

# Institutionen för datavetenskap

Department of Computer and Information Science

Final Thesis

Semantic Analysis Of Multi Meaning Words Using  
Machine Learning And Knowledge Representation

by

Marjan Alirezaie

LiU/IDA-EX-A- -11/011- -SE

2011-04-10



## Linköpings universitet

Linköpings universitet  
SE-581 83 Linköpings, Sweden

Linköpings universitet  
581 83 Linköpings



**Final Thesis**

Semantic Analysis Of Multi Meaning Words Using  
Machine Learning And Knowledge Representation

by

**Marjan Alirezaie**

LiU/IDA-EX-A- -11/011- -SE

Supervisor, Examiner : **Professor Erik Sandewall**

Dept. of Computer and Information Science  
at Linköpings Universitet





## Abstract

The present thesis addresses machine learning in a domain of natural-language phrases that are names of universities. It describes two approaches to this problem and a software implementation that has made it possible to evaluate them and to compare them.

In general terms, the system's task is to learn to 'understand' the significance of the various components of a university name, such as the city or region where the university is located, the scientific disciplines that are studied there, or the name of a famous person which may be part of the university name. A concrete test for whether the system has acquired this understanding is when it is able to compose a plausible university name given some components that should occur in the name.

In order to achieve this capability, our system learns the structure of available names of some universities in a given data set, i.e. it acquires a grammar for the microlanguage of university names. One of the challenges is that the system may encounter ambiguities due to multi meaning words. This problem is addressed using a small ontology that is created during the training phase.

Both domain knowledge and grammatical knowledge is represented using decision trees, which is an efficient method for concept learning. Besides for inductive inference, their role is to partition the data set into a hierarchical structure which is used for resolving ambiguities.

The present report also defines some modifications in the definitions of parameters, for example a parameter for entropy, which enable the system to deal with cognitive uncertainties. Our method for automatic syntax acquisition, ADIOS, is an unsupervised learning method. This method is described and discussed here, including a report on the outcome of the tests using our data set.

The software that has been implemented and used in this project has been implemented in C.

**Keywords :** Machine Learning, Supervised Learning, Unsupervised Learning, Ontology, Decision Tree, ADIOS, Grammar Induction



## Acknowledgements

I would like to thank Professor. Erik Sandewall, for his supervision and his support during the pursuit of this work. Under his guidance, I acquired confidence that served me throughout the work. In addition, his concise but advantageous comments marvelously showed me the right way of thinking to design the phases of the project step by step.

I would also like to thank all of the people that have worked on the various software packages that I have used, such as LaTeX and C# compiler, as well as the people who have worked on the preparation of the various corpora I have used.

# Contents

<b>List of Figures</b>	<b>ix</b>
<b>List of Tables</b>	<b>x</b>
<b>1 Introduction</b>	<b>1</b>
1.1 What is a Language? . . . . .	1
1.2 What is a Grammar? . . . . .	2
1.3 Motivation, Problem Statement . . . . .	2
1.4 Method . . . . .	3
1.5 Thesis Outline . . . . .	5
<b>2 Language Learning Algorithms</b>	<b>7</b>
2.1 Introduction . . . . .	7
2.2 Machine Learning . . . . .	8
2.2.1 Learning Method Categorization . . . . .	9
Supervised Learning . . . . .	9
Unsupervised Learning . . . . .	9
Reinforcement Learning . . . . .	10
2.2.2 Applications of machine learning . . . . .	10
2.3 Natural Language Processing . . . . .	11
2.3.1 Syntactic Analysis . . . . .	12
2.3.2 Semantic Analysis . . . . .	15
2.3.3 Word Sense Ambiguity . . . . .	15
2.4 Knowledge Representation . . . . .	18

2.4.1	Ontology . . . . .	20
<b>3</b>	<b>Decision Tree</b>	
	<b>Supervised Learning Algorithm</b>	<b>23</b>
3.1	Introduction . . . . .	23
3.2	Decision Tree Learning . . . . .	23
3.2.1	ID3, Iterative Dichotomiser 3 . . . . .	26
<b>4</b>	<b>ADIOS</b>	
	<b>Unsupervised Learning Algorithm</b>	<b>29</b>
4.1	Introduction . . . . .	29
4.2	Grammar Induction . . . . .	30
4.2.1	Methods of Grammar Induction . . . . .	30
4.3	The ADIOS Algorithm . . . . .	31
4.3.1	The ADIOS Algorithm Structure . . . . .	32
4.3.2	The ADIOS Algorithm Phases . . . . .	33
4.3.3	The ADIOS Algorithm Parameters . . . . .	36
<b>5</b>	<b>Methodology</b>	<b>39</b>
5.1	Introduction . . . . .	39
5.2	Decision Tree Induction . . . . .	39
5.2.1	Data Input/Output Structure . . . . .	40
5.2.2	Program Modules and Components of the Application . . . . .	42
5.2.3	Evaluating the Usefulness of the Learner . . . . .	46
5.3	ADIOS Algorithm . . . . .	48
5.3.1	Setup . . . . .	48
<b>6</b>	<b>Conclusion</b>	<b>55</b>
6.1	Comparing two implemented learning methods . . . . .	55
6.2	Future work . . . . .	57
	<b>Bibliography</b>	<b>58</b>

# List of Figures

1.1	A simple view of relations among features . . . . .	5
1.2	Scope of the work . . . . .	5
2.1	Converting a list of tokens to a parse tree(hierarchical analysis)	13
2.2	A parse tree for a simple sentence . . . . .	14
3.1	Creating a Classifier from a Set of Examples . . . . .	24
3.2	Classifying a New Case . . . . .	24
3.3	ID3 flow chart . . . . .	28
4.1	A set of nodes is identified as a pattern when a bundle of paths traverses it . . . . .	35
5.1	Relation between the $\alpha$ parameter and the measurements performed . . . . .	50
5.2	Relation between the $\eta$ parameter and the measurements performed . . . . .	51

# List of Tables

2.1	Words with the greatest number of senses in the Merriam-Webster Pocket Dictionary . . . . .	16
5.1	List of main attributes . . . . .	40
5.2	Samples of records in data set . . . . .	41
5.3	List of available structures in data set . . . . .	43
5.4	The first level of Entropy Calculation . . . . .	45
5.5	Matrix for evaluating the Learner . . . . .	46
5.6	Evaluatinn results of the tree . . . . .	47
5.7	Relation between the $\alpha$ parameter and the measurements performed. . . . .	49
5.8	Relation between the $\eta$ parameter and the measurements performed. . . . .	49
5.9	Number of Nodes . . . . .	52
5.10	ADIOS-final parameter settings . . . . .	53

# Chapter 1

## Introduction

This work is an investigation into cognitive methods and their effects on the process of language learning. The analysing methods that we have used are based on machine learning with special focus on natural language processing and computational linguistics. Unlike other major projects working directly on natural languages and grammars, this thesis addresses a special domain of work and it simulates a new language based on the natural language. The following sections give short introductions and an outline of the entire report.

### 1.1 What is a Language?

Language is a systematic means of communicating by the use of sounds or conventional symbols in the form of sentences generated from a sequence of words which are known in formal grammars as terminal symbols [1]. The meaning of sentences of a language adheres to the meaning of terminal symbols or morphemes which are known as the smallest linguistic units that has meaning. In addition, the order of words in a sentence plays a major role in the meaning of a sentence.

One of the outstanding features of an intelligent agent is conversation. It



is a process in which agents should be able to make novel sentences related to the topic of the conversation. To have such agents acting like human, we should implement a linguistic process that creates a meaningful sentence. Human languages have structures that determine the formation of novel sentences. These structures provide some rules for an agent equipped by a lexicon, to create correct sentences. It leads us to a formal definition of a language grammar.

## 1.2 What is a Grammar?

A grammar is a representation or a description of a language [1]. In linguistics, a grammar is a set of structural rules that governs the composition of sentences, phrases, and words in a given natural language. It includes several topics, in particular morphology, syntax and semantic analysis.

Recent research in the realm of linguistics, especially computational linguistics, has concentrated on the cognitive process of grammar induction and its automation. Simply, the goal of these researches is discovering a grammar by performing a set of operations on a corpus of data. In recent decades, many machine learning approaches have been released to improve the accuracy and performance of computations in the automatic process of grammar induction. For advancement in this respect, many linguists, psychologists, cognitive scientists and other researchers apply neuro-cognitive techniques to linguistics problems. Because of achievements acquired in this domain, focusing on automatic induction of linguistic structure of natural languages by humans seems resonable. The scientific question of this thesis is indirectly related to this issue.

## 1.3 Motivation, Problem Statement

Agents that understand sentences in a natural language and machines that are able to communicate in human languages are really exciting if they can understand concepts undelying of our words. Imagine computers working as advisers of their end users offering their recommendations using speech.

Or think about a machine that can write a meaningful essay related to a given topic. The road towards such goals may be divided into several phases.

One of the most important research topics in this respect is semantic analysis. To be able to understand the concept of a text or a piece of speech, a system needs to know the constructor units of the context. It means that the first requirement for such a system is a domain vocabulary. More known words, higher level of understanding [2]. However, the real vocabularies are huge. A high school graduate has learned on average, about 60,000 words [3]. Therefore, when designing an agent that can learn the meanings of such a large volume of words, it would be interesting to study the process of learning in children, the word storing structures and the data retrieval methods they use. Conceptual analysis in human brains is performed by using morphological structures of words and their lexical representation [2].

In this thesis, we are going to model the syntax acquisition in an agent and focus on methods for resolving ambiguities in multi meaning words. A target application with a data set containing the international universities names is defined. The task for our system is to construct a correct name according to the meanings of the words in it and based on what it has learnt. More specifically it shall guess possible names for a university according to the constituent words which may be e.g. a country name, a city name or the name of a scientist. In this process it should get help from the grammar which is the output of the training phase.

## 1.4 Method

In the first step of approaching the chosen problem, we gathered about 1500 university names from Europe, America and Asia. This study showed that the most frequently used objects for naming a university are the following:

- Type of institution(university, college,...)
- Country name

- City name
- Famous persons(scientist, poets, leaders)
- Major disciplines offered by the university(science, engineering, economics, art, etc.)
- Dates(important dates in a history of a country may be chosen as a university name)

We used two different types of learning methods. One of them is a supervised learning technique using a decision tree and another one is the unsupervised technique which implements the ADIOS algorithm and induces the grammar existing in data.

For the supervised learning phase, we need to have a set of labeled data as the input for the algorithm. The above mentioned set plays the role of input for our supervised learning method. In this set the class number of each university name syntax is indicated and the output of the application is a decision tree which is in the form of if-else statements. This output can be embedded in most programs in structural languages such as Java or C#.

In the unsupervised training phase of the work we still need a set of input data. In this case the set should not be labeled. The task of the algorithm is to induce the hidden grammar in our data. The output of the work will therefore be a set of classes demonstrating the syntax of different university names.

The application should learn the sequence and order of words in a name. When we observe a set of objects or a sequence of words, we perceive the relations between them. This suggests that we should have a relational hierarchy for the connections between features in a name. We can see the relational structure as a semantic network, whereby we can represent semantic relations among concepts as well. Figure 1.1 shows a simple example of such relations in our problem domain.

