained in the BS dataset.

Table 26: Summary of the performance achieved when introducing syntactical and semantic information.

	Semantically	Syntactically	Syntactically And Semantically
NB cross-validation	Best	Best	Best
SVM cross-validation			Best
Aleph cross-validation		Best	Best
NB time split			Best
SVM time split		Best	Best
Aleph time split			Best

Some limitations of the syntactic approach are introduced by the use of the Link Parser, in particular time constrains and limited vocabulary. This parser allows us to define how much time to spend analyzing each sentence, and when the time is over, an empty or non-accurate syntactic analysis is provided. This can be solved by increasing the time limit, at least for those difficult to analyze sentences. If this was not an option, the use of other more superficial parsers, as the one presented by Jacquemin [J98], could be considered. In the case of the HOLJ dataset we are also planning in future to use the syntactic analyses already provided in the annotated corpus.

The vocabulary the Link Parser recognizes can also be extended, adding to it the technical terms necessary for the particular domain being analyzed. However, even when a particular word is not recognized, if its part of speech can be deduced from the context, the parser can still produce a correct syntactic analysis. Extension to the Link Parser vocabulary

in the biomedical domain have been studied by Pyysalo et al. [PGPBJSK04] without showing a significant improvement in the syntactic analyses.

6.3. Contribution of Relational Learner

Our last contribution is the use of the relational learner Aleph for the task of Sentence Selection. To our best knowledge its use for Sentence Selection has not been tried before, probably because Aleph is not a machine learning tool commonly used for ATC. But again, the particular characteristics of Sentence Selection make it different from ATC, and feasible to apply this approach. In our experiments, Aleph achieved accuracy and precision comparable to other state of the art machine learning algorithms.

As a relational learner, Aleph uses a FOL representation, which permits us to define predicates indicating the relation between two words. In our experiments, we used this representation for the examples, together with simple predicates to define the type of rules to learn, and some background knowledge (as for the "lexexist" predicate and the interaction verbs). Aleph was then able to learn rules using one or more predicates given in the examples and/or the background knowledge, generalizing them by using variables, and combining them by the means of those variables in order to define transitive relations between more than 2 words.

The rules learned by Aleph covered few positive examples each, making it possible to identify the characteristics these few examples had in common. When the whole set of rules was tested in the left out testing set, each managed to again identify a few examples following its pattern, and classified it correctly. However, in general, these rules did not manage to identify all the positive examples in the testing sets, and therefore the

performance was affected by the low recall. This low generalization of the rules was mainly a problem for the representation equivalent to the bag of words in the time split experiment. The recall (and F1-measure) greatly increased when using our syntactically enriched representation. Even when adding the syntactic relations could make some rules more specific, it can also make other rules more general by allowing the use of variables in them.

As presented in section 6.1, the introduction of hierarchical semantic knowledge was also tried with Aleph in the BS dataset experiments. For the case of the genes or proteins names, the replacement and addition approaches were used, obtaining the latter the best performance. When a gene or protein name is found in the examples, a new predicate containing the generalization term instead of the world itself was created and either used to replace the original one or added to the representation. In the cases where the new predicates were added, Aleph decides whether to use the generalized predicate or the original one.

A different approach was used to add the "interacts" verbs generalization. When one of such verbs (as defined by several instances of the "interaction" background predicate) is found in the examples, Aleph evaluates the possibilities of leaving it as it appears, replacing it by a variable, or linking it to another word or variable by the use of the "interacts" predicate. The option that obtains the best gain is then dynamically chosen. This alternative permits Aleph to include more complicated predicates in one step, which to some extent helps it fighting the greedy nature of its learning.

7. Conclusions and Future Work

We have presented here all the work done in this thesis.

We started by introducing the general concepts of Sentence Selection and ATC, differentiating them and giving their particular characteristics and application areas, the tools most commonly used for them, and the classical BOW text representation and its limitations. We presented some recent research in these areas, with special emphasis on the use of alternative text representations and the use of background knowledge.

Then we presented our approach to Sentence Selection, which consists of using a semantically and syntactically enriched text representation by means of technical dictionaries and syntactic analysis obtained from automatic parsing. This representation can be either used with the state of the art machine learning tools generally used for ATC, or as suggested here within a FOL representation and a relational learning system.

We have tried our approach in several experiments in three different datasets, two from the biomedical information extraction domain and the other from the legal judgments text summarization domain. Our experiments showed that our approach performed better than the classical BOW text representation. The use of the suggested relational representation and relational learner showed to be the best over all in our more extensively studied dataset. Table 27 summarizes the performance of the semantically and syntactically enriched representations over the different datasets and experiments. The higher the number of "+" displayed the better the performance with respect to the BOW.

Table 27: Summary of the semantically and syntactically enriched representations performance

	Semantically	Syntactically	Syntactically And Semantically
NB	+	+++	+++
SVM		++	+++
Aleph	++	+	+++

We have analyzed in detail the particular results obtained with each experiment, and we have discussed on a more general level the contributions that this work presents. We have also presented the limitations of each of the steps we have performed during our experiments and when appropriate we have suggested changes or alternatives with explanations of why we believe it to be better. As a summary we repeat here the main limitations:

- limitations related to the use of the syntactic parser:
    - lack of specific technical vocabulary, which could be extended
    - not all the relations are discovered or correct
    - the type of links and morphological information has not been taken into consideration, but it is available and could be used.
- limitations related to the use of semantic knowledge:
    - technical dictionaries are not complete
    - only few technical categories have been generalized
    - the hierarchies were limited to only three or four levels.

There are many things we would like to do as continuation of our research. We have already mentioned many of them when appropriate, and we now repeat them all:

- to use in the representation all the links together as we have been doing, but to include their type (noun phrase, prepositional phrase or subject-object relation) and the morphological information about the words, i.e. which ones are nouns, adjectives and verbs,
- to use the same list of gene/proteins that was used when creating the datasets, in order to recognize at least all the same names that were originally considered,
- to take advantage of the HOLJ corpus annotations, obtaining a more complete and accurate syntactic analysis,
- to consider different levels in a hierarchy of the interaction verbs.

As we have already mentioned, we believe it would be worth to use the technical dictionaries and generalization lists as background knowledge with an LSI representation.

Another way the syntactic and semantic features could be used is as different views in a co-training system. Co-training systems assume that the features of the different views are independent [NG00]. As the syntactic and semantic features presented in this disertation are highly dependent, the different views would have to be created in a way that minimize the dependency.

