

*Master Thesis*  
*Software Engineering*  
*Thesis no: MSE-2007:22*  
*June 2006*



# **Pattern Acquisition Methods for Information Extraction Systems.**

**Michał Marcińczuk**

School of Engineering  
Blekinge Institute of Technology  
Box 520  
SE - 372 25 Ronneby  
Sweden

This thesis is submitted to the School of Engineering at Blekinge Institute of Technology in partial fulfillment of the requirements for the degree of Master of Science in Software Engineering. The thesis is equivalent to 20 weeks of full time studies.

**Contact Information:**

Author(s):

Michał Marcińczuk

Address: ul. Dobra 21, 66-400 Gorzów Wlkp., Poland

E-mail: marcinczuk@gmail.com

External advisor(s):

Maciej Piasecki, Ph.D.

Company/Organisation: Wrocław University of Technology, Poland

Address:

Phone:

University advisor(s):

Niklas Lavesson

School of Engineering, BTH

School of Engineering  
Blekinge Institute of Technology  
Box 520  
SE - 372 25 Ronneby  
Sweden

Internet : [www.bth.se/tek](http://www.bth.se/tek)  
Phone : +46 457 38 50 00  
Fax : +46 457 271 25

# ABSTRACT

This master thesis treats about Event Recognition in the reports of Polish stockholders. Event Recognition is one of the Information Extraction tasks. This thesis provides a comparison of two approaches to Event Recognition: manual and automatic. In the manual approach regular expressions are used. Regular expressions are used as a baseline for the automatic approach. In the automatic approach three Machine Learning methods were applied. In the initial experiment the Decision Trees, naive Bayes and Memory Based Learning methods are compared. A modification of the standard Memory Based Learning method is presented which goal is to create a classifier that uses only positives examples in the classification task. The performance of the modified Memory Based Learning method is presented and compared to the baseline and also to other Machine Learning methods. In the initial experiment one type of annotation is used and it is the meeting date annotation. The final experiment is conducted using three types of annotations: the meeting time, the meeting date and the meeting place annotation. The experiments show that the classification can be performed using only one class of instances with the same level of performance.

**Keywords:** Natural Language Processing, Information Extraction, Patterns Acquisition, Linguistic Patterns, Memory Based Learning, Event Recognition

# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Background . . . . .	1
1.2	Motivation and Goal . . . . .	2
1.3	Related Works . . . . .	2
1.4	Aims and Objectives . . . . .	3
1.5	Research Questions . . . . .	3
1.6	Expected Outcomes . . . . .	3
1.7	Research Methods . . . . .	3
1.8	Thesis Outline . . . . .	5
<b>2</b>	<b>Information Extraction</b>	<b>7</b>
2.1	Introduction . . . . .	7
2.2	Basic Definitions . . . . .	8
2.3	Information Extraction System Architecture . . . . .	9
<b>3</b>	<b>Measuring Information Extraction Systems Performance</b>	<b>11</b>
3.1	Basic Metrics . . . . .	11
3.2	Cross Validation . . . . .	13
<b>4</b>	<b>Event Recognition Task</b>	<b>14</b>
4.1	Event Recognition as a Classification Task . . . . .	14
4.2	Instance Features . . . . .	16
<b>5</b>	<b>Methods of Patterns Acquisition</b>	<b>19</b>
5.1	Manual pattern creation . . . . .	20
5.2	Automatic pattern creation . . . . .	22
5.3	Decision Trees C4.5 . . . . .	23
5.4	Memory-Based Learning . . . . .	24
5.4.1	The IB1 algorithm . . . . .	24
5.4.2	The IGTREE algorithm . . . . .	25
5.4.3	Memory-Based Learning in Semantic Labeling . . . . .	27
5.5	Bootstrapping . . . . .	27
5.5.1	Example of Bootstrapping . . . . .	28
5.5.2	Mutual Bootstrapping and Meta Bootstrapping . . . . .	30
5.6	Conclusion . . . . .	30

<b>6</b>	<b>Experiment</b>	<b>32</b>
6.1	Introduction . . . . .	32
6.2	Preparation of the Training Set . . . . .	32
6.3	Baseline . . . . .	35
6.3.1	Results for <i>Zgromadzenie godzina</i> . . . . .	36
6.3.2	Results for <i>Zgromadzenie data</i> . . . . .	37
6.3.3	Results for <i>Zgromadzenie miejsce</i> . . . . .	37
6.4	Initial Experiment . . . . .	38
6.5	Experiment with Memory Based Learning . . . . .	40
6.6	Memory Based Learning Modification . . . . .	41
6.7	Experiment with Modified Memory Based Learning Method . . . . .	44
<b>7</b>	<b>Results and Conclusion</b>	<b>46</b>
7.1	Performance of the Modified Memory Based Learning . . . . .	47
7.1.1	Results for <i>Zgromadzenie godzina</i> . . . . .	47
7.1.2	Results for <i>Zgromadzenie data</i> . . . . .	49
7.1.3	Results for <i>Zgromadzenie miejsce</i> . . . . .	51
7.1.4	Summary of the Results . . . . .	53
7.2	Comparison to the Baseline . . . . .	53
7.3	Conclusion . . . . .	54
7.3.1	RQ1 How can the process of patterns acquisition for Events Recognition from the reports of Polish stockholders be automated? . . . . .	54
7.3.2	RQ2 What are the approaches to patterns acquisition for the Event Recognition task? . . . . .	54
7.3.3	RQ3 How can Event Recognition be treated as a classification task in order to apply Decision Trees, Naive Bayes and Memory Based Learning methods? . . . . .	55
7.3.4	RQ4 What performance can be obtained by applying common machine learning algorithms to pattern acquisition for event recognition for the Polish stockholders reports task? . . . . .	55
7.4	Future Work . . . . .	56
7.4.1	Question to answer . . . . .	56
7.4.2	Place for improvements . . . . .	56
<b>A</b>	<b>Stock Exchange Schema Annotation</b>	<b>57</b>
<b>B</b>	<b>Stock Exchange Schema Annotation</b>	<b>58</b>
<b>C</b>	<b>Example of training instances</b>	<b>59</b>

# Chapter 1

## Introduction

### 1.1 Background

Natural languages are languages developed by humans for a general-purpose communication. People use natural languages in their daily life in the spoken and the written form to communicate between each other. There are many natural languages over the world and each of them is defined by its own set of rules. The set of rules is called grammar. Grammar defines language syntactic that is the acceptable sequence of characters from the language alphabet.

However, the fact that a message is correct in terms of grammar rules does not guarantee that it will be understandable for someone who knows that language. The semantic information makes the message understandable. Semantic is the meaning of any sequence of characters (words) from the language alphabet. The ability to interpret the fragments of texts allow human to find the relevant information and understand its meaning.

Natural Language Processing (NLP) is a domain that studies the problems of the automated generation and understanding of the natural human languages. NLP is a subdiscipline of the Artificial Intelligence field. One of the major task in NLP is the Information Extraction. The main goal of the Information Extraction Systems is to retrieve the relevant information from the unstructured but computer-readable documents.

Information Extraction Systems have to face with many problems, like:

- Text Segmentation [36],
- Part of Speech Tagging [10], [34],
- Word Sense Disambiguation [21],
- Coreference Resolving [32],
- Named Entity Recognition [35], [17],
- Events Recognition [38], [28],
- Terminology Extraction [33], [6], [7].

Most of the problems can be defined as a classification task. For example in Text Segmentation we have to determine if a character represents the end of a segment (token or sentence) or not. The goal of Part of Speech Tagging is to classify the tokens in terms of the grammar and the syntactic characteristic. Coreference resolving determines if a pair of the named entities refers to the same object. In turn Named Entity Recognition and Events Recognition are not typical classification tasks.

Events Recognition is a specific case of the Named Entity Recognition. The Named Entity Recognition relies on finding all instances of a given entity type (for example dates). In turn, in Events Recognition the role of the instance is taken into account (for example, the date of a meeting and the date of a trust deposit). There are two approaches to Named Entity Recognition. The first approach makes use of the linguistic patterns that are used to describe the information structure and the context by a pseudo language. In the other approach the problem is treated as a classification task and the Machine Learning techniques are applied [18]. Both strategies have pros and cons, for example, the linguistic patterns have to be created manually by domain experts what is very time consuming but this approach gives better results than the Machine Learning techniques [23].

## 1.2 Motivation and Goal

**The motivation** to develop a method for *patterns acquisition for Event Recognition* task is to create a system that will monitor the condition of the stock issuers by analyzing the reports published by them. The condition of the company can be used to determine the risk and profitability of investments. According to Polish law [4] every stock issuer is obligated to publish the reports about 26 types of events defined by the act. The idea is to analyze those reports in order to track the condition of the stock issuers in the time. Because of the huge number of reports (about 2.000 - 6.000 reports are published every month) there is a need to automate the process by developing the Information Extraction system.

**The motivation** to develop a method for *automatic pattern acquisition* is to reduce the effort needed in the manual approach to construct a set of the extraction patterns for every annotation type.

**The goal** of the thesis is to develop and evaluate a method for the automatic pattern acquisition for Event Recognition task from the reports of Polish stockholders.

## 1.3 Related Works

Kupsc et al. [24] present an Information Extraction system that retrieves medical data from Polish mammographic reports. It is based on the SProUT platform and makes use of manually created patterns. The extraction process was divided into two steps. In the first step extraction rules were used to retrieve small pieces of information. After this all the information were combined into one structure.

Burnside et al. [12] present another approach to Information Extraction from mammographic reports for English language that makes use of Bayesian network with manually created model and manually assigned probabilities.

Within the series of Message Understanding Conferences (MUCs) several Information Extraction systems were developed. In the MUC-3 and MUC-4 the task was to extract information about the terrorist activities from the news reports. In turn the task for MUC-7 was to retrieve information about the airplane crashes and the rocket/missiles launches from the news reports. Systems created for the purpose of Message Understanding Conferences used manually created rules.

Bennett et al. [8] created Information Extraction system namely RoboTag that used decision trees (C4.5) to recognize the beginnings and the endings of annotations. Then a heuristic method was used to pair the beginnings and the endings.

IdentiFinder [9] and The Kent Ridge Digital Labs System [48] represent a group of systems that apply the statistical methods like Hidden Markov Models. In this approach the model represents the probabilities that sequence of tokens belong to the given class. The states represents classes (for instance, a person name, an organization name or non-an-entity) and the transitions determine the probability that the state will be changed if given token occurs.

## 1.4 Aims and Objectives

- to gather concepts of automatic pattern acquisition methods,
- to develop a method for automatic pattern acquisition for Event Recognition,
- to evaluate the developed method on the reports of Polish stockholders,
- to compare the results of the developed method with the manual approach.

## 1.5 Research Questions

The main research question is:

**RQ1** How can the process of patterns acquisition for Events Recognition from the reports of Polish stockholders be automated?

In order to answer the main research question, the minor research question have been stated:

**RQ2** What are the approaches to patterns acquisition for the Event Recognition task?

**RQ3** How can Event Recognition be treated as a classification task in order to apply Decision Trees, Naive Bayes and Memory Based Learning methods?

**RQ4** What performance can be obtained by applying common machine learning algorithms to pattern acquisition for event recognition for the Polish stockholders reports task?

## 1.6 Expected Outcomes

- an annotated training set of the Polish reports form the exchange stock domain,
- a methods for automatic pattern acquisition for Event Recognition from the reports of Polish stockholders,
- a stand alone application that will support the evaluation of the developed methods,
- an evaluation of the developed methods on the set of the documents from the stock exchange domain.

## 1.7 Research Methods

The main activities and artifacts have been presented on the Figure 1.1.

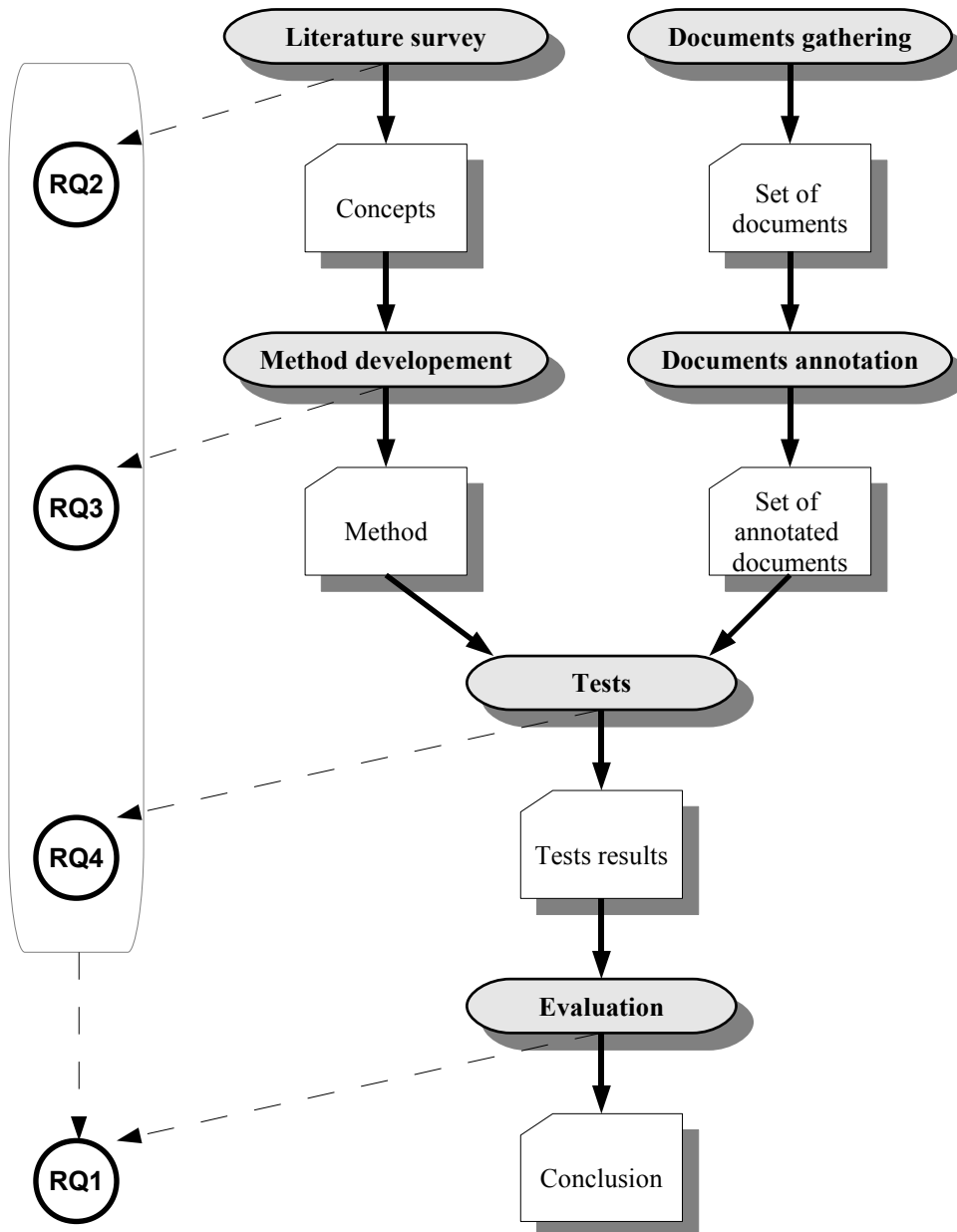


Figure 1.1: The work flow with the main activities and artifacts

Firstly, I conducted a literature survey regarding related works. The goal of the survey was to answer the Research Question 2 i.e. *What are the methods of patterns acquisition?* The output of the survey was a list of ideas how the automation of the pattern acquisition can be obtained.

In the following step, basing on the list of possible solutions the methods for the manual and the automatic patterns acquisition were developed. In the manual approach the regular expressions were used. In the automatic approach the Decision Trees, Naive Bayes and Memory Based Learning (MBL) methods were used. The goal of this step was to answer the Research Question 3 i.e. *How can Event Recognition be treated as a classification task in order to apply Decision Trees, Naive Bayes and Memory Based Learning methods?*

Simultaneously to the literature survey and the methods development, I was gathering the documents in order to create the training set with the reports about annual meetings of the stockholders. After gathering the relevant documents, each of them was manually verified and annotated in terms of the date, the time and the place of the annual meeting.

In the next step, the developed methods and the training set of documents were used in the tests. The goal of this step was to develop a software that will support the process of testing the methods with different configuration sets. The output of this step was a set of testing results. The results of this step was the answer for the Research Question 4 i.e. *What performance can be obtained by applying common machine learning algorithms to pattern acquisition for event recognition for the Polish stockholders reports task?*

In the last step of the work the results of the experiments were discussed and the automatic approach was compared with the base line. The base line for the experiment was the manual approach. The goal of the last step was to answer the main Research Question 1 *How can the process of patterns acquisition for Events Recognition from the reports of Polish stockholders be automated?*.

## 1.8 Thesis Outline

**Chapter 1: Introduction** presents the Research Plan of my thesis. This chapter contains the background, the motivation, the goal, the aims and objectives, the expected outcomes and the research methods.

**Chapter 2: Information Extraction** presents the basic definitions regarding Natural Language Processing domain and introduces the general concept of the Information Extraction system.

**Chapter 3: Measuring Information Extraction Systems Performance** presents the overview of the widely used metrics in the NLP domain and provides the motivation of the metrics choice that were used to compare the performance of the developed methods.

**Chapter 4: Event Recognition Task** provides the description of the Event Recognition as a classification task. Three concepts from the literature are presented that treats Event Recognition as the classification task. The potential and used instance features are presented.

**Chapter 5: Methods of Patterns Acquisition** presents the methods (manual and automatic) of pattern acquisition and concepts how the pattern acquisition process can be automated.

**Chapter 6: Experiment** presents the results of the initial experiment where the baseline is presented. Three Machine Learning methods are compared and the modification of the Memory Based Learning method is described.

**Chapter 7: Results and Conclusion** presents the performance of the modified Memory Based Learning methods, the discussion of the results, the answers for the Research Questions and the future work.

## Chapter 2

# Information Extraction

### 2.1 Introduction

Information Extraction (IE) is one of the Natural Language Processing (NLP) tasks. The purpose of Information Extraction, in terms of the NLP domain, is to create a system that process the digital documents written in any of the natural languages and identify relevant information. In other words such system will be able to discover semantic information in the documents. The Information Extraction System can be also perceived as a set of linguistic tools and resources that combined are used to performed specific task related to the natural language processing.

In general the IE System can be presented as a black box (Figure 2.1) that uses linguistic resources to fulfill given task. A user delivers documents in a natural language to the IE system and receives the result of the processing. The results depend on the task characteristic.

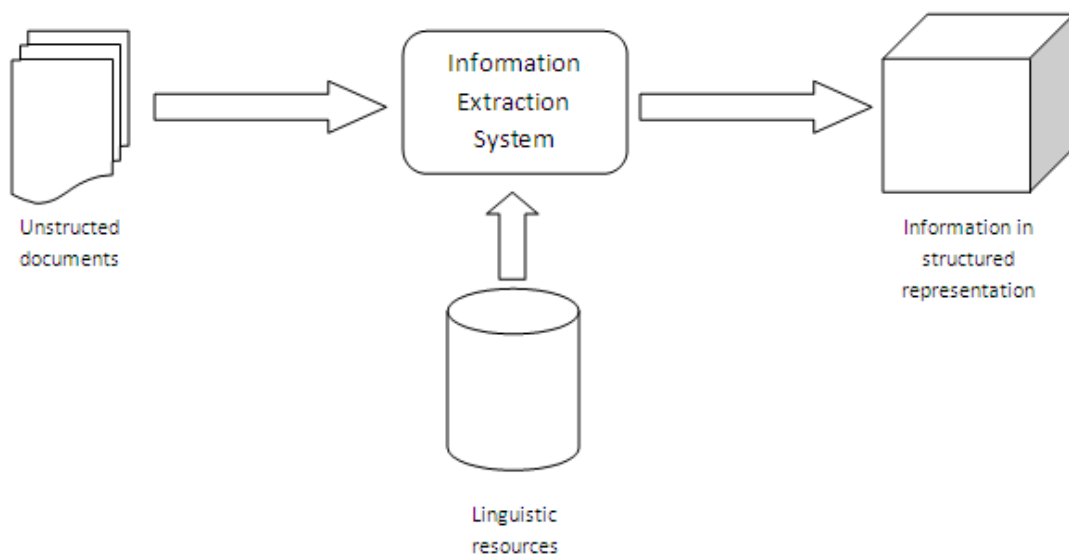


Figure 2.1: Information Extraction System as a black box

## 2.2 Basic Definitions

This section provides basic terminology used in the Natural Language Processing domain.

**Annotation** is an information that is assigned to a chunk of document. The annotation can provide various information about the chunk like text characteristic, grammatical class or semantic information. Each annotation is linked with a token or a sequence of the tokens.