For both the learning methods that we use, we also create a semi-ontology

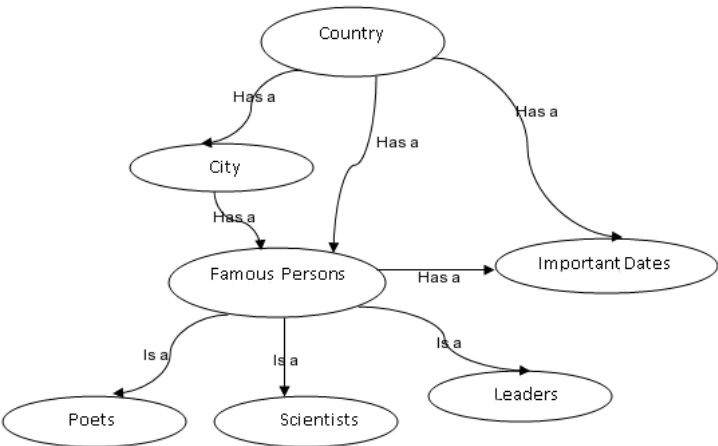


Figure 1.1: A simple view of relations among features

structure and use it for solving some problems regarding the ambiguity of multi meaning words in the system.

### 1.5 Thesis Outline

The figure 1.2 shows the main domain of work in this thesis and the organization of this report:

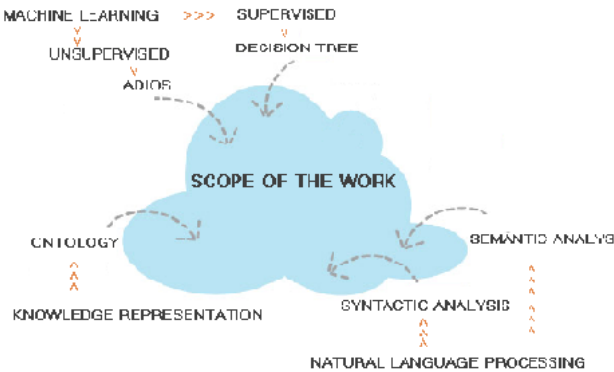


Figure 1.2: Scope of the work

Chapter 2 provides a background for machine learning and different types of learning. It then gives a short introduction to the supervised and unsupervised learning algorithms that are implemented in this thesis. The rest of the chapter focuses on Natural Language Processing in relation to research in machine learning. It is followed by a short discussion of the two main NLP topics, namely syntactic and semantic analysis, and also an introduction of the problem of Word Sense Disambiguation. Knowledge Representation is the final part in chapter 2 which focuses on the knowledge, methods and structures that are used for retrieving knowledge and for semantic processing for example for ontology.

Chapter 3 introduces decision trees as the supervised method of learning that we have used in this work. In this section we describe ID3 and the parameters that are needed in its implementation.

The third part, Chapter 4, is dedicated to the ADIOS algorithm which is the unsupervised learning method that we use in this project. Grammar induction is discussed in section 4.2 which is followed by short descriptions of some similar algorithms. The following part introduces the details of the algorithm, its structures, phases and parameters.

The rest of this report describes and discusses our implementation and experiments.

Chapter 5 represents the details of the work in supervised learning methods using decision trees, and the conclusions that we have drawn from the implementation and the experience with it. The precision of the work is measured using our selected test cases. This chapter continues with our work on unsupervised learning using the ADIOS method, with the main emphasis on grammar induction.

Finally, Chapter 6 summarizes the results and the comparisons between the two methods in this work. Future work and related interesting fields of works are introduced in the last section of this chapter.

## Chapter 2

# Language Learning Algorithms

### 2.1 Introduction

This chapter contains techniques that we will use to implement the application which is expected to solve the problem statement of this thesis. We discuss theories related to some of the methods that are needed for a better understanding of points and techniques we will employ in next chapters.

The structure of this chapter is as follows. Section 2.2 is a brief discussion of machine learning. This section has some sub parts introducing the main categorization of learning methods, the supervised decision tree learning algorithm, and the unsupervised ADIOS method. Applications of machine learning are introduced in the last part of this section. Section 2.3 is about Natural Language Processing, with sub sections on machine learning techniques dedicated to natural language analysis and on the linguistic problems of syntax analysis, semantic analysis and word sense disambiguation. Finally we address knowledge representation as the topic of section 2.4, with a short introduction of ontology which is a hierarchical structure of concepts.

## 2.2 Machine Learning

By using algorithms, computers can solve problem statements. An algorithm is a sequence of commands run to transform input data into desired output forms. For some tasks we are interested in finding the most efficient algorithm with respect to time, space or both. For some other tasks, however, there is no specific algorithm. For example, in signature recognition the input is a bitmap file containing a signature and the output should be yes or no indicating whether the file belongs to a specified person or not [4]. In such cases we would like machines to be able to learn the solution by processing samples of input data for such problems. This type of algorithms are trustful, because we believe that there is a process which explains the data we observe and that in a reasonable amount of data we can identify certain patterns. Although we may not be able to identify details of patterns completely, at least we can detect certain regularities with an acceptable certainty [5].

By definition, machine learning is a scientific discipline studying how computer systems can improve their performance based on previous experiences. The type of prediction which uses machine learning methods on large databases is called data mining [5]. To be more intelligent, systems should have the ability to learn. It means that applications should be able to adapt themselves to changes that may occur in environments. In situations in which agents work in unstable environments, designers can not consider all possible events and states. Instead, by using machine learning algorithms agents are capable of learning different situations in the environment so they can foresee future events [4].

Building systems that are able to adapt to their environments and learn from past experiences is interesting and has become a main goal in many research fields such as computer science, neuroscience, mathematics and cognitive science. The most popular fields of machine learning studies are in Computer Vision, Robotics, Speech Recognition and Natural Language Processing (NLP).

### 2.2.1 Learning Method Categorization

The core task of machine learning algorithms is making inference from samples using mathematical and statistical methods. There is a general categorization for such algorithms [4]:

- Supervised learning
- Unsupervised Learning
- Reinforcement Learning

#### Supervised Learning

The supervised learning algorithm tries to learn parameters for a function that maps inputs to desired outputs. It is the core of classification methods. In a classification problem, a set of input-output examples of data is available, called training set. The learner uses it for adjusting the parameters of the mapping function. After building this function, the classifier is ready to receive an input and transform it using the generated function. The output shows the closest class to which the input belongs [5]. In other words, the learning algorithm is equipped with evidence about the classification of data. This type of learning includes most of the traditional types of machine learning [6].

The decision tree learning method is based on supervised learning and is our selected algorithm in the first phase of this work.

#### Unsupervised Learning

In unsupervised learning the learner simply receives input data without supervised target outputs. This method tries to find patterns in data and determines by itself how they are organized. Clustering is a very classic example of these learning algorithms in which the learners are given unlabelled input only [5]. As the collection of labelled data is hard or limited in quantity, unsupervised learning methods are more interesting for most applications.



As mentioned above, clustering is a standard technique for unsupervised learning. In this method, data points are grouped together according to their similarities and closeness in some selected feature space. This can be accomplished by algorithms such as K-means in which the number of clusters is determined in advance, or a neural network based method, or even a hierarchical clustering algorithm [6].

ADIOS learning is a type of unsupervised learning that we use in the second phase of this work.

### **Reinforcement Learning**

This is a type of learning where the learner interacts with the environment. In this method the learner tries to learn from the sequence of its actions. In other words, it essentially works based on trial and error learning as it chooses its actions based on its past experiences as well as its new choices [7]. In this work we do not have any need for this type of learning.

#### **2.2.2 Applications of machine learning**

Machine learning research has developed learning algorithms that are used in commercial systems for pattern recognition, speech recognition, computer vision, natural language processing and a variety of other tasks such as data mining to discover hidden patterns and regularities in large data sets [8].

Machine learning combines Computer Science and Statistics. Nowadays there is however an emerging third aspect of machine learning which is related to the study of human and animal learning in Neuroscience, Psychology, Linguistic and other related fields. To date, the knowledge that has been acquired from this approach to machine learning is however limited in comparison with the amount of knowledge we have gained from Statistics and Computer Science, the limiting factor being the current understanding of Human learning [8].

Nevertheless, the synergy between these two sides of learning is growing and there are some significant real-world applications. For example, in computer vision many applications ranging from face recognition systems to systems that automatically classify microscopic images of cells are developed using machine learning. In Robot control, machine learning enables systems to acquire control strategies for helicopter flight and aerobatics [8]. Natural language processing or NLP is a rich application domain for the field of machine and human learning, including activities such as Speech Recognition, Machine Translation, Auto Text Summarization and especially semantic analysis which is the main focus of this thesis.

## 2.3 Natural Language Processing

Natural Language Processing is an area of research that has developed algorithms that allow computers to process and understand human languages. Its broad range is from basic research in computational linguistics to applications in human language technology, and covers fields such as sentence understanding, grammar induction, word sense disambiguation, and automatic question answering [9]. The main goal of such analysis is to obtain a suitable representation of text structure and make it possible to process texts based on their content. This is required in various applications, such as spell and grammar checkers, text summarization, or dialogue systems.

The automatic processing of languages contains several levels of analysis as follows:

- Morphological Analysis
- Syntactic Analysis
- Semantic Analysis
- Knowledge Representation

**Morphological Analysis** is related to the grammatical forms of words. It uses a set of tags that identify the possible grammatical classes of a given word. Automatic analysis of word forms is needed for grammar checker

tools.

**Syntactic Analysis** checks whether a given stream of input words is a correct sequence according to the grammar of a given language. The derivation tree is the most widely used structure for syntax analysis purposes. It provides a description of the syntactic structure of a sentence, whereby agents can understand relationships among words. Syntactic Analysis is a building block in machine translation systems. In addition, dialogue systems and systems for punctuation correction work on a syntactic analysis base.

**Semantic Analysis** is the most complex phase of language processing as it works on the results of all mentioned disciplines. No completely adequate general solutions have been proposed for this topic. Many algorithms and theories have been released, but much work is needed to optimize the solutions and overcome many problems caused by errors on lower processing levels, specially in machine translation systems.

**Knowledge Representation** is a main pillar of artificial intelligent applications which need reasoning and inferring knowledge from information. The mentioned phases and techniques are not enough to understand the content of a text properly. To do it, we also need to process certain knowledge and facts about the world. These facts and concepts such as "birds can fly" really help for reasoning and extract the concept and meaning of a sentence or a text. For this purpose, we need to gather data and information, and to represent them suitably.

We continue this discussion with a focus on syntactic and semantic analysis by considering the concept of knowledge representation and its use in implemented system.

### 2.3.1 Syntactic Analysis

We typically use a grammar to define the syntax of a language. In other words, a grammar is a specification of a language syntax, but not of its semantics. Generally to specify the syntactic structure of a language we

use a special type of grammar, context free grammar or CFG, which is defined by the following set (T, N, P, S):

- T is a set of terminals(vocabularies of the grammar or tokens)
- N is a set of non-terminals(variables representing language constructs)
- P is a set of production rules(defines possible children for each non-terminal)
- S is a start symbol which belongs to the non terminal set(N) and is a root for any parse tree

According to this type of grammars we expect a text to contain syntactically correct sentences. Syntactic analysis or parsing is a type of processing a text which has been made from a sequence of tokens(words). The goal of this analysis is determining the grammatical structure applied in the text. Interpreters and compilers use parsers for checking the syntax of sentences. This checking is done by using data structures known as parse trees or abstract syntax trees which are hierarchical structures whose interior nodes are marked by non terminals and whose leaf nodes are labelled by terminals of the grammar.