In addition to this future work, we would also like to validate our Sentence Selection approach by using it in a complete Information Extraction or Text Summarization system. If, as indicated by our results, our approach obtains higher recall and/or precision than the

standard BOW representation, then we could expect to extract information in a more complete or precise way, and to obtain more accurate summaries.

There is also one more approach that has caught our interest during our project, and that we are considering as future work. It is that of propositionalization, in which a given FOL representation is converted into propositional logic. Kramer [K00] presented a way to select from the whole space of predicates those that would be useful to include in the conversion. In our experiments we have already discovered many interesting FOL predicates that we could convert into propositional clauses by simply evaluating their logic value in the examples. Each of these new clauses could then be used as a binary feature in a vector space or probabilistic model. For instance, using the predicates in rule 5 from example 5 (`pos(S):-link(S,B,geneprot),link(S,B,bdependent)`) we could add in the representation a boolean feature that indicates whether the word "`bdependent`" happens in the sentence and is syntactically related to another word that is itself syntactically related to a gene or protein name. In this way we believe we could capture at least part of the power of our relational representation and use it in state of the art classification methods such as SVM and Naïve Bayes.

Finally, we would like to mention another application in which we consider that our relational representation could be useful. It is the clustering of search engine results. Most search engines available nowadays return a plain list of the web pages hits described by its title and a couple of sentences (snippets). When the query words are rather general, the returned result could contain documents of many diverse topics. For example, when the search word is "seal", it could return document related either to the animal or to the stamp of authenticity. Some research works, as the one by Cutting et al. [CKPT92] and several by

Weiss with other authors [WS03] [OW04] [OSW04], have taken the approach of clustering the returned web pages according to their content. For this purpose they represent each result by only the returned snippets, which is a very small and incomplete representation of the document contain. These sentences lack the context in which they occurred in the document and therefore cannot capture its semantics. In such a case, any extra help in figuring out the sentence context and its semantics would be beneficial. We believe that the sentence syntactic structure could do so, for example the knowledge that the word "seal" is the subject of a sentence could help determine that the document most probably refers to the animal and not to the stamp. In the future we would also like to experiment in the Clustering of Web Pages research area using our syntactic relations representation.

References

[ATV00] Pieter W. Adriaans, Marten Trautwein and Marco Vervoort. Towards High Speed Grammar Induction on Large Text Corpora. In: *SOFSEM 2000: Theory and Practice of Informatics*, Lecture Notes in Computer Science 1963, V. Hlavác, K.G. Jeffery and J. Wiedermann (editors), 2000.

[B93] Eric Brill, "Transformation-based error-driven parsing". In Proceedings of the 3rd International Workshop on Parsing Technologies, Tilburg, The Netherlands. 13—25, 1993.

[CNPB00] C. Cardie, V. Ng, D. Pierce, and C. Buckley, "Examining the Role of Statistical and Linguistic Knowledge Sources in a General-Knowledge Question-Answering System". *Proceedings of the Sixth Applied Natural Language Processing Conference (ANLP-2000)*, 180--187, Association for Computational Linguistics / Morgan Kaufmann, 2000.

[CM06] Maria Fernanda Caropreso and Stan Matwin. "Beyond the Bag of Words: a Text Representation for Sentence Selection". Canadian Artificial Intelligence 2006, Quebec, 2006.

[CM07] Maria Fernanda Caropreso and Stan Matwin. "Incorporating Syntax and Semantics in the Text Representation for Sentence Selection". Recent Advances in Natural Language Processing, Borovets, Bulgaria, 2007.

[CMS01] Maria Fernanda Caropreso, Stan Matwin, and Fabrizio Sebastiani, "A learner-independent evaluation of the usefulness of statistical phrases for automated text categorization". In Amita G. Chin (ed.), *Text Databases and Document Management: Theory and Practice*, Idea Group Publishing, Hershey, US, 2001, pp. 78-102.

[C95a] William W. Cohen Fast effective rule induction in ICML 1995: 115-123. 1995.

[C95b] W. W. Cohen. Text categorization and relational learning. In A. Prieditis and S. J. Russell, editors, Proceedings of ICML-95, 12th International Conference on Machine Learning, pages 124--132, Lake Tahoe, US, 1995. Morgan Kaufmann Publishers, San Francisco, US.

[C95c] Cohen, W. Learning to classify English text with ILP methods. In De Raedt, L., editor, Proceedings of the 5th International Workshop on Inductive Logic Programming, pages 3-24. 1995.

[C96] William W. Cohen (1996): Learning Trees and Rules with Set-valued Features in AAAI/IAAI, Vol. 1 1996: 709-716.

[CH98] W. W. Cohen, Haym Hirsh. Joins that Generalize: Text Classification Using Whirl. In Proceedings of the Fourth Conference on Knowledge Discovery and Data Mining. August, 1998.

[CS99] W. W. Cohen & Y. Singer (1999): Context-sensitive learning methods for text categorization in ACM Trans. Inf. Syst. 17(2): 141-173 (1999).

[CJS02] Copeck, T., Japkowicz, N., and Szpakowicz, S. Text Summarization as Controlled Search. Proceedings of the Canadian Society for Computational Studies of Intelligence (AI'2002). 268-280.

[CKPT92] Cutting, D. R., Karger, D. R., Pedersen, J.O. and Tukey, J.W. Scatter/Gather: A cluster-based approach to browsing large document collections. In Proceedings of the 15th International ACM SIGIR Conference on Research and Development in Information Retrieval. 1992.

[DDLFH90] S. C. Deerwester, S. T. Dumais, T. K. Landauer, G. W. Furnas, and R. A. Harshman. Indexing by latent semantic analysis. Joumal of the American Society of Information Science. 1990.

[DLW00] Y. Diao, H. Lu, and D. Wu. A comparative Study of Classification Based Personal E-mail Filtering. In Proceedings of PAKDD-00, 4th Pacific-Asia Conference on Knowledge Discovery and Data Mining. 2000.

[D94] S. T. Dumais. Latent Semantic Indexing (LSI): TREC-3 Report. NIST Special Publication 500-226: Overview of the Third Text REtrieval Conference (TREC-3) 1994.