**Corpus** is "a collection of written texts" [3].

**Entity** is an object of interest like a person, a date, a location or a thing [1].

**Event** is an activity or occurrence of interest [1], i.e. stockholders meeting.

**Fact** is any relation between two or more entities [1].

**Feature** is a measurable linguistic characteristics that is used to represent a training instance [19]. For example feature can be length of the instance or the base of the proceeding token. Let's consider example sentence "*Mr. Thomas was hired by Electromax Inc.*" where *Thomas* is the instance that represent a person name. Table 2.1 presents example features that can be used to learn name identification.

Feature	Value
proceeding token value	Mr.
following token value	was
instance capitalization	first capital letter

Table 2.1: Example instance features.

**Information extraction** is a process of information extraction from given text and putting them into the structured, easy to process by computers, form [1].

**Information Extraction system** is a set of tools used to extract information from the documents [7].

**Linguistic pattern** is a pattern that defines a repeatable structure of a text in terms of linguistic characteristics. Linguistic pattern makes use of text characteristic as well as syntactic and semantic information about the text. They are used to describe the text structures in which given type of information can appear in the documents.

**Named entity** is a named object of interest like person or company name [1].

**Question Answering system** is a system that provide an answer for given question written in natural languages [47]. The system analyze a question and search the knowledge base for the correct answer.

**Tag** , according to the Oxford Dictionary [3], is "a label providing identification or giving other information". In the Information Extraction domain the tags are largely related to the words part of speech categories [7].

**Token** is the smallest, atomic part of text that can be processed by Information Extraction System.

**Pattern** , according to the Oxford Dictionary [3], is "a regular or discernible form or order in which a series of things occur" or "a model, design, or set of instructions for making something".

## 2.3 Information Extraction System Architecture

The common architecture of Information Extraction systems consist of 4 main modules shown on the Figure 2.2.

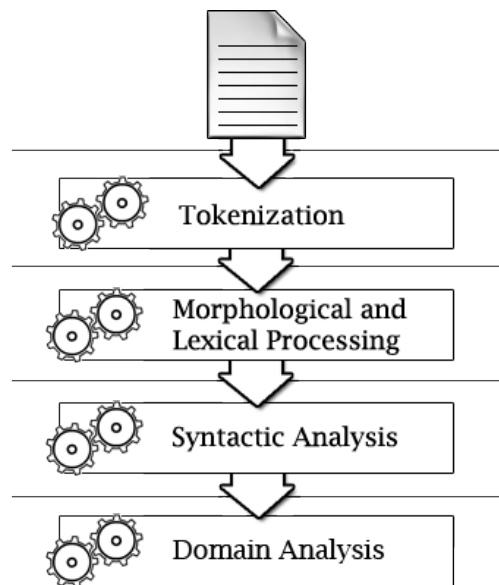


Figure 2.2: General Information Extraction System architecture.

The general architecture of IE systems is based on a pipeline processing where set of modules is executed in given order. An output of one module is an input for another. The order in which the subsequent modules are executed is essential because some modules provide or require an information that is required or provided by other modules. Depending on the task specification and the language characteristic different types of modules are plugged into the pipeline. Figure 2.3 presents some basic modules that are used in Information Extraction Systems.

Overview of IE modules [7]:

**Text segmentation** that includes:

**Word segmentation** is a process of dividing text into words. This problem is trivial for languages that has explicit word boundaries like white spaces. However in some languages (for examples Chinese) there is no explicit boundaries and to solve this problem more complex techniques must be applied.

**Sentence segmentation** is a process of dividing text into sentences.

**Part of Speech Tagging** [10], [34] is a process of determining word part of speech, based on definition and the context in which the word appear (like relation to other words in the sentence). In addition the POS tagging identifies the case, the number and the person for every token. It is common that after POS tagging some words get more than one interpretation. For example word *play* can be used as:

- verb and represent an action of playing a game or on a musical instrument,
- noun and represent a piece of writing performed in a theater.

**Full Parsing** is a process of transformation the sentence into a tree structure that depict relations between tokens [16]. In such structure node children represent arguments of the predicate in the node.

**Coreference resolution** [32] is the process of identification different words (in general) in a document that refers to the same instance. For example, in a fragment of a document *This is Mark. He is my best friend.* words *Mark* and *He* refer to the same person.

**Named Entity Recognition/Identification** [35], [17] also known as Named-entity Recognition, is a process that leads to locate and classify atomic elements in the text into predefined categories like people, companies, locations, expressions of time, monetary values and many others. During this process some semantic information is attached to the document.

**Event Recognition** [38], [28] is a process similar to *Named-entity identification*, however, the information attached to elements of the text is more precise and takes into consideration not only semantic class but also meaning of the element. For example if we have a document that treats about a concert announcement and it contains two dates: one represents the concert date and the other the date up till tickets can be bought. In this case both dates belongs to the same semantic category but have different meanings.

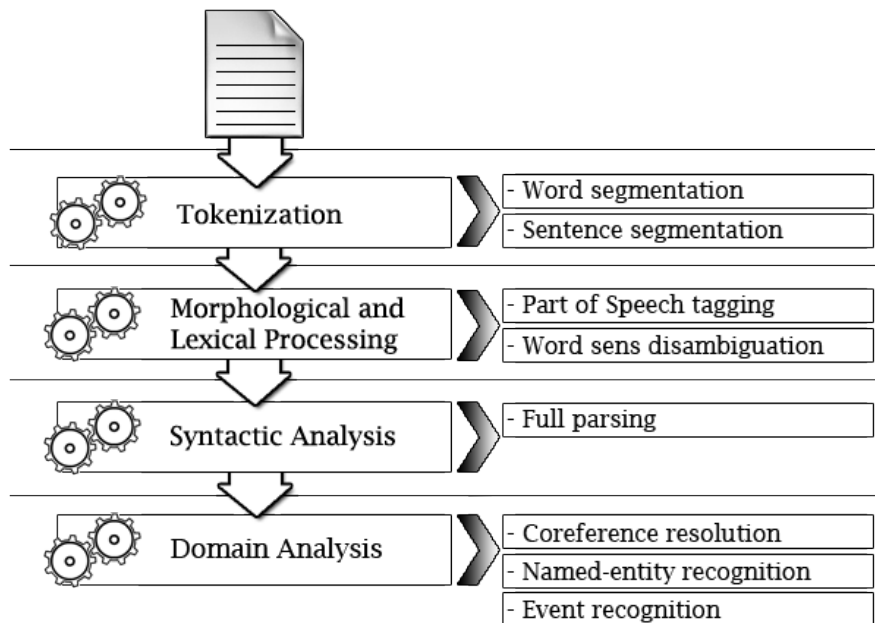


Figure 2.3: Modules in Information Extraction System.

## Chapter 3

# Measuring Information Extraction Systems Performance

### 3.1 Basic Metrics

The common way used to compare the performance of different Information Extraction systems is to compare the ratio of correctly and incorrectly classified positive instances [7]. The negative instances are not taken into consideration in the evaluation because the number of negative instances is very prevalent. Also the accuracy of classifying the positive instances is much more important for Information Extraction than the classification of the negative instances.

The results of the classification task are divided into four groups which are used to evaluate the system performance [31]. They are as follows :

**True Positives (TP)** - positive instances correctly classified as positive,

**True Negative (TN)** - negative instances correctly classified as negative,

**False Positives (FP)** - negative instances incorrectly classified as positive,

**False Negative (FN)** - positive instances incorrectly classified as negative.

There are two metrics that represent the system performance with respect to the number of TP. It is Precision (P) and Recall (R). In addition third metric namely F-measure is used that combines the value of the Precision and the Recall.

Below the equations of the Precision, the Recall and the F-measure are presented:

- precision - defined as:

$$P = \frac{\text{number of relevant instances in the result}}{\text{number of instances in the result}} = \frac{TP}{TP + FP}$$

- recall - defines as:

$$R = \frac{\text{number of relevant instances in the result}}{\text{number of all relevant instances in the set}} = \frac{TP}{TP + FN}$$

- F-measure - is a combination of precision and recall and is defines as [7]:

$$F = \frac{(\beta^2 + 1)PR}{\beta^2 * P + R},$$

where  $\beta$  is a parameter that represents relative importance of  $P$  and  $R$ . Typically  $\beta$  has the value 1, what means that precision and recall are equally import. This is the weighted harmonic mean of the Precision and the Recall.

The common way to present the tests result for different configuration of learning and testing parameters is a plot of precision versus recall. Figure 3.1 show an example plot where each point on the plot represents single test case with some configuration of parameters. This type of plot depicts the relation between recall and precision - the higher recall is the lower precision. This relation can be approximated to the curve presented on the figure 3.2.

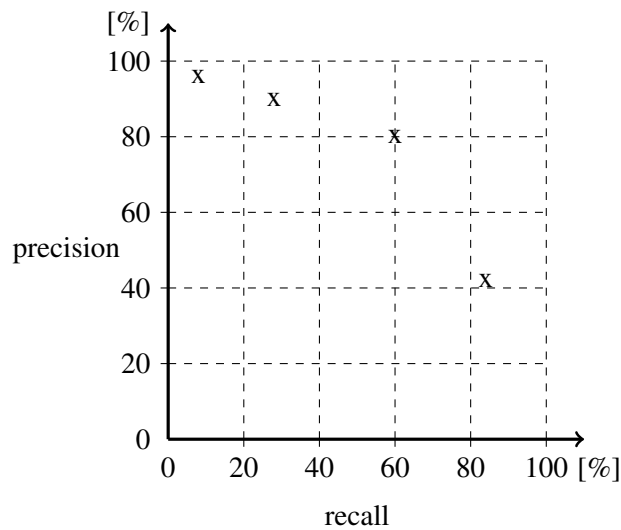


Figure 3.1: An example of precision versus recall plot.

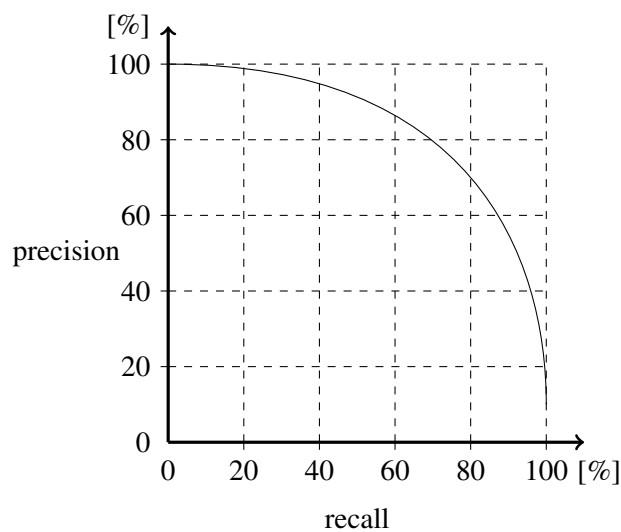


Figure 3.2: An example of precision versus recall plot approximated to curve.

## 3.2 Cross Validation

When a system is using rules learned with any machine learning method or technique instead of rules created manually there is a need to take into consideration one more factor that is related to the problem of estimating generalization error. In this case the whole set of instances cannot be used in the learning process because there is no point in testing the system on the same set. Such result will be misleading because the systems performance will be based on testing the "known" instances. More important is how the system will cope with the unknown instances. One of the common methods in this case is Cross-Validation (CV) [22]. The general idea of CV is to divide the set of instances into several subsets and test the system using different combination of the subsets as the training and the testing sets. For example in 10-fold Cross Validation the set of instances is divided into 10 subsets and in each test 9 subsets are used as the training set and the 10th as testing set. In every test different subset from the 10 is used in the testing.

Metrics presented above are widely used in comparing Information Extraction systems' performance. However those values cannot be directly compared between different systems unless the tests were conducted on the same set of instances. Moreover the top score is not always 100%. It is caused by the fact that natural language is not always unambiguous and can be interpreted in different ways by different people. So there is a need to estimate the top score by tests conducted by an independent group of people. In other words the system cannot learn to recognize relevant information if group of people is not coherent with the interpretation. However the process of estimating the top score is very time-consuming and require additional resources.

## Chapter 4

# Event Recognition Task

### 4.1 Event Recognition as a Classification Task

In the naive approach Event Recognition can be treated as a single-classification task that relies on testing every subsequence of tokens in the document. The concept of the single-classification task is presented in the Figure 4.1c. The idea of this approach is to construct two-class classifier and classify every subsequence of the document as an annotation or non-annotation. In this approach the main problem is the huge number of instances to test. This is because the length of the annotations is not fixed and different subsequences should be tested. We do not know the length of the annotation so we have to test all 1-token, 2-tokens, ..., n-tokens subsequences, where the n is the document length equal to the number of tokens. We can limit the search space by testing only sequences which length is between the minimum and the maximum annotation length observed in the training set.

Event Recognition can be also treated as a multi-classification task. The idea is to decompose Event Recognition into several classification tasks and then merge the partial results into whole annotations. Bennett et al. [8] used two classifiers to recognize the beginnings and the endings of annotations. The concept of this approach is presented on the Figure 4.1a. After selecting potential candidates for annotation boundaries the heuristic rule is used to determine which pairs of the starting and the ending boundaries create an annotation. Each potential beginning and ending has a certainty factor, a value between 0 and 1 that represents the probability of correct classification. The heuristic rule used the values of certainty factors to rank the annotations. This concept could be modified by using another classifier instead of the heuristic rule to determine which pairs create an annotation.

Another concept of multi-classification approach is presented by Pradhan et al. [37]. The used IOB representation (abbreviation stands for Inside Outside Begin) and created one three-class classifier. Each token is classify as being the beginnings of the annotation, being inside or outside the annotation. Then a heuristic rule is used to decide which sequences create correct annotation. This concept is presented on the Figure 4.1b.

I decide to use the concept presented in [8] because the process is described in details and, in turn, some aspects of the IOB presentation [37] are not clarified. In the IOB approach the process of selecting the final annotations is not fully clarified. I also decide not to use the naive approach because of two reasons. The first reason is the huger number of instances. The second reason is the potential problem with the representation of the training instances.

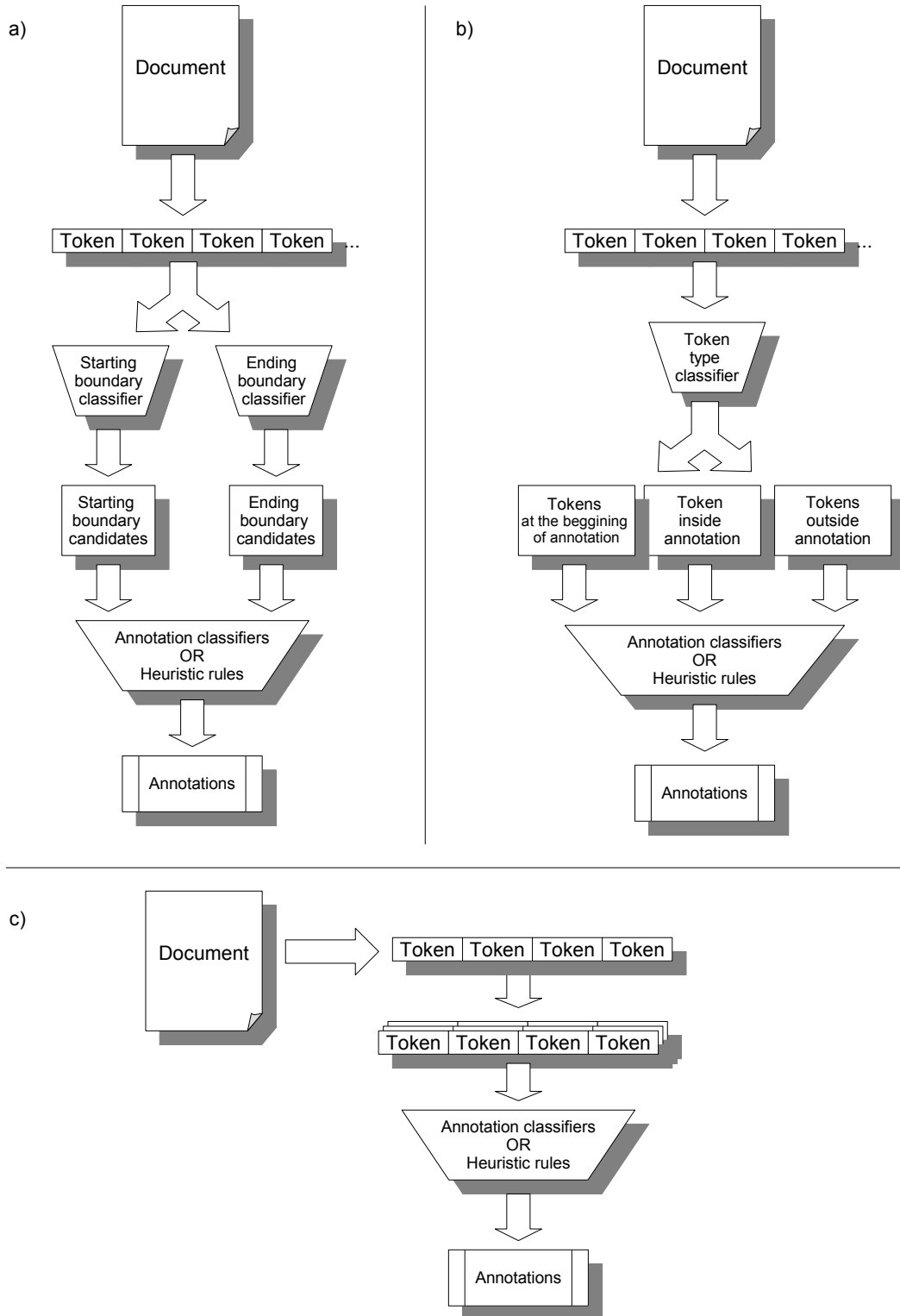


Figure 4.1: Annotation recognition concepts

## 4.2 Instance Features

The features that might be used to represent the learning and testing instances (the starting and the ending boundaries of an annotation):

- Description of token characters - refers to the token structure. I have distinguished 7 groups of tokens in terms of their structure. The groups are presented in the Table 4.1.

Group	Description
LowerCase	a sequence of lower case letters
UpperCase	a sequence of upper case letters
InitCap	a capitalized word
MixedCase	a sequence of lower and upper case letters
Number	a sequence of digits
Symbol	a sequence of characters other than letters and digits
Mixed	a sequence of letters, digits and/or symbols

Table 4.1: Groups of tokens in terms of their structure

- Grammatical class - a tag that represents token number, case, gender, person, degree, aspect and other. Full list of grammatical categories and their values is presented in [39].
- Morphological base form - is used to generalize the variety of word forms.
- Shallow/full parsing - can be used to create predicate-argument model of a sentence that provides features regarding semantic role of the sentence elements [44]. However, I did not use this feature because there is no effective parser for Polish language available.
- Semantic groups - can be used to measure the similarity between words. Words from the same semantic category seems to be more similar then belonging to two other categories. For instance word 'august' is more similar to 'may' then to 'cat'. This is intuitive because words 'august' and 'may' are the names of months. Unfortunately this feature was not be used because the semantic word net is being constructed [29].

Below is the final list of features used in the experiment:

- **base of outer tokens** - the base of  $o$  proceeding (for the starting boundary) or following (for the closing boundary) tokens.
- **grammatical category of outer tokens** - the grammatical category of  $o$  proceeding (for the starting boundary) or following (for the closing boundary) tokens,
- **type of outer tokens** - the structure category of  $o$  proceeding (for the starting boundary) or following (for the closing boundary) tokens,
- **base of inner tokens** - the base of  $i$  first (for the starting boundary) or last (for the closing boundary) tokens,
- **grammatical category of inner tokens** - the grammatical category of  $i$  first (for the starting boundary) or last (for the closing boundary) tokens,
- **type of inner tokens** - the structure category of  $i$  proceeding (for the starting boundary) or following (for the closing boundary) tokens,

- **base of proceeding prepositions** - base of  $p$  tokens that are prepositions that occur on the left side of the boundary,
- **base of proceeding nouns** - base of  $n$  tokens that are nouns that occur on the left side of the boundary.

*The outer tokens* feature is used to catch the structure of the text that proceeds and follows the annotation. In turn *the inner tokens* feature is used to catch the structure of the annotation. The structure of the annotation is very simplified because it concerns only peripheral tokens. The base of the tokens is used to store keywords that are typical for the annotation or the context, however we do not know which of the tokens are keywords. For example, in the following location annotation “ul. Nowowiejska 12” *ul* is a keyword but *Nowowiejska* is a street name and can be replaced by another name. The same is with the number that follows the street name. In order to catch the characteristic of the non-keywords the *grammatical category* is used.

*The bases of the proceeding prepositions* and *nouns* are used to catch the role of the annotation. The nouns are good determinants of the event type that the text fragment treats about. In turn the preposition determines the type of the information that follows it.

Parameters  $o$ ,  $i$ ,  $p$  and  $n$  determine final number of features used to represent single instance boundary.