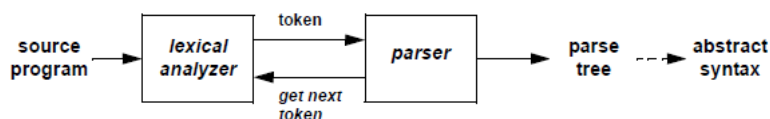


Figure 2.1: Converting a list of tokens to a parse tree(hierarchical analysis)

As it is clear from figure 2.1 the lexical analyser that is used by any syntactic analyser separates the sequence of input characters into tokens. The parser marks each token with a syntactic tag. From a linguistic point of view, a tag denotes subjects, objects, main verbs, etc. But according to a grammar definition, we can have tags with different roles but with same purposes [10]. For example, in a human language, a sentence may contain noun phrase, a verb and a noun phrase in that order. A simple grammar

is some thing like this:

$S \rightarrow NP\ V\ NP$   
 $NP \rightarrow A\ N$   
 $V \rightarrow \text{eats} \mid \text{loves} \mid \text{hates}$   
 $A \rightarrow \text{a} \mid \text{the}$   
 $N \rightarrow \text{dog} \mid \text{cat} \mid \text{rat}$

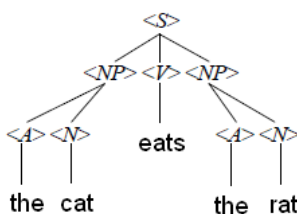


Figure 2.2: A parse tree for a simple sentence

For processing a sentence like *"the cat eats the rat"* the parse tree in figure 2.2 shows that the sentence is syntactically correct as all its leaves belongs to the terminal set. In this grammar,  $S$  is the start symbol,  $NP$ ,  $V$ ,  $A$  and  $N$  are non-terminals, and the terminal tokens are the rest of words such as *eats*, *loves*, *hates*, *a*, *the*, *dog*, *cat* and *rat*.

Grammatical ambiguity occurs when we encounter several derivations of a sentence. Solutions involving constraints on grammars' production rules exist to overcome these issues. But, especially in human languages, we have to find a way to apply both syntactic and semantic restrictions as early as possible in the language analysis process in order to restrict the growth of ambiguities [11].

In this thesis, we create a new language with its production rules, terminal and non-terminal sets. The syntactic analyser part of the program constructs a parse tree to check whether the input is correct or not with respect to syntax. Solutions used to resolve probable ambiguities in the grammar are a mixture of general methods such as left derivation parse trees, and techniques using the meaning of words and the semantic net be-

hind each token in the system. The details of this solutions are discussed in later chapters.

### 2.3.2 Semantic Analysis

In designing and implementing cognitive systems we study interaction and consider its implementation for intelligent agents. One of the fundamental means of interaction is language [12].

The bottleneck of many applications in which semantic processing is the main task is lexical, i.e multiple meanings of a word. Notice then that *meaning* is a notion in semantics that is classically defined as having two components [13]:

- Reference: anything in the referential realm (anything, real or imagined, that a person may talk about) is denoted by a word or expression. In a simpler definition, the reference of a word is the thing it refers to.
- Sense: the system of paradigmatic and syntagmatic relationships between a lexical unit and other lexical units in a language. In other words, it is that part of the expression that helps us to determine the thing it refers to.

### 2.3.3 Word Sense Ambiguity

A language can be ambiguous, so that many words of this language may be interpreted in multiple ways and have multiple meanings depending on the context in which they occur. There are many words with a large number of common meanings. Some such English words are listed in Table 2.1 [14].

There are three types of lexical ambiguity: Polysemy, Homonymy and Categorical Ambiguity.

Polysemous words are those whose several meanings are related to one another. For example, the verb *open* has these related meanings: *expanding*, *revealing*, *moving to an open position* and so on.

Word	No.Of Senses
Go	63
fall	35
run	35
draw	30
way	31
strike	24
wing	20

Table 2.1: Words with the greatest number of senses in the Merriam-Webster Pocket Dictionary

Homonymous words have multiple meanings with no relations among them. As an example we can point to the word *bark*, with two different meanings, *the noise a dog makes* and *the stuff on the outside of a tree*.

Categorical ambiguous words are those whose syntactic categories may be different. The word *sink* as a verb means *becoming submerged* and as a noun means *plumbing fixture*.

Before considering these definitions, we have to take notice of the environment and domain we are working with and identify the type of ambiguity we may encounter in the problem statement, and then apply suitable solutions [14].

In this work, as we said in the introduction chapter, the domain of words contains some different categories such as country and city names, name of famous persons in the history of a country and fields of education courses (discipline) names. Among these categories of objects we may encounter both homonymous and polysemous words. For example, the word *Wood* can be used as a persons' family name and also as a geographical area with many trees (name of a region) or even as a discipline. Besides these issues, we found another type of word ambiguity which may occur in multi lingual systems. A word can be the name of a person in one language while it

means a name of another entity such as a discipline or a region in another language.

Word sense disambiguation (WSD) is the ability to computationally determine the sense of a word (commonly accepted meaning of a word), which is activated by its usage in a particular context [15].

A more formal definition of WSD is as follows: we can view a text  $T$  as a sequence of words  $(w_1, w_2, \dots, w_n)$ , and we can formally describe WSD as the task of assigning the appropriate sense(s) to all or some of the words in  $T$ , that is, to identify a mapping  $A$  from words to senses, such that  $A(i) \subseteq \text{SensesD}(w_i)$ , where  $\text{SensesD}(w_i)$  is the set of senses encoded in a dictionary  $D$  for word  $w_i$ , and  $A(i)$  is that subset of the senses of  $w_i$  which are appropriate in the context  $T$ . The mapping  $A$  can assign more than one sense to each word  $w \in T$ , although typically only the most appropriate sense is selected, that is,  $|A(i)| = 1$  [15].

In another view, WSD can be considered as a classification task, so that the word senses are the classes and a classification method is used to assign each occurrence of a word to one or more classes based on the evidence from the context and from external knowledge sources [15].

To be able to disambiguate words, most language processing systems have focused on both the context in which the word occurs and local appearance of the word in a sentence using methods of semantic interpretation and knowledge representation. With respect to the semantic analysis of languages and word disambiguity, several techniques have been used. Some of the more widespread techniques are as follows [14]:

Symbolic machine learning such as ID3 decision tree algorithms, graph based clustering and classification such as Ward's hierarchical clustering, statistical-based multivariate analyses such as latent semantic indexing and multi dimensional scaling regressions, artificial neural network-based computing and evolution-based computing such as genetic algorithms [15]. For this thesis we choose the symbolic machine learning method ID3 which will be discussed in chapters 3 and 5.



What is common for all these methods is the result of the semantic analysis process which can be represented in the form of semantic networks, decision rules or predicate logics. Recently, to develop and empower the existing knowledge bases, many researches have focused on integrating such results with human-created knowledge structures such as ontologies in order to have a more precise and reliable information retrieving process from large scale knowledge bases. In the following section we shall focus on the different types of knowledge resources that may be used for semantic processing tasks and the representation of knowledge in those structures.

## 2.4 Knowledge Representation

Knowledge is the fundamental component of WSD. Until now, there is no agreement on the definition of knowledge, but the following definition of knowledge is mostly acceptable. It is awareness, consciousness or familiarity of a fact or situation gained by experience or learning. The knowledge helps to understand the meaning of a subject with the ability to use it for a specific purposes [16]. There are different types of knowledge sources providing data which essentially associate senses with words. One may distinguish between structured and unstructured resources, with the following sub categories [15].

**Unstructured Resources** include:

**Corpora:** are samples or collections of real world texts and are used in corpus linguistics for learning language models. They can be used for both supervised and unsupervised learning approaches [15]. Corpora can be in a raw or annotated representation. The Brown Corpus with a million words, The British National Corpus (BNC) with a 100 million word collection of written and spoken samples of the English language, the American National Corpus which includes 22 million words of written and spoken American English language are examples of raw corpora. SemCor is a sense-tagged corpus with 352 texts tagged with around 234,000 sense annotations is an example of large annotated corpus. Corpus annotation assigns a part-of-

speech tag and morphological features to each word.

**Collocation resources:** These types of resources register the tendency for words to occur regularly with others. Some examples are the Word Sketch Engine, JustTheWord and the WebIT corpus which contain frequencies for sequences of up to five words in a one trillion word corpus derived from the Web.

**Other resources:** such as word frequency lists, stoplists, domain labels, etc.

**Structured Resources** include:

**Thesauri:** containing information about relationships such as synonymy, antonymy and further relations. One of the thesauri mostly used in the field is WSD Rogets International Thesaurus containing 250,000 word entries.

**Machine-readable dictionaries (MRDs):** have been used for natural language processing since the 1980s, when the first dictionaries were made available in electronic format. The Oxford Advanced Learners Dictionary of Current English is an example of this type of knowledge source. WordNet is one of the most utilized resources for word sense disambiguation in English and is one step beyond common MRDs. It encodes a rich semantic network of concepts and it is therefore usually described as a computational lexicon.

**Ontologies:** These are specifications of conceptualizations(an abstract view of the world) of specific domains of interest, usually including a taxonomy and a set of semantic relations. WordNet and its extensions can also be considered as ontologies. The Unified Medical Language System(UMLS), which includes a semantic network providing a categorization of medical concepts is another example.

As ontology is a base structure in this work, we shall continue the discussion of ontologies and the ways we can make them and work with them in the next section.

### 2.4.1 Ontology

The root of the word ontology refers to the major branch of philosophy known as metaphysics and deals with issues concerning what entities exist or can be said to exist, and how these entities can be grouped through a hierarchy according to their similarities and differences [17]. An ontology is a structured system of categories or semantic types whereby the knowledge about a certain domain can be organized through the categorization of the entities of the domain which are known as "types" in the ontology [18]. In other words, ontology is the study of the categories of things that exist or may exist in some domain [17].

The major difference between ontology information and common information is that an ontology expresses semantic structure [19]. The goal of building an ontology is having a structure determining the set of semantic categories which properly reflects the particular conceptual organization of the domain of information on which the system works, and finally obtaining a sensible optimization of the quantity and quality of the retrieved information [18]. In most Language Engineering tasks such as content-based tagging, word sense disambiguation and multilingual translations, ontologies have a crucial role for identifying the lexical contents of words.

The best formal definition of ontology is the one offered by John F. Sowa, a computer scientist who developed the theory of conceptual graphs. He defines it as "a catalogue of the type of things that are assumed to exist in a domain of interest D, from the perspective of a person who uses a language L for the purpose of talking about D" [18]. For example, if the ontology only contains animals, it does not have any reason to give information about vehicles, unless they are categorised as animals. For the specific purpose of this thesis, the domain that we are going to focus on contains names of cities and countries, the educational majorities, and names of scientist and other famous persons. The task in the phase of designing the system's ontology is highlighting those connections and regularities that are the most needed for the given purpose.

The degree of complexity of ontology design depends on the type of knowl-

edge represented. It means that the knowledge may be general knowledge or domain-specific (terminological) knowledge. Terminological knowledge is structured and homogeneous while general knowledge is very loosely structured and heterogeneous. Another parameter affecting the design process of an ontology concerns the choice between a multi-purpose or a usage-specific ontology [19]. For instance, in this work we are interested in extracting information of correlations between words in an university name. An ontology which is particularly suitable to this goal should include fine-grained classifications of cities, regions, scientists and educational disciplines. It should take into account particular relations between these entities as well.

Designing a usage-specific ontology for an area of terminological knowledge has some advantages such as efficiency of representation and easy implementation. However, its main weakness is the inability to be cross-domain portable. This means that very specific ontologies are not easily reusable which causes developers to do some adjusting or even design a new structure for every new system with a specific domain of work [18]. General ontologies which allow resource sharing and are able to be used across multiple domains should in principle be the solution to this problem [20].



## Chapter 3

# Decision Tree Supervised Learning Algorithm

### 3.1 Introduction

This chapter focuses on the machine learning algorithm for supervised learning that is implemented in this work. Section 3.2 introduces the decision tree, its features and parameters.

### 3.2 Decision Tree Learning

Decision tree learning is a common method used in data mining. It is one of the most widely used supervised learning methods whose goal is learning the relations among features of data from a set of labelled or pre-classified examples. In the first step, a set of attributes of related data is chosen. Then, for each pre-classified sample a record of values corresponding to the selected features is created. The set of all such samples which is used to learn the relations is known as the training set. This means that the

training set is the input to a learning system and the resulting output is a classifier. This generated classifier is then used for classifying new or unseen samples of data [21]. Figures 3.1 and 3.2 show these two phases.