[DPHS98] S. T. Dumais, J. Platt, D. Heckerman, and M. Sahami. Inductive learning algorithms and representations for text categorization. In G. Gardarin, J. C. French, N. Pissinou, K. Makki, and L. Bouganim, editors, Proceedings of CIKM-98, 7th ACM International Conference on Information and Knowledge Management, pages 148{155, Bethesda, US, 1998. ACM Press, New York, US.

[F87] J. L. Fagan. Experiments in automatic phrase indexing for document retrieval: a comparison of syntactic and non-syntactic methods. PhD thesis, Department of Computer Science, Cornell University, Ithaca, US, 1987.

[F68] Fillmore, Charles J. The Case for Case. In Bach and Harms (Ed.): Universals in Linguistic Theory. New York: Holt, Rinehart, and Winston, 1-88. 1968.

[FC04] George Forman and Ira Cohen. Learning from Little: Comparison of Classifiers Given Little Training. ECML/PKDD, Pisa Italy, 2004

[F94] J. Furnkranz. Inductive Logic Programming (a short introduction and a thesis abstract). *ÖGAI-Journal* 13(3-4):3-8, 1994.

[F98] J. Furnkranz. A study using n-gram features for text categorization. Technical Report TR-98-30, Oesterreichisches Forschungsinstitut Artificial Intelligence, Wien, AT, 1998. http://www.ai.univie.ac.at/cgi-bin/tr-online?number+98-30.

[F99] J. Furnkranz. Exploiting structural information for text classification on the WWW. *Proceedings of IDA '99, 3rd symposium on Intelligent Data Analysis*, Amsterdam, NL, 1999, 487-497.

[FMR98] J. Furnkranz, T. M. Mitchell, and E. Rilof. A case study in using linguistic phrases for text categorization on the WWW. In Proceedings of the 1st AAAI Workshop on Learning for Text Categorization, pages 5{12, Madison, US, 1998.

[GOS04] M. Goadrich, L. Oliphant & J. Shavlik (2004). Learning Ensembles of First-Order Clauses for Recall-Precision Curves: A Case Study in Biomedical Information Extraction. *Proceedings of the Fourteenth International Conference on Inductive Logic Programming*, Porto, Portugal.

[HG05] Ben Hachey and Claire Grover. Sequence Modelling for Sentence Classification in a Legal Summarisation System. In: *Proceedings of the 2005 ACM Symposium on Applied Computing*, 2005.

[J96] Christian Jacquemin. *What is the tree that we see through the window: A linguistic approach to windowing and term variation.* Information Processing and Management, 32(4):445--458, 1996.

[J98] Joachims, Thorsten. Text categorization with support vector machines: learning with many relevant features. In Proceedings of ECML-98, 10th European Conference on Machine Learning (Chemnitz, DE, 1998), pp. 137–142.

[J99] Joachims, Thorsten. Transductive Inference for Text Classification using Support Vector Machines. Proceedings of the International Conference on Machine Learning (ICML), 1999

[JM00] Dan Jurafsky, James Martin et al., "Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics and Speech Recognition". Prentice Hall, 2000.

[K98] Kirkpatrick, B. Roget's Thesaurus of English Words and Phrases. Harmondsworth, Middlesex, England: Penguin. 1998.

[K00] Stefan Kramer: Thesis: Relational learning vs. propositionalization. AI Commun. 13(4): 275-276, 2000.

[LFL98] Landauer, T. K., Foltz, P. W., & Laham, D. (1998). Introduction to Latent Semantic Analysis. *Discourse Processes,* **25**, 259-284.

[LSD01] Latinne, P., Saerens, M. and Decaestecker, C. Adjusting the Outputs of a Classifier to New a Priori Probabilities May Significantly Improve Classification Accuracy: Evidence from a Multi-Class Problem in Remote Sensing. Proceedings of the Eighteenth International Conference on Machine Learning,(ICML). 2001.

[L92a] Lewis D D, "Representation and Learning in Information Retrieval", Ph.D. dissertation, University of Massachusetts, 1992

[L92b] D. D. Lewis. An evaluation of phrasal and clustered representations on a text categorization task. In N. J. Belkin, P. Ingwersen, and A. M. Pejtersen, editors, Proceedings of SIGIR-92, 15th ACM International Conference on Research and Development in Information Retrieval, pages 37-50, Kobenhavn, DK, 1992. ACM Press, New York, US.

[L98] Lewis, D. D. 1998. Naive (Bayes) at forty: The independence assumption in information retrieval. In Proceedings of ECML-98, 10th European Conference on Machine Learning (Chemnitz, DE, 1998), pp. 4–15.

[LC90] D. D. Lewis and W. B. Croft. Term clustering of syntactic phrases. In Proceedings of SIGIR-90, 13th ACM International Conference on Research and Development in Information Retrieval, pages 385{404, Bruxelles, BE, 1990.

[MBK94] Maarek, Y., Berry, D.M. & Kaiser, G.E.: GURU: Information Retrieval for Reuse, in P.Hall (ed.), Landmark Contributions in Software Reuse and Reverse Engineering, 1994

[MG98] Mladenic, D. and Grobelnik, M. Word sequences as features in text learning. Proceedings of *ERK-98, the seventh Electrotecnical and Computer Science Conference* (pp. 145-148). Ljubljana, Slovenia. 1998

[MG99] Mladenic, D. and Grobelnik, M. Feature Selection for unbalanced class distribution and Naïve Bayes. In Proceedings of the 16th International Conference on Machine Learning, 1999. pp. 258--267. Morgan Kaufmann.

[M90] Miller, G. WordNet: an On-line Lexical Database. International Journal of Lexicography. 1990.

[M96] Mitchell, T. M. 1996. Machine learning. McGraw Hill, New York, US.

[MBSC97] M. Mitra, C. Buckley, A. Singhal, and C. Cardie, "An Analysis of Statistical and Syntactic Phrases". *5TH RIAO Conference, Computer-Assisted Information Searching On the Internet*, 200-214, 1997.

[N00] Nédellec C., "Bibliographical Information Extraction in Genomics" in IEEE Intelligent Systems: Trends & Controversies - Mining Information for Functional Genomics, N. Shadbolt (éd.), p. 76-78, may-june, 2000

[NOB01] Nédellec C., Ould Abdel Vetah M., and Bessières P., "Sentence Filtering for Information Extraction in Genomics: A Classification Problem," *Proceedings of the International Conference on Practical Knowledge Discovery in Databases* (PKDD'2001), pp. 326–338, Springer Verlag, LNAI 2167, Freiburg, September, 2001.