Table 4.2 presents value of features for example instance. Parameters were set as follows  $a=3$ ,  $i=2$ ,  $p=1$  and  $n=1$ .

*Zarząd Prokom S.A. zwołuje na dzień <date>20 marca 2007 roku</date> Walne Zgromadzenie.*

*Prokom S.A. board of management announced the Annual Meeting for Stockholders at <date>20th March 2007</date>.*

Feature description	Starting boundary	Ending boundary
base of 1st outer token	dzień	walny
base of 2nd outer token	na	zgromadzenie
base of 3rd outer token	zwoływać	.
grammatical category of 1st outer token	subst:sg:nom:m3	adj:sg:nom:n:pos
grammatical category of 2nd outer token	prep:acc	subst:sg:nom:n
grammatical category of 3rd outer token	in:sg:ter:imperf	interp
type of 1st outer token	LowerCase	LowerCase
type of 2nd outer token	LowerCase	LowerCase
type of 3rd outer token	LowerCase	Symbol
base of 1st inner token	20	rok
base of 2nd inner token	marzec	2007
grammatical category of 1st inner token	ign	subst:sg:gen:m3
grammatical category of 2nd inner token	subst:sg:gen:m3	ign
type of 1st inner token	Number	LowerCase
type of 2nd inner token	LowerCase	Number
base of 1st preceding preposition	na	na
base of 1st preceding noun	zwoływać	zwoływać

Table 4.2: Instance features

## Chapter 5

# Methods of Patterns Acquisition

When first Information Extraction systems were created the highest priority had their accuracy in extracting relevant information. The rules to identify relevant information were created by hand by linguistics engineers supported by domain experts. This was very laborious process but eventually in many cases lead to satisfactory results [7], [20].

The approach has changed when the Information Extraction systems with manually created patters achieved relatively high performance. Best reported performance of IE system with hand-coded rules was F-measure of 98.50 for person, 96.96 for place, and 92.48 for entity in MUC-6 task [23]. Researchers started thinking about automating this process in order to shorten the time that was required to create the extraction patterns. The need to automate pattern creation was obvious and simple. An architecture of IE system and major components (Tokenization, Morphological and Lexical Analysis, Syntactic Analysis and Semantic Analysis) were general and once created could be used in every Information Extraction system. However, Domain Analysis in which extraction patterns are widely used is more complex. Every domain and task requires separate set of patterns and rules to identify relevant information. It is natural that extraction patterns created in order to identify acts of terrorism in articles cannot be used to extract information about diseases from medical reports. This kind of resources could not be shared by IE systems that work in different domains and every task required individual care and creation of specific extraction patterns what was time-consuming. This is why automation in pattern creation is much desired.

First experiments with Machine Learning techniques showed very promising results. Despite the fact that the manual approach gives better results in information extraction task, automated methods are still in the center of interests and many works are done on how to improve those methods.

In the first section of this chapter the process of manual patterns creation is presented with description of *top-down* and *bottom-up* approaches. Then the idea of automatic approach to this problem is presented and compared with manual approach. Then the following sections contain overview of state-of-the-art works and research that are related to automatic process of pattern extraction. Those sections present used methods and techniques from Machine Learning (*Decision Diagrams*, *Memory-Based Learning* that is also knows as *Instance-Bases Learning*, *Neural Networks*) that are used to teach the system how to identify relevant information. Also some techniques for automate evaluation of created patterns' accuracy (*Bootstrapping*) and interactive methods (*Active Learning*) are presented.

## 5.1 Manual pattern creation

The process of the manual patterns creation starts from gathering a set of task representative documents. Then the linguistic engineers supported by domain expert construct a set of basic patterns or rules that identify the relevant information. The form of the patterns or rules can be one of those presented in Chapter 4 or very similar to them. The initial set of patterns or rules is tested on gathered documents in terms of precision and recall. Then the results are verified by domain experts and linguistic engineers modify the set of patterns if it is required. Then the modified patterns are tested and so on. The process is repeated until satisfied level of precision and recall is achieved.

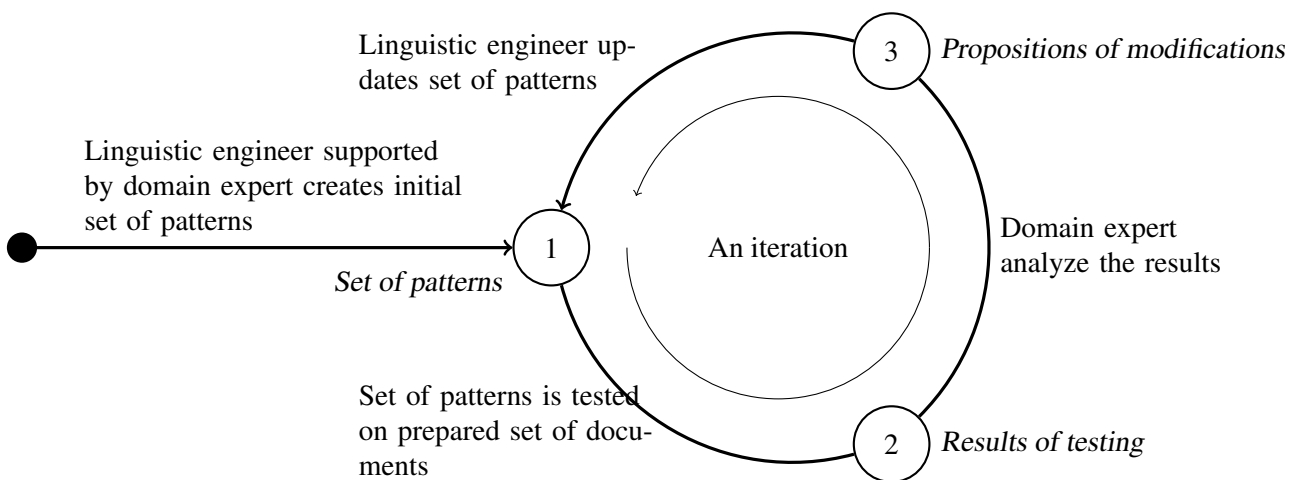


Figure 5.1: A process of manual patterns creation.

There are two main strategies used in following iteration in patterns specification:

**top-down** — initial set of patterns or rules is very general and match all known examples. In following iterations the set of patterns or rules become more specific in order not to match negative examples. In this strategy we start from high recall and low precision, and try to increase the precision as high as possible but also not to decrease to much the recall. Figure 5.2 presents how the precision and recall may change in following iteration using this strategy.

**bottom-up** — initial set is very small and contains very specific patterns or rules that match only most common examples. In following iterations the set is extended by new patterns or rules, or existing one are generalized in order to match uncommon and rare examples. In this strategy we start from high precision and low recall, and try to increase the recall as high as possible but also not to decrease to much the precision. Figure 5.3 presents how the precision and the recall may change in the following iteration using this strategy.

There is always a problem if a pattern or rule should be more general or more specific. On the one hand specific rules match most positive examples and doesn't match negative but on the other hand they ignore positive examples that are a bit different. Similar situation is with general rules that can "predict" similar examples that were not taken into consideration before but also can match irrelevant information. There is always need to make tradeoff between rules precision and recall what can be presented by vertical, dotted line on Figures 5.2 and 5.3.

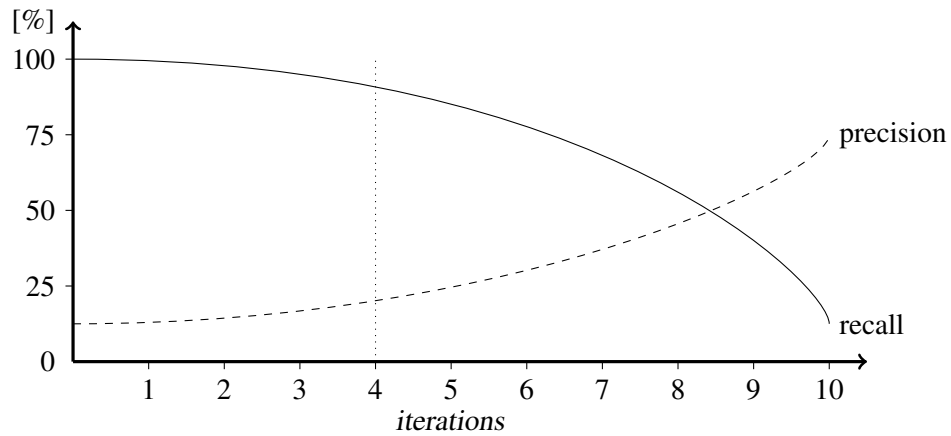


Figure 5.2: An example how the precision and recall may change in following iteration using top-down strategy

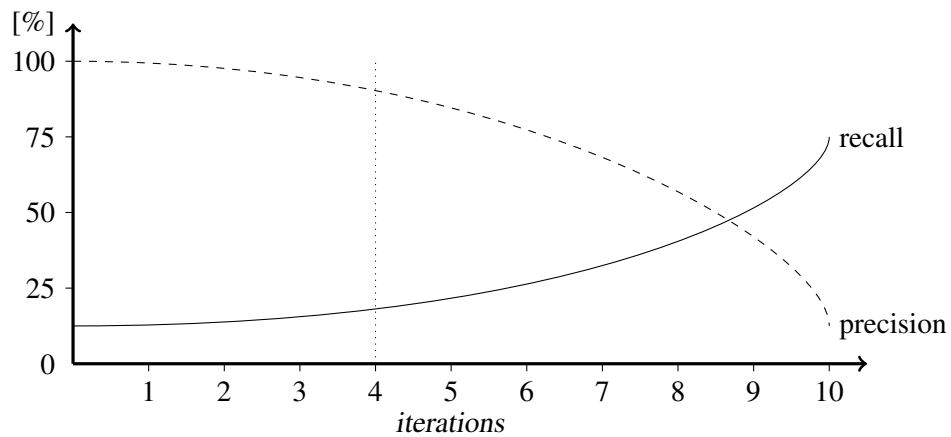


Figure 5.3: An example how the precision and recall may change in following iteration using bottom-up strategy

#### Advantages of manual approach:

- requires only small set of representative documents,
- the set of documents doesn't have to be annotated (however, annotations can help in patterns evaluation),
- natural and intuitive for human representation of patterns and rules,
- so far this approach gives better results than automatic approach.

#### Disadvantages of manual approach:

- requires domain experts that should be involved in the process,
- very time-consuming and laborious,
- the more patterns are required the more difficult is to make changes.

## 5.2 Automatic pattern creation

Automatic pattern creation, like manual approach, also starts from gathering a set of task representative documents. In the next step all documents have to be manually annotated. The annotations should be created in a way that will allow the system to process them automatically. It is a good practice to annotate the documents by group of independent people and compare the results in order to avoid errors that can be made by one person. It is essential to have error-free training set.

The learning process starts from creating a set of training instances. A training instance is the representation of the training example in terms of the features and their values. Each training example must be described by the same set of features. Then a Machine Learning method is applied to learn the patterns from the set of training instances. According to Daelemans and Hoste work [14] there is no one Machine Learning method that is considered to be the best one for Semantic Labeling task. Different methods have been applied in the Semantic Labeling, for examples Bennett et al. applied C4.5 decision trees to create classifiers that predict starting and ending boundary of an annotations. In turn Gallippi [19] presents different approach in which Semantic Labeling task is broken down into *delimitation* and *classification*. The *delimitation* is done by using a set of POS based hand-coded templates. Then the ID3 decision trees are learn to recognize proper annotations. Leek in his master thesis [27] makes use of Hidden Markov Models to extract genes name and location from medical documents. The annotations and their context are described in terms of hidden states and transitions between them. Vilain and Day [46] present a finite-state automates based approach in which the patterns are created automatically by searching the space of all possible patterns and minimizing the estimated error. Bosch et al. [45] applied Memory Based Learning (also known as Instance Based Learning) supported by features selection and parameter optimization strategies.

As was mentioned before no one method is considered to be the best one for Semantic Labeling task. The performance of different approaches cannot be compared because the testing environments were different in each case. There is a need to perform such experiment that will investigate the performance of different Semantic Labeling methods and will be conducted in the same testing environment what means that the same set of training and testing data will be used.

Another important fact that should be taken into consideration while conducting an experiment is parameters confirmation and feature selection. The set of available parameters depends on the Machine Learning method and the same configuration for different types of annotations may report different performance in terms of precision and recall. The same is with set of features that are used to represent training examples. Those two factors caused the need to try several configurations of parameters and features before evaluating the method.

There are several strategies used in the learning process:

**One Try** — the learning and matching parameters are set manually and the system is started to learn the patterns. This strategy is applied for example in AutoSlog [40] developed by E. Riloff.

**Patterns Bootstrapping** — the set of patterns is created in an iterative manner. The general idea is following: in every iteration set of patterns is created. Then each pattern is evaluated in terms of precision and recall in order to determine the best ones. Only best patterns are chosen and added to the final set of patterns and the process is repeated. This strategy is presented in [26], [13], [43], [25].

**Parameters Optimization** - the initial values of the learning and matching parameters are set. Then, the system evaluates set of different parameters configuration in order to choose the most optimum. The parameters for which best results were achieved are used as final system configuration. This strategy is described by Antal van den Bosch et al. in [45].

**Features Selection** — the tests are performed with different set of features and the set for which best results are reported is selected. Bosch et al. used this strategy in their work [45].

Following sections of this chapter presents works that treat about automatic pattern creation. The goal of the revision is to presents the variety of methods and techniques that can be apply to solve the problem of Semantic Labeling.

### 5.3 Decision Trees C4.5

Bennett, Aone and Lovell created a multilingual information extraction system (called RoboTag) that uses decision trees C4.5 to learn extraction patterns [8]. The task of Semantic Labeling is broken down into two classification problems and two decision trees are created. One to recognize where the annotation starts and the other where the annotation finishes.

The training examples are represented by the set of following features:

**Token Type:** characteristic of the token like upper case, lower case, capital and so on,

**POS:** part of speech like verb, noun, adverb and so on,

**Location:** semantic information related to location like city, province, country and so on,

**Semantic Type:** other semantic information connected to the token like person, name, title and so on.

The set of parameters that can have impact on the results is following:

**Radius:** number that determines how many proceeding and following tokens are used to create learning tuple. Radius of 1 means that only one proceeding and one following token is taken into consideration. The higher is the value the more contextual is the tuple.

**Sampling Ratio:** determines the ratio of negative examples to positive examples. These parameters are used to limit number of negative examples learned by decision trees. Remembering all negative examples may lead to decrease the accuracy by making the decision trees too conservative.

**Certainty Factor:** number between 0 and 1 that determines decision trees pruning. Lower value means more pruning.

In the experiment RoboTag was trained using 300 instances from the MUC-6 Wall Street Journal corpus to tag 30 documents. The best results were obtained with a sampling ratio of 10, a radius of 2 and certainty factor of 0.75 for pruning and are shown in the Table 5.1.

Annotation Type	Training Instances	Testing Instances	Recall	Precision	F-measure
Person	1978	372	93.5	95.9	94.7
Place	2495	110	95.5	89.7	92.5
Entity	3551	448	79.7	83.4	81.5
Total	8024	930	87.1	89.2	88.1

Table 5.1: The RoboTag best results.

## 5.4 Memory-Based Learning

The idea of Memory-Based Learning (MBL) is to store all training examples as they are in the memory [11]. No further generalization or reduction of the learned data is performed and no information is lost. When new instance is labeled the most *similar* instances are retrieved from the training set and they are used to determine the label of the new instance. Following subsections contain description of two algorithms (IB1 and IGTREE) that are used to determine the label of test instance.

### 5.4.1 The IB1 algorithm

The idea of IB1 [11] algorithm is similar to k-Nearest Neighbors (k-NN) algorithm. According to k-NN, each instance is perceived as a point in n-dimensional space where n is the number of features that describe the instance. When a new instance is classified the algorithm looks for the k similar instances (Nearest Neighbors) in the set of training examples. The similarity is based on the distance in the k-dimensional space between instances. If all instance's features are numerical then the distance can be computed as a Euclidean distance (5.1) where  $x_i$  and  $y_i$  refers to value of i-th feature of x and y instance.

$$\Delta(X, Y) = \sqrt{\sum_{i=1}^n (x_i - y_i)^2} \quad (5.1)$$

In NLP domain most of the features are symbolic what requires some modification in k-NN algorithm. In IB1 algorithm the distance between instances is computed as a number of features that have the same value (or number of features that differ). The distance (similarity) can be computed using equation 5.2 and 5.3.

$$\Delta(X, Y) = \sum_{i=1}^n \delta(x_i, y_i) \quad (5.2)$$

$$\delta(x_i, y_i) = \begin{cases} 1 & \text{if } x_i = y_i \\ 0 & \text{otherwise} \end{cases} \quad (5.3)$$

In the k-NN and IB1 algorithms  $k$  refers to the number of Nearest Neighbors that are taken into consideration while determining class (label) of test instance. The test instance has been assigned the label that is most frequent in the k-NN set. In case of tie (two or more labels are equally frequent) the label that is more frequent in the training set is chosen. If this doesn't solve the tie then the label that is first seen in the training material is chosen.

### The IB1-IG algorithm

It is common that some features are more important than others it is intuitive to use weights while computing the distance. The IB1-IG algorithm is a modified version of the IB1 algorithm in which the distance is computed with respect to features' weights. The distance is computed using equation 5.4 where  $w_i$  represent the weight of  $i$ -th feature.

$$\Delta(X, Y) = \sum_{i=1}^n w_i \times \delta(x_i, y_i) \quad (5.4)$$

In the TiMBL software package 4 different algorithms are implemented to determine features weights:

- Information Gain (IG)
- Gain Ratio (GR)
- Chi-squared weights ( $\chi^2$ )
- Shared Variance (SV)

The concept of those algorithms is briefly described in [11].

### Modified Value Difference Metric

So far when comparing two values of symbolic features the result could be either 0 (when equal) or 1 (when different). This approach is simple but doesn't reflect the real similarity between symbolic features. For instance word *Monday* is more similar to word *Friday* than to word *car* because *Monday* and *Friday* belong to the same class that represent days of week. To capture this relation between symbolic features Stanfill with Waltz and Cost with Saltzberg developed Modified Value Difference Metric (MDVM). In MDVM the distance (similarity) between two symbolic values is computed by comparing the associated class distribution. The distance in MDVM is defined in equation 5.5 where  $v_1$  and  $v_2$  are values of symbolic feature.

$$\Delta(v_1, v_2) = \sum_{c \in U} |P(c|v_1) - P(c|v_2)| \quad (5.5)$$

### 5.4.2 The IGTREE algorithm

The idea of the IB1 algorithm is very simple. The training phase is relatively fast, however testing phase can be very slow what is a big disadvantage of the algorithm. Testing (finding k Nearest Neighbors) requires computation of the distance between testing instance and each instance in the memory what makes the computation time dependent on the number of features and number of learned examples. Therefore, to achieve better performance in testing, Daelemans, Van den Bosch and Weijters developed IGTREE algorithm that is a fast approximation of IB1-IG. All learned examples are stored in an oblivious decision tree. In oblivious trees every level contains test on some feature (unlike to e.g. C4.5 in which at the same level in different branches different features can be tested). Features in the tree are ordered according to their weights, with highest-weighted feature on the top. IB1-IG algorithm takes into account relative difference between features' weights. In IGTREE the relative difference between features is lost and only determines on which levels the features are tested. Every node represents a class that is most frequent among set of training instances that match the path from root to that node. IGTREE has one more feature that IB1-IG doesn't have. IGTREE allows pruning the training instances by cutting of some subtrees. For instance, subtree in which all nodes represent the same class can be replaced by one node representing the same class. This allows minimizing the size of the tree.

Figure 5.5 present an example of IGTREE structure. This example is based on an example from [11]. The task is to determine words' functions in the sentence. Figure 5.4 presents 3 training examples and 7 rules generated from those examples. Each learned instance is represented by 5 features (columns 1-5):

- 1 - relative position of the word to the verb,
- 2 - the verb in the sentence,
- 3 - preposition that appears before the word,

- 4 - the word that is the subject of testing,
- 5 - phrase.

Each instance is classified as one of the following classes:

- NP-SBJ - Noun Phrase-Subject,
- NP-OBJ - Noun Phrase-Objects,
- PP-TMP - Prepositional Phrase-Temporal.

On the Figure 5.5 on the left side the number inform with feature is tested on that level in the tree. Each path from node to the leaf represents one training instance. Some subtrees contain edges with the same class. In pruning process those subtrees are removed. Removed branches are presented on the figure as dotted lines. When new instance is tested their features are compared with features in the tree in given order starting from the root and is continue until all features are matched or any of them is mismatched. The class from node in which testing process was finished has been assigned to the instance.