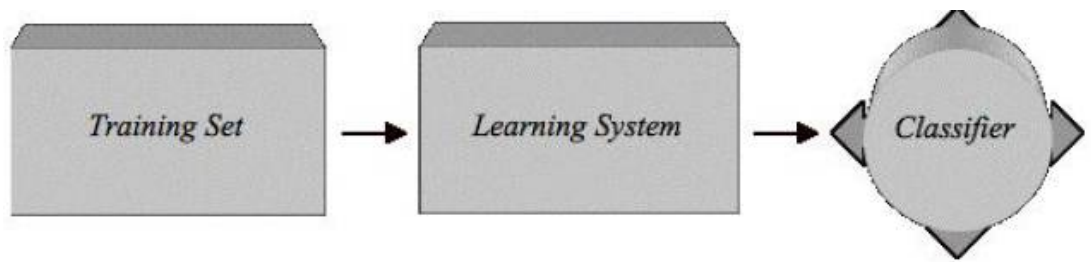


Figure 3.1: Creating a Classifier from a Set of Examples

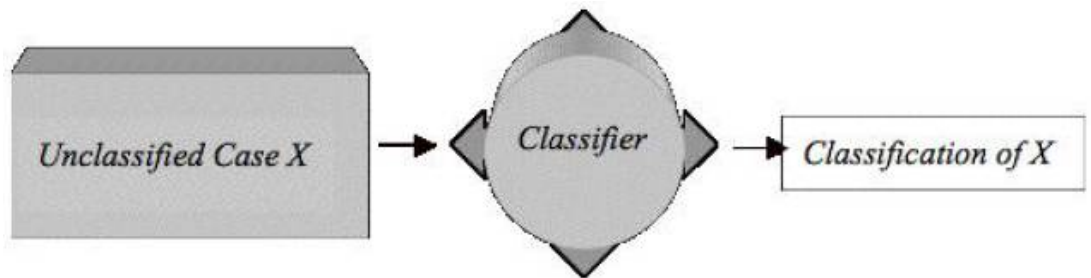


Figure 3.2: Classifying a New Case

A decision tree as used in a supervised learning system contains a hierarchical data structure based on a divide-and-conquer strategy [22]. By using decision trees the learner builds a tree structure from the training data set. This hierarchy is easy to understand and can be converted to a set of simple rules [5]. The internal nodes of this tree are tests and correspond to one of the input variables; for each of the possible values of interior nodes there are edges to its children. Its leaf nodes are categories of patterns representing values of the target variable. The output is a class number which is assigned to the input pattern by grouping the pattern down through the

tests in the tree [23]. In brief, a decision tree is a classifier in the form of a tree structure in which each branch node shows a choice among a number of alternatives and each leaf node represents a decision [24].

Decision trees are useful as classifiers in large data sets [22]. They are also commonly used in decision analysis to identify a strategy leading to the goal with the highest probability. On the other hand, NLP is fundamentally a classification paradigm at least with respect to the disambiguation and the segmentation (determining the correct boundary of a segment from a set of possible boundaries) tasks of linguistic problems [23]. The use of decision trees is therefore a candidate method for analysing some linguistic problems, especially for the syntax analysis phase of language learning.

The most important issue in learning decision trees is selecting an attribute at each node for testing. The level of certainty of a particular prediction which can be measured as a number from 1 (completely uncertain) to 0 (completely certain) has a direct effect on this selection. This parameter is affected by the information gain, the piece of information acquired. It is clear that the concept of information gain is useful for making machine learning predictions [22].

Entropy (a term used in information theory) is a probability-based measure of uncertainty. It can also be understood as a measure of the randomness of a variable. For example, the entropy of an event with the probability of 50/50 is 1. If the probability is 25/75 then the entropy is little lower [24]. In artificial intelligent applications doing stochastic modelling such as pattern recognition, medical diagnostics and semantic analysis of words, this metric is widely used [22].

The entropy concept is relevant for the learning of decision trees since these characterizing a sequence of decisions and minimum entropy is desirable in each step. The following formula is used for calculating entropy:

$$Entropy(X) = - \sum_{x \in \mathcal{X}} p(x) \log_2 p(x). \quad (3.1)$$

where  $p$  is the probability of the event  $X$



The minus sign is used to create a positive value for entropy. The base 2 in the logarithm function is used since we consider the information in the form of bits. Simply, this determines the number of bits that are necessary to show a piece of information. Now it is clear why we are looking for minimum entropy values. With a minimum entropy value we can find the attribute that best reduces the amount of further information we need to classify our data [25].

### 3.2.1 ID3, Iterative Dichotomiser 3

The most popular algorithm for generating decision trees is ID3 (Iterative Dichotomiser 3). This algorithm was invented by Ross Quinlan and is a precursor of the C4.5 algorithm. It takes all unused attributes and calculates their entropies. After this step, it chooses the attribute with the minimum entropy value. Finally it makes node which contains the selected attribute [22]. This algorithm tends to come up with the "next best" attribute in the data set to use as a node in the tree. Therefore by the logic supporting the entropy formula, ID3 is able to find the best attribute to classify records in the data set [25].

The ID3 algorithm is based on notions of information theory. Measures in information theory refer to the amount of information we gain from different parts of data by considering the cost of time. In each context, different messages have different information content measured. With decision trees, we define the measurement as the information on a given split in the tree. If some attribute is better at determining an answer, it gives more information. In this way, decision trees are made based on the best information given by the attributes. The theory of the ID3 algorithm says that the best attributes will make the best trees.

In this thesis we used ID3 as a learning algorithm for decision trees. In our implementation the decision tree is constructed using the ID3 algorithm that splits the data by the feature with the maximum information gain recursively for each branch.

This algorithm is described by the flow chart in figure 3.3.

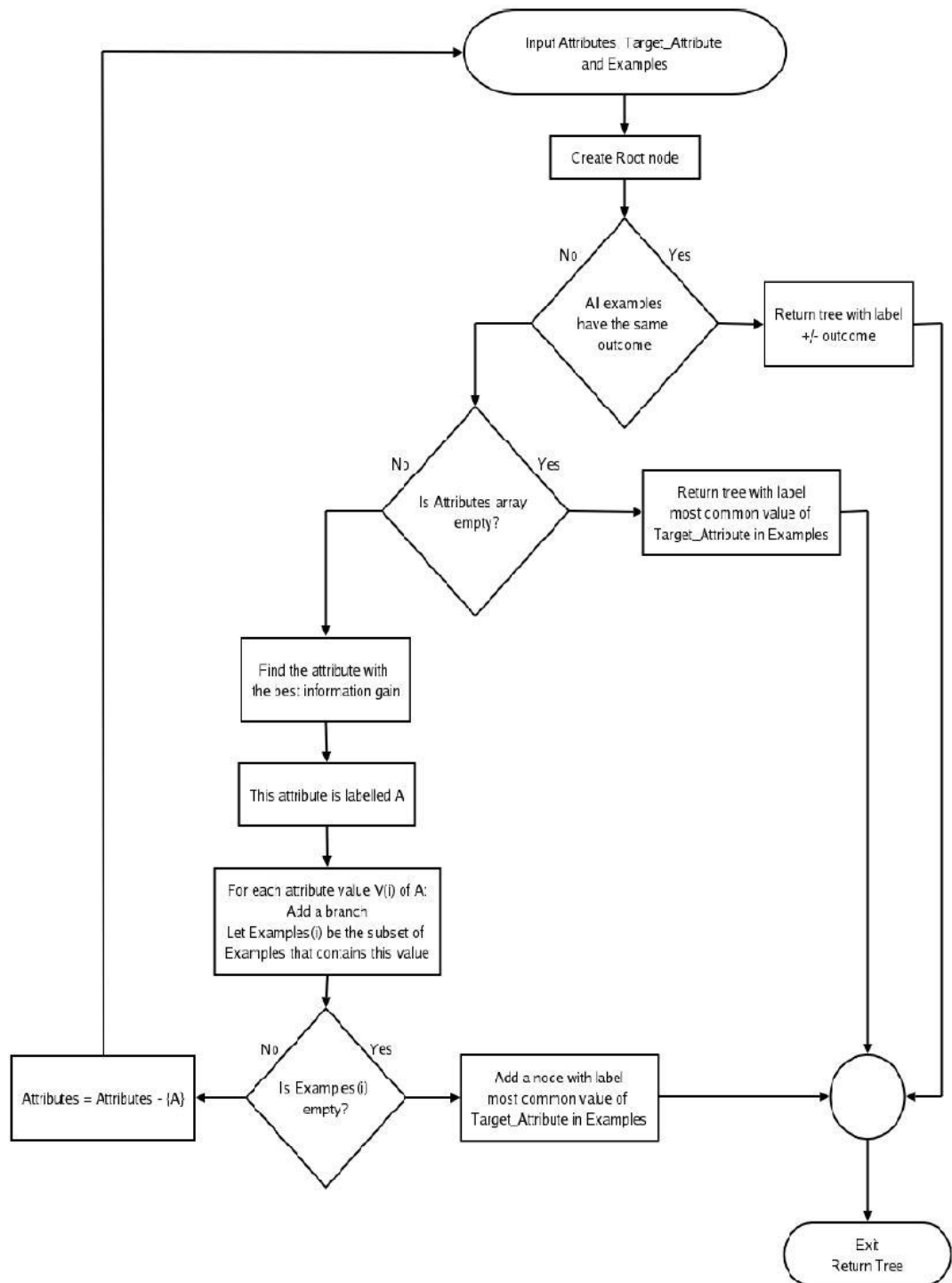


Figure 3.3: ID3 flow chart

## Chapter 4

# ADIOS

# Unsupervised Learning Algorithm

### 4.1 Introduction

As we mentioned in previous chapters, statistical methods of machine learning are divided into two main categories, supervised and unsupervised ones. Most current language learning processes are supervised and algorithms work on bracketed data set in the training phase. However, gathering such a data set is not always easy or it may even be impossible for some problem statements of language acquisition. Because of such lack of data sets we are encouraged to focus on unsupervised approaches. On the other hand, language acquisition has been one of the main research issues in cognitive science and most proposed computational models are inferior in scale and accuracy compared to those methods of computational linguistics that are based on machine learning techniques. Among these, unsupervised language learning and especially grammar induction represent steps towards the cognitive basis of human language acquisition.

This chapter concentrates on unsupervised language learning. As the first step it introduces grammar induction in section 4.2. The ADIOS algorithm will be discussed in section 4.3 as the selected unsupervised method that we used in this work. In the next part, section 4.4, we examine the problem statement of this thesis according to ADIOS as the empirical work on unsupervised language learning.

## 4.2 Grammar Induction

Grammatical induction, also known as grammatical inference or syntactic pattern recognition, refers to the process in machine learning and linguistics related to the learning of a grammar from a set of observations of actual sentences. In general this set of observations can be grammatically correct and, if available, incorrect sentences. The result of the grammar inference process is usually a grammar that we can use to describe the structure of acceptable sentences. Also, we will be able to build correct novel sentences belonging to the acceptable sentences list of the language [26]. A learner in language grammar induction uses a corpus generated by a grammar  $G_0$  to build a new grammar  $G$  that is intended to approximate  $G_0$  in some sense [27].

If we consider the process of acquiring language of human children, the whole data in this process are usually positive examples (correct sentences). To have a learning approach which is more similar to the cognitive human language acquisition process, we just use positive samples of data in our data set and assume that all sentences entered to the system are correct. The rate of learning by such samples of data is higher than by negative ones [28].

### 4.2.1 Methods of Grammar Induction

Until now many different algorithms for grammar induction have been released. The following are some of them [28]:

***Koza's genetic programming method*** is an evolutionary method that

builds a tree for grammatical structure. This tree contains terminal nodes that are lexical items from raw text corpora. This approach is more useful in bioinformatics applications in comparison with applications in language processing area.

