ZÁPADOČESKÁ
UNIVERZITA
V PLZNI

Named Entity Recognition

PhD Study Report

Michal Konkol

Technical Report No. DCSE/TR-2012-04
June, 2012

Distribution: Public

Copies of this report are available on
http://www.kiv.zcu.cz/publications/
or by surface mail on request sent to the following address:

University of West Bohemia in Pilsen
Department of Computer Science and Engineering
Univerzitní 8
30614 Pilsen
Czech Republic

Contents

1 Introduction

"The lurking suspicion that something could be simplified is the world's richest source of rewarding challenges."
Edsger Dijkstra

Named Entity Recognition (NER) is one of the important parts of Natural Language Processing (NLP). NER is supposed to find and classify expressions of special meaning in texts written in natural language. These expressions range from proper names of persons or organizations to dates and often hold the key information in texts.

NER can be used for different important tasks. It can be used as a self-standing tool for full-text searching and filtering. Also it can be used as a preprocessing tool for other NLP tasks. These tasks can take advantage of marked Named Entities (NE) and handle them separately, which often results in better performance. Some of these tasks are Machine Translation, Question Answering, Text Summarization, Language Modelling or Sentiment Analysis.

NER task was firstly introduced at MUC-6 in 1995. Since that time it has moved from rule-based systems to statistical systems with variety of advanced features. The state-of-the-art performance is around 90% for English and 70% for Czech. The performance for other languages greatly varies depending on properties of a given language. It is thus very important to find new approaches to fill this gap in performance between different languages. Following example shows a typical output of a NER system.

```
<organization>The European Union</organization> was formally
established when the <other>Maastricht Treaty</other> came
into force on <date>1 November 1993</date>.
```

1.1 Outline

The second chapter will be devoted to measures for NER. We believe that it is important to define the evaluation techniques before some results are presented.

The third chapter describes the factors that may influence the performance of NER system. These factors are usually given and independent on the method. It is important that the results must be interpreted in context and not only as absolute numbers.

The fourth chapter introduces the methods that are used for NER. For each of these methods one or two examples of a NER system are given.

The fifth chapter covers the most used features for NER. The features seems to have at least the same importance as the methods.

The last chapter summarizes the open challenges of NER task.

2 Measures

"Measure what is measurable, and make measurable what is not so."
Galileo Galilei

This chapter will be devoted to performance measurement. In any area of research it is important to evaluate and compare results of new methods. There is thus a need to use some objective measure (or measures), which would well cover the purpose of the research.

Unlike other NLP tasks (e.g. Machine Translation) NER uses only one standard method of measurement, which is generally accepted. This method uses three metrics to describe the performance of NER system, each for different aspect of the task. These metrics are called precision, recall and F-measure (also F-score or F_1 score).

We will define these measures on a general classification of objects into two classes; positive and negative. Then there exist four following classes of classification results.

- Positive (P) - positive object marked as positive.
- Negative (N) - negative object marked as negative.
- False positive (FP)- negative object marked as positive.
- False negative (FN) - positive object marked as negative.

This is well shown on fig. 2.1, where the curves show the distribution of positive and negative objects, the dotted line shows the decision threshold of classifier. In the areas marked as FN and FP are some objects marked incorrectly.

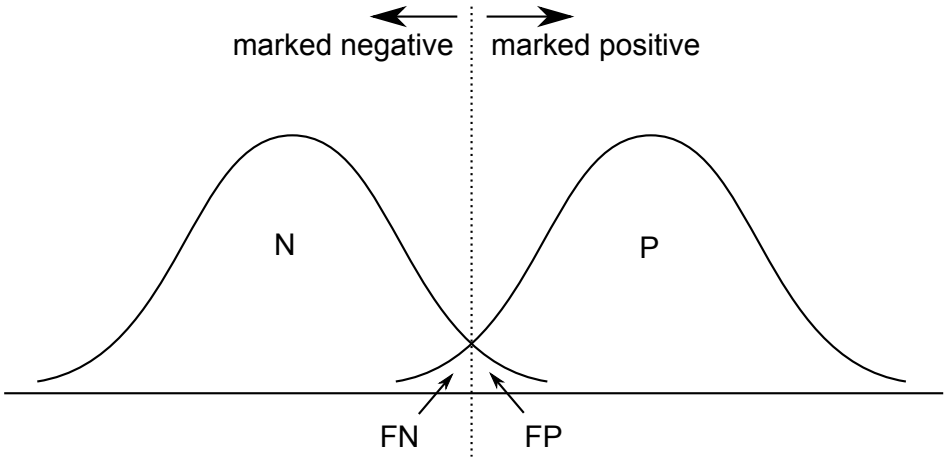


Figure 2.1: Precision and recall

Now we can easily define precision, recall and F-measure as follows.