[N01] Nigam, K. Using Unlabeled Data to Improve Text Classification. Doctoral Dissertation, Computer Science Department, Carnegie Mellon University. Technical Report CMU-CS-01-126. 2001.

[NG00] Nigam, K and Ghani, N. Understanding the behavior of co-training. In KDD-2000 Workshop on Text Mining, 2000.

[NMTM00] Nigam, K., McCallum, A., Thrun, S. and Mitchell, T. Text Classification from Labeled and Unlabeled Documents using EM. Machine Learning, 39(2/3). pp. 103-134, 2000.

[NGL97] Ng, H. T., Goh, W. B., and Low, K. L. 1997. Feature selection, perceptron learning, and a usability case study for text categorization. In Proceedings of SIGIR-97, 20th ACM International Conferen ce on Research and Development in Information Retrieval (Philadelphia, US, 1997), pp. 67–73.

[NSC07] Nastase, V., Sayyad, J. and Caropreso, M. F. "Using Dependency Relations for Text Classification". University of Ottawa, SITE, Technical report TR-2007-12 http://www.site.uottawa.ca/eng/school/publications/techrep/2007/TR-2007-12.pdf

[OSW04] Osinski, S., Stefanowski, J., and Weiss, D. Lingo : Search Results Clustering Algorithm Based on Singular Value Decomposition. Intelligent Information Processing and Web Mining, Proceedings of the International IIS: IIPWM'04 Conference, Zakopane, Poland, 2004, pp. 359-368.

[OW04] S. Osinski and D. Weiss. *Conceptual clustering using lingo algorithm: Evaluation on open directory project data.* In IIPWM04, 2004.

[O05] Ould, M. Apprentissage Automatique Applique a l'Extraction d'Information a Partir de Textes Biologiques. PhD Thesis. L'Universite Paris-Sud. France. 2005

[OCMNM03] Ould, M., Caropreso, F., Manine, P., Nedellec, C., Matwin, S., "Sentence Categorization in Genomics Bibliography: a Naïve Bayes Approach", Informatique pour lèanalyse du transcriptome, Paris, 2003.

[P80] M.F. Porter, 1980, An algorithm for suffix stripping, *Program*, **14**(3) pp 130−137.

[PGPBJSK04] Pyysalo, S., Ginter, F., Pahikkala, T., Boberg, J., Jarvinen, J., Salakoski, T., and Koivula, J. Analysis of Link Grammar on Biomedical Dependency Corpus Targeted to Protein-Protein Interactions. Proceedings of the International Workshop on Natural Language Processing in Biomedicine and its Applications (JNLPBA). 2004. p. 15-21

[RC01] Soumya Ray, Mark Craven. Representing Sentence Structure in Hidden Markov Models for Information Extraction. Proceedings of the 17th International Joint Conference on Artificial Intelligence (IJCAI-2001)

[R03] Rigouste, L. Evolution of a text Summarizer in an Automatic Evaluation Framework. Master's thesis. University of Ottawa. 2003.

[SM99] Sam Scott, Stan Matwin. Feature Engineering for Text Classification. Proceedings of ICML-99, 16th International Conference on Machine Learning, 1999.

[S02] Fabrizio Sebastiani, "Machine learning in automated text categorization". *ACM Computing Surveys*, 34(1):1-47, 2002

[SS98] R.E. Shapire and Y. Singer. Improved boosting algorithms using confidence-rated predictions. *Computational Learning Theory*, 1998, 80-91.

[S96] SIEMENS, R. G. Lemmatization and Parsing with TACT Preprocessing Programs. *CHWP* A.1. 1996

[S03] Siolas, G. Modèles probabilistes et noyaux pour l'extraction d'informations à partir de documents. Thèse de doctorat de l'Université Paris 6. July 2003.

[ST91] D. Sleator and D. Temperley. 1991. Parsing English with a Link Grammar. Carnegie Mellon University Computer Science technical report CMU-CS-91-196, October 1991.

[S93] Srinivasan, 1993.
http://web.comlab.ox.ac.uk/oucl/research/areas/machlearn/Aleph/aleph_toc.html

[TG03] Temkin JM, Gilder MR. Extraction of protein interaction information from unstructured text using a context-free grammar. Bioinformatics. 19(16):2046-53, 2003.

[TM02] Simone Teufel and Marc Moens. Summarizing scientific articles – experiments with relevance and rhetorical status. Computational Linguistics, 28(4):409-445,2002.

[T01] Turney, P.D. Mining the Web for synonyms: PMI-IR versus LSA on TOEFL, Proceedings of the Twelfth European Conference on Machine Learning, ECML-2001.

[WADJOGH99] S. M. Weiss, C. Apte, F. J. Darneau, D. E. Johnson, F. J. Oles, T. Goetz, T. Hampp. Maximizing text-mining performance. *IEEE Software Systems*, 1999, 14(4): 63-69.

[WS03] Weiss, D. and Stefanowski, J. Web search results clustering in Polish: experimental evaluation of Carrot. In: Advances in Soft Computing, Intelligent Information Processing and Web Mining, Proceedings of the International IIS: IIPWM'03 Conference, Zakopane, Poland, 579 (vol. XIV), 2003, pp. 209-220.

[W83] Terry Winograd, "Language as a Cognitive Process." Addison-Wesley Publishing Company, 1983, vol. 1, Syntax.

[WF05] Ian H. Witten and Eibe Frank. Data Mining: Practical machine learning tools and techniques, 2nd Edition, Morgan Kaufmann, San Francisco, 2005.

[X03] Quianjun Xu. Many Features, Little Data: Feature Selection for Small Data Sets Using Probabilistic Background Knowledge. M.S. thesis, University of Maryland, Baltimore County. 2003.

[Y94] Yiming Yang. Expert network: effective and efficient learning from human decisions in text categorisation and retrieval. In Proceedings of SIGIR-94, 17th ACM International Conference on Research and Development in Information Retrieval Dublin, IE, 1994, pp. 13–22.

[Y99] Yiming Yang. An Evaluation of Statistical Approaches to Text Categorization. Information Retrieval, 1(1/2):69--90. 1999.

[YL99] Yiming Yang and Xin Liu. A re-examination of text categorization methods. Proceedings of SIGIR-99, 22nd ACM International Conference on Research and Development in Information Retrieval, pages 42--49, 1999.