***ABL or Alignment-Based Learning*** is another grammar induction method containing two phases: alignment learning and selection learning. In the first phase, the algorithm tries to find constituents in corpora while in the second phase it selects the best generated constituent that results from the learning phase.

***EMILE algorithm*** is based on a distributional model and consists of the two phases of clustering and rule induction. It is based on the idea that expressions occurring in the same context can cluster together and are substitutable in similar contexts. After finding classes of expressions that cluster together, then in the induction rule phase we can induce rules from their contexts.

In the next section we introduce the unsupervised grammar induction algorithm, ADIOS, which we used in our implementation.

### 4.3 The ADIOS Algorithm

Automatic Distillation of Structure or ADIOS is an algorithm for analysing contexts such as texts and producing meaningful information about the structures which build those contexts. For example, in processing languages in the form of texts, ADIOS can accept the text as its input and infer the grammatical rules that are manifested as patterns in the text. Indeed, with a language processing view, this method infers the underlying rules of grammar autonomously without previous information about an input text. These retrieved rules are useful for generation of grammatically and conceptually true sentences [26].

With a different perspective, this algorithm can be called a probabilistic model, as it looks for redundancy which makes itself known as a pattern in a given corpus. Being intrinsically suitable for any set of sentential data with a finite lexicon is one of the main reasons of choosing this algorithm.

In this work, we have a very limited number of tokens. Until now ADIOS has been applied to contexts which are generated from artificial grammar. It is also applicable in other similar research such as learning language of genes in biology, or language of notes in music. Actually, according to the theory in distributional structures cited by Harris in his book [29], there are two common features in all the above applications and also in our work. It means that we know that ADIOS is a suitable algorithm for problems that have the following attributes in their data [29]:

- In these contexts the different parts of a language occur in certain positions which are relative to other elements.
- The limitation in distribution of classes can not be discarded for some other purposes like semantic needs.

#### 4.3.1 The ADIOS Algorithm Structure

As we introduced above, the ADIOS algorithm is a statistical-structural algorithm. The input of the algorithm is usually a set of sentences called a corpus. This suggests that the first phase of work should be loading a corpus of data into a data structure [26]. This data structure is a directed pseudo graph. We call it a directed pseudo graph as it contains loops or edges with the same vertices. Each node of this graph is mapped with a unique word in the corpus. Depending on the domain of the problem, these nodes may represent e.g. part-of-speech, or proteins.

There are two special nodes in this data structure that all paths in the graph contains, namely, the BEGIN and END nodes. Paths in the graph which start by the BEGIN node and end with the END node correspond to sentences in the entered corpus [28]. In the first phase, each sentence is mapped to a path in the graph. As the algorithm progresses, nodes in the paths of the graph are replaced by patterns and equivalence classes and the paths in the graph becoming compressed. This process is repeated until a specific criteria is reached [26].

The meaning of the term "pattern" we used above, is a sequence of nodes

---

**Algorithm 4.3.1** The Initialization of ADIOS algorithm

---

INITIALIZATION()

```

1  /* N is the number of sentences in the corpus
   */
2  for  $m \leftarrow 1$  to  $N$ 
3  do LOADSENTENCE( $m$ )
4  /* It loads the sentence  $m$  as a path onto a
   pseudo graph whose vertices are the unique
   words in the corpus */
5
```

---

which can be terminal nodes, equivalence classes or other patterns. The redundancy in paths is reduced by replacing terminal nodes with non-terminal patterns. Also "Equivalence Classes" are considered as sets of terminal and non-terminal nodes occurring in the same contexts and are distilled by the grammar [28].

### 4.3.2 The ADIOS Algorithm Phases

There are three different phases for the algorithm: Initialization, Pattern Distillation, and Generalization. All these phases are generally represented in algorithm 4.3.2. The pseudo code in algorithm 4.3.1 shows the first phase. In the *initialization* step all sentences which will be translated to paths are loaded into the graph. The words mapped to vertices are labelled and the edges linking them together are created and represent sentences in the corpus. As we can see in the pseudo code in algorithm 4.3.2 the two next phases are called in a loop until the algorithm reaches a significant pattern [26].

In the *distillation phase* the algorithm looks for sub-paths which are shared by some other partially aligned paths. In this phase we do a segmentation on input sentences. This segmentation is performed using a statistical segmentation criterion, MEX (Motif EXtraction), which scans the graph for patterns [30]. As we represent in figure 4.1 these common sub-paths in the



---

**Algorithm 4.3.2** The ADIOS algorithm

---

```

ADIOS()
1  INITIALIZATION()
2  repeat
3      /* N is the number of paths in the graph */
4      for  $m \leftarrow 1$  to  $N$ 
5          do PATTERN-DISTILLATION( $m$ )
6              /* Identifies new significant patterns in search
               path m using MEX criterion */
7              GENERALIZATION( $m$ )
8              /* Generate new pattern candidates for search
               path m */
9
10 until /* until no further significant patterns are
        found */

```

---

graph help us extract the patterns. We may encounter many sub-paths common to different groups of paths. A probability factor based on the number of edges in a sub-path is suitable for identifying candidate patterns [31]. For each extracted pattern a new node  $P$  is constructed and all nodes in the pattern are replaced with  $P$ . This segmentation process causes the graph to be converted to a hierarchical structure without any change to the size of the language. The stop criteria of this phase of the algorithm is reaching to the state where no new patterns can be found [32].

In the *generalization phase* whose pseudo code is in algorithm 4.3.3, the algorithm finds the most significant pattern, then searches for elements that are in the complementary distribution, in the context of this pattern. It means that the algorithm generalizes the pattern over the graph by creating equivalent classes from all of the variable node elements in the pattern. This generalization is simply performed by a segment along a candidate path, and then looking for other paths coinciding with this segment in all places of nodes but one [31]. For example, if we have a pattern such as "the x dog" as the selected significant one, then all nodes that can fill the x

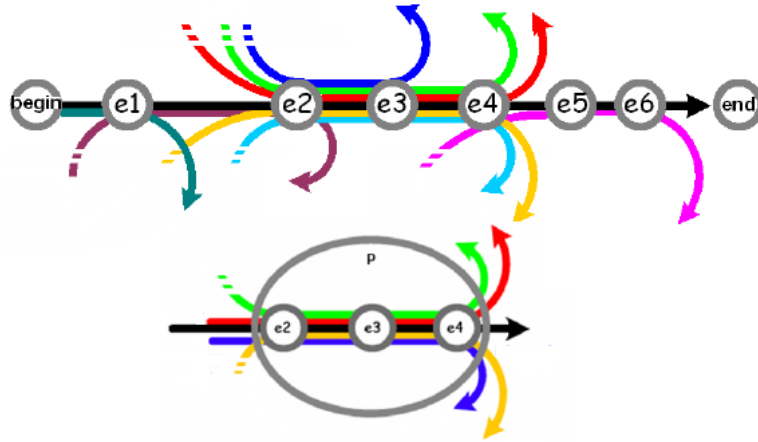


Figure 4.1: A set of nodes is identified as a pattern when a bundle of paths traverses it

---

**Algorithm 4.3.3** The Pattern Distillation of ADIOS algorithm

---

PATTERNDISTILLATION()

```

1  /* N is the number of paths in the graph */
2  for m ← 1 to N
3  do for i ← 1 to LENGTH(m)
4      do for j ← i + 1 to LENGTH(m)
5          do s ← GETSEGMENT(i, j)
6              p = MEX(s)
7              /* Find the leading significant pattern (p) by
                  performing MEX on the search segments(i,j)
                  */
8              REWRITEGRAPH(p)
9              /* Rewire the graph, creating a new node for
                  P, replace the string of nodes comprising P
                  with the new node P */

```

10

---

slot are assigned to an equivalence class, and then sequences of nodes are replaced with their equivalent patterns. This yields the graph rewired [30].

---

**Algorithm 4.3.4** The Generalization of ADIOS algorithm

---

GENERALIZATION( $m$ )

```

1  /* m is a search path                                     */
2  for  $i \leftarrow 1$  to  $LENGTH(m) - L - 1$ 
3  do  $s \leftarrow SLIDECONTEXT$ 
4      for  $j \leftarrow i + 1$  to  $i + LENGTH(s) - 2$ 
5      do /* Here we define a slot at j define the gener-
           alized path consisting of all paths with same
           prefix and same suffix and perform MEX on
           the generalized path                                */
```

---

### 4.3.3 The ADIOS Algorithm Parameters

The entire process of the ADIOS algorithm containing the repeating search for patterns and equivalence classes is governed by three parameters:

The  $\eta$  and  $\alpha$ , which control the definition of pattern significance, and  $L$ , which sets the width of the context window where equivalence classes are sought.

As we mentioned in the pattern distillation phase, the main task of the Motif EXtraction (MEX) algorithm is finding patterns that different paths converge onto at the beginning of the pattern and diverge from at the end of the pattern (Figure 4.1) [32]. The MEX algorithm searches for a decrease ratio of sub-sequences of a sub-path as a candidate pattern. The nodes outside a candidate pattern have certain probabilities which are associated with incoming and outgoing edges. An increase in the number of edges going into or out of the given node causes a decrease in the ratio of these probabilities. This trend indicates the boundary of a significant pattern. The ADIOS algorithm works with the  $\eta$  criteria as its "cutoff parameter"

for the decrease ratio. A sub-path is considered as a candidate pattern if its decrease ratio is less than the predefined value [30].

There is another parameter measuring the significant level for the decrease ratio. It is the  $\alpha$  value by which we can test the significance. If the significance is not less than  $\alpha$  then the pattern is rejected and is not suitable to be a candidate [32].



# Chapter 5

## Methodology

### 5.1 Introduction

In this chapter we focus on the implementation phase of the work. Section 5.2 is dedicated to the decision tree induction using the supervised learning method introduced in chapter 3. The details of the supervised algorithm is discussed in subsections of this part. Section 5.2.1 describes the input and output file structures. The discussion will be continued by presenting the algorithm and the way we use it for learning the syntactic class of the language. Modules of the program and components of the application will be introduced in section 5.2.2. Section 5.2.3 concentrates on evaluating and measuring the performance of the learner.

Section 5.3 starts the discussion on ADIOS. The details of the work, the challenges we encountered in parameter settings of the algorithm and issues we addressed to improve the results are discussed in part 5.3.1.

### 5.2 Decision Tree Induction

As we mentioned in chapter 3, we use the ID3 algorithm which is widely used for statistical classification and which chooses the feature with the

highest normalized information gain to split the training data. The algorithm then recurses on each training data subset until no more features can be used. We use ID3 to build a decision tree for our data set and by interpreting the result, we are able to induce the grammar of the artificial language. Because of its simplicity, this algorithm is suitable for an implementation in which the main focus is on detecting ambiguity and handling the missing data.

The results obtained in this work are very precise and the created decision tree allows us to illustrate relations among the attributes (predictors) used for classification and determine which of these variables are more useful in the classification process.

### 5.2.1 Data Input/Output Structure

A decision tree as a supervised learning method needs the training data which are imported into the working environment for manipulation. In our case, we first gathered all universities names from formal academic sites such as *webometrics* (<http://www.webometrics.info/>) that contain university names according to their world ranking. The total number of names in the training data set used to initially verify the decision tree algorithm reached 1500 names selected from universities in different parts of the world. We chose names from various parts to make the learner be familiar with different structures in name phrases which are dependent language grammars.

This trivial data set has 11 attributes listed in table 5.1. Each of them

---

Country	City	Province	Region
Person Designation	Geo Designation	Discipline	Abbreviation
Affiliation	Organized-as	Sub type	

---

Table 5.1: List of main attributes

contains its corresponding values that we will use to build the semi ontology structure for the learner.