1	2	3	4	5	class	example
-1	visited	-	Mike	NP	NP-SBJ	Mike visited John on Monday.
1	visited	-	John	NP	NP-OBJ	
2	visited	on	Monday	PNP	PP-TMP	
-1	arrived	-	John	NP	NP-SBJ	John arrived on Sunday.
1	arrived	on	Sunday	PNP	PP-TMO	
-1	left	-	Mike	NP	NP-SBJ	Mike left before midnight.
1	left	before	midnight	PNP	PP-TMP	

Figure 5.4: Examples of training instances for words function classification task and their symbolic representation. Example is based on [11].

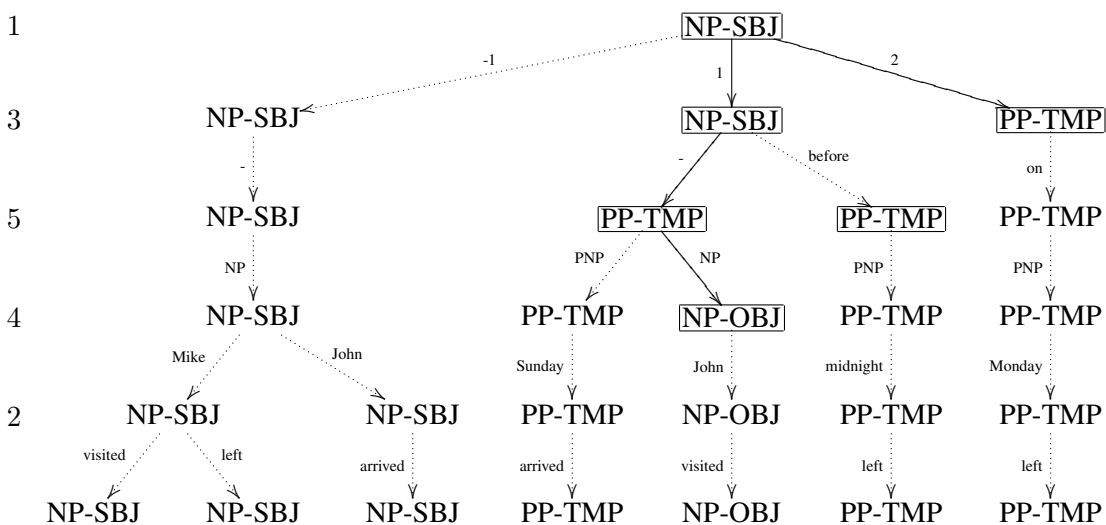


Figure 5.5: Example of IGTree that contains training instances from Figure 5.4

### 5.4.3 Memory-Based Learning in Semantic Labeling

Antal van den Bosch et al. [45] applied Memory-Based Learning in Semantic Labeling task supported by parameters optimization and feature selection. The results of the experiments confirm that the parameters and features configuration has impact on the classification results.

The training instances contained following features:

- a verb,
- distance to the verb measured in chunks,
- Noun Phrase (NP) chunks,
- Verb Phrase (VP) chunks,
- attenuated words,
- Part of Speech (PoS) tags,
- a chunk tag,
- passive main verb,
- current clause,
- role pattern.

As a method for feature selection *bi-directional hill-climbing* was used. This method starts from empty set of features. Then all combination of chosen (this set is empty in the initial step) and available features are tested and the best combination is chosen and used in the following iteration. The process is repeated until no better results are achieved.

Parameter optimization was done using a heuristic method called *iterative deepening*. The method starts from a set of experiments with different parameters configuration. Each parameter configuration is learned using a small set of training data and tested on another set of instances. Only best configurations are kept and the others are removed from the pool of experiments. In the following iterations training and testing sets are expanded and the process is repeated until only one configuration is left or all training instances are used.

Combination of *feature selection* and *parameter optimization* were used in order to determine the best configuration. Table 5.2 presents the results on the following steps of the optimization process. The 51.8% of F-measure for initial set of features and parameters values was improved to 60.9%. This experiment depicts the fact that *parameter optimization* and *feature selection* have impact on the performance.

## 5.5 Bootstrapping

In Computer Science bootstrapping is known as an iterative approach to developing systems and computer software. It is a process that starts from developing a simple version of the system or tool. Then the simple tool or system is used to develop more complex tool or system that serves the same purposes. This approach is used, for example, while developing compilers for computer languages. At the beginning simple compiler for a new computer language is written and compiled using existing language. Then another version of the compiler is written, but this time in the new language and is compiled in the earlier developed compiler. Now we have simple compiler of the new language written in that language. Steps of writing new compiler and compiling the codes in

No.	Precision	Recall	F-measure	Method
1	51.6	51.9	51.8	feature selection
2	57.3	52.7	54.9	parameter optimization
3	58.8	54.2	56.4	feature selection
4	59.5	53.9	56.5	parameter optimization
5	64.3	54.2	58.8	classifier stacking
6	66.3	56.3	60.9	parameter optimization
7	66.5	56.3	60.9	feature selection

Table 5.2: Effect of feature selection and parameter optimization using Memory-Based Learning

previous version are repeated until we get final version of the compiler. The latest version of the compiler is written in the new language and compiled by itself.

Bootstrapping approach is also used in Information Extraction domain. It is used to automate creation of linguistic resources (for example lexicon of geographic names [26]), classification (text categorization [13], subjective tenses classification [43]), word sense disambiguation [25] and many other purposes.

The key idea of bootstrapping is:

1. In the initial step some training examples from whole set of documents are prepared manually. Those hand-annotated documents are commonly called seeds.
2. The examples are used to create rules (depending on the task it might be rules for information extraction, word sense disambiguation or others).
3. The generated rules are applied to other documents from the set.
4. The results are verified by human and right examples from the result are added to the initial set of examples.
5. The rules are generated once again from the modified set of examples and verified.
6. The steps of rules creation and verification are repeated in a loop until exit condition is satisfied. The exit condition may be for example the number of iterations or number of annotated documents is achieved. In every iteration the set of annotated documents is expanded by new examples what cause those new rules may be created from the training set.

Figure 5.6 depicts key steps in the process of bootstrapping. On the figure ovals symbolize activities and rectangles symbolize artifacts created during the activities. Solid lines show main flow of the process and dotted lines presented modifications of set of unannotated documents that occurs during the process.

### 5.5.1 Example of Bootstrapping

In [26] Lee and Lee present their bootstrapping approach to annotating geographic entity names. In the initial step they use entries from gazetteer as a seed to annotate geographic names in raw documents. Authors point out that the gazetteer may contain many ambiguous entities that should not be used as seeds (for example *U.S.* in *U.S. Navy* should not be annotated as COUNTRY because it is part of organization) in order to avoid noises in the annotation in the first phases of bootstrapping.

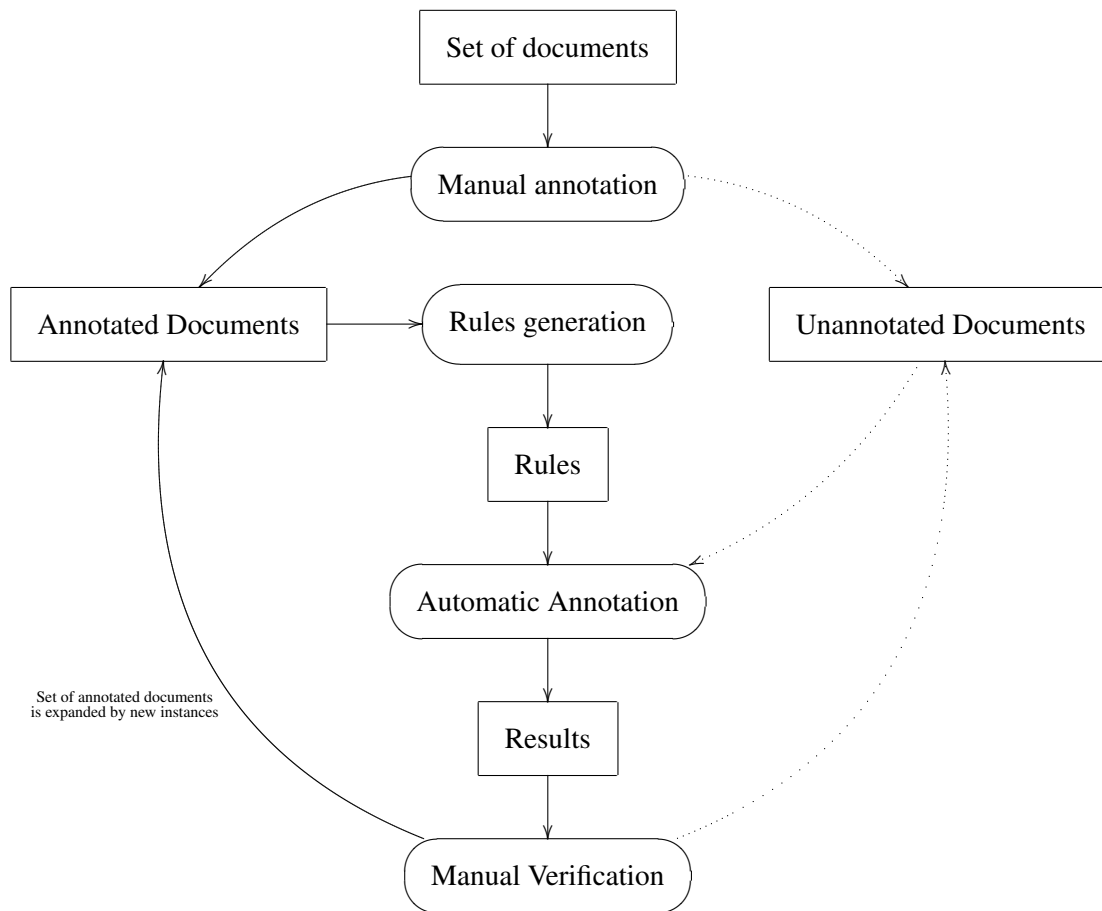


Figure 5.6: The idea of bootstrapping

In the next step boundary patterns are generated from the annotated documents. Each annotation is described by pair of starting ( $bp_s$ ) and ending ( $bp_e$ ) boundary pattern. The patterns are defined as follow:

$$bp_s = [f(w_{-2})f(w_{-1})f(w_{-0})]$$

where:  $f(w_{-0})$  is the first token of the annotation and  $f(w_{-1})$ ,  $f(w_{-2})$  are preceding tokens,

$$bp_e = [f(w_{+0})f(w_{+1})f(w_{+2})]$$

where:  $f(w_{+0})$  is the last token of the annotation and  $f(w_{+1})$ ,  $f(w_{+2})$  are subsequent tokens.

Example of boundary patterns from [26]:

(...)in the cities of <CITY>Los Angeles</CITY>, San Diego and (...)

Will be generated following patterns:

opening boundary patterns	closing boundary patterns
[%cities %of %los]	%angeles %, %san
[%cities %of &initcap]	&initcap %, %san
[@city1 %of &initcap]	&initcap %, \$city_gaz
[@city1 %of \$city_gaz]	\$city_gaz %, \$city_gaz

Then each pattern is scored using following measures:

- $pos(bp)$  - number of matched entities that are already annotated as the same entity type,
- $neg(bp)$  - number of matched entities that are already annotated as another entity type,
- $unk(bp)$  - number of matched entities that have not been annotated yet.

The boundary score  $Score(bp)$  is computer as follows:

$$Score(bp) = \frac{pos(bp)}{pos(bp) + neg(bp) + unk(bp)}$$

The score is used in the next step to select only promising candidates. There are two thresholds: bottom  $\theta_b$  and top  $\theta_t$ . The new entity is select only when at least starting boundary score or ending boundary score is higher than  $\theta_t$  and both boundary patterns scores are higher than  $\theta_b$ , what can be written as following condition:

$$[(Score(bp_s) > \theta_t) \vee (Score(bp_e) > \theta_t)] \wedge (Score(bp_s) > \theta_b) \wedge (Score(bp_e) > \theta_b)$$

In each iteration the thresholds values are controlled. The bootstrapping process start with values  $(\theta_b, \theta_t) = (0.90, 0.90)$ . In the subsequent iterations value of the bottom threshold id decreased to  $(0.90, 0.15)$  and then value of top threshold is decreased. The bootstrapping is finished when the thresholds arrive at  $(0.51, 0.15)$ . The thresholds values are decreased in each iteration in order to extract new entities.

### 5.5.2 Mutual Bootstrapping and Meta Bootstrapping

In paper [42] Riloff and Jones present an approach to automate creation of words lexicon. The approach is based on the idea of bootstrapping. Authors point out two important facts. One is that many IE system use both lexicons of known phrases and dictionaries of extraction patterns to recognize relevant noun phrases. The other one is that while creating extraction patterns, new patterns can extract new phrases that can later create new extraction patterns. Basing on those two facts Riloff and Jones presented approach in which both lexicon of phrases and dictionary of extraction patterns are created simultaneously. This approach is called Mutual Bootstrapping.

According to [42] at the initial steps of the process two artifacts are required: set of unannotate documents and list of relevant words, called seeds. Authors use AutoSlog ([40], [41]) to create extraction patterns from the documents for every noun phrase. Then the mutual bootstrapping procedure identifies best extraction pattern for known noun category. Later the pattern is used to extract new nouns. Created pattern is added to dictionary and extracted nouns to lexicon and the process is repeated.

Authors pointed out weak point of the Mutual Bootstrapping approach. The performance of this method can be rapidly decreased when a noise in the dictionary (noncategory words) occurs. In order to make the Mutual Bootstrapping more robust Riloff and Jones introduced second level of bootstrapping called Meta Bootstrapping. After performing inner bootstrapping (mutual) in the outer bootstrapping procedure 5 most reliable entries from the lexicon are identified and retained for the permanent semantic lexicon. Then the Mutual Bootstrapping process is restarted.

## 5.6 Conclusion

Many different methods for patterns acquisition have been developed over last years. However, there is no one answer for question: *Which method is the best/most suitable/most powerful for*

*concrete task/domain?* Daelemans and Hoste [14] formulated thesis that the variation in results between different supervised learning methods are much less than the variance in results for each method but with different set of parameters value. In other words appropriate parameters' configuration for any method has bigger influence on the performance than choosing concrete method.

## Chapter 6

# Experiment

### 6.1 Introduction

The practical part of this master thesis consists of three parts:

- preparation of the corpus used for testing,
- implementation of the system that will learn patterns and propagate annotations,
- testing the system with different set of features and testing parameters.

Following section of this chapter provide description of the components, techniques and methods used in the experiment conducted within the master thesis.

### 6.2 Preparation of the Training Set

The training set was created from documents regarding stock exchange domain. Each document contains a report posted by stock issuer. In Poland every unit that issue stocks is obligated by law to present current and periodical information about the issuer according to act [4]. The issuer is obligated to inform about 26 types of event defined by the act. Within the experiment three types of events were chosen and relevant documents were manually annotated. The appendix A contains brief description of the chosen events and their attributes.

All documents were gathered from the website that publishes and stores reports published by stock issuers. Each report is stored as XML document and beside report title and content contains other information that is not relevant in the experiment. At first for every document XSLT transformation was applied in order to extract the report title and content from the document. Then every document was tagged using TaKIPI.

TaKIPI is a morpho-syntactic tagger for Polish language [34]. It applies the algorithm of Induction of Decision Trees called C4.5r8. Tagger is based on both handmade rules and rules that were automatically acquired. The purpose of TaKIPI is to assign set of relevant morpho-syntactic tags for each token in a document and determine which of them is proper in given context.

The set of documents contains more than 11.000 documents that treat about different events from stock exchange domain. The first goal was to filter this set in order to separate documents about Annual Meeting of Stockholders from others. Two methods were used to achieve this goal:

- *Simple search by keywords* - keywords related to the Annual Meeting of Stockholders were search in raw files (Figure 6.2), for example *Walne Zgromadzenie* (The Annual Meeting), *odbędzie się* (will be held), *przebieg spotkania* (the meeting agenda) and so on.

- *Propagation of the annotations and synchronization between the events and the annotations* - this method was used in later iterations because it requires at least one annotation for given type, however, the more annotations, the higher chances to propagate them on other instances. Existing annotations were used to locate similar instances of the annotation type in other documents. Figure 6.2 presents an interface of the component to annotation propagation. *The propagation component* learned existing examples and search whole set of documents for similar instances. All found instances were verified by user that was deciding whether the annotation was correct or not. Then the set with correct annotations was added to existing ones and the process could be repeated. User could set the values of *starting* and *ending boundary threshold* that were responsible for matching generalization. *The processes of learning* and *matching* are described in sections 6.4 and 6.6, respectively. Propagated annotations were later synchronized with their events. Each *annotation type* has assigned an event type, for instance *Zgromadzenie data* (a meeting date), *Zgromadzenie godzina* (a meeting time), *Zgromadzenie miejsce* (a meeting place) have assigned the event called *Zgromadzenie* (a meeting), see Figure 6.2.

When a new instance was located in a new document then the document was manually examined in order to annotate other instances of given type. This assured the annotations completeness.

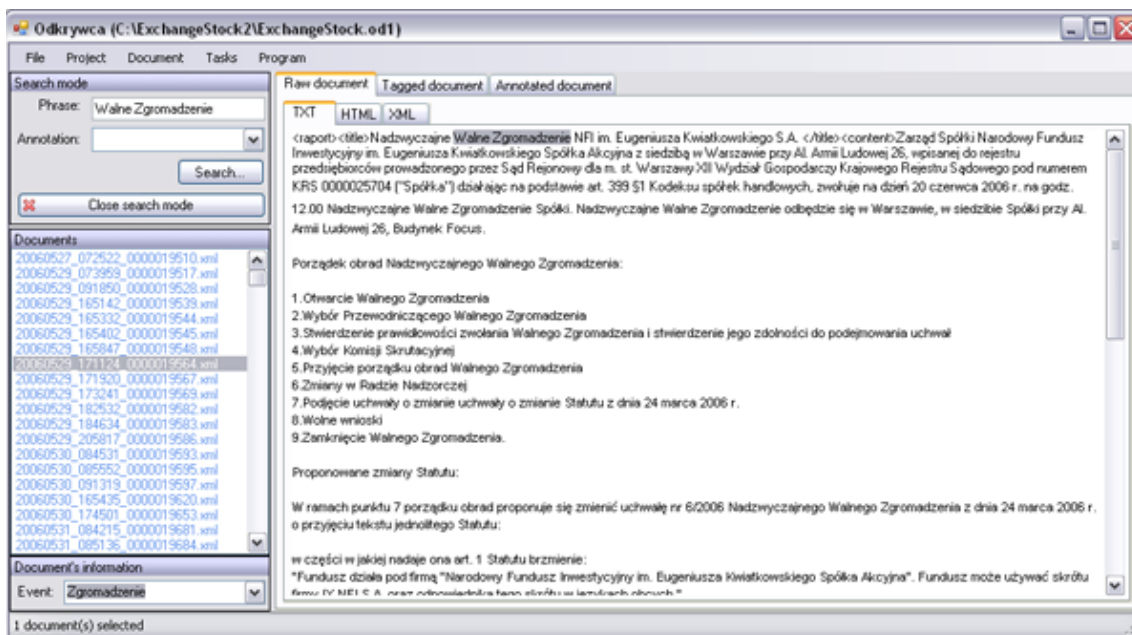
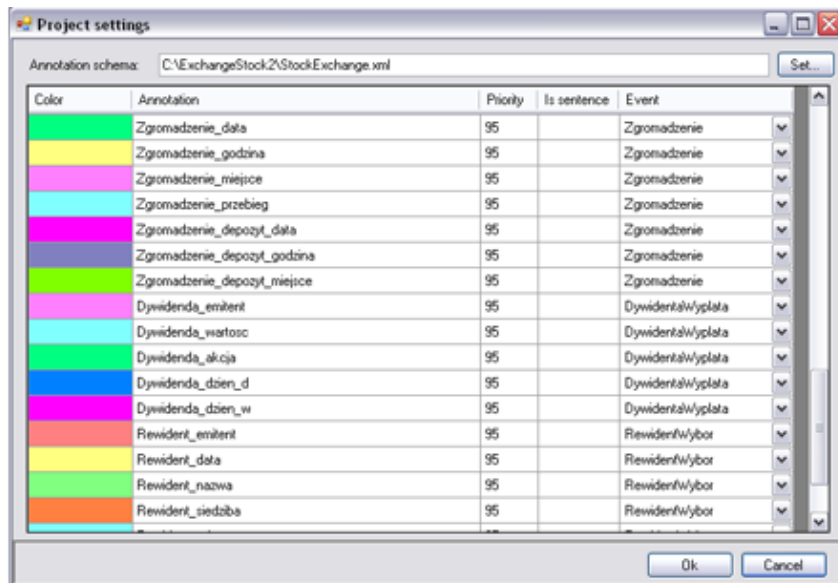
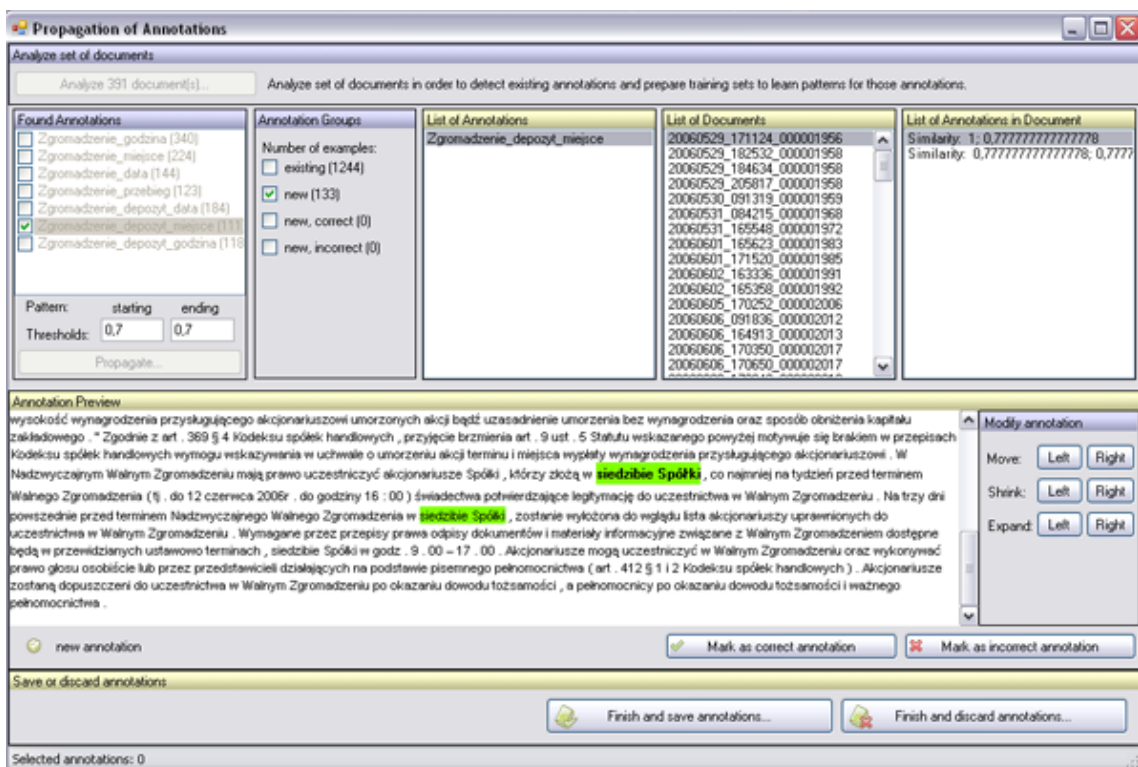


Figure 6.1: Documents filtering by a keywords searching (*Odkrywca*).