$$\text{Precission} = \frac{P}{P + FP}$$

$$\text{Recall} = \frac{P}{P + FN}$$

$$\text{F-measure} = \frac{2P}{2P + FP + FN}$$

Precision is a measure of trust, that the objects marked as positive are really positive. Recall is a measure of trust, that all the positive objects are marked. It is obvious that precision and recall describe different aspects of results. Moreover, these measures are competing. As shown on fig. 2.2, if the decision threshold is moved to the left, there will be fewer FN objects and more FP objects, resulting in high recall and lower precision. This is important in evaluation of a classifier, because high recall (resp. precision) classifier can be better for various tasks. F-measure is a harmonic mean between precision and recall and is something like overall perspective.

The last piece of information needed for NER performance evaluation is to define, what is counted as P, N, FP and FN. This definition slightly differs between NER conferences. The following sections will describe evaluation

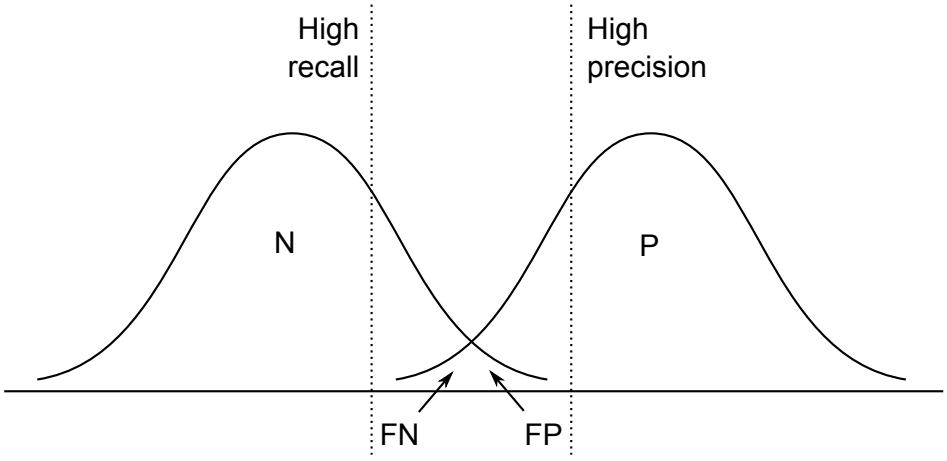


Figure 2.2: Precision and recall change

methods used on major NER conferences and also the evaluation method used in this thesis.

2.1 MUC-6 Evaluation

The NER task was introduced at MUC-6[1]. So the initial evaluation technique was defined at this conference. The choice of evaluation metrics was based on other information retrieval tasks. Since that time precision, recall and F-measure are used as a standard in NER.

At MUC-6 span (or text) and type of the entity was handled separately. The type is counted as correct, if it is the same as the original type and the span overlaps the entity. The text is considered correct, if it matches the original text of the entity. The text comparison involves operations like trimming or removing unimportant parts (ltd., the, etc.). Each entity can have an alternative text, which is also considered as correct.

Three numbers were counted for both type and span: COR (correct answers), POS (number of original entities) and ACT (number of guesses). The overall results of the system were acquired by adding these number for type and span together. The precision, recall and F-measure were then computed in a standard way using these numbers.

The evaluation techniques for all MUC tasks are covered in The Message Understanding Conference Scoring Software User's Manual [1].

2.2 CoNLL Evaluation

The CoNLL 2002[2] and 2003[3] have used an exact match evaluation. The entity is considered correct only if it has exactly the same span and type.

The advantage of this method is, that it is clear, simple and gives a lower estimate of the evaluated system. The disadvantage is, that in some cases it is too strict. If the original entity is "The United States" and "United States" is marked by the system, then "The United States" is considered as FN and "United States" as FP. The result is, that the system is penalized in two ways for almost good answer.

2.3 ACE Evaluation

The Automatic Content Extraction program consists of various NLP tasks. There are two tasks directly focused on NER, Entity Detection and Tracking (EDT)[4] and Time Expression Recognition and Normalisation (TERN)[5]. Both tasks extends the standard definition of NER tasks with deeper level of detail.