For some pre processing purposes, we made an excel file to have a visual evaluation of data. These spreadsheets also helped us to get some statistical information such as detecting null values for some predictors in some records. In table 5.2 we show a small capture of data in the excel file.

As we mentioned in chapter 3, separating data into training and test

Country	City	Province	Region	Person Designation	Geo Designation	Discipline	Abbreviation	Affiliation	organized-as	Sub type
Germany	Berlin			Humboldt						University
Germany	Mainz			JohannesGutenberg						University
Hungary	Budapest			Semmelweis						University
Romania	Bucharest					EconomicStudies				Academy
Ukraine	Dnipropetrovsk					Metallurgical		National		Academy
Turkey	Trabzon				Karadeniz	Technical				University
Azerbaijan	Baku					Economic		State		University
Bulgaria	Sofia			St.KlimentŲhridski						University
Russia	Moscow					Social		State	Russian	University
Ukraine	Ternopil			IvanPul'uj		Technical		State		University
Ukraine	Kiev				Kiev	TechnologiesAndDesign		National		University
UnitedKingdom	Cambridge									University
Russia	Voronezh							State		University
Ukraine	Odesa					FoodTechnologies		National		Academy
Russia	Moscow					InternationalRelations		State		Institute
Bulgaria	Bourgas							Free		University
Belarus	Minsk					Technical		National		University
Russia	Moscow					Social		State	Russian	University
Belarus	Minsk					Technical		National	Belarussian	University
Armenia	Yerevan							State		University
Armenia	Yerevan					Engineering		State		University
Austria	Graz					Medical				University
Austria	Graz									University
Austria	Graz									University
Austria	Graz									University
Austria	Innsbruck									University
University										
Sweden	Jonkoping									University
Moldova	Chisinau					EconomicStudies				Academy
Sweden	Linkoping									University

Table 5.2: Samples of records in data set

sets is an important part of evaluating data mining models in every machine learning task. Data used to discover a predictive relationship among data are called the training data set. A set which is independent of the training data, but with same probability distribution is called a test set. In particular, for supervised learning methods, our goal is to find the model having the best performance on new data. If we want to compare different models according to their performances, the simplest approach is evaluating their error function using data which is independent of that used for training. It means that the performance of a model is evaluated by an error function using an independent validation set, and the model having the smallest error with respect to the validation set is selected.



The performance of the selected model should be confirmed by measuring its performance on the test set. After the model has been processed by the training set, it is time to test the model by making predictions against the test set. We have to make sure that data are similar in the test and training sets. The major part of the data is assigned to training, and a smaller portion is used for the testing set. Choosing data randomly guarantees that the types of data in both of sets are similar. This similarity helps the learner to have a better modeling and to minimize the future misclassifications.

Our data set is available in the form of a text file readable for our C# application. This text file is extracted from the mentioned excel file. The data set was split into training, validation and test groups. 50% of the original data (about 750 records of names) is used for training, 25% for validation and 25% for testing. We created a new target value called *classNO* indicating the class to which the record belongs. We found about 57 different groups of structures underlying the training set and for each record of data we set the value of *classNO* attribute by a suitable number. The list of these structures is represented in table 5.3. Also, the primary data set and its finalized version are accessible via:

<http://www.ida.liu.se/ext/reseda/y2011/n001/page.html>

### 5.2.2 Program Modules and Components of the Application

According to the phases of decision tree learning, the algorithm starts to build a decision tree by calling the *createDecisionTree* method. Before calling this method, the application starts to create some data structures we need during the work. As the created tree is built and processed according to some criteria being considered in structures working as the ontology of the work, these structures should be built before creating any tree. The *readData* method is the procedure doing this. It processes the data set, line by line, and creates a domain for each attribute in the system. It also checks the dictation of values; if they have same dictation then it creates

Structure	Description	classNO
X	X = Abbreviation	17
X Y Z	X = Abbreviation, Y = City, Z = subtype	37
X-Y Z of P	X = Abbreviation, Y = Discipline, Z = subtype, P = City	54
X Y of Z	X = Abbreviation, Y = subtype, Z = Discipline	4
X Y Z of P	X = Affiliation, Y = Discipline, Z = subtype, P = City	5
X Y Z P	X = Affiliation, Y = Discipline, Z = subtype, P = City	32
X Y Z of P	X = Affiliation, Y = Discipline, Z = subtype, P = Country	33
X and Y Z of P	X = Affiliation, Y = Person Designation, Z = subtype, P = City	31
X Y of Z - P	X = Affiliation, Y = subtype, Z = Country, P = City	34
X Y Z P K	X = City, Y = affiliation, Z = Person Designation, P = Discipline, K = subtype	22
X Y Z of P	X = City, Y = affiliation, Z = subtype, P = Discipline	13
X Y Z	X = City, Y = geo Designation, Z = subtype	23
X Y Z	X = City, Y = OrganizedAs, Z = subtype	26
X Y Z P K	X = City, Y = Person Designation, Z = Affiliation, P = Discipline, K = subtype	29
X Y Z	X = City, Y = subtype, Z = Person Designation	12
X Y Z P	X = Country, Y = affiliation, Z = Discipline, P = subtype	10
X Y of Z	X = Country, Y = subtype, Z = Discipline	11
X Y in Z	X = Discipline, Y = subtype, Z = City	39
X Y of Z	X = Discipline, Y = subtype, Z = Province	40
X Y Z of P	X = geo Designation, Y = Affiliation, Z = subtype, P = Discipline	30
X Y Z	X = geo Designation, Y = Discipline, Z = subtype	6
X Y Z	X = OrganizedAs, Y = Discipline, Z = subtype	25
X Y Z in P	X = OrganizedAs, Y = Discipline, Z = subtype, P = City	20
The X Y	X = OrganizedAs, Y = subtype	43
X Y at Z	X = OrganizedAs, Y = subtype, Z = City	16
X Y of Z	X = OrganizedAs, Y = subtype, Z = Discipline	21
The X Y of Z	X = OrganizedAs, Y = subtype, Z = Discipline	41
X Y of Z at P	X = OrganizedAs, Y = subtype, Z = Discipline, P = City	7
X Y of Z	X = OrganizedAs, Y = subtype, Z = Province	35
X Y Z P	X = OrganizedAs, Y = affiliation, Z = Discipline, P = subtype	14
The X Y	X = Person Designation Y = subtype	42
X Y Z	X = Person Designation, Y = Affiliation, Z = subtype	38
X Y Z of P	X = Person Designation, Y = Affiliation, Z = subtype, P = City	28
X Y Z P	X = Person Designation, Y = Affiliation, Z = subtype, P = City	55
X Y Z of P	X = Person Designation, Y = Affiliation, Z = subtype, P = Country	51
X Y Z P	X = Person Designation, Y = City, Z = Affiliation, P = subtype	27
X Y Z P of K	X = Person Designation, Y = OrganizedAs, Z = Affiliation, P = subtype, K = Discipline	36
X Y of Z P	X = Person Designation, Y = subtype, Z = Discipline, P = City	49
X Y Z	X = Person Designation, Y = Discipline, Z = subtype	24
X Y Z of P	X = Person Designation, Y = City, Z = subtype, P = Discipline	9
X Y of Z	X = Person Designation, Y = subtype, Z = City	8
X Y in Z	X = Person Designation, Y = subtype, Z = City	19
X Y of Z	X = Person Designation, Y = subtype, Z = Discipline	18
X Y of Z	X = Person Designation, Y = subtype, Z = Province	50
X Y Z	X = Province, Y = Discipline, Z = subtype	15
X Y Z	X = Province, Y = Person Designation, Z = subtype	1
The X Y of Z	X = Province, Y = subtype, Z = City	44
X Y of Z	X = Province, Y = subtype, Z = Discipline	48
X Y Z	X = subtype, Y = Person Designation, Z = City	45
X Y of Z	X = subtype, Y = Person Designation, Z = City	46
X of Y of Z	X = subtype, Y = Discipline, Z = City	2
X of Y of Z	X = subtype, Y = Discipline, Z = Country	3
X of Y in Z	X = subtype, Y = Discipline, Z = City	52
X of Y Z	X = subtype, Y = Province, Z = City	47
X of Y in Z	X = subtype, Y = Province, Z = City	53
X Y	X = Region, Y = subtype	56
X of Y	X = subtype, Y = Province	57

Table 5.3: List of available structures in data set

some relations between their domains. This domain structure is handled by the *DataPoint* class containing some sequential structures for values. For each record of data, this object keeps a sequence of index values belonging to the series of predictors.

*TreeNode* is an entity with its specific attributes which are *entropy*, a sequence of *data*, *decompositionAttribute*, *decompositionValue*, *children* and *parent*. Each *TreeNode* object holds its own values for its attributes. In this phase, we consider a *TreeNode* object called *root* and add every created datapoint object as a new data of the root object.

The *decomposeNode* method is the main part of the algorithm. It initializes all attributes of the root node with suitable values by calling the *calculateEntropy* method. These two methods are frequently called to make the final decision tree according to the ID3 algorithm.

Unlike most decision tree implementations which produce a hierarchical tree structure as their output, here, we build the output as a separate program module. As this module makes an *if-else* structure for the classification, it is able to be added to any C# source code. During this part of the work, where the restrictions of the algorithm are considered, we have also added some new constraints related to the multi meaning words issues. It means that we changed the ID3 and added new condition checking code to it.

The first phase of entropy calculation shows the *City* as the first partitioning attribute in the decision tree. Table 5.4 lists the first level of Entropy and Gain Information values for all attributes. As we can see, the maximum value belongs to the *City* attribute. To repeat the partitioning of data set, we have to do the same calculation but excluding the chosen attributes (*City*) from the list. For some values of the *City* attribute the tree can directly lead us to the related *classNo*, but for most other values it needs to find a second informative attribute to partition the data set. The *Person Designation* is the second one giving the best information by traversing the tree. The conditional structure which is the result of the recursion algorithm orders the attributes according to the information they

provide. The order of them are as follows:

*City, Person Designation, Country, Province, Region , Geo Designation, Discipline, Abbreviation, Affiliation, organized-as, Sub type*

---

Entropy(S)	4.2922
Gain(S, Country)	1.5919
Gain(S, City)	3.5879
Gain(S, Province)	0.6160
Gain(S, Region)	0.0333
Gain(S, Person)	0.4647
Gain(S, GEO)	0.1053
Gain(S, Discipline)	0.9970
Gain(S, Abbreviation)	0.11232
Gain(S, Affiliation)	0.4538
Gain(S, OrganizedAs)	0.4423
Gain(S, Subtype)	0.3381

---

Table 5.4: The first level of Entropy Calculation

By creating an ontology structure in parallel with adding values of predictors in the public *DataPoint* object, we can specify the type of values by applying an *IS-A* label. In this way we are able to gather the grammatical information of a word which has different roles in the domain. For example, at the time of adding a new value to the *DataPoint* object we do some processing and search the whole structure to find a word which is similar in dictation with the new one. If the search produces a result then we make a relation between these two similar values. It virtually makes an ontology structure which has the value as its root and has *IS-A* edges from this root to the attribute names.

### 5.2.3 Evaluating the Usefulness of the Learner

After designing the hierarchical structure, there is a need to present a suitable measurement for the learner. The evaluation process requires to test the classifier on a test set that is independent of the training set used to generate the classifier. For this purpose of empirical evaluation, a confusion matrix which is a visualization tool used in supervised learning is applied. The dimension of this matrix is  $n$  by  $n$ , where  $n$  is the number of classes (57 classes for our case). Each row of the matrix represents the instances in a predicted class, and each column represents instances in an actual class. Every confusion matrix shrinks to a special one depicted in the table 5.5 which is used in predictive analysis in machine learning tasks. This matrix contains two rows and two columns representing the number of true positives, false positives, true negatives and false negatives.