About 390 relevant documents were collected. Then the next goal was to analyze every single document and mark the relevant information. The *Annual Meeting of Stockholders* event contains 7 types of annotations: *a meeting date*, *a meeting time*, *a meeting place*, *a meeting agenda*, *a trust deposit date*, *a trust deposit time* and *a trust deposit place*. Figure 6.2 presents the graphical interface of the component that was used to create document annotations. The component takes as an input filename of tagged with TaKIPI file and as an output create xml file with created annotations.

Figure 6.2: The schema annotation (*Odkrywca*).Figure 6.3: The component used to annotate documents through propagation (*Odkrywca*).

The annotation component allows creating two types of annotations: sentence and document annotations. The difference between those types of annotations is in the annotation range. Sentence annotations must be assigned to a sequence of tokens within one sentence. Document annotations can be assigned to every sequence of tokens in a document. Low performance of the *sentencer* caused that all annotations were added as document annotations. The *sentencer* had problems with determining if given *full stop* represents the end of a sentence or is part of a date or hour. In all cases the date and hour were treated as two or more sentences what made impossible to mark them as one sentence annotation.

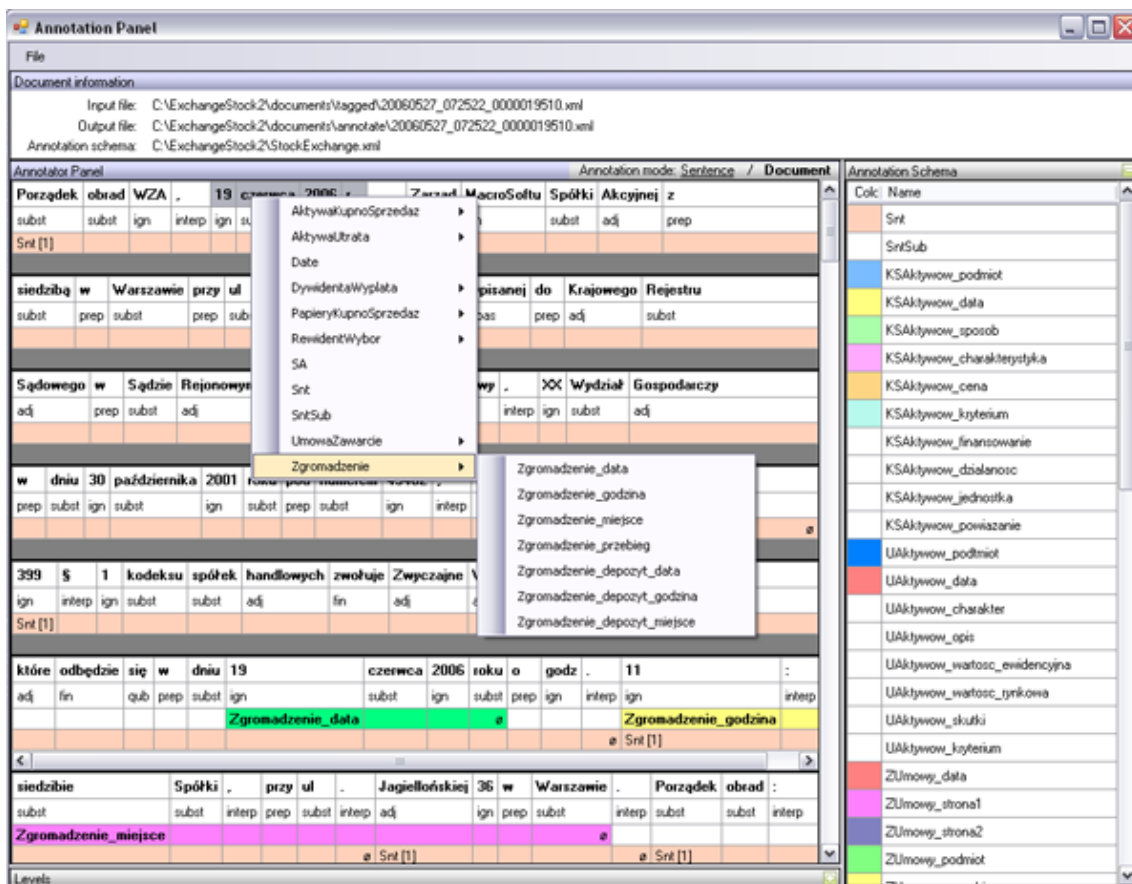


Figure 6.4: The component used to annotate documents (*Odkrywca*).

### 6.3 Baseline

As a baseline for the automatic approach I used manually created patterns. The patterns were constructed using regular expressions. I spent the same number of hours to create the regular expressions for each annotation type in order not to favour any of them. By spending the same number of hours for each annotation type the results will reflect the relative complexity of each annotation type.

The process of creating the regular expressions was performed in two steps. The goal of the first step was to write the *regular expression* that will match all instances of given type, like

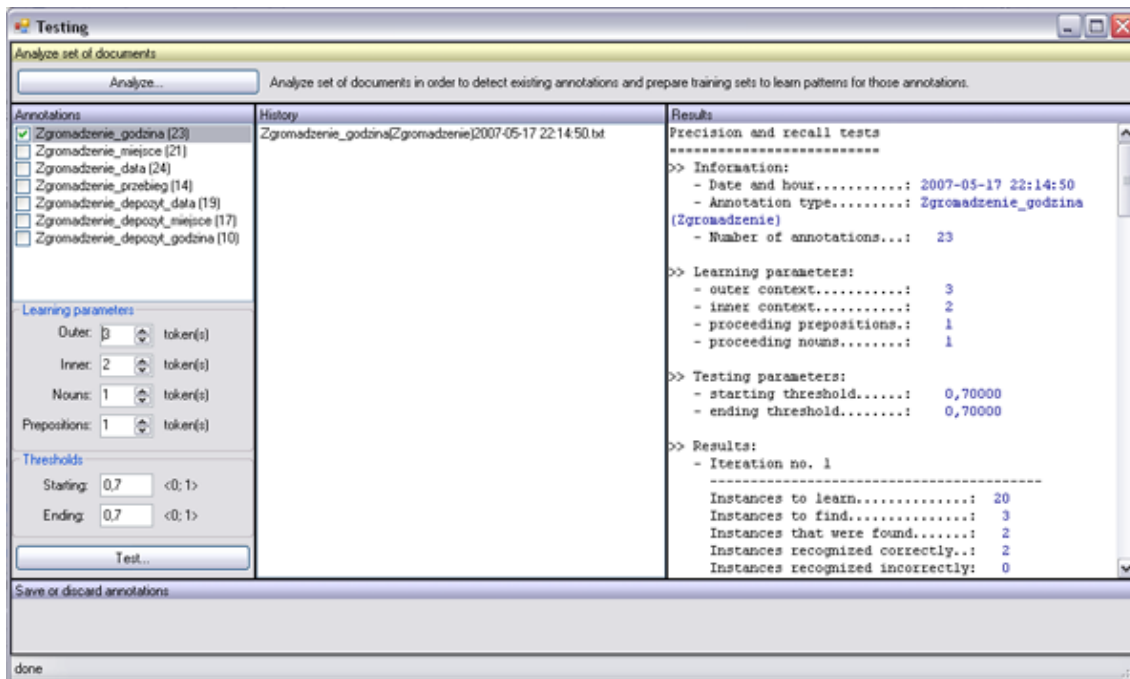


Figure 6.5: The component used to test pattern propagation (*Odkrywca*).

dates or time expressions. In other words the first step was concentrated on achieving the highest recall as possible with no respect to the precision. In the second step the regular expressions were extended by the description of the context in which the relevant annotation may appeared. This step was concentrated on improving the precision.

In order to support the process of creating the regular expressions special component was developed. The Figure 6.6 presents the interface of the component. The top part of the component contains the results of matching (the precision, the recall, the number of correct and incorrect matching for every pattern in the set). The bottom part the component presents list of not found, correctly and incorrectly matched instances.

### 6.3.1 Results for *Zgromadzenie godzina*

The regular expression for the meeting time was written as three expressions that cover different instances. The table below presents results for every single expression and also aggregate the results for all three expressions.

Regex	Precision	Recall	F-measure
1	93.38	92.16	92.77
2	100.00	29.76	45.87
3	100.00	10.70	19.34
all	93.74	97.91	95.78

The results for regular expressions that match the *Zgromadzenie godzina* annotations.



```
( (\s|\.) ( (o\s*(godz\.|godzinie)) | (na\s*(godzinę|godz\.)) ) \s* )
(?<annotation>\d{1,4} (\s:\d\d| ([\.,:\s])? (\d\d| Ā|oo))?) )

( (zwołuje|zwołaniu|zwołanego|zwołane|zwołanym|zwołania|zwołał)
\s\S*) {1,10} \s (godz\.|godzina) \s
(?<annotation>\d{1,4} (\s:\d\d| ([\.,:\s])? (\d\d| Ā|oo))?) )

(odbędzie\ssię) (\s\S*) {1,10} \s (o\s*godzinie|godzina) \s
(?<annotation>\d{1,4} (\s:\d\d| ([\.,:\s])? (\d\d| Ā|oo))?) )
```

Figure 6.7: The regular expressions that match the *Zgromadzenie godzina* annotations.

```
(na\s*dzień|w\s*dniu|odbędzie\s*(się)?|w\s*dniu
|odbędzie\s*sie\s*dnia|zwołanego\s*na|rozpoczną\s*sie) \s*
(?<annotation>\d{1,2} (\s*|\.|/)) (\w*|\d{2}) (\s*|\.|/)\d{4} (\sroku|\sr)? )
```

Figure 6.8: The regular expression that match the *Zgromadzenie data* annotations.

the best regular expression for which the highest recall was obtained. Other regular expression achieved relatively low results, each of them matched only couple examples.

The low result for this type of annotation is caused by the annotation structure variety. The location is a combination of a street name, a building name and number, a city and a postal code. The parts of the location appeared in different order and combination what made it difficult to write general regular expression. The solution for this problem could be in changing the level of details. Instead of representing whole address as a one annotation, decompose it into smaller parts, like street name, building and apartment number and so on.

```
(w|we) \s* (?<annotation>(siedzibie\s*[sS]półki\s*((,\s*)?w\s*))?)? \w*
(,\s*przy) \s* (?<street>ul\.\s*\w*\s*(nr\s*)? (?<number>\d*(/\d*))?) )
```

Figure 6.9: The regular expressions that match the *Zgromadzenie miejsce* annotations.

Regex	Precision	Recall	F-measure
1	17.91	16.10	16.96

The results for regular expressions that match the *Zgromadzenie miejsce* annotations.

## 6.4 Initial Experiment

Because of the numerous number of Machine Learning methods and limited time for the thesis I performed initial experiment in which I chose three most popular Machine Learning methods to create a classifier for starting boundary of the meeting date annotation. I chose the meeting date annotation in the initial experiment because referring to the baseline this type of annotation has average structure complexity.

I decide to test following Machine Learning methods:

- Decision Trees (C4.5) - this method has been already applied to Named Entity Recognition for English language by Bennett et al. [8],

- Memory Based Learning (Lazy Learning) - this method has been already applied to Named Entity Recognition for English and German language by Meulder and Daelemans [30] as well to other Information Extraction tasks (Semantic Labelling [45], Grammatical Relation Finding [11] Part of Speech Tagging [15]),
- Naive Bayes - this method has been already applied to Named Entity Recognition for Chinese language by Li et al. [49].

In the initial experiment I used the WEKA software that provides an environment to test Machine Learning methods and contains the implementation of those three methods [5].

The training set used in the experiment contained 302003 instances (424 positives and 301579 negative instances). The training instances represent the beginning boundary of the meeting time annotation. The testing set was encoded into WEKA arff file format. Test for each method was performed using default parameters.

	Precision	Recall	F-measure
C4.5	90,50	84,00	87,13
MBL (IB1)	88,70	91,00	89,84
NB	58,80	97,20	73,27

Table 6.1: Comparison of three ML methods

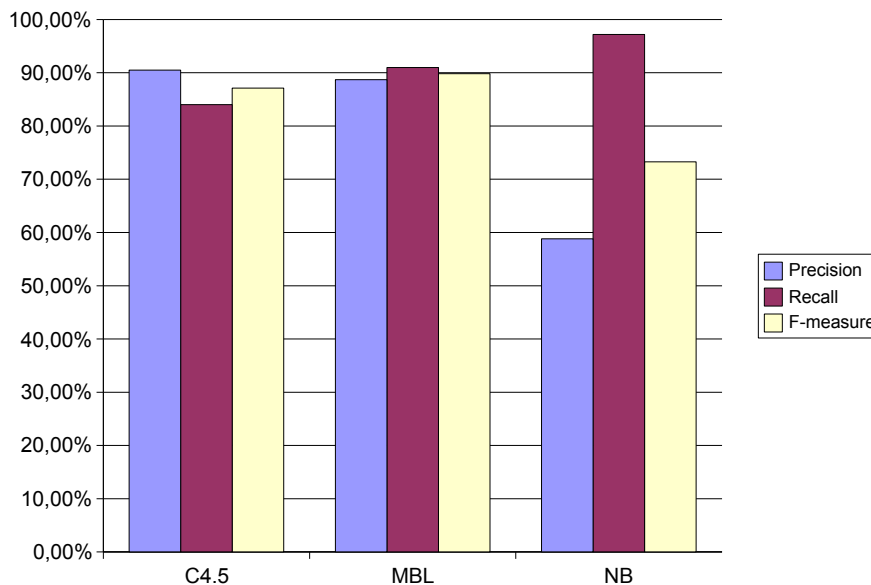


Figure 6.10: Comparison of three ML methods

The results of the initial experiments are presented in the Table 6.10 and at the Figure 6.10. The best precision of 90,50% was obtained for the Decision Trees and the recall of 97,20% for the Naive Bayes classifier. However, the best F-measure value of 89,84% was obtained for the Memory Based Learning. The Decision Trees obtain subtly lower result of 87,13%. What more, the execution time for the Memory Based Learning was extremely higher then time needed to perform 10-fold Cross Validation for the Decision Trees. The execution time of the 10-fold

Cross Validation for Decision Trees was less than 15 minutes and for the Memory Based Learning more than 24 hours. This is caused by the fact that Memory Based Learning relies on storing all training instances in the memory. However, most of the training instances (99,85%) represented the negative examples. This brings a two question:

1. Can be the same accuracy obtained using the Memory Based Learning method with lower number of negative instances?
2. Can be the performance of the Decision Trees improved by parameters optimization or in another way?

Because of the time limitation I decide to investigate only one case. I chose the MBL because it achieved the highest performance and the performance factor is much more important the the execution time. The second case will be included in the future work.

Finally, in the following experiments I was trying to find the answer for the following question:

Can be the same accuracy obtained using the Memory Based Learning method with lower number of negative instances?

## 6.5 Experiment with Memory Based Learning

In order to answer the question stated after the initial experiment I have conducted second experiment in which I analyzed the impact of the number of negative instances on the performance for the Memory Based Learning method. The goal of this experiment was to investigate if the execution time could be shorten by reducing the number of training instances but keeping the same performance.

The experiment contained 4 test cases. In all testes the 10-fold Cross Validation was used and average vales of precision, recall and F-measure were noticed. In the first run the balanced number of positives and negatives instances was used. For each positive instance one randomly taken negative instance was used. In the following runs 1%, 10% and 100% of all negative instances were used. The exact number of negative instances in each test case is presented in the Table 6.2.

	Meeting date		
	positives	negatives	total
Balanced	424	424	848
1%	424	3015	3439
10%	424	30157	30581
100%	424	301579	302003

Table 6.2: Number of positive and negative instances for the meeting date annotation

The second experiment showed that using 10% of all negative instances the execution time is shoren about 10 times and usind 1% about 100 times. However, the lower number of negative instances has also negative imapct on the performance what can be observed at the Figure 6.11. The execution time for the second test case (1% of the negative instances) was similar to the Decision Trees execution time but the performance is much lower.

The reduction of negative examples shorten the execution time as it was expected but also lowered the performance what was unacceptable.

After this experiment I have stated following question:

	Precision	Recall	F-measure
Balanced	0,50	100,00	1,00
1%	15,32	98,53	26,51
10%	62,42	94,48	75,17
100%	88,70	91,00	89,84

Table 6.3: The impact of the negative instances

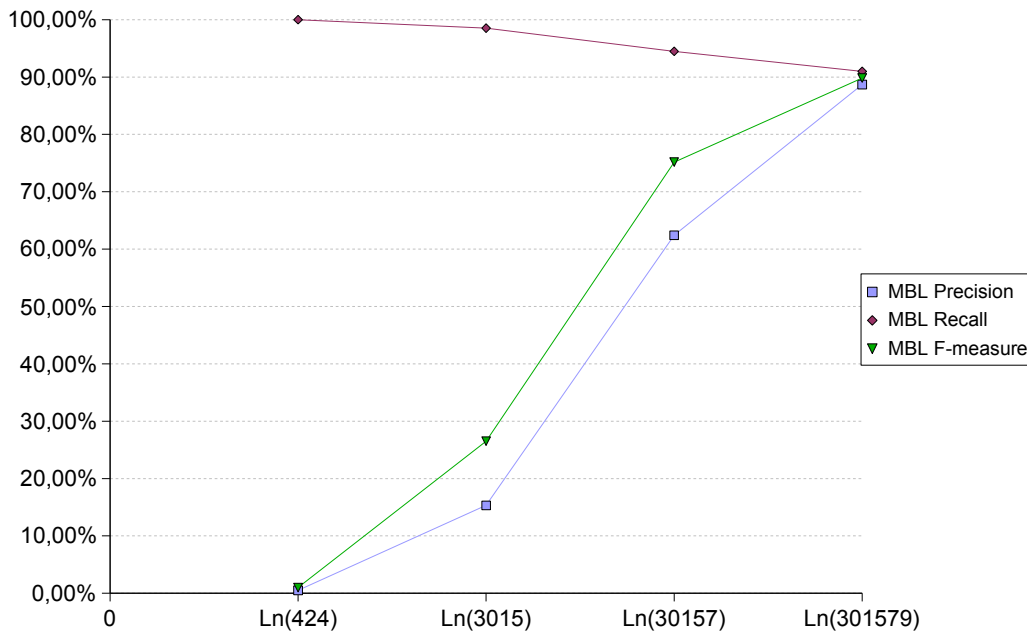


Figure 6.11: The impact of the negative instances

How can I use only positive examples to classify new instances and what will be the performance of such approach in terms of the accuracy and the execution time?