[ZH00] Sarah Zelikovitz and Haym Hirsh. Improving Short-Text Classification Using Unlabeled Background Knowledge to Assess Document Similarity. Proceedings of the Seventeenth International Conference on Machine Learning, ICML-2000

[ZH01] Sarah Zelikovitz and Haym Hirsh. Using LSI for Text Classification in the Presence of Background Text. Proceedings of CIKM-01, 10th ACM International Conference on Information and Knowledge Management. 2001.

[ZH02] Sarah Zelikovitz and Haym Hirsh. Integrating Background Knowledge into Nearest-Neighbor Text Classification. Proceedings of the 6th European Conference on Case Based Reasoning. Springer Verlag. 2002.

Appendix A Example of parsing using the Link parser:

```
Phrase: The big dog chased the black cat in the park.


Result:
++++Time                                    0.01 seconds (327.70 total)


Found 2 linkages (2 with no P.P. violations)


  Linkage 1, cost vector = (UNUSED=0 DIS=0 AND=0 LEN=20)


    +-------------------------------Xp-------------------------------+
    |                           +----------MVp----------+            |
    +--------Wd-------+         +---------Os---------+   |            |
    |      +----Ds----+         |     +-----Ds-----+    +---Js---+   |
    |      |    +--A--+---Ss--+      |     +---A--+    |  +--Ds-+   |
    |      |    |     |       |      |     |      |    |  |     |   |
LEFT-WALL the big.a dog.n chased.v the black.a cat.n in the park.n .


Constituent tree:
(S (NP the big dog)
   (VP chased
       (NP the black cat)
       (PP in
           (NP the park)))
   .)
```

```
  Linkage 2, cost vector = (UNUSED=0 DIS=1 AND=0 LEN=17)


    +------------------------------Xp-------------------------------+
    +--------Wd-------+       +--------Os--------+                  |
    |      +----Ds----+       |     +-----Ds-----+    +---Js---+    |
    |      |    +--A--+---Ss--+     |     +---A--+-Mp-+  +--Ds-+    |
    |      |    |     |       |     |     |      |    |  |     |    |
LEFT-WALL the big.a dog.n chased.v the black.a cat.n in the park.n .


Constituent tree:


(S (NP the big dog)
   (VP chased
       (NP (NP the black cat)
           (PP in
               (NP the park))))
   .)
```

Appendix B. Other experiments we have tried.

In this section we present some more preliminary experiments that we performed. They were discontinued for not showing promising results. The first two subsections correspond to experiments performed on the BS dataset.

B.1. Ripper and Flipper

As introduced in chapter 2, Ripper and Flipper are Inductive Rule Learning systems. One characteristic of Ripper makes it particular suitable for the ATC task: it permits to define attributes of type Set, which makes it easy to use the classic BOW representation (by simply defining one of such attributes). Each example and the rules learned are then represented by mentioning the words contained in the set attribute. This approach is presented by Cohen in his [C96] paper.

The following is an example of rules learned by Ripper in our Bacillus Subtilis dataset. Each line shows the class (Y or N), the number of examples covered (first the ones from the mentioned class, then the ones not from that class) and the word (if that words is present in the WORDS set attribute, then the example is classified as belonging to the class).

Example B1. Some rules learned by Ripper

```
Y 121 53 IF WORDS  ~  is .
Y 34 11 IF WORDS  ~  expression .
Y 22 9 IF WORDS  ~  but .
Y 17 3 IF WORDS  ~  gerE .
Y 21 13 IF WORDS  ~  transcription .
Y 12 2 IF WORDS  ~  spoIIE .
Y 10 2 IF WORDS  ~  after .
N 180 90 IF .
```

It can be noticed from that example that the default configuration of Ripper tends to learn rules that cover many examples, making them too general and not suitable for our task. Even when we know it is possible, we did not attempt to change Rippers parameters in order to obtain more specific rules. Table B1 present some preliminary results when performing 1 run of 5-fold cross-validation in the BS dataset.

Table B1. Ripper preliminary results (1 run of 5-fold cross validation)

		words
Ripper	Accuracy	0.65
	Precision	0.65
	Recall	0.64
	F1-measure	0.63

Flipper is a FOL version of Ripper. Cohen [C95c] experimented with Flipper in order to introduce context into ATC, For example they defined predicates to indicate the order of the words in a documents and the closeness between two words.

The representation we used with Flipper was similar to the one we used for the relational learner Aleph. To compare its power with Ripper and the BOW approach, each sentence was represented by several predicates each of them relating the sentence with a word that appear in it. And to further explode its relational characteristics, predicates expressing different syntactic links between two words in a sentence were used (unlike our experiments with Aleph were there was only one predicate expressing any kind of syntactic link.)

As it was the case with Ripper, the default configuration of Flipper creates very general rules, not suitable for our experiments. In particular we always obtained the rule that classifies everything as a negative example. Inspecting the output from Flipper's run, we found out that a more specific rule set was created, but discarded as the result of a pruning stage. Not being able to disable this pruning option, we decided to simply extract the learned rules from the output file as they were before the pruning stage took place.

We tried several representations both using only words and links, when leaving the genes/proteins names in the sentences and when replacing them by the syntactic marker "geneprot", and also when replacing interaction verbs by a general marker "interaction". Example 6 shows some rules learned by Flipper when using both markers replacements.

Example B2. Some rules learned by Flipper (before being pruned)

```
c(D):-n(D,interaction), pp(D,W11,interaction),
adj(D,W11,geneprot), n(D,activity);7,0.

c(D):-v(D,is),v(D,interact), n(D,sporulation);8,0.

c(D):-n(D,transcription),pp(D,geneprot,W11), v(D,are);7,1.

c(D):-n(D,interaction),v(D,is),adj(D,geneprot,W11),
subj(D,W12,geneprot);12,0.

c(D):-n(D,interaction),pp(D,W11,interaction),v(D,was),
subj(D,W13,geneprot);5,0.

c(D):-n(D,transcription),pp(D,geneprot,W11),
subj(D,is,W11);8,0.

c(D):-n(D,interaction),v(D,is),subj(D,W11,interaction),
adj(D,geneprot,W12);7,1.

c(D):-n(D,interaction),n(D,fusion);11,0.

c(D):-n(D,interaction),n(D,protein);13,3.

c(D):-n(D,interaction),pp(D,W11,interaction),
n(D,promoters);6,2.

c(D):-n(D,transcription),pp(D,geneprot,W11),v(D,interact),
pp(D,W13,transcription);18,1.

c(D):-v(D,is),adj(D,factor,W11);11,1.

c(D):-n(D,interaction),pp(D,interaction,W11),a(D,geneprot),
subj(D,W13,geneprot), n(D,promoter);3,0.

c(D):-v(D,is),a(D,e);11,2.

c(D):-;272,334.
```

It can be observed in this example that, similar to the results obtained with Aleph, each rule is very specific covering only few examples (the coverage is indicated at the end of the rule.) However, after only few rules, Flipper decides that any other example would be

considered as a negative one. As it is expected, this last rule causes it not to perform well in the classification of unseen sentences.