The evaluation of EDT task did not used standard metrics. The evaluation is based on a special scoring system, where each type of error and also each type of entity has different weight. The scoring system is very complex. On one hand, it can be adjusted and used to properly evaluate systems regarding various needs. On the other hand, the weights must be the same to compare two systems and it is hard to get direct feedback.

[1] http://www-nlpir.nist.gov/related_projects/muc/muc_sw/muc_sw_manual.html

2.4 Our evaluation

Each entity is defined by two attributes: type and span. Both these attributes are important, but in many cases, you prefer to have correct type. Sometimes the span of entity is hard to guess even for humans. This can be exemplified on entities in table 2.1, where not correctly marked span gives important information.

Entity	Marked as
Západočeská univerzita v Plzni	Západočeská univerzita
University of West Bohemia in Pilsen	*University of West Bohemia*
Kongres Spojených států amerických	Kongress
United States Congress	Congress
IBM Česká republika	IBM
IBM Czech Republic	*IBM*

Table 2.1: Examples of entities marked with not correct span, which can give a valid information.

This is the motivation to use extended model, where the partially good span can be taken into account. In this thesis, we categorize entities into the following bins. It is not necessary to keep track of N, because it is not needed for our measures.

- Correct - the entity is marked on correct span with correct type.
- Partially correct - the entity is marked with correct type, but the span is not exact.
- Not correct - something is marked, but it is not an entity.
- Not marked - entity is not marked.

We then use these bins to compute two versions of all previously mentioned measures. These versions differ in the way, how the bins are mapped to P, FP and FN. In both versions correct is mapped to P, not correct to FP and not marked to FN. The first version is strict and maps the partially good to FP. The second version is lenient and maps partially good to P. All is summarized in the following formulas.

Strict

$$\text{Precission} = \frac{Correct}{Correct + Not\ correct + Partially\ correct}$$

$$\text{Recall} = \frac{Correct}{Correct + Not\ marked}$$

$$\text{F-measure} = \frac{2 \cdot Correct}{2 \cdot Correct + Not\ correct + Not\ marked + Partially\ correct}$$

Lenient

$$\text{Precission} = \frac{Correct + Partially\ correct}{Correct + Not\ correct + Partially\ correct}$$

$$\text{Recall} = \frac{Correct + Partially\ correct}{Correct + Not\ marked}$$

$$\text{F-measure} = \frac{2 \cdot (Correct + Partially\ correct)}{2 \cdot (Correct + Partially\ correct) + Not\ correct + Not\ marked}$$

3 Performance influencing factors

"Success is a science; if
you have the conditions,
you get the result."
Oscar Wilde

There are many factors that can radically change the performance of a NER system. The most important of these factors are the language and domain of the processed texts and the information we are looking for.

3.1 Language

The language itself is obviously one of the most important factors. The first systems based on rules were build for a specific language and it was not possible to easily alter them to different language. With the advent of systems based on machine learning, it was possible to choose features independent on the language and use the system for different languages. The performance of the systems is significantly affected by the language and for some languages the difference in performance is more then 20% [6].

The majority of systems were logically created for English. But many languages have at least some experiments with state-of-the-art methods. The results for various languages are presented in tab. 3.1, but keep in mind that these results are affected by several conditions.

We are obviously interested in Czech. So far, there are only two published results [10][15]. The best performance for all entities using similar level of detail as CoNLL is F-measure 71%. There is a big gap between state-of-the-art English and Czech NER. Even for many other languages the difference is quite big.

Language	F-measure
Arabic [7]	79.21 %
Bulgarian [8]	89 %
Chinese [9]	≅ 90 %
Czech [10]	71 %
Dutch [11]	77.05 %
English [12]	88.76 %
German [12]	72.41 %
Greek [13]	71–94 %
Hungarian [14]	91.95 %
Spanish [11]	81.39 %

Table 3.1: Results of NER for various languages.

3.2 Domain

The domain of corpora can highly influence the performance of NER system. Some domains seems to be easier for NER then others, e.g. news articles and texts from social networks.

The systems are usually trained on one domain and it would be desirable to directly use them for other domains. Unfortunately, a fundamental performance degradation was detected if the domains are slightly different. Drop in F-measure from 90.8% (CoNLL) to 64.3% (Wall Street Journal) was reported in [16]. Similar performance degradation was reported in [17] for NER trained on MUC-6 corpus and used for more informal texts like emails. The domain adaptation problem is one of the challenges of NER [18].