## 6.6 Memory Based Learning Modification

This section presents the modification of the standard Memory Based Learning methods. The goal of the modification is to develop a method that will classify new instances using only positive instances. The motivation in using only the positive instances is that the number of negative instances is prevalent (more than 99%) what makes the execution time very long.

The main change in the learning process relies on storing only positives instances and ignoring the negatives. This has huge impact on the testing process. The k-NN could not be used because only one class of instances is stored. This required new approach in the testing process. The modified testing process is illustrated on the example of the *Zgromadzenie data* annotation. The goal is to annotate the date of the annual meeting. The example sentence is following:

Walne Zgromadzenie odbędzie się 27.11.2007 o godz. 11:00 w siedzibie spółki.

The Annual Meeting will be held on 27.11.2007 at 11:00 in the registered office.

Only a fragment of the sentence will be taken into consideration in this example, however, in fact, whole sentences is processed. The following sequence of the tokens is used in the example:

Zgromadzenie	odbędzie	się	27	.	11	.	2007	o	godz
--------------	----------	-----	----	---	----	---	------	---	------

Figure 6.12: The sequence of the tokens used in the example.

The matching process starts from computing the similarities to *starting* and *ending boundaries* patterns for every single token in the document. The similarity between two instances X and Y is calculated from the Equation 6.1. The X is the token for which the similarity is being calculated and the Y is the token that is a *starting* or *ending boundary* in the training set.

$$\Delta(X, Y) = \frac{\Delta_{outer}(X, Y) + \Delta_{inner}(X, Y) + \Delta_{prep}(X, Y) + \Delta_{noun}(X, Y)}{2w_o o + 2w_i i + w_p p + w_n n} \quad (6.1)$$

The numerator in the Equation 6.1 is a sum of the values that represent the similarities between the instances X and Y in respect to different features.  $\Delta_{outer}(X, Y)$  concerns *the outer context*,  $\Delta_{inner}(X, Y)$  *the inner context*,  $\Delta_{preps}(X, Y)$  *the proceeding prepositions*, and  $\Delta_{nouns}(X, Y)$  *the proceeding nouns*.

The denominator has the value for which two instances are equal.  $w_o$ ,  $w_i$ ,  $w_p$ ,  $w_n$  represent the weights of the *outer context*, *inner context*, *prepositions* and *nouns characteristic*, respectively. The  $w_o$  and  $w_i$  weights are multiply by two because each token from the *outer* and *inner contexts* is represented by two features (the token base and the grammatic category). Those features has the same weights because none of them is considered to be more important. However, this should be experimentally verified if those features weights are equally important for different types of an annotations.

The initial values of the weights were set as following  $w_o = w_i = 1$  and  $w_p = w_n = 3$ . The weights of *the prepositions* and *nouns characteristics* are higher then the weights of *the inner* and *outer context* because of the reason that the nouns are a good determinant of the event and the prepositions determine the role of the noun argument. Let's consider an example, the announcement of the Annual Meeting contains at least two dates: the date of the meeting and the trust deposit date. The dates can appear in the following contexts: *zgromadzenie odbędzie się 27.11.2007* (the meeting will be held on 27.11.2007) and *złożyć depozyt do dnia 14.11.2007* (the trust should be held in till 14.11.2007), respectively. In those examples the nouns determine the role of the dates. In turn, in the following sentences *zgromadzenie zacznie się o 12:00* (the meeting will start at 12:00) and *depozyt należy złożyć do 12:00* (the trust should be held in before 12:00) the prepositions determine the role of the hours.

The  $\Delta_{outer}(X, Y)$  function returns the tokens similarity in respect to the *outer context characteristic*. The *outer context characteristic* takes into consideration *o* first tokens base and grammatical class from outer context.

$$\Delta_{outer}(X, Y) = \sum_{j=1}^p [w_o(\delta(base(X_{-j}), base(Y_{-j})) + \delta(gram(X_{-j}), gram(Y_{-j})))] \quad (6.2)$$

The  $\Delta_{inner}(X, Y)$  function returns tokens similarity in respect to the *inner context characteristic*. The *inner context characteristic* takes into consideration *i* first tokens base and grammatical class from inner context.

$$\Delta_{inner}(X, Y) = \sum_{j=0}^{i-1} [w_i(\delta(base(X_{-j}), base(Y_{-j})) + \delta(gram(X_{-j}), gram(Y_{-j})))] \quad (6.3)$$

The  $\Delta_{preps}(X, Y)$  function returns tokens similarity in respect to the *proceeding prepositions characteristic*. The *proceeding prepositions characteristic* takes into consideration  $p$  proceeding preposition base.

$$\Delta_{prep}(X, Y) = \sum_{i=1}^p w_p \delta(\text{base}(\text{prep}(X, i)), \text{base}(\text{prep}(Y, i))) \quad (6.4)$$

The  $\Delta_{nouns}((X, Y))$  function returns tokens similarity in respect to the *proceeding nouns characteristic*. The *proceeding nouns characteristic* takes into consideration  $n$  proceeding nouns base.

$$\Delta_{noun}(X, Y) = \sum_{i=1}^n w_n \delta(\text{base}(\text{noun}(X, i)), \text{base}(\text{noun}(Y, i))) \quad (6.5)$$

The  $\delta(x, y)$  function is a binary function that returns  $1$  when both values are equal and  $0$  in other case.

$$\delta(x, y) = \begin{cases} 1 & \text{if } x = y \\ 0 & \text{otherwise} \end{cases} \quad (6.6)$$

The  $\text{gram}(X_{-j})$  function returns the grammatical class of given token.

The  $\text{base}(X_{-j})$  function returns the base of given token.

The  $X_{-j}$  in equations refers to  $j$ -th token on the left side of the token  $X$ . Similarly the  $X_{+j}$  refers to  $j$ -th token on the right side of the  $X$  token.  $X_{+0}$  refers to the token  $X$ .

Similarity to Pattern 3	0.0	0.0	0.0	0.0	0.0	0.0	0.1	0.3	0.9	0.5
Similarity to Pattern 2	0.0	0.0	0.0	0.0	0.0	0.3	0.7	0.4	0.3	0.0
Similarity to Pattern 1	0.0	0.0	0.0	0.0	0.0	0.1	0.2	0.8	0.4	0.0
Similarity to Pattern 3	0.0	0.3	0.3	0.4	0.8	0.4	0.0	0.0	0.0	0.0
Similarity to Pattern 2	0.1	0.3	0.7	0.5	0.3	0.1	0.0	0.0	0.0	0.0
Similarity to Pattern 1	0.0	0.2	0.3	0.9	0.4	0.2	0.0	0.0	0.0	0.0
Sequence of Tokens										

Figure 6.13: The similarities for the *starting* and the *ending boundaries* for the example.

After calculating *starting* and *ending boundary similarities* for every token and every pattern two arrays are constructed. The first array contains *maximum values of starting boundary similarities* and the other one *maximum values of ending boundary similarities*.

In the next step *the array of maximum values of starting boundary similarities* is search for the first sequence of tokens for which the similarities are higher than *the starting boundary threshold*. When the sequence is found then the token with highest value of the similarity is chosen. If there are two or more tokens with the highest similarity then the most left is chosen. Next, for the chosen token *the ending boundary* is search. The token that can be *the ending boundary* is searched in the sequence of the tokens from *the current token position + the minimal length of the annotation to the current token position + the maximum length of the annotation*. The token with highest *ending boundary similarity* is chosen. If *the ending boundary similarity* is higher than *ending boundary threshold* the token becomes the *ending boundary* and new annotation is created. If there is no token that meets this condition then the process of searching for *the starting boundary* is being continued from the token that follows the sequence of tokens.

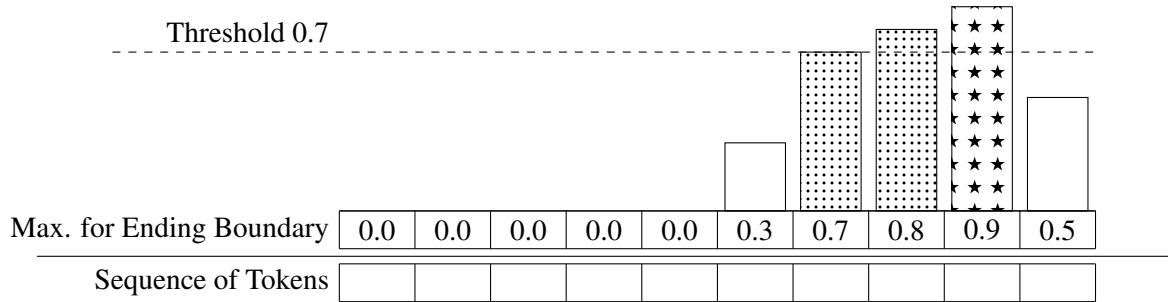


Figure 6.14: The maximum similarities for *the ending boundary* for the example.

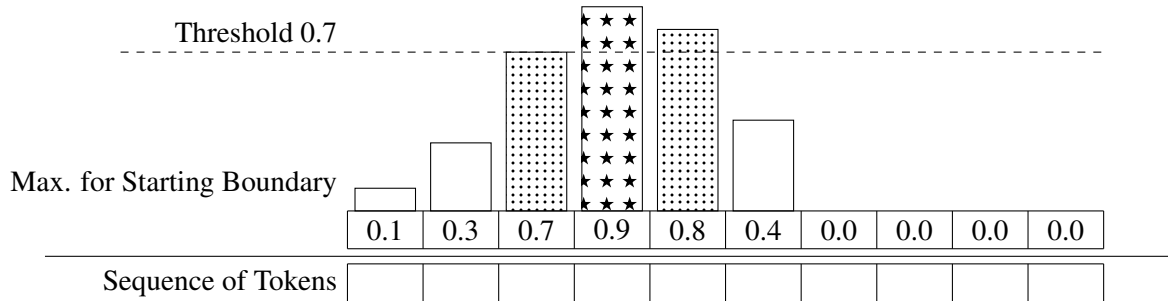


Figure 6.15: The maximum similarities for *the starting boundary* for the example.

## 6.7 Experiment with Modified Memory Based Learning Method

The performance of the modified Memory Based Learning (ExMBL) method was measured and compare to the results obtained by the standard Memory Based Learning method. The performance of ExMBL is presented in the Table 6.4. The Figure 6.17 presents the comparison of both methods. The results obtained by the modified MBL are as good as for the standard MBL method with all negative instances. The execution time was the same as for standard MBL using balanced number of negative examples. The results shows that the performance of the modified method remained the same and the execution time was shortened from few hours to several minutes.

	Precision	Recall	F-measure
ExMBL	92.16	88.28	90.16

Table 6.4: Performance of the modified Memory Based Learning methods

The results confirmed that using only positives examples we can obtain the the same level of performance as when we are using negative examples as well. The next question that was stated was:

Can be the performance improved by changing the methods parameters (i.e. the thresholds, the context size, the number of proceeding nouns and prepositions)?

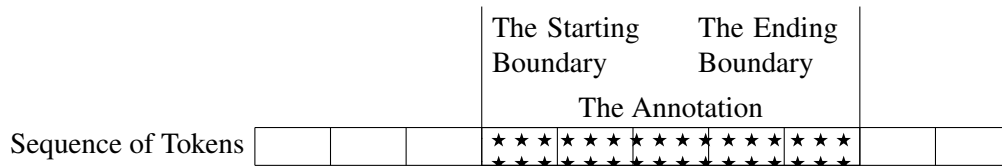


Figure 6.16: The result of finding the starting and ending boundaries for the annotation.

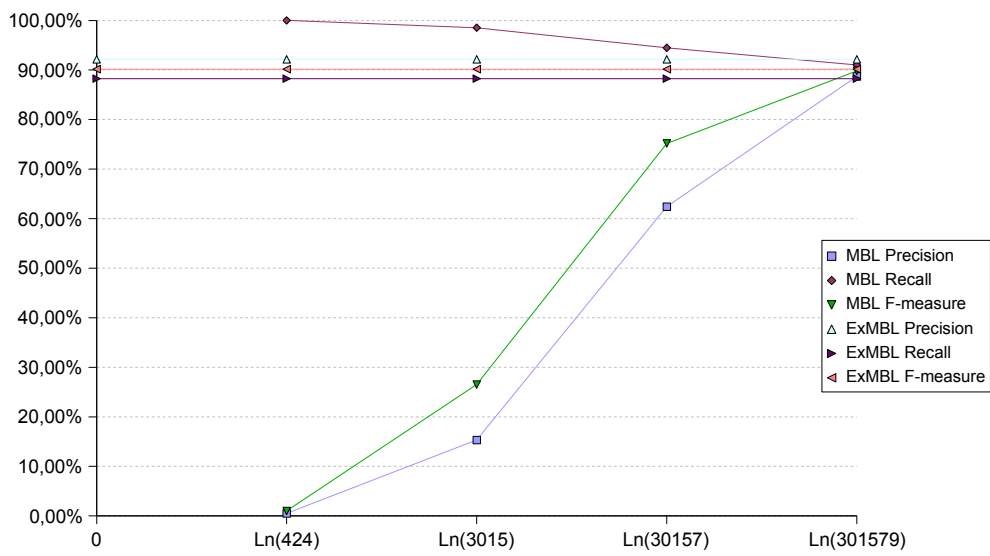


Figure 6.17: Standard MBL and modified MBL comparison

## Chapter 7

# Results and Conclusion

The performance of the modified Memory Based Learning methods was tested on three types of annotations. As a method to estimate generalization error was used 10-fold Cross Validation. The training set contains 390 documents that treat about Annual Meetings of Stockholders. Those documents contain at most 7 types of annotations but only three of them were used in the tests:

- *Zgromadzenie\_godzina* - time of Annual Meeting, **374** instances,
- *Zgromadzenie\_data* - date of Annual Meeting, **424** instances,
- *Zgromadzenie\_miejsce* - place of Annual Meeting, **398** instances.

For each type of annotation several tests with different configuration of learning and training parameters were performed. The initial values of parameters were set as follows:

Parameter	Description	Value
<i>Outer</i>	number of tokens in the outer context	3
<i>Inner</i>	number of tokens in the inner context	2
<i>Preps</i>	number of proceeding prepositions	1
<i>Nouns</i>	number of proceeding nouns	1
<i>Starting</i>	value of the threshold for starting boundary	0.8
<i>Ending</i>	value of the threshold for ending boundary	0.8

Table 7.1: Initial values of learning and testing parameters.

The testing strategy was following:

- all parameters were set to initial values,
- for each feature set of tests were performed in which the value of the feature was changed from the starting to the ending value with given step. Table 7.2 presents the range of parameters values.

Each annotation was tested with 31 different configurations of parameters what gives 93 tests. The following subsection contains results from the tests. The last subsection discusses the results.

The results presented in this chapter concerns the performance of full annotation recognition. This means that the performance of recognition the starting and the ending boundaries are not presented. The presented performance is the performance of both boundaries recognition and the heuristic rule used to pair the boundaries, as it was described in the previous chapter.

Parameter	From	To	Step
<i>Outer</i>	1	5	1
<i>Inner</i>	1	5	1
<i>Preps</i>	1	3	1
<i>Nouns</i>	1	3	1
<i>Starting</i>	0.50	0.95	0.05
<i>Ending</i>	0.50	0.95	0.05

Table 7.2: The range of the parameters values used in the experiment.

## 7.1 Performance of the Modified Memory Based Learning

### 7.1.1 Results for *Zgromadzenie godzina*

This section presents the average values of Precision, Recall and F-measure in 10-fold Cross Validation for the *Zgromadzenie godzina* annotation.

Training Parameters				Thresholds		Results		
Outer	Inner	Prep	Noun	Starting	Ending	Precision	Recall	F-measure
1	2	1	1	0.8	0.8	10.27	42.43	16.39
2	2	1	1	0.8	0.8	96.42	86.87	91.32
3	2	1	1	0.8	0.8	99.31	80.10	88.58
4	2	1	1	0.8	0.8	98.42	<b>87.06</b>	<b>92.33</b>
5	2	1	1	0.8	0.8	<b>99.65</b>	76.18	86.18

The average values of Precision, Recall and F-measure in 10-fold Cross Validation for the *Zgromadzenie godzina* annotation and changing value of the *Outer* parameter.

The range of the F-measure for different values of *the outer context* is wide and obtained values from 16.39% to 92.33%. The results shows that the wider *the outer context*, the better the precision. Wide context makes the patterns more specific and this causes that at some level the recall value is decreasing. In this case the optimal value for *the inner context* was 4.

Training Parameters				Thresholds		Results		
Outer	Inner	Prep	Noun	Starting	Ending	Precision	Recall	F-measure
3	1	1	1	0.8	0.8	98.33	85.12	91.21
3	2	1	1	0.8	0.8	<b>99.31</b>	80.10	88.58
3	3	1	1	0.8	0.8	96.91	<b>86.81</b>	<b>91.52</b>
3	4	1	1	0.8	0.8	97.03	83.17	89.50
3	5	1	1	0.8	0.8	98.26	82.34	89.51

The average values of Precision, Recall and F-measure in 10-fold Cross Validation for the *Zgromadzenie godzina* annotation and changing value of the *Inner* parameter.

The obtained values of F-measure for different values of *the inner context* vary from 88.58% to 91.21%. However, the F-measure for the initial configuration was the smallest and the best result was obtained for *the inner context* equal to 3.

Training Parameters				Thresholds		Results		
Outer	Inner	Prep	Noun	Starting	Ending	Precision	Recall	F-measure
3	2	1	0	0.8	0.8	98.63	<b>91.95</b>	<b>95.14</b>
3	2	1	1	0.8	0.8	<b>99.28</b>	80.16	88.62
3	2	1	2	0.8	0.8	97.31	77.32	86.08
3	2	1	3	0.8	0.8	98.31	76.54	85.96

The average values of Precision, Recall and F-measure in 10-fold Cross Validation for the *Zgromadzenie godziny* annotation and changing value of the *Nouns* parameter.

The highest precision was obtained for one proceeding noun what confirms the importance of the noun as a determinant of the annotation role. However, the nouns caused that the patterns were too specific and for one or more nouns the recall was decreasing.

Training Parameters				Thresholds		Results		
Outer	Inner	Prep	Noun	Starting	Ending	Precision	Recall	F-measure
3	2	0	1	0.8	0.8	99.30	81.46	89.42
3	2	1	1	0.8	0.8	<b>99.38</b>	80.16	88.62
3	2	2	1	0.8	0.8	97.07	<b>90.40</b>	<b>93.54</b>
3	2	3	1	0.8	0.8	97.80	80.71	88.31

The average values of Precision, Recall and F-measure in 10-fold Cross Validation for the *Zgromadzenie godziny* annotation and changing value of the *Preps* parameter.

The results are similar to the results for different number of nouns in terms of the precision. The highest recall was obtained for 2 prepositions what can mean that not only one but a sequence of the prepositions is a good delimiter of the annotation role.

Training Parameters				Thresholds		Results		
Outer	Inner	Prep	Noun	Starting	Ending	Precision	Recall	F-measure
3	2	1	1	0.50	0.80	85.75	82.27	83.67
3	2	1	1	0.55	0.80	85.75	82.27	83.67
3	2	1	1	0.60	0.80	88.53	82.27	84.92
3	2	1	1	0.65	0.80	88.53	82.27	84.92
3	2	1	1	0.70	0.80	<b>96.70</b>	<b>82.78</b>	88.92
3	2	1	1	0.75	0.80	98.75	82.51	<b>89.65</b>
3	2	1	1	0.80	0.80	99.35	79.60	88.10
3	2	1	1	0.85	0.80	99.35	79.60	88.10
3	2	1	1	0.90	0.80	99.18	67.56	80.11
3	2	1	1	0.95	0.80	99.18	67.56	80.11

The average values of Precision, Recall and F-measure in 10-fold Cross Validation for the *Zgromadzenie godziny* annotation and changing value of the *Starting* parameter.

Training Parameters				Thresholds		Results		
Outer	Inner	Prep	Noun	Starting	Ending	Precision	Recall	F-measure
3	2	1	1	0.80	0.50	7.21	6.79	6.99
3	2	1	1	0.80	0.55	7.21	6.79	6.99
3	2	1	1	0.80	0.60	20.77	19.37	20.02
3	2	1	1	0.80	0.65	20.77	19.37	20.02
3	2	1	1	0.80	0.70	96.64	<b>84.83</b>	<b>90.29</b>
3	2	1	1	0.80	0.75	97.15	83.79	89.84
3	2	1	1	0.80	0.80	99.35	79.60	88.10
3	2	1	1	0.80	0.85	99.35	79.60	88.10
3	2	1	1	0.80	0.90	<b>100.00</b>	58.71	73.59
3	2	1	1	0.80	0.95	<b>100.00</b>	58.71	73.59

The average values of Precision, Recall and F-measure in 10-fold Cross Validation for the *Zgromadzenie godziny* annotation and changing value of the *Ending* parameter.