Some preliminary results when performing 1 run of 5-fold cross-validation in the BS dataset are presented in Table B2. Unfortunately none of the proposed representations could even equal the Ripper performance with the BOW and therefore we decided to abandon these experiments and we moved to using Aleph.

Table B2. Flipper preliminary results (1 run of 5-fold cross validation)

	Words		Links		
	No rep.	genes rep.	No rep.	genes rep.	genes and inter. rep.
Accuracy	0.62	0.60	0.59	0.61	0.59
Precision	0.65	0.64	0.66	0.61	0.64
Recall	0.51	0.53	0.38	0.68	0.42
F1-measure	0.57	0.57	0.48	0.63	0.48

B.2. Grammar Induction

Temkin and Gilder [TG03] present an approach to IE using a context-free grammar. They attempt to extract protein, genes, and small molecules (PGSMs) interaction information from biomedical articles. For this purpose they design and implement a system consisting of three stages: technical dictionaries, lexical analyzer and a context free grammar parser. It utilizes two dictionaries, one for identifying the PGSM names and the other for

identifying the interaction verbs. These dictionaries are used by the lexical analyzer that scans the sentences and returns tokens specifying the type of word found, as for example a PGSM name, an interaction verb, or a preposition, in this last case also indicating of which kind. The tokens are then received by the parser and if they respect the given grammar then it is most probably the case that the sentence denotes an interaction between the mentioned PGSMs. In their experiments they show that the system recognizes many of these sentences. Unfortunately it also recognizes sentences in which the relationship is not truth, and it missed some relevant relations. They report recall and precision of approximately 64% and 70% respectively.

For their system the grammar was built by hand, and it was our idea to try to learn it automatically. For that purpose we experimented with the automatic grammar learner Emile [ATV00]. Unfortunately we could only get access to the interaction verbs dictionary, but neither to the PGSMs dictionary nor to the lexical analyzer even when they are supposed to be available by request. The 100 abstracts dataset they used for testing was also provided, but without any annotation of the sentences where they found the interacting PGSMs. Therefore we decided to use the BS dataset, with the SwissProt list, and the interaction verbs list from this work.

Emile automatically learn a context free grammar from the positive and negative examples, and even when it managed to learned a grammar from our sentences dataset, the obtained grammar consists of too many nodes what makes it too specific and not easy to understand. In order to obtain a more general grammar we then simplified the sentences in a similar way to the one used by Temkin and Gilder, by replacing each interaction word by the token INTERACT and each protein name found in the SwissProt list by the token

PROTEIN and discarding all the other words. We then learned the underlying context free grammar, but still obtained very specific and difficult to understand rules.

We evaluated the last set of learned rule in the same training set to have an idea of how well this set could be, but the results (shown in Table B3) were discouraging low for values on the training set. For these reason and the inability of learning general and easy to understand rules, we decided to abandon this approach.

Table B3. Performance of Emile on the BS dataset (same as training set)

Emile	Accuracy	77%
	Precision	65%
	Recall	72%
	F1-measure	68%

B.3.LSI for background knowledge

The original idea of the experiments reported in this section was to study the way of improving text classification by adding highly semantic background knowledge.

As introduced in chapter 3, Zelikovitz and Hirsh [ZH01] explores the inclusion of background knowledge into a LSI representation to help ATC. They show that any text can be used as background knowledge and that there is no need of these documents being related to the categories that will be used later in the classification task

Following Zelikovitz and Hirsh approach, we considered the possibility of including as background knowledge the general thesaurus Roget [K98]. This thesaurus contains, among other things, an example sentence showing the context of use of each entry. We experimented including all these sentences into the LSI matrix representation for several different datasets. We then compared the ATC results obtained when using the features generated in these cases to those obtained when not using the background knowledge. The use of the Roget thesaurus in this way resulted in a benefit only when the datasets were very small. This agrees with what Zelikovitz and Hirsh found in their research.

These experiments were at that time abandoned as we moved the focus of our research towards Sentence Selection in technical documents. After working extensively in this area, we realize that in this particular context the use of LSI together with technical dictionaries rather than a general thesaurus could be of help. We therefore would like to consider this approach in the future.

In Appendix C we present the details of our LSI for background knowledge experiments. It includes and introduction to LSI, a review of the [ZH01] paper, a descriptions of the data sets and background knowledge used, and the general setting for the experiments, and an analysis of the results. In addition to the experiments using background knowledge, we also analyzed the performance of LSI in different collections and using different learning algorithms besides the k-Nearest Neighbor usually used with the LSI representation.

Appendix C. Latent Semantic Indexing in Text Classification: an analysis of its performance in different classifiers and the usefulness of Roget Thesaurus as background knowledge

The research presented in this Appendix addresses the usefulness of Latent Semantic Indexing (LSI) in text classification using several Machine Learning algorithms. From our experiments, it seems like its benefits are only associated to a poor BOW representation, when the collection is big enough there seems to be no improvement in using LSI. We also extend the research on the usefulness of background knowledge when using a LSI representation. In particular we study the benefits of a general thesaurus as background knowledge.

C1 Introduction

The original idea was to study the way of improving text classification by adding highly semantic background knowledge.

Following Zelikovitz and Hirsh paper [ZH01], we decided to add the background knowledge to a LSI representation. Any text can be used for this purpose, there is no need of these documents being related to the categories that will be used later in the classification task.

To start with, some of the experiments from [ZH01] were repeated, and then expanded to add experiments of our own interest, including background knowledge from the Roget thesaurus. These experiments were performed on the same small collections that [ZH01] presented, which are available on the Internet.