	Predicted Negative	Predicted Positive
Negative Cases	True Negative(TN)	False Positive(FP)
Positive Cases	False Negative(FN)	True Positive(TP)

Table 5.5: Matrix for evaluating the Learner

According to the above matrix, we define the following parameteres of evaluation:

- accuracy: is the proportion of correctly classified samples among all classified samples.

$$accuracy = (TN + TP) / (TN + FP + FN + TP) \quad (5.1)$$

- precision: is the proportion of returned results that are in fact valid (assigned the correct class).

$$precision = (TP) / (TP + FP) \quad (5.2)$$

- recall: is a complementary metric representing what proportion of the cases really belonging to a class were identified as such by the classifier.

$$recall = (TP)/(TP + FN) \quad (5.3)$$

- f-measure or balanced F-score: is the harmonic mean of precision and recall and can assess the commitment between both criteria.

$$fmeasure = 2/((1/Precision) + (1/Recall)) \quad (5.4)$$

After converting the 57 by 57 confusion matrix into a binary table, we got the following depicted results for above measuring parameters.

- Number of True Positives(TP): 806
- Number of False Positives(FP): 84
- Number of True Negative(TN): 541
- Number of False Negative(FN): 69
- The total number of cases analysed(TN + TP + FN + FP): 1500
- The total number of correct predictions(TN + TP): 1347
- The total number of incorrect predictions(FN + FP): 153

As we can see in Table 5.6, both precision and recall are high (the F-measure of 0.913 is as the support of this fact) and it results in a balance in the prediction and discrimination relationship.

Precision	Recall	F-measure	error rate	accuracy rate
0.905	0.921	0.913	153/1500 = 0.102	1347/1500 = 0.898

Table 5.6: Evaluatinn results of the tree

## 5.3 ADIOS Algorithm

This part describes implemented experiments and their interpretations which are based on ADIOS algorithms. The data set is the same as in the previous study. The subsections discuss the experimental methods followed by a discussion on results.

### 5.3.1 Setup

This application has also been implemented in C#. We applied the ADIOS algorithm with different values we set for  $\alpha$  and  $\eta$  parameters on our available data set. After each change we made in parameters values, we ran the algorithm again and saved the resulted grammars as sets of patterns and equivalence classes. The range of values we used for the  $\alpha$  and  $\eta$  parameters is 0.03, 0.33, 0.63 and 0.93. By applying these values, sentences which are reduced to their possible minimum paths rebuild the data set.

To describe more, we have to recall the details of the ADIOS algorithm mentioned in chapter 4. For each university name that we have in the original data set, a set of terminal nodes exists. These nodes are objects replaced with their corresponding patterns and equivalence classes and are finally saved into a new file when it has been reduced to its minimal path.

The different compounds we made by values of  $\alpha$  and  $\eta$ , yielded various outputs. One way which lets us measure the performance of the algorithm is evaluating the compression rate of terminal nodes representing a sentence. According to definitions, the compression score is calculated by dividing the number of nodes in the ADIOS paths by the number of nodes in the original data set.

$$\text{Compression-score} = \text{ADIOSNumberofNodes} / \text{OriginalNumberofNodes} \quad (5.5)$$

The compression is usually reported as a percentage. As the above formula shows, the greater compression is achieved when the above ratio gives a smaller value. The number of nodes is obviously less than the original number of nodes if some of the original terminal nodes are replaced by patterns.

According to the definition we offered for the parameters of the ADIOS algorithm, we can say that if ADIOS can distill a sentence, then there must have been patterns that have met the criteria of the  $\alpha$  and  $\eta$  parameters.

Finally we focus on the effects of parameter values on compression, number of patterns and number of equivalence classes. In Table 5.7 the relation between the  $\alpha$  parameter, compression and the number of patterns and equivalence classes is shown. It is obvious that increasing the  $\alpha$  value causes all other measuring parameters to grow. Figure 5.1 also shows the trends. It should be mentioned that the downward trend of the compression score represents an increase for the compression rate. Similarly, we

$\alpha$	Compression	Equivalence Classes	Patterns
0.03	0.93	15	27
0.33	0.81	22	31
0.63	0.72	26	38
0.93	0.60	33	42

Table 5.7: Relation between the  $\alpha$  parameter and the measurements performed.

have the same relations but between the value of  $\eta$  value and the mentioned measuring parameters. Table 5.8 and Figure 5.2 show this.

$\eta$	Compression	Equivalence Classes	Patterns
0.03	0.97	9	14
0.33	0.71	23	29
0.63	0.63	33	35
0.93	0.55	42	47

Table 5.8: Relation between the  $\eta$  parameter and the measurements performed.



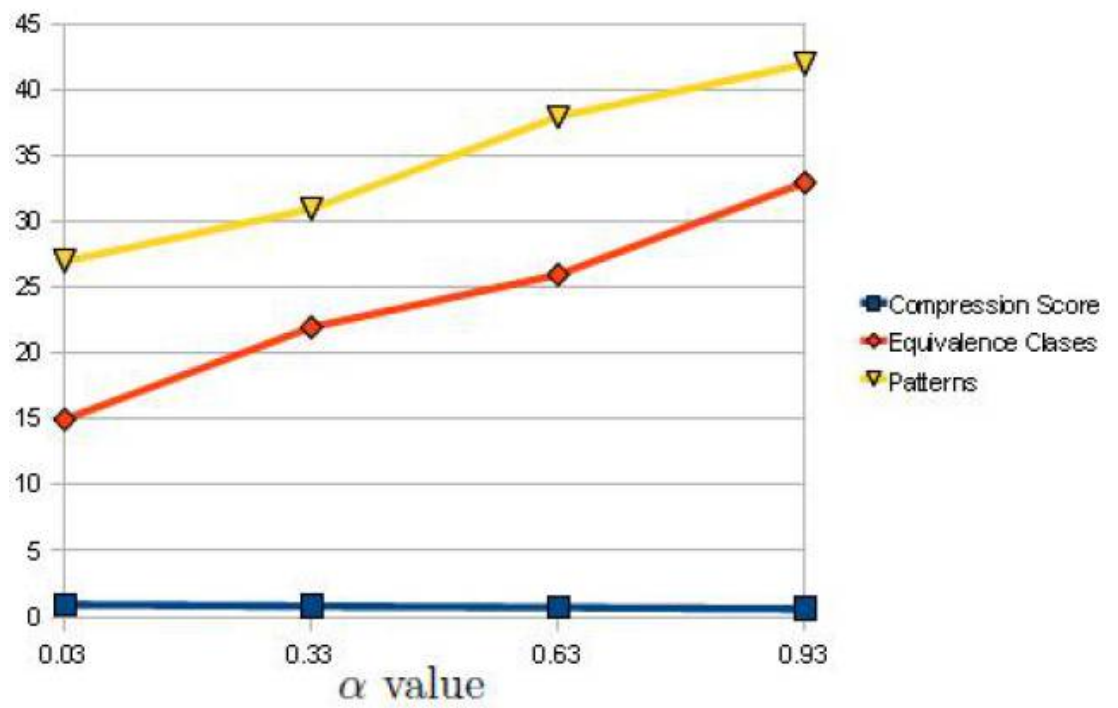


Figure 5.1: Relation between the  $\alpha$  parameter and the measurements performed

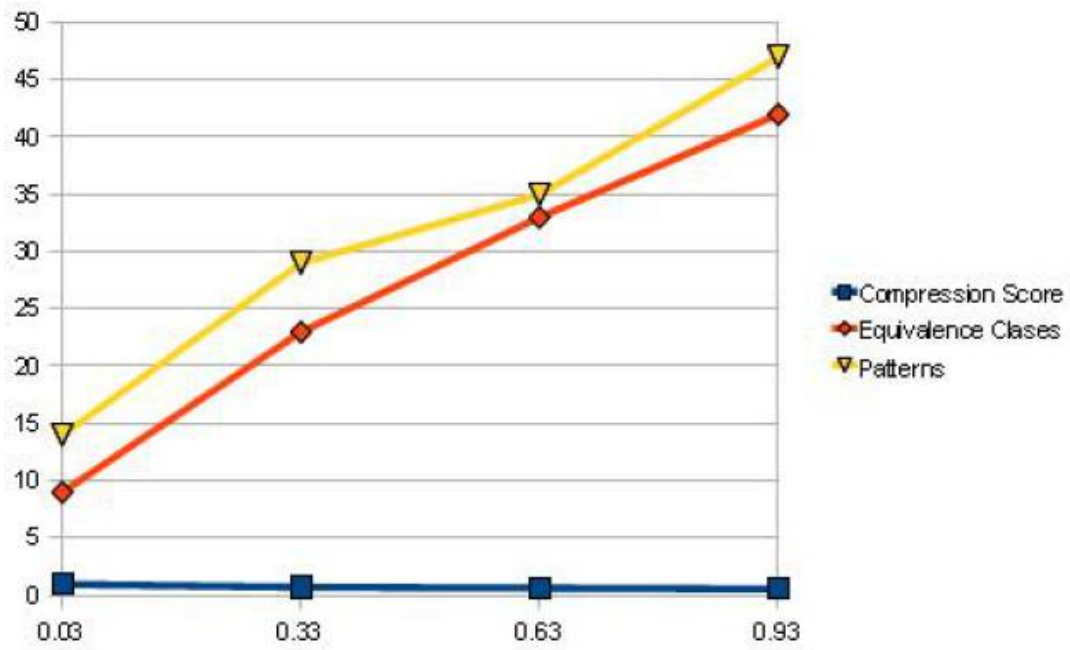


Figure 5.2: Relation between the  $\eta$  parameter and the measurements performed

The number of nodes in the initial graph of the ADIOS algorithm is equal to the total number of unique words in the university names or sentences of our data set. This number is about 1263 with the following details: If

---

Type of words	Number
Country	83
City	836
Province	67
Region	11
Person Designation	134
GEO Designation	46
Discipline	85
Abbreviation	35
Affiliation	19
OrganizedAs	29
Subtype	18

---

Table 5.9: Number of Nodes

we compare the above results with those we got in the supervised learning method we see an obvious difference among the number of patterns found in both methods(57 detected patterns versus 42 induced patterns). To reach a more acceptable result in ADIOS, it is useful to remember that ADIOS is a data driven method and increasing the number of sentences and loading them into the graph will improve its result sensibly.

To prove this fact, we added 300 other universities names into the original data set and reloaded the graph of ADIOS. At this time the number of universities reached 2800 and the total number of nodes in the graph reached 2339. In this new data set we could induced the true number of patterns by the settings in Table 5.10.

What we inferred from above outcomes is that ADIOS is not guaranteed to induce the complete grammar from the data. The dependency of the

---

$\alpha$	$\eta$	Compression	Equivalence Classes	Patterns
0.93	0.93	0.45	48	55

---

Table 5.10: ADIOS-final parameter settings

grammar on the number of words (tokens) we can use in the training set is one of the reasons for this weakness. It means that the algorithm is sensitive to the corpus size even if we set the optimum values for  $\alpha$  and  $\eta$  parameters to increase the compression, number of patterns and number of equivalence classes.



## Chapter 6

# Conclusion

### 6.1 Comparing two implemented learning methods

The task of this work was learning a new language with simple domains of words and grammar. We studied two models of learning. The findings of this study reveal that the decision tree method as the supervised learning model can accurately predict outcomes if an ideal data set is used for building the tree. With a suitable data set we can trust the list of grammars extracted. Among the features of data we chose in the data set, it is crucial to detect more important ones and also to indicate irrelevant attributes. The entropy measurement helps us find the order of attributes for the classification task. If we are able to detect unnecessary features and remove them, we can improve the result. The bottleneck in this respect was the existence of multi meaning words which got different roles in the records of data according to their meanings. The fact that records in our data set are marked by true grammatical classes benefitted the ontology structure which was produced during the building phase of the decision tree. We were aware of the classes of records and placed their words in separate parts of the ontology hierarchy, and traversed this structure to find the similar words having been seen previously. After finding a match word, we

related these two positions in the ontology structure. By manipulating the entropy calculation, this relation can obtain a more precise result when we encounter words with more than one meaning. Indeed, this ontology provides a classification for word senses. This classification is useful for making explicit aspects of these words meanings that are relevant for overcoming ambiguity.