What was expected *the thresholds for starting and ending boundaries* were responsible for the generalization. The higher the thresholds, the lower the generalization. Low generalization means that the patterns were too specific. The more specific patterns, the higher the precision and lower the recall.

### 7.1.2 Results for *Zgromadzenie data*

This section presents the average values of Precision, Recall and F-measure in 10-fold Cross Validation for the *Zgromadzenie data* annotation.

Training Parameters				Thresholds		Results		
Outer	Inner	Prep	Noun	Starting	Ending	Precision	Recall	F-measure
1	2	1	1	0.8	0.8	61.37	76.85	68.01
2	2	1	1	0.8	0.8	87.13	<b>83.60</b>	<b>84.93</b>
3	2	1	1	0.8	0.8	<b>97.53</b>	71.17	82.62
4	2	1	1	0.8	0.8	80.91	69.33	74.54
5	2	1	1	0.8	0.8	85.77	65.46	74.06

The average values of Precision, Recall and F-measure in 10-fold Cross Validation for the *Zgromadzenie data* annotation and changing value of the *Outer* parameter.

The results are similar to the results for *Zgromadzenie godziny* annotation. The only difference is that the best value of F-measure was obtained for *the outer token* equal 2.

Training Parameters				Thresholds		Results		
Outer	Inner	Prep	Noun	Starting	Ending	Precision	Recall	F-measure
3	1	1	1	0.8	0.8	79.10	69.87	74.00
3	2	1	1	0.8	0.8	<b>97.53</b>	72.17	82.62
3	3	1	1	0.8	0.8	94.98	<b>83.33</b>	<b>88.58</b>
3	4	1	1	0.8	0.8	96.63	76.41	85.07
3	5	1	1	0.8	0.8	94.24	76.68	84.26

The average values of Precision, Recall and F-measure in 10-fold Cross Validation for the *Zgromadzenie data* annotation and changing value of the *Inner* parameter.

The results are similar to the results for the *Zgromadzenie godzina* annotation.

Training Parameters				Thresholds		Results		
Outer	Inner	Prep	Noun	Starting	Ending	Precision	Recall	F-measure
3	2	1	0	0.8	0.8	91.01	<b>85.26</b>	<b>87.79</b>
3	2	1	1	0.8	0.8	<b>97.53</b>	72.17	82.62
3	2	1	2	0.8	0.8	83.11	65.98	73.34
3	2	1	3	0.8	0.8	82.84	64.82	72.50

The average values of Precision, Recall and F-measure in 10-fold Cross Validation for the *Zgromadzenie data* annotation and changing value of the *Nouns* parameter.

The results are similar to the results for the *Zgromadzenie godzina* annotation.

Training Parameters				Thresholds		Results		
Outer	Inner	Prep	Noun	Starting	Ending	Precision	Recall	F-measure
3	2	0	1	0.8	0.8	<b>97.66</b>	<b>75.75</b>	<b>85.04</b>
3	2	1	1	0.8	0.8	97.53	72.17	82.62
3	2	2	1	0.8	0.8	82.27	70.51	75.63
3	2	3	1	0.8	0.8	89.58	68.15	76.98

The average values of Precision, Recall and F-measure in 10-fold Cross Validation for the *Zgromadzenie data* annotation and changing value of the *Preps* parameter.

The best results were obtained when the prepositions were omitted. This means that the prepositions are not crucial to identify the *Zgromadzenie data* annotations.

Training Parameters				Thresholds		Results		
Outer	Inner	Prep	Noun	Starting	Ending	Precision	Recall	F-measure
3	2	1	1	0.50	0.80	74.34	68.00	70.83
3	2	1	1	0.55	0.80	74.34	68.00	70.83
3	2	1	1	0.60	0.80	81.21	71.77	75.93
3	2	1	1	0.65	0.80	91.21	71.77	75.93
3	2	1	1	0.70	0.80	83.60	71.77	76.98
3	2	1	1	0.75	0.80	85.04	70.33	76.75
3	2	1	1	0.80	0.80	97.53	<b>72.17</b>	<b>82.62</b>
3	2	1	1	0.85	0.80	97.53	<b>72.17</b>	<b>82.62</b>
3	2	1	1	0.90	0.80	<b>97.80</b>	32.92	48.62
3	2	1	1	0.95	0.80	<b>97.80</b>	32.92	48.62

The average values of Precision, Recall and F-measure in 10-fold Cross Validation for the *Zgromadzenie data* annotation and changing value of the *Starting* parameter.

Training Parameters				Thresholds		Results		
Outer	Inner	Prep	Noun	Starting	Ending	Precision	Recall	F-measure
3	2	1	1	0.80	0.50	8.59	7.12	7.77
3	2	1	1	0.80	0.55	8.59	7.12	7.77
3	2	1	1	0.80	0.60	43.98	36.61	39.89
3	2	1	1	0.80	0.65	43.98	36.61	39.89
3	2	1	1	0.80	0.70	80.70	65.76	72.33
3	2	1	1	0.80	0.75	81.41	65.52	72.45
3	2	1	1	0.80	0.80	97.53	<b>72.17</b>	<b>82.62</b>
3	2	1	1	0.80	0.85	97.53	<b>72.17</b>	<b>82.62</b>
3	2	1	1	0.80	0.90	<b>98.03</b>	60.19	74.25
3	2	1	1	0.80	0.95	<b>98.03</b>	60.19	74.25

The average values of Precision, Recall and F-measure in 10-fold Cross Validation for the *Zgromadzenie data* annotation and changing value of the *Ending* parameter.

The results are similar to the results for the *Zgromadzenie godzina* annotation.

### 7.1.3 Results for *Zgromadzenie miejsce*

This section presents the average values of Precision, Recall and F-measure in 10-fold Cross Validation for the *Zgromadzenie miejsce* annotation.

Training Parameters				Thresholds		Results		
Outer	Inner	Prep	Noun	Starting	Ending	Precision	Recall	F-measure
1	2	1	1	0.8	0.8	9.76	19.06	12.87
2	2	1	1	0.8	0.8	76.00	50.28	60.35
3	2	1	1	0.8	0.8	<b>84.30</b>	49.00	<b>61.67</b>
4	2	1	1	0.8	0.8	65.55	<b>56.96</b>	60.88
5	2	1	1	0.8	0.8	67.56	52.07	58.70

The average values of Precision, Recall and F-measure in 10-fold Cross Validation for the *Zgromadzenie miejsce* annotation and changing value of the *Outer* parameter.

The results are similar to the results for the *Zgromadzenie godzina* and the *Zgromadzenie godzina* annotation. The only difference is in the value for which best results are obtained.

Training Parameters				Thresholds		Results		
Outer	Inner	Prep	Noun	Starting	Ending	Precision	Recall	F-measure
3	1	1	1	0.8	0.8	66.46	<b>62.95</b>	64.60
3	2	1	1	0.8	0.8	<b>84.30</b>	49.00	61.67
3	3	1	1	0.8	0.8	76.06	57.58	<b>65.47</b>
3	4	1	1	0.8	0.8	79.18	51.82	62.50
3	5	1	1	0.8	0.8	82.21	48.71	61.02

The average values of Precision, Recall and F-measure in 10-fold Cross Validation for the *Zgromadzenie miejsce* annotation and changing value of the *Inner* parameter.

The results are similar to the results for the *Zgromadzenie godzina* and the *Zgromadzenie godzina* annotation. The only difference is in the value for which best results are obtained.

Training Parameters				Thresholds		Results		
Outer	Inner	Prep	Noun	Starting	Ending	Precision	Recall	F-measure
3	2	1	0	0.8	0.8	71.88	<b>54.08</b>	61.47
3	2	1	1	0.8	0.8	<b>84.33</b>	49.00	<b>61.67</b>
3	2	1	2	0.8	0.8	65.33	51.18	57.37
3	2	1	3	0.8	0.8	67.00	<b>54.08</b>	61.47

The average values of Precision, Recall and F-measure in 10-fold Cross Validation for the *Zgromadzenie miejsce* annotation and changing value of the *Nouns* parameter.

Training Parameters				Thresholds		Results		
Outer	Inner	Prep	Noun	Starting	Ending	Precision	Recall	F-measure
3	2	0	1	0.8	0.8	68.76	43.47	53.06
3	2	1	1	0.8	0.8	<b>84.30</b>	49.00	61.67
3	2	2	1	0.8	0.8	68.16	<b>61.73</b>	<b>64.72</b>
3	2	3	1	0.8	0.8	71.79	58.76	64.57

The average values of Precision, Recall and F-measure in 10-fold Cross Validation for the *Zgromadzenie miejsce* annotation and changing value of the *Preps* parameter.

Training Parameters				Thresholds		Results		
Outer	Inner	Prep	Noun	Starting	Ending	Precision	Recall	F-measure
3	2	1	1	0.50	0.80	35.75	21.63	26.82
3	2	1	1	0.55	0.80	35.75	21.63	26.82
3	2	1	1	0.60	0.80	60.45	36.77	45.50
3	2	1	1	0.65	0.80	60.45	36.77	45.50
3	2	1	1	0.70	0.80	65.53	39.81	49.30
3	2	1	1	0.75	0.80	65.53	39.81	49.30
3	2	1	1	0.80	0.80	84.30	<b>49.00</b>	<b>61.67</b>
3	2	1	1	0.85	0.80	84.30	<b>49.00</b>	<b>61.67</b>
3	2	1	1	0.90	0.80	<b>86.67</b>	43.80	57.92
3	2	1	1	0.95	0.80	<b>86.67</b>	43.80	57.92

The average values of Precision, Recall and F-measure in 10-fold Cross Validation for the *Zgromadzenie miejsce* annotation and changing value of the *Starting* parameter.

Training Parameters				Thresholds		Results		
Outer	Inner	Prep	Noun	Starting	Ending	Precision	Recall	F-measure
3	2	1	1	0.80	0.50	2.48	2.88	2.66
3	2	1	1	0.80	0.55	2.48	2.88	2.66
3	2	1	1	0.80	0.60	10.57	12.02	11.23
3	2	1	1	0.80	0.65	10.57	12.02	11.23
3	2	1	1	0.80	0.70	75.03	<b>64.69</b>	<b>69.41</b>
3	2	1	1	0.80	0.75	75.03	<b>64.69</b>	<b>69.41</b>
3	2	1	1	0.80	0.80	84.30	49.00	61.67
3	2	1	1	0.80	0.85	84.30	49.00	61.67
3	2	1	1	0.80	0.90	<b>89.90</b>	41.81	56.79
3	2	1	1	0.80	0.95	<b>89.90</b>	41.81	56.79

The average values of Precision, Recall and F-measure in 10-fold Cross Validation for the *Zgromadzenie miejsce* annotation and changing value of the *Ending* parameter.

The results are similar to the results for the *Zgromadzenie godzina* and the *Zgromadzenie godzina* annotation. The only difference is in the value for which best results are obtained.

#### 7.1.4 Summary of the Results

The Table 7.3 presents the training and matching parameters for which the best F-measure value was obtained for different types of annotations.

Annotation Type	Training Parameters				Thresholds		Results
	Outer	Inner	Prep	Noun	Starting	Ending	F-measure
Zgromadzenie godzina	3	2	1	0	0.80	0.80	95.14
Zgromadzenie data	3	3	1	1	0.80	0.80	88.58
Zgromadzenie miejsce	3	3	1	1	0.80	0.80	65.47

Table 7.3: The summary of the best parameters configurations for the test annotation types.

The results strongly depends on the annotations characteristics. The *Zgromadzenie godzina* annotation has the simplest pattern in terms of the number of unique ways how it can be presented. In turn the *Zgromadzenie data* annotation has little more complex structure. The most complex structure had the *Zgromadzenie miejsce* annotation that was a combination of a street, a home number, an apartment number, a building name, a city and a postal code. The elements could appear in different order what caused that more unique patterns were required. Some of the combinations were unique or appeared only in few instances and the system could not learn the patterns to identify them. This is the reason why the *Zgromadzenie miejsce* annotation, comparing to other two annotations, obtained so low result.

The example instances of each type are printed in the Appendix 3.

## 7.2 Comparison to the Baseline

The performance of the modified Memory Based Learning method was relatively good comparing them to the results of the regular expressions. However, the performance of the regular expressions are a matter of time. This means that they could be improved if more effort was spent on

Annotation Type	Memory Based Learning	Regular Expressions
Zgromadzenie godzina	95.14	95.78
Zgromadzenie data	88.58	73.75
Zgromadzenie miejsce	65.47	16.96

Table 7.4: The comparison of the Memory Based Learning and the regular expressions results.

creation of the regular expressions. In turn, the performance of the automatic approach could be experimentally improved by testing another parameters configuration or introducing new features. The good side of this solution is that no human interference is required in the parameters optimization. The bad side of this solution is that the solution space is very broad and a test for one configuration takes about 15 minutes. If we consider only the ranges for the parameters that were used in the experiments it gives us  $5 \cdot 5 \cdot 4 \cdot 4 \cdot 10 \cdot 10 = 40.000$  combinations of the parameters and it would take about 10.000 hours to test them all. What more, there is no guarantee that the better configuration exists.

## 7.3 Conclusion

### 7.3.1 RQ1 How can the process of patterns acquisition for Events Recognition from the reports of Polish stockholders be automated?

The results shows that the standard Memory Based Learning method can obtained high results but the execution time is not accepted. The proposed modification of the standard Memory Based Learning method shortened the execution time to acceptable level and the performance remained at the same level. This shows that modified Memory Based Learning as well as Decision Tress can be applied to automatic creation of the pattern for the Event Recognition purpose.

The observations from the experiments are that the performance of the automatically created linguistic patterns for the Semantic Labeling task depends on the following factors:

- annotation characteristic - in how many different ways can be the annotation written and in how many different ways it can appear in a sentence,
- feature selection - which features best describe given type of annotation,
- parameters configuration - what is the best configuration of training and matching parameters for given type of annotation,
- data representation - what kind of preprocessing of the document was performed.

### 7.3.2 RQ2 What are the approaches to patterns acquisition for the Event Recognition task?

Nowadays, the Event Recognition task is in the center of interest among many linguistic engineers. In general there are two approaches to Event Recognition: *manual* and *automatic*. First Information Extraction Systems made use of hand-coded *extraction patterns* that were written as *regular expressions* or using self-made pseudo languages. This approach allowed obtaining good results in terms of precision and recall. However, it was very time-consuming and involved

domain experts. There was need to shorten the time that was required to create the extraction patterns.

Manually created patterns have main disadvantage – they are very specific. In turn, a method that will automatically extract patterns from one type of documents (i.e. *stockholders meeting*) might be later use to extract patterns from other types of documents from the same documents (i.e. *stock sales*) or even from another domain.

Many experiments were conducted that applied different methods of Machine Learning, like Decision Trees (ID3 and C4.5), Hidden Markov Models and Memory Based Learning. According to the literature no one method is concerned as the best one for Event Recognition task. The results could not be compared because the experiments were conducted in different environments, what means that different training sets were used. There is a need to test different methods using the same training data in order to make the comparison.

The number of tools for processing Polish language is very limited. The only available components are TaKIPI (a part of speech tagger for Polish language) [34] and Manufakturzysta (a unofficial component to annotate the documents tagged by TaKIPI). There is also a thesaurus for Polish language [29] but the interface of the thesaurus is still unavailable.

### 7.3.3 RQ3 How can Event Recognition be treated as a classification task in order to apply Decision Trees, Naive Bayes and Memory Based Learning methods?

In order to apply a Machine Learning method to Event Recognition it must be defined as a classification task. In the naive approach the Event Recognition task can be treated as a single-classification task that relies on testing every subsequence of the tokens in the document and decide if it is an annotation or not. However, this simple approach has two problems. One of them is the huge number of instances to test. The other one is the potential problem with the feature instances.

In the other approach the Event Recognition can be treated as a multi-classification task. Bennett et al. [8] used two classifiers to recognize the beginnings and the endings of the annotations. Then a heuristic rule was used to decide which potential boundaries create an annotation.

Pradhan et al. [37] used one classifier that classify each token as beginning of the annotation, being inside the annotation or being outside the annotation. Then a heuristic rule was used to decide which sequences of tokens create the annotation.

### 7.3.4 RQ4 What performance can be obtained by applying common machine learning algorithms to pattern acquisition for event recognition for the Polish stockholders reports task?

In the initial experiment the highest performance was obtained for Memory Based Learning ( $F = 89.84\%$ ) and Decision Trees ( $F = 87.13\%$ ). The Naive Bayes obtained the lowest performance ( $F = 73.27$ ).

The MBL obtained best results but the processing time was more than 24 hours. The reason of the long processing time is the prevalent number of negative examples. There are 1000 more negative examples than positives. The second experiment showed that the reduction of the negative instances has unfavorable impact on the performance.

The performance of the modified Memory Based Learning method was tested on the training set concerning the stock exchange domain. The three annotations types were used in the tests: *the meeting date*, *the meeting time* and *the meeting place*. The highest values of F-measure that were obtained for modified Memory Based Learning are as follows: *the meeting time* 95%, *the meeting date* 88% and *the meeting place* 65%. Comparing to the manual approach in one case

the performance was comparable (*the meeting time 95%*) and in the two other cases higher (*the meeting date and the meeting place*).

## 7.4 Future Work

### 7.4.1 Question to answer

- Can be the performance of the Decision Trees improved by parameters optimization or in another way?

### 7.4.2 Place for improvements

- Make use of the Sentencer, a component that divide document into sentences. This would allow creating patterns for the sentence context instead of the document context. In current approach this component was not been used because of the low precision in sentence delimiter recognition. For example a number "1.200.000 PLN" was treated as three sentences: "1", "200", "000 PLN" despite it was part of a sentence and represent amount of money.
- Make use of the Words Similarities Matrix. In current approach when base of the words were compared the result could be either 0 or 1. However, some words are more or less similar. For example, when we look at the date annotation it contains a name of a month, let's say it is *January*. Now, when we compare this to two words *March* and *Paris* we can see that the base is completely different but *January* is more similar to *March* than *Paris*. Introducing the Matrix of similarities the comparison may be more accurate.
- Make use of the Polish wordnet i.e. plWordNet [29], [2]. WordNet is a semantic web for polish words. The purpose of the wordnet is the same as the purpose of the Matrix of similarities. In the current approach for each noun only base and grammatical class is taken into consideration. It could be extended by semantic class of the token.
- Apply automatic *feature selection* and *parameters optimization* in order to get the best training and matching configurations for given type of annotations.
- Decompose the complex annotations to the atomic elements. This concern annotations like location or address that contains elements like a street name, a building name, a city and so on.

## Appendix A

# Stock Exchange Schema Annotation

This appendix contains description of schema annotation that was used in the experiment performed within this master thesis. The description is based on act Dz.U.05.209.1744 from Polish law that defines 26 types of information that the issuer is obligated to publish. Below brief description of chosen tasks is provided and main attributes are listed.

**Zgromadzenie** - information about planning Annual Meeting of Stockholders,

- *Zgromadzenie\_data* - Annual Meeting date,
- *Zgromadzenie\_godzina* - Annual Meeting hour,
- *Zgromadzenie\_miejsce* - Annual Meeting place,
- *Zgromadzenie\_przebieg* - Annual Meeting agenda,
- *Zgromadzenie\_depozyt\_data* - trust deposit date,
- *Zgromadzenie\_depozyt\_godzina* - trust deposit hour,
- *Zgromadzenie\_depozyt\_miejsce* - trust deposit place.

**UmowaZawarcie** - issuer has sign a contract,

- *ZUmowy\_data* - date when the contract was sign,
- *ZUmowy\_strona1* - one side of the contract (issuer),
- *ZUmowy\_strona2* - another side of the contract (company with which issuer sign the contract),
- *ZUmowy\_podmiot* - subject of the contract,
- *ZUmowy\_warunki* - conditions of the contract,
- *ZUmowy\_kary* - penalties,
- *ZUmowy\_wartosc* - value of the contract.

**DywidendaWyplata** - dividend payout,

- *Dywidenda\_emitent* - name of the issuer,
- *Dywidenda\_wartosc* - total value of the dividend,
- *Dywidenda\_akcja* - value per stock,
- *Dywidenda\_dzien\_d* - date when the decision who get the dividend will be made,
- *Dywidenda\_dzien\_w* - date when the dividend will be paid.