In the presence of unsatisfying results, new research questions were formulated, and the project changed focus to analyze the performance of LSI in different collections and using different learning algorithms in addition to the k-Nearest Neighbor usually used with the LSI representation. These new experiments were performed in subsets of the Reuters collection that we considered more convenient for the particular task at hand. Despite our effort, they did not show good results and we moved back to the smaller collections from the [ZH01] paper.

In the next section we present some preliminaries to this research, including an introduction to LSI, a review of the [ZH01] paper, and descriptions of the data sets and background knowledge used. We then describe our experiments in detail, present and analyze our results. And finally there are the conclusions and future work.

C2 Preliminaries

Latent Semantic Analysis (LSA) [LFL98] is an automatic mathematical/statistical technique for inferring contextual usage of words. It takes as input text parsed into words and separated into meaningful samples such as sentences or paragraphs, represented in a matrix of words by samples, with each cell containing the frequency with which each word appears in each sample. By applying Singular Value Decomposition (SVD) (a form of factor analysis), the matrix is decomposed into the product of three other matrices, two describing respectively the rows and columns of the original matrix as orthogonal factors, and the other one is a diagonal matrix containing scaling values (the singular values).

Given a rectangular matrix X, for example a terms by documents matrix of dimensionality $t \times d$, it can be decomposed using SVD as

$$X = T \cdot S \cdot D$$

such that the dimensionality of T is $t \times r$, the dimensionality of D is $r \times d$, and they are both orthogonal matrices, and S is a diagonal matrix with dimensionality $r \times r$, where r is the rank of X.

The matrix X can be approximated keeping only the k largest singular values of S along with their corresponding columns in T and D. The approximated matrix obtained in this way is the matrix of rank k that is closest (in the least square sense) to X.

Latent Semantic Indexing (LSI) [DDLFH90] uses LSA in order to represent the documents in a new vector space of reduced dimensionality. The idea is that by using an approximation of the original terms by documents matrix that contains only the first k linear components, the main associations among terms (and also between terms and documents) can be captured, ignoring minor differences in terminology. This introduces higher semantic meaning into the new features of the representation, and results in the possibility of documents that do not share terms still being considered close to each other.

Zelikovitz and Hirsh [ZH01] explores the inclusion of background knowledge into LSI to help text classification. The background knowledge in their experiments is of different types, sometimes being simple unlabeled examples from the same domain and other times coming from different domains. Since the matrix SVD performed by LSI does not take into consideration the label of the documents, it is simple to add these new documents to enrich the representation. The empirical results show that "background knowledge is most useful when the training set is small."

From Zelikovitz and Hirsh work you used the following small size collections:

- **Physics**: paper titles of 2 physic classes, with abstracts being used as background knowledge,
- **News**: articles from 20 different newsgroups, with unlabeled examples being used as background knowledge, and
- **Ads**: online advertisements, with documents coming from a different site (of similar nature) being used as background knowledge.

We also used the Reuters collection, which is a collection of news articles that has been used as a standard to test Text Classification methods. It consists of 135 thematic categories with the number of examples per class varying from 1 to thousands. Several splits in training/test/unused examples have been proposed, ModApté being the most commonly used.

The Rogets Thesaurus was used in our experiments to incorporate background knowledge from a general perspective. This thesaurus consists of 990 head nouns and many synonyms and other related words for each of them.

C3 Experiments and Results

C3.1 Experiments in the small collections

Our first set of experiments consisted of duplicating some of the experiments from [ZH01], obtaining the classification performance from the small collections used in that paper with and without the assistance of the background knowledge from the unused documents. We also experimented using Roget's thesaurus as background knowledge, in

two different ways: using each entry as a document and splitting each entry in paragraphs (each paragraph being considered as a separated document.)

These experiments were performed using the Telcordia LSI[14] system, and a Perl script was written in order to combine the results following the 30 Nearest Neighbor approach also used in [ZH01]. The experiments were run in the whole corpus, randomly selecting 1/5 as the test set and the rest as the training set.

The results in table C1 show the accuracy for the News and Adds datasets. It can be seen that the performance when using background knowledge slightly deteriorates in the case of the News dataset, while it improved with the Ads dataset, mainly when the Roget's thesaurus entries were split in paragraphs.

Table C1: Accuracy for the News and Adds datasets.

		30-NN
NEWS	**Without background**	0.933
	With Roget as background	0.928
	With Roget (paragraphs) as background	0.928
	With unlabeled examples as background	0.928
ADS (class employment)	**Without background**	0.738
	With Roget as background	**0.753**
	With Roget (paragraphs) as background	**0.825**
	With similar documents as background	**0.750**

[14] Telcordia LSI http://lsi.research.telcordia.com/

Thinking that the results may be biased by the partition, we did the experiment in the Physics dataset using 5-fold cross-validation. The results are shown in table C2.

Table C2: Accuracy for the Physics dataset (5-fold cross-validation)

		30-NN
Physics	**Without background**	0.831
	With Roget as background	0.606
	With Roget (paragraphs) as background	0.603
	With abstracts as background	**0.866**

In this case, the abstracts as background knowledge improved performance, probably given the overlap between the words in the titles and the abstracts. On the other hand, Roget's thesaurus as background knowledge deteriorated performance. We expected not to obtain any improvement in this case, since these are scientific articles and not many of the words in its titles would appear in a generic thesaurus.

C3.2 Experiments with Reuters

Given that the previous collections are rather small and that they use many abbreviations (and sometimes stemming,) we decided to perform the same kind of experiments in the Reuters collection, which is big and uses clean English that could, we thought, take better advantage of the thesaurus usage (since the decomposition performed by LSI will match together only the same terms.)

These experiments were again performed using the LSI system, and the 30 Nearest Neighbor Perl script. The experiments were run in the Mod Apte split (which divides the

training and test set) using a one class against the rest approach for the 10 most populated classes in the collection. The examples from the other classes were left unused, and they served as background knowledge.

The results in table C3 show the accuracy for 4 of the most populated classes using different backgrounds. Only for the Interest class the Roget's thesaurus background increased performance, the use of background knowledge deteriorated performance in the other cases.

Table C3: Accuracy in 4 classes from the Reuters collection (ModApte split)

	30-NN			
	Interest	**Ship**	**Trade**	**Wheat**
Without background	0.588	0.574	0.657	0.550
With Roget as background	**0.607**	0.494	0.657	0.550
With Roget (paragraphs) as background	0.588	0.551	---	---
With unused documents as background	0.579	0.540	---	---

In general, the performance in this corpus was very low, which made us think of new research questions, like how is the LSI representation performing compared to bag of words? and how would the LSI representation perform with other Machine Learning algorithms? These questions are addressed in the following sections.