3.3 Searched entities

Another important aspect are the types of entities we are looking for. Some categories of NEs are easier to find then others, e.g. countries are easier then organizations [2][3]. Of course, this depends on the definition of the class.

For example the datetime class can be defined very strictly to contain only absolute dates (2007; 5.3.2001; June 5, 2004) or it can include relative dates (next Saturday, in December), obviously the second case is harder. This can be applied to other classes more or less.

There also exist different levels of detail. On one hand, some authors use coarse-grained categories. The annotation scheme defined at MUC-6[1] has three types of NEs, where each category has two or three subtypes. At CoNLL[2][3], there were only four categories. On the other hand, some authors have defined many detailed categories. A finer-grained categories were used by BBN for Question Answering[19]. A hierarchy of 150 (later 200) categories was proposed by [20]. The Czech named entity corpus defines ten categories and 62 subcategories [15]. Generally, it is harder to classify entities into more categories [20][15].

The common NE categories are Person, Organization, Location (GPE), Date (and time), Numbers (of different kinds) and Miscellaneous. Another branch of NER is focused on biology and thus uses categories like Protein, DNA etc. [21]

4 Methods

"An algorithm must be seen to be believed."
Donald Knuth

Many different approaches were used for NER. This chapter is an attempt to choose the most important methods and describe their usage, training and other properties.

4.1 Method division

There are various aspects which can be used to divide or describe the methods. The divisions presented in the following paragraphs are compilation of notations used in NER or generally NLP field. They are not rigid, but they should give some idea about the methods.

All the methods have two basic development steps, creation and usage. The first division is based on creation phase. We say that the system is *handcrafted*, if all the parameters (in this meaning parameter can be even a rule) of the system are made or set by human. The system is denoted as *machine learning* system, if the parameters are somehow estimated by a computer.

The machine learning systems can be further divided based on the data they need to find the parameters. If the system needs corpus with already marked entities, then the system uses *supervised learning*. If the system needs only small amount of marked examples and then tries to improve the performance using unmarked text, then it uses *semi-supervised learning*. The last option is *unsupervised learning* which estimates the parameters from unmarked text.

The methods can be also divided by their usage to *deterministic* and *stochastic*. The difference between these groups is simple, the stochastic models are based on probability distributions while the deterministic are not. The difference can be seen on the results of these methods. The stochastic methods assign a set of labels and their probabilities for each word, while the deterministic methods assign for each word only one label.

There is also a term *rule-based* system. This term suggests another division based on how exactly is the NER task executed. Rule-based systems recognizes NEs by applying rules. The other category would be for example classification-based, but this term is not commonly used.

In many cases not all of the presented terms are mentioned. For example when a rule-based system is mentioned, it is automatically considered as hand-crafted deterministic rule-based system. Similarly if we talk about a system based on some classification method, we consider that it uses some machine learning method to find the parameters for the classifier.

4.2 Rule-based Methods

4.2.1 Dictionaries

The simplest method which can be used are dictionaries. NER based on dictionaries tries to find NE in the dictionary for each word (or group of words). If it finds some NE, then it is marked. Dictionaries of NEs are often called *gazetteers*.

The performance of this method corresponds to its simplicity. Even a very large gazetteers contains only a small portion of all used NEs. NEs are also used in various forms (e.g. Pilsner University), not only the basic form (e.g. University of West Bohemia), so it is necessary to enumerate all possible forms in the dictionary. For inflectional languages it is necessary to enumerate all word forms or to use lemmatization or stemming.

This method is not usually used separately, but is often used as part in more complex systems. Extended version of this method was used in [22].

4.2.2 Regular Expressions

One of the most widely used tools for text processing (not only NER) are regular expressions (RE). REs are a grammar classified as regular in Chmosky hierarchy. That means that they can be processed by finite state automaton and that their processing is very fast. The grammar itself slightly differs from one implementation to another and will not be described here.

REs were used as a part of many systems and some of the simpler rule based systems can be built only on REs.

4.2.3 Context Free Grammars

Context Free Grammars (CFG) are more general level of grammar than regular expressions in Chomsky hierarchy. CFG can create more complex rules then RE and thus have advantage for rule-based systems. On the other side, the complex rules written in CFG cannot be applied as machine learning feature without loosing adaptability. The more complex rules tend to be more dependent on a particular domain or preprocessing tool (e.g. tokenizer).