However, as we mentioned in the introduction section, we intended to focus on the cognitive aspect of natural language learning. By this perspective, supervised learning solutions are not useful, as most of the time the labeled data sets are not available for learners. The essence of natural language learning leads us to unsupervised learning solutions. The ADIOS algorithm as one of these learning methods has less accuracy in some situations but it is more operational as it is not dependent on a labelled data set.

In comparing ADIOS with other language learning methods for inducing grammars, we can easily find a more cognitive perspective in this algorithm. It uses a graph which is similar to the structure created in the human brain during the learning in the first years of life. It induces the grammar from raw contexts by utilizing some adjustable criteria, and therefore that the accuracy of this statistical data-driven algorithm is in the top list in comparison with other natural language learning algorithms. In this thesis, the goal of the ADIOS algorithm implementation was creating an engine which learns a semi natural language grammar and that can evaluate a new sentence which was not in the training set. If we make this engine more abstract, it will be a candidate for use in other natural language processing applications such as machine translation, human-computer interaction applications, semantic web tasks and etc.

Nevertheless, a defect of ADIOS is its dependency on the size of the data set and the number of words and tokens in sentences. It means that the grammar induced by a short data set is untrustworthy for its generalization. As we observed in the experiments that were described in the previous chapter, to get the true number of classes of grammars we had to extend the data set almost twice. Therefore, to induce the authentic generalizable grammars we have to be assured about the size of the data set and the

diversity of records as well.

## 6.2 Future work

The dream behind this work is improving human-computer interaction using a linguistic perspective. This field of work is one of the most interesting ones in this decade and is mainly used in applications and utilities which are available in mobile phones, ATMs and many other devices providing interaction and communication between people and computers. The human is in one side of this issue and therefore we have to concentrate on more cognitive-based approaches to learning. Improving the interaction will require the use of natural language learning by machines so that they can learn to understand the concepts of speeches or texts they receive from humans. As all of us know, one of the fundamental sources of emotions is language and semantic analysis strongly helps to understand the relations between sentences and meaning of ambiguities in multi meaning words. Embedding such an automatic inducing grammar in interactive applications and devices opens a new gate to this world.

Other types of work in which the ADIOS algorithm can be applied are in text mining and machine translation as well as other applications that need to create rule based data sets or large knowledge bases from large contents. In other words, automatic grammar induction provides a wealth of applications for this unsupervised learning algorithm.




# Bibliography

- [1] *Introduction to Automata Theory, Languages, and Computation*. Addison Wesley, 2000.
- [2] Dominiek Sandra. The morphology of the mental lexicon. *Language and Cognitive Processes*, 1994.
- [3] *The Language Instinct*. HarperCollins - New York, 1994.
- [4] *Machine Learning*. Tom M. Mitchell, 1997.
- [5] *Introduction to Machine Learning*. Ethem Alpaydin.
- [6] Dan Klein. *The Unsupervised Learning Of Natural Language Structure*. PhD thesis, Stanford University, 2005.
- [7] F. Woergoetter and B. Porr. Reinforcement learning. *Scholarpedia*, 2008.
- [8] Tom M. Mitchell. The discipline of machine learning. Technical report, School of Computer Science Carnegie Mellon University-Pittsburgh-PA 15213, 2006.
- [9] *The Handbook of Computational Linguistics and Natural Language Processing*. Wiley-Blackwell, 2010.
- [10] Pasi Tapanainen and Timo Jarvinen. Syntactic analysis of natural language using linguistic rules and corpus-based patterns. *International Conference on Computational Linguistics Proceedings*, 1994.

- 
- [11] Yiming Yang. Combining prediction, syntactic analysis and semantic analysis in chinese sentence analysis. *Computing Research Laboratory, New Mexico State University*, 1994.
  - [12] Fuji Ren Shingo Kuroiwa Jiajun Yan, David B. Bracewell. A semantic analyzer for aiding emotion recognition in chinese. *Computational Intelligence, International Conference on Intelligent Computing, ICIC*, 2006.
  - [13] Day Jr Paul C. Jordan J. Douglas Wingate Eugene E. Loos Susan Anderson, Dwight H. Glossary of linguistic terms. Technical report, SIL International, 2004.
  - [14] *Semantic Interpretation and the Resolution of Ambiguity*. Cambridge University Press, 1992.
  - [15] Roberto Navigli. Word sense disambiguation: A survey. *ACM Comput. Surv.*, 41(2):1–69, 2009.
  - [16] H. Srimathi. Article:knowledge representation of lms using ontology. *International Journal of Computer Applications*, 6(3):35–38, September 2010. Published By Foundation of Computer Science.
  - [17] *Social Ontology: Recasting Political Philosophy Through a Phenomenology of Whoness*.
  - [18] Alessandro Lenci. Building an ontology for the lexicon: Semantic types and word meaning.
  - [19] Mingxia Gao, Chunnian Liu, and Furong Chen. An ontology search engine based on semantic analysis.
  - [20] *WordNet An Electronic Lexical Database*. Cambridge, MA: MIT Press, 1998.
  - [21] Jelber Sayyad Shirabad. *Supporting Software Maintenance by Mining Software Update Records*. PhD thesis, University of Ottawa, 2003.
  - [22] James Bedford. Machine learning, decision trees and entropy. *Wordpress-CS*, 2010.

- 
- [23] L. Findlater W. Olive H. Hamilton. E. Gurak. Overview of decision trees. Technical report, University of Regina, 2001.
- [24] Bill Wilson. Induction of decision trees. Technical report, UNSW's CRICOS, 2008.
- [25] Christopher Roach. Building decision trees in python. *ONLAMP*, 2006.
- [26] *Constructing a Language: A Usage-Based Theory of Language Acquisition*. Harvard University Press, Cambridge, MA., 2001.
- [27] Shimon Edelman. Language acquisition in humans and computers. Technical report, University of Cornell, 2005.
- [28] M. R. Vervoort. *Games, Walks and Grammars. PhD thesis*. PhD thesis, Universiteit van Amsterdam, 2000.
- [29] *Papers On Syntax*. Springer, 1981.
- [30] Matan Mussel Ben Sandbank Eytan Ruppim Shimon Edelman Jonathan Berant, Yaron Gross. Learning syntactic constructions from raw corpora. Technical report, Tel Aviv University and Cornell University, 2003.
- [31] Matan Mussel Jonathan Berant, Yaron Gross. Boosting unsupervised grammar induction by splitting complex sentences on function words. Technical report, Tel Aviv University and Cornell University, 2003.
- [32] Matan Mussel Ben Sandbank Eytan Ruppim Shimon Edelman Jonathan Berant, Yaron Gross. Automatic acquisition and efficient representation of syntactic structures. Technical report, Tel Aviv University and Cornell University, 2007.

 <b>Avdelning, Institution</b> Division, Department  IDA, Dept. of Computer and Information Science 581 83 LINKÖPING		<b>Datum</b> Date  2011-04-10
<b>Språk</b> Language  <input type="checkbox"/> Svenska/Swedish <input checked="" type="checkbox"/> Engelska/English  <input type="checkbox"/> _____	<b>Rapporttyp</b> Report category  <input type="checkbox"/> Licentiatavhandling <input checked="" type="checkbox"/> Examensarbete <input type="checkbox"/> C-uppsats <input type="checkbox"/> D-uppsats <input type="checkbox"/> Övrig rapport <input type="checkbox"/> _____	<b>ISBN</b> _____  <b>ISRN</b> _____  <b>Serietitel och serienummer ISSN</b> Title of series, numbering _____
<b>URL för elektronisk version</b> <a href="http://www.ep.liu.se/exjobb/ida/202011/dd-d/xxx/">http://www.ep.liu.se/exjobb/ida/202011/dd-d/xxx/</a>		
<b>Titel</b> TITEL  Title      Semantic Analysis Of Multi Meaning Words Using Machine Learning And Knowledge Representation  <b>Författare</b> Marjan Alirezaie Author		
<b>Sammanfattning</b> Abstract  <p>The present thesis addresses machine learning in a domain of natural-language phrases that are names of universities. It describes two approaches to this problem and a software implementation that has made it possible to evaluate them and to compare them.</p> <p>In general terms, the system's task is to learn to 'understand' the significance of the various components of a university name, such as the city or region where the university is located, the scientific disciplines that are studied there, or the name of a famous person which may be part of the university name. A concrete test for whether the system has acquired this understanding is when it is able to compose a plausible university name given some components that should occur in the name. In order to achieve this capability, our system learns the structure of available names of some universities in a given data set, i.e. it acquires a grammar for the microlanguage of university names. One of the challenges is that the system may encounter ambiguities due to multi meaning words. This problem is addressed using a small ontology that is created during the training phase.</p> <p>Both domain knowledge and grammatical knowledge is represented using decision trees, which is an efficient method for concept learning. Besides for inductive inference, their role is to partition the data set into a hierarchical structure which is used for resolving ambiguities.</p> <p>The present report also defines some modifications in the definitions of parameters, for example a parameter for entropy, which enable the system to deal with cognitive uncertainties. Our method for automatic syntax acquisition, ADIOS, is an unsupervised learning method. This method is described and discussed here, including a report on the outcome of the tests using our data set.</p> <p>The software that has been implemented and used in this project has been implemented in C.</p>		
<b>Nyckelord</b> Machine Learning, Supervised Learning, Unsupervised Learning, Ontology, Decision Tree, ADIOS, Grammar Induction Keywords		



# Copyright

## Svenska

Detta dokument hålls tillgängligt på Internet - eller dess framtida ersättare - under 25 år från publiceringsdatum under förutsättning att inga extraordinära omständigheter uppstår.

Tillgång till dokumentet innebär tillstånd för var och en att läsa, ladda ner, skriva ut enstaka kopior för enskilt bruk och att använda det oförändrat för ickekommersiell forskning och för undervisning. Överföring av upphovsrätten vid en senare tidpunkt kan inte upphäva detta tillstånd. All annan användning av dokumentet kräver upphovsmannens medgivande. För att garantera äktheten, säkerheten och tillgängligheten finns det lösningar av teknisk och administrativ art.

Upphovsmannens ideella rätt innefattar rätt att bli nämnd som upphovsman i den omfattning som god sed kräver vid användning av dokumentet på ovan beskrivna sätt samt skydd mot att dokumentet ändras eller presenteras i sådan form eller i sådant sammanhang som är kränkande för upphovsmannens litterära eller konstnärliga anseende eller egenart.

För ytterligare information om *Linköping University Electronic Press* se förlagets hemsida <http://www.ep.liu.se/>

## English

The publishers will keep this document online on the Internet - or its possible replacement - for a period of 25 years from the date of publication barring exceptional circumstances.

The online availability of the document implies a permanent permission for anyone to read, to download, to print out single copies for your own use and to use it unchanged for any non-commercial research and educational purpose. Subsequent transfers of copyright cannot revoke this permission. All other uses of the document are conditional on the consent of the copyright owner. The publisher has taken technical and administrative measures to assure authenticity, security and accessibility.

According to intellectual property law the author has the right to be mentioned when his/her work is accessed as described above and to be protected against infringement.

For additional information about the *Linköping University Electronic Press* and its procedures for publication and for assurance of document integrity, please refer to its WWW home page: <http://www.ep.liu.se/>