C3.3 Incorporating LSI representation into other classifiers

In order to test the performance of the LSI representation with Machine Learning algorithms other than 30 Nearest Neighbor, we used the SVD matrix obtained when using the LSI system. The documents represented by these factors were written in an .arff file and

load into the Weka system. The following Machine Learning algorithms were tried: k Nearest Neighbor (k-NN), Naïve Bayes, and Support Vector Machine (SVM).

The results in table C4 show the accuracy for the Interest class, in the test set, using the same split as before, with and without Roget's thesaurus as background knowledge. The performance when using background knowledge increased with all the classifiers that were tested in this case. The increase was significant when using SVM.

Table C4: Accuracy using the LSI representation in different classifiers

		30-NN	**NBayes**	**SVM**
Interest	**Without background**	0.458	0.327	0.178
	With Roget as background	0.467	0.336	**0.234**

C3.4 Comparing LSI and Bag of Words representations

From the previous sections, and previous research, we could think that the background knowledge is useful in some cases when using the LSI representation, but how useful is this representation by itself? In this section we compare the performance of the widely used BOW representation with that of the LSI representation.

Again, in order to obtain the representations, we used the LSI system, that provided us with both the SVD matrix and the BOW matrix. And again, the documents represented by the LSI factors or those represented by words were written in an .arff file and loaded into the Weka system and several Machine Learning algorithms were tried.

Table C5 shows the results in the Grain class of the Reuters collection when using the BOW and the LSI representations and different classifiers. These experiments were performed using 10-fold cross validation.

Table C5: Accuracy for the BOW and LSI representations (10-fold cross-validation)

		30-NN	Naïve Bayes	SVM
Grain	**BOW**	96.432	83.013	97.157
	LSI	96.366	**89.137**	97.259

The performance when using LSI did not vary much with the 30-NN and SVM classifiers. It improved significantly with Naïve Bayes when using LSI, but it has to be noticed that it did so by increasing only the number of negative examples correctly classified. Because of the big imbalance of the Reuters collection there is lot of opportunity of this to happen. We therefore decided to perform new experiments in a two-classes balanced subsets of Reuters.

The following table shows the accuracy for 2 of these subsets when using the BOW and the LSI representations and different classifiers. These experiments were performed using 5-fold cross validation.

Table C6: Accuracy for the BOW and LSI representations (5-fold cross-validation)

		30-NN	SVM	Naïve Bayes	Tree
Wheat-Corn	**BOW**	60.4555	73.913	70.1863	76.3975
	LSI	54.0206	67.2165	57.1134	62.8866
Grain-Crude	**BOW**	93.9139	98.1273	97.4719	---
	LSI	85.1402	95.4206	71.9626	---

The performance when using the LSI representation was always worse than when using the BOW representation. Our reasoning was that it might be because of the high correlation between the words and the class in the Reuter collection (i.e.: the documents from the class corn would very often content either the word corn or maize.) Since this is not the case in other collections, we wanted to see what would happen if that was not the case, and we built new subsets of Reuters where the words closely related to the class were removed.

Similar experiments as before were performed in these collections and the results are shown in the following table.

Table C7: Accuracy for the BOW and LSI representations (without keywords)

		5-NN	30-NN	SVM	Naïve Bayes	Tree
Wheat-Corn (without keywords)	**BOW**	60.2484	59.4203	62.1118	64.8033	63.147
	LSI	55.6701	60.0000	59.7938	55.8763	48.2474
Grain-Crude (without keywords)	**BOW**	92.2285	92.0412	97.3783	96.9101	95.1311
	LSI	85.0467	83.1776	95.5140	68.6916	93.6449

As expected, the performance deteriorated when using the BOW representation (with respect to the one showed in table C6). Unfortunately it also happened when using the LSI representation, and the performance of LSI was still worse than that of BOW in every case (except for the Wheat-Corn without keywords subset with 30-NN).

C3.5 New experiments with small collections

Our last attempt to try to explain the very low performance of the LSI representation with respect to the BOW representation attributes it to the size of the collection. Given the large number of documents and their length in the subsets tried, the BOW representation may be doing already very well. Also, the small number of factors used in the LSI representation may not be enough to capture all the information.

We decided then to go back to the small collections used in [ZH01], and we tried the BOW and the LSI representations with several classifiers in the Ads and News collections using 5-fold cross validation. The results are shown in the following table.

Table C8: Accuracy for the BOW and LSI representations in the small datasets.

		5-NN	**30-NN**	**SVM**	**Tree**
Ads	**BOW**	87.9971	---	92.9308	---
	LSI	74.7059	---	89.4853	---
News	**BOW**	78.8321	83.2117	91.9708	72.1168
	LSI	88.064	80.2038	92.1397	84.2795
	LSI with background	88.7918	80.2038	91.8486	***91.9942***

The performance of LSI was still worse than that of BOW in the Ads collection, but very promising with the News collection. The best performance for the News collection was obtained by SVM using LSI.

With Nearest Neighbor the results varied when using k=5 or k=30, the performance decreased a little when using LSI for k=30, but increased significantly when using LSI for

k=5, being these results the best for the classifier and the second best after SVM (when not using background knowledge.)

In the case of Decision Trees, LSI performed much better than BOW even when the algorithm is not based in comparing distances in the vector space representation. We believe that might be caused by a smoothing in the hypothesis space produced by bringing semantically related terms together. The Roget's background knowledge was of great help in this case, bringing the performance very close to that of SVM.

C4 Conclusions and future work

It seems like quite a bit of research has been done in LSI for text retrieval, but not much about LSI usefulness in text classification. In addition, LSI research has not study the benefits of this representation in Machine Learning algorithms other than k-NN.

From our experiments, it seems like its benefits are only associated with a poor BOW representation, when the collection is big enough there seems to be no improvement in using LSI (it may makes it even worse.)

We showed some evidence that for certain collections, using the LSI representation may be beneficial even for some algorithms not based on comparing distances in the vector space (as Decision Trees.) We plan to keep researching this issue with other small collections (and smaller subsets of Reuters) and different classifiers.

With respect to the usefulness of background knowledge in text classification, after our experiments we can say that it depends on the collection and the source of the background knowledge itself.