CFG have been used in many systems in the rule based era of NER. Examples of these systems can be found on MUC-6 and MUC-7 conferences [23][24].

4.3 Statistical Methods

Statistical methods for NER are modelling the probability distribution $p(\mathbf{y}|\mathbf{x})$, where $\mathbf{y}$ is the sequence of NE classes and $\mathbf{x}$ is the sequence of words. Some methods classify each word separately, other methods are classifying the whole sequence.

There are two different approaches to classification called generative and discriminative. The generative classifiers are based on the Bayes theorem (4.1) and are modelling $p(x|y)$ and $p(x)$. The typical generative classifier is Naive Bayes. The discriminative approach models directly $p(y|x)$ and one of the typical examples is Maximum Entropy.

$$p(y|x) = \frac{p(x|y)p(x)}{p(y)} \tag{4.1}$$

It is also necessary to adopt some model for handling multi word NEs. All the described methods assign probabilities of labels to each word. If there are two words with the same assigned class, then by using simple labelling (one label for one class) it is not possible to distinguish, if the second word is start of another entity or continuation of the first. An example in Czech follows (in this case, the English translation does not have the same problem).

Ukázal dopis od Petra Pavlovi.
He showed the letter from Peter to Paul.

The easiest way to handle it is to ignore it. It is not very common, that one entity of the same type directly follows another (of course it depends on language). So it is possible to mark all consecutive occurrences as one entity. Another way is to encode one entity type with multiple labels. A common way is to use BIO (or IOB) model [2][25], where B stands for begin, I for inside and O for outside. All NE classes then have two labels (e.g. person_B, person_I, city_B) and the O is for non-entity class. Some authors are extending BIO model with E (end) tag [26].

The following sections will describe the most used classification methods.

4.3.1 Hidden Markov Models

Markov Models are modelling a Markov process and are based on a state graph. Markov process is a stochastic process for which the state transmission probability distribution depends only on the present state. Hidden Markov Models are modelling a process, where the states are not directly observable. Hidden Markov Model is fully described by following properties.

- $X = \{x_1, \ldots, x_n\}$ – Set of observations.
- $Y = \{y_1, \ldots, y_m\}$ – Set of states.
- y_0 – Initial state.

- $p(y_k|y_{k-1})$ – The state transition probability distribution.
- $p(x|y_k, y_{k-1})$ – The observation emission probability distribution.

For NER the states are the NE classes and observations are words. The Viterbi algorithm is then used to find the sequence of states (NE classes) with highest probability. The Baum-Welch algorithm can be used to improve the parameters of HMM using unmarked texts. A typical example of a HMM system is [27]. HMM were only used in combination with other classifier at CoNLL 2003 [3]. Recent systems often prefer Conditional Random Fields, which have similar ability to handle sequences.

4.3.2 Support Vector Machines

The Support Vector Machines will be described in the simplest possible way following the original description [28]. We will assume only binary classifier for classes $y = -1, 1$ and linearly separable training set $\{(x_i, y_i)\}$. It means that the conditions (4.2) are met.

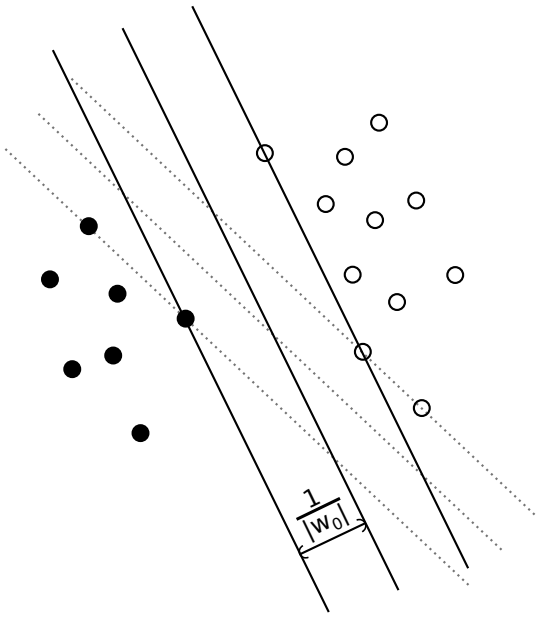


Figure 4.1: Optimal (and suboptimal) hyperplane.