## Appendix B

# Stock Exchange Schema Annotation

An example of the report from testing an annotation using 10-fold Cross Validation.

```
Precision and recall tests
=====
>> Information:
  - Date and hour.....: 2007-05-21 03:36:23
  - Annotation type.....: Zgromadzenie_godzina (Zgromadzenie)
  - Number of annotations...: 357

>> Learning parameters:
  - outer context.....: 3
  - inner context.....: 2
  - proceeding prepositions.: 1
  - proceeding nouns.....: 1

>> Testing parameters:
  - starting threshold.....: 0,80000
  - ending threshold.....: 0,80000

>> Results:
  - Iteration no. 1
  -----
  Instances to learn.....: 321
  Instances to find.....: 36
  Instances that were found.....: 30
  Instances recognized correctly..: 30
  Instances recognized incorrectly: 0
  Precision.....: 100,000%
  Recall.....: 83,333%
  F-measure.....: 90,909%

[the results for iterations 2-10]

>> Summary:
  - Precision:
    min.....: 96,429%
    max.....: 100,000%
    avg.....: 99,310%
  - Recall:
    min.....: 71,053%
    max.....: 86,486%
    avg.....: 80,103%
  - F-measure:
    min.....: 83,077%
    max.....: 92,754%
    avg.....: 88,589%
  - End time.....: 2007-05-21 03:51:38
  - Process time.....: 00:15:14.1745184
```

## Appendix C

# Example of training instances

This appendix contains example instances of *Zgromadzenie data*, *Zgromadzenie godzina* and *Zgromadzenie miejsce* annotations that were used in the testing process.

Left Context	Annotation	Right Context
Walne Zgromadzenie Akcjonariuszy, które odbędzie się w dniu Łąkowej 39/44, zwołuje Zwyczajne Walne Zgromadzenie na dzień Zwyczajne Walne Zgromadzenie, które odbędzie się w dniu że Zwyczajne Walne Zgromadzenie Spółki odbędzie się w dniu Polska S.A. informuje, że w dniu Zgromadzenie), które odbędzie się w dniu Walnego Zgromadzenia Akcjonariuszy, które odbędzie się w dniu Zgromadzenia Akcjonariuszy Spółki, które odbędzie się w dniu .A., które odbyło się w dniu Zgromadzenia Akcjonariuszy Spółki, które odbędzie się w dniu §22 Statutu Spółki, zwołuje na dzień Walne Zgromadzenie Spółki, które odbędzie się w dniu uchwał Zarząd Spółki COMP SA informuje, iż na dzień podjętych uchwał na Zwyczajnym Walnym Zgromadzeniu w dniu Spółki zwołuje Nadzwyczajne Walne Zgromadzenie Spółki na dzień ("Spółka") informuje o zwołaniu na dzień Kodeksu spółek handlowych, zwołuje na dzień które odbędzie się w dniu Zgromadzenie Akcjonariuszy Spółki, które odbędzie się w dniu Walne Zgromadzenie Spółki. NWZ odbędzie się w dniu statutu Spółki, zwołuje Nadzwyczajne Walne Zgromadzenie na dzień oraz §16 Statutu, zwołuje na dzień Nadzwyczajne Walne Zgromadzenie Akcjonariuszy Spółki na dzień iż na dzień Walne Zgromadzenie Funduszu. Obrady rozpoczną się w dniu zawiadania wszystkich Akcjonariuszy, że zwołuje na dzień	19 czerwca 2006 roku 29.06.2006r. 22 czerwca 2006r. 22.06.2006 roku 30 czerwca 2006 roku 8 listopada 2006 roku 23 czerwca 2006r. 23 czerwca 2006r. 9 czerwca 2006r. 19 czerwca 2006r. 18 lipca 2006r. 22 czerwca 2006r. 22 czerwca 2006 roku 19 czerwca 2006r. 8 grudnia 2006r 12 grudnia 2006 roku 20 grudnia 2006r. 14 grudnia 2006r 18 grudnia 2006 roku 18 grudnia 2006 roku 19 grudnia 2006r 19 grudnia 2006 roku 22 grudnia 2006r 21 grudnia 2006 roku 21 grudnia 2006 roku 29.12.2006r	o godz. 11:00 w o godzinie 12.00 w , o godz. 12. o godzinie 12.00 w siedzibie , o godzinie 16.30 o godzinie 9.00 w sali o godz. 10.00 o godz. 10.00 o godz. 13.00 o godz. 11.00 o godz. 10:00 o godz. 12.00 , na godzinę 11, w Warszawie Napodstawie §39 ust. . na godz. 10.00 , na godzinę 10.30, na godz. 11.00 , początek o godz. 12 , początek obrad o godzinie 12 , o godzinie 15.00 ,, nagodz. 11 , o godz. 11.00 . o godz. 10.00 zwołane zostało o godzinie 11.15 . Nadzwyczajne WalneZ

Table C.1: Examples of training instances of the *Zgromadzenie data* annotation.

Left Context	Annotation	Right Context
odbędzie się w dniu 12 czerwca 2006 roku o godzinie zwołyje na dzień 28 czerwca 2006r. na godz.	11.00	w siedzibie Spółki w Gdyni przy ul
zwołyje na dzień 28 czerwca 2006 roku, o godz.	1100	Zwyczajne Walne Zgromadzenie
Walne Zgromadzenie na dzień 30 czerwca 2006 roku, na godzinę	11.00	Zwyczajne Walne Zgromadzenie
zwołyje na dzień 29 czerwca 2006r.(czwartek), na godz.	15.00	. Zgromadzenie odbędzie się
Zwyczajne Walne Zgromadzenie na dzień 30 czerwca 2006r. na godz.	11:15	,XV Zwyczajne Walne Zgromadzenie
obrady Walnego Zgromadzenia rozpoczyna się 27 czerwca 2006r. o godz.	11.00	., które odbędzie się
Statutu Spółki, zwołyje na dzień 28 czerwca 2006r. o godz.	11.00	w sali konferencyjnej
Walne Zgromadzenie Akcjonariuszy Spółki na dzień 07.07.2006r. o godz.	1000	w siedzibie Spółki w
Chelmie, które odbędzie się w dniu 30 czerwca 2006r. o godzinie	11.00	, w sali konferencyjnej
Walne Zgromadzenie Akcjonariuszy na dzień 29 czerwca 2006r. o godz.	1200	w Warszawie, Plac Żelaznej
Walne Zgromadzenie Akcjonariuszy na dzień 30 czerwca 2006r. na godz.	14.00	w Warszawie przy ulicy
Akcjonariuszy, które odbędzie się 30 czerwca 2006r.(piątek) o godz.	12.00	, które odbędzie się
	1000	w Biurze Spółki

Table C.2: Examples of training instances of the *Zgromadzenie godzina* annotation.

Left Context	Annotation	Right Context
o godz. 11:00 w odbędzie się w o godz. 12.00 w o godz. 10.00 w które odbędzie się w odbędzie się w o godzinie 12.00 w o godz. 12.00, w odbędzie się w o godz. 12.00 w o godzinie 11.00 w które odbędzie się w	siedzibie Spółki, przy ul. Jagiellońskiej 36 w Warszawie Sanockim Domu Kultury w Sanoku przy ul. Mickiewicza nr 24 siedzibie Spółki, w Kaliszu, ul. Majkowska 13 siedzibie Spółki w Sosnowcu przy ul. Gen. Grota Roweckiego 130 Warszawie, ul. Domaniewska 41, Budynek Taurus siedzibie Spółki w Kielcach przy ul. Zagnańskiej 27 Gdańsku przy ul. Łąkowej 39/44 biurze Spółki w Warszawie, pod adresem: Warszawa, ul. Waliców 11 Warszawie, w siedzibie Spółki przy Al. Armii Ludowej 26, Budynek Focus Częstochowie przy ul. Dąbrowskiego 18a, z następującym porządkiem siedzibie Spółki przy ul. Rzepakowej 2 Hotelu Ideal przy ulicy Bolesława Prusa 1 w Pruszkowie	. Porządek obrad: 1. Otwarcie o godzinie 12.00 . Porządek obrad: 1. Otwarcie .1. Otwarcie .W załączonym pliku . Porządek obrad: 1. Otwarcie z następującym porządkiem . Porządek obrad: 1. Otwarcie . Porządek , zwołał Zwyczajne Walne . Porządek obrad:

Table C.3: Examples of training instances of the *Zgromadzenie miejsce* annotation.

# List of Figures

1.1	The work flow with the main activities and artifacts . . . . .	4
2.1	Information Extraction System as a black box . . . . .	7
2.2	General Information Extraction System architecture. . . . .	9
2.3	Modules in Information Extraction System. . . . .	10
3.1	An example of precision versus recall plot. . . . .	12
3.2	An example of precision versus recall plot approximated to curve. . . . .	12
4.1	Annotation recognition concepts . . . . .	15
5.1	A process of manual patterns creation. . . . .	20
5.2	An example how the precision and recall may change in following iteration using top-down strategy . . . . .	21
5.3	An example how the precision and recall may change in following iteration using bottom-up strategy . . . . .	21
5.4	Examples of training instances for words function classification task and their symbolic representation. Example is based on [11]. . . . .	26
5.5	Example of IGTREE that contains training instances from Figure 5.4 . . . . .	26
5.6	The idea of bootstrapping . . . . .	29
6.1	Documents filtering by a keywords searching ( <i>Odkrywca</i> ). . . . .	33
6.2	The schema annotation ( <i>Odkrywca</i> ). . . . .	34
6.3	The component used to annotate documents through propagation ( <i>Odkrywca</i> ). . . . .	34
6.4	The component used to annotate documents ( <i>Odkrywca</i> ). . . . .	35
6.5	The component used to test pattern propagation ( <i>Odkrywca</i> ). . . . .	36
6.6	The Component used to test regular expression patterns ( <i>Odkrywca</i> ). . . . .	37
6.7	The regular expressions that match the <i>Zgromadzenie godzina</i> annotations. . . . .	38
6.8	The regular expression that match the <i>Zgromadzenie data</i> annotations. . . . .	38
6.9	The regular expressions that match the <i>Zgromadzenie miejsce</i> annotations. . . . .	38
6.10	Comparison of three ML methods . . . . .	39
6.11	The impact of the negative instances . . . . .	41
6.12	The sequence of the tokens used in the example. . . . .	42
6.13	The similarities for the <i>starting</i> and the <i>ending boundaries</i> for the example. . . . .	43
6.14	The maximum similarities for the <i>ending boundary</i> for the example. . . . .	44
6.15	The maximum similarities for the <i>starting boundary</i> for the example. . . . .	44
6.16	The result of finding the starting and ending boundaries for the annotation. . . . .	45
6.17	Standard MBL and modified MBL comparison . . . . .	45

# List of Tables

2.1	Example instance features. . . . .	8
4.1	Groups of tokens in terms of their structure . . . . .	16
4.2	Instance features . . . . .	18
5.1	The RoboTag best results. . . . .	23
5.2	Effect of feature selection and parameter optimization using Memory-Based Learning . . . . .	28
6.1	Comparison of three ML methods . . . . .	39
6.2	Number of positive and negative instances for the meeting date annotation . . . . .	40
6.3	The impact of the negative instances . . . . .	41
6.4	Performance of the modified Memory Based Learning methods . . . . .	44
7.1	Initial values of learning and testing parameters. . . . .	46
7.2	The range of the parameters values used in the experiment. . . . .	47
7.3	The summary of the best parameters configurations for the test annotation types. . . . .	53
7.4	The comparison of the Memory Based Learning and the regular expressions results. . . . .	54
C.1	Examples of training instances of the <i>Zgromadzenie data</i> annotation. . . . .	60
C.2	Examples of training instances of the <i>Zgromadzenie godzina</i> annotation. . . . .	61
C.3	Examples of training instances of the <i>Zgromadzenie miejsce</i> annotation. . . . .	62

# Bibliography

- [1] The "Message Understanding Conference (MUC)" web page [http://www-nlpir.nist.gov/related\\_projects/muc](http://www-nlpir.nist.gov/related_projects/muc).
- [2] The Polish Wordnet <http://plwordnet.pwr.wroc.pl>.
- [3] The oxford web dictionary. The Oxford Web Dictionary <http://www.askoxford.com>.
- [4] Rozporządzenie ministra finansów z dnia 19 października 2005 r w sprawie informacji bieżących i okresowych przekazywanych przez emitentów papierów wartościowych. Rozporządzenie Ministra Finansów z dnia 19 października 2005 r w sprawie informacji bieżących i okresowych przekazywanych przez emitentów papierów wartościowych, Dziennik Ustaw z 2005 r. Nr 209 poz. 1744, <http://www.abc.com.pl/serwis/du/2005/1744.htm>.
- [5] Weka 3 - data mining with open source machine learning software. Weka 3 - Data Mining with Open Source Machine Learning Software <http://www.cs.waikato.ac.nz/ml/weka/>.
- [6] James Allen. *Natural Language Understanding*. The Benjamin/Cummings Publishing Company, Inc., 1995.
- [7] Douglas E. Appelt and David J. Israel. Introduction to information extraction technology. IJCAI-99 Tutorial, 1999.
- [8] Scott W. Bennett, Chinatsu Aone, and Craig Lovell. Learning to tag multilingual texts through observation. *Proceeding of the Second Conference on Empirical Methods in NLP*, page 8, 1997.
- [9] Daniel M. Bikel, Richard Schwartz, and Ralph M. Weischedel. An algorithm that learns what is in a name. In *Machine Learning*, volume 34, pages 211–231, 1999.
- [10] Eric Brill. A simple rule-based part of speech tagger. *Proceedings of the Third Conference on Applied Computational Linguistics, Trento, Italy*, 1992.
- [11] Sabine Nicole Buchholz. *Memory-Based Grammatical Relation Finding*. PhD thesis, Tilburg University, 2002.
- [12] E. Burnside, D. Rubin, and R. Shachter. A bayesian network for mammography. In *Proceedings of the American Medical Informatics Association Symposium*, pages 16–100, 2000.
- [13] Wenliang Chen, Jingbo Zhu, Honglin Wu, and Tianshun Yao. *Computational Linguistics and Intelligent Text Processing*, chapter Automatic Learning Features Using Bootstrapping for Text Categorization, pages 571–579. Springer Berlin / Heidelberg, 2004.
- [14] Walter Daelemans and VPeronique Hoste. Evaluation of machine learning methods for natural language processing tasks. *Proceedings of the Third International Conference on Language Resources and Evaluation (LREC 2002)*, page 755–760, 2002.

- [15] Walter Daelemans and Antal van den Bosch. *Memory-Based Language Processing*. Cambridge University Press, 2005.
- [16] Alberto Lavelli Fabio Ciravegna and Giorgio Satta. Efficient full parsing for information extraction. *Atti dell'Incontro dei Gruppi di lavoro dell'Associazione Italiana per l'Intelligenza Artificiale (AI\*IA) di Apprendimento Automatico e Linguaggio Naturale (AA-NL n°97)*, 1997.
- [17] Xiaoshan Fang and Huanye Sheng. *Advances in Information Systems: Second International Conference, ADVIS 2002, Izmir, Turkey, October 23-25, 2002. Proceedings*, chapter Pattern Acquisition for Chinese Named Entity Recognition: A Supervised Learning Approach, pages 166–175. Springer Berlin / Heidelberg, 2002.
- [18] Dayne Freitag. Machine learning for information extraction from online documents. Master's thesis, School of Computer Science Carnegie Mellon University, 1996.
- [19] Anthony F. Gallippi. A synopsis of learning to recognize names across languages. In *Meeting of the Association for Computational Linguistics*, 1996.
- [20] Ralph Grishman and Beth Sundheim. Message understanding conference - 6: A brief history. *Proceedings of COLING-96*, 1:466–471, 1996.
- [21] Nancy Ide and Jean Véronis. Introduction to the special issue on word sense disambiguation: the state of the art. *Computational Linguistics*, 24:1–40, 1998.
- [22] Ron Kohavi. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *IJCAI*, pages 1137–1145, 1995.
- [23] George Krupka. Sra: Description of the sra system as used for muc-6. *Proceedings of Sixth Message Understanding Conference (MUC-6)*, page 15, 1995.
- [24] A. Kupść., A. Marciniak, A. Mykowiecka, J. Piskorski, and T. Podsiadły-Marczykowska. Information extraction from mammographic reports. In *KONVENS 2004, Osterreichische Gesellschaft fur Artificial Intelligence*, pages 113–116, 2004.
- [25] Anh-Cuong Le, Akira Shimazu, and Le-Minh Nguyen. *Computer Processing of Oriental Languages. Beyond the Orient: The Research Challenges Ahead*, chapter Investigating Problems of Semi-supervised Learning for Word Sense Disambiguation, pages 482–489. Springer Berlin / Heidelberg, 2006.
- [26] Seungwoo Lee, , and Gary Geunbae Lee. *Information Retrieval Technology*, chapter A Bootstrapping Approach for Geographic Named Entity Annotation, pages 178–189. Springer Berlin / Heidelberg, 2005.
- [27] Timothy Robert Leek. Information extraction using hidden markov models. Master's thesis, University of California, San Diego, 1997.
- [28] K. Lerman, A. Plangprasopchok, and C. A. Knoblock. Semantic labeling of online information sources. In *International Journal on Semantic Web and Information Systems, Special Issue on Ontology Matching*, 2007.
- [29] Derwojedowa Magdalena, Piasecki Maciej, Szpakowicz Stanisław, and Zawisławska Magdalena. Polish wordnet on a shoestring. In W G. Rehm, A. Witt, and L. Lemnitzer, editors, *Proceedings of Biannual Conference of the Society for Computational Linguistics and Language Technology*, pages 169–178. Universität Tübingen, 2007.

- [30] Fien De Meulder and Walter Daelemans. Memory-based named entity recognition using unannotated data, 2003.
- [31] Tom M. Mitchell. *Machine Learning*. McGraw-Hill Science/Engineering/Math, 1997.
- [32] Vincent Ng and Claire Cardie. Improving machine learning approaches to coreference resolution. *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 104–111, 2002.
- [33] Anselmo Peas, Felisa Verdejo, and Julio Gonzalo. Corpus-based terminology extraction applied to information access. *Proceedings of the Corpus Linguistics 2001 conference*, pages 458–465, 2001.
- [34] Maciej Piasecki and Grzegorz Godlewski. Reductionistic, tree and rule based tagger for polish. *Kłopotek, M.A., Wierzchon, S.T., Trojanowski, K., eds. Proceedings of Intelligent Information Processing and Web Mining 2006*, page 10, 2006.
- [35] Jakub Piskorski. *Intelligent Media Technology for Communicative Intelligence*, chapter Named-Entity Recognition for Polish with SProUT, pages 122–133. Springer Berlin / Heidelberg, 2005.
- [36] J.M. Ponte and W.B. Croft. Text segmentation by topic. *Proceedings of the First European Conference on research and advanced technology for digital libraries*, pages 120–129, 1997.
- [37] Sameer Pradhan, Kadri Hacioglu, Wayne Ward, James H. Martin, and Daniel Jurafsky. Semantic role chunking combining complementary syntactic views. In *Proceedings of the 9th Conference on Computational Natural Language Learning (CoNLL)*, pages 217–220, 2005.
- [38] Sameer Pradhan, Wayne Ward, Kadri Hacioglu, James H. Martin, and Daniel Jurafsky. Semantic role labeling using different syntactic views. In *Annual Meeting of the ACL Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, 2005.
- [39] Adam Przepiórkowski. *The IPI PAN Corpus: Preliminary version*. Institute of Computer Science, Polish Academy of Sciences, Warsaw, 2004.
- [40] Ellen Riloff. Automatically constructing a dictionary for information extraction tasks. *National Conference on Artificial Intelligence*, pages 811–816, 1993.
- [41] Ellen Riloff. Automatically generating extraction patterns from untagged text. *Thirteenth National Conference on Artificial Intelligence (AAAI-96)*, page 1044–1049, 1996. Portland, OR.
- [42] Ellen Riloff and Rosie Jones. Learning dictionaries for information extraction by multi-level bootstrapping. *Proceedings of the Sixteenth National Conference on Artificial Intelligence (AAAI-99)*, page 6, 1999.
- [43] Ellen Riloff, Janyce Wiebe, and Theresa Wilson. Learning subjective nouns using extraction pattern bootstrapping. *Proceedings of CONLL-03, 7th Conference on Natural Language Learning*, pages 25–32, 2003.
- [44] Kiyoshi Sudo, Satoshi Sekine, and Ralph Grishman. An improved extraction pattern representation model for automatic ie pattern acquisition. *CiteSeer*, page 8, 2003.

- [45] Antal van den Bosch, Sander Canisius, Iris Hendrickx, Walter Daelemans, and Erik Tjong Kim Sang. Memory-based semantic role labeling: Optimizing features, algorithm, and output. page 4, 2004.
- [46] Marc Vilain and David Day. Finite-state phrase parsing by rule sequences. In *Proceedings of COLING*, 1996.
- [47] Min Wu, Xiaoyu Zheng, Michelle Duan, Ting Liu, and Tomek Strzalkowski. Question answering by pattern matching, web-proofing, semantic form proofing. *TREC 2003*, pages 578–585, 2003.
- [48] Shihong Yu, Shuanhu Bau, and Paul Wu. Description of the kent ridge digital labs system for muc-7. In *Proceedings of the MUC-7*, 1998.
- [49] Li Zhang, Yue Pan, and Tong Zhang. Focused named entity recognition using machine learning. In *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 281–288. ACM Press, 2004.