$$\begin{array}{rcrll} \mathbf{w}\cdot\mathbf{x_i}+b & \leq & -1 & \text{if} & y_i = -1 \\ \mathbf{w}\cdot\mathbf{x_i}+b & \geq & 1 & \text{if} & y_i = 1 \end{array} \tag{4.2}$$

Thanks to the choice of y labels, we can rewrite the conditions (4.2) in one equation (4.3) that covers all objects in the training set.

$$y_i \cdot (\mathbf{w}_0 \cdot \mathbf{x} + b_0) \geq 1 \tag{4.3}$$

SVM are based on the search of the optimal hyperplane (4.4) that separates both classes with the maximal margin. We need to measure the distance between the classes in the direction given by $\mathbf{w}$. The formula for this is (4.5).

$$\mathbf{w}_0 \cdot \mathbf{x} + b_0 = 0 \tag{4.4}$$

$$d(\mathbf{w}, b) = \min_{x;y=1} \frac{\mathbf{x}\cdot\mathbf{w}}{|\mathbf{w}|} - \max_{x;y=-1} \frac{\mathbf{x}\cdot\mathbf{w}}{|\mathbf{w}|} \tag{4.5}$$

The optimal hyperplane maximizes the distance $d(\mathbf{w}, b)$ and can be expressed as (4.6). Therefore the parameters $\mathbf{w}_0$ and b_0 can be found by maximizing $|\mathbf{w}_0|$. For better orientation the optimal hyperplane (and also one suboptimal) is shown on figure 4.1.

$$d(\mathbf{w}_0, b_0) = \frac{2}{|\mathbf{w}_0|} \tag{4.6}$$

The classification is then done by looking on which side of the hyperplane the object is. Mathematically written as (4.7).

$$l(\mathbf{x}) = \text{sign}(\mathbf{w}_0 \cdot \mathbf{x} + b_0) \tag{4.7}$$

The best presented result for Czech NER was achieved with SVM [10]. SVM are also often used in systems which combine multiple classifiers [29], because they are able to generalize very well and the principle is quite different from other methods.

4.3.3 Maximum Entropy

The most uncertain probability distribution is the uniform one, because then everything has the same probability. If some constraints are added to the model, the model has to be modified to satisfy these constraints, but there is infinite number of probability distributions satisfying them. The principle of maximum entropy [30] says that the best distribution is the most uncertain one subject to the constraints. A constraint is given in the following form.

$$E_{\tilde{p}}(f_i) = E_p(f_i) \tag{4.8}$$

Where $E_{\tilde{p}}(f_i)$ is the expected value of feature f_i observed from data and $E_p(f_i)$ is the expected value of maximum entropy model. The features are in the following form.

$$f(x,y) = \begin{cases} 1 & \text{if } y \text{ is PERSON and } x \text{ starts with capital letter} \\ 0 & \text{otherwise} \end{cases} \tag{4.9}$$

Where parameter y is a class of a NE and x is the classified object, in our case word (or lemma). It is not necessary to have only binary features, but all feature values have to be positive.

For named entity recognition we want to find conditional probability distribution $p(y|x)$, where y is class of word and x are words used for classification. Following the principle of maximum entropy we want $p(y|x)$ to have maximum entropy of all possible distributions.

$$\arg\max_{p(y|x)} H(p(y|x)) = -\sum_{x\in\Omega} p(x) \sum_{y\in\Psi} p(y|x) \log p(y|x)$$

Because $H(p(y|x))$ is a concave function, there is only one maximum. It can be shown by Lagrange method [31] that the best probability distribution has the following parametric form.

$$p(y|x) = \frac{1}{Z(x)} \exp \sum_{i=1}^{n} \lambda_j f_j(x,y) \tag{4.10}$$

$$Z(x) = \sum_{y} \exp \sum_{i} \lambda_i f_i(x, y)$$

$Z(x)$ is only a normalizing factor which ensures that $p(y|x)$ is a probability distribution. $\Lambda = \{\lambda_0, \ldots, \lambda_n\}$ are parameters and have to be set properly to gain maximum entropy. The parameters are found using Generalized Iterative Scaling[32], Improved Iterative Scaling[31], Limited memory BFGS[33][34] or other minimization method[35].

ME is one of the most popular and successful methods. On CoNLL 2003 five of 16 systems used ME [3]. A typical pure ME classifier is presented in [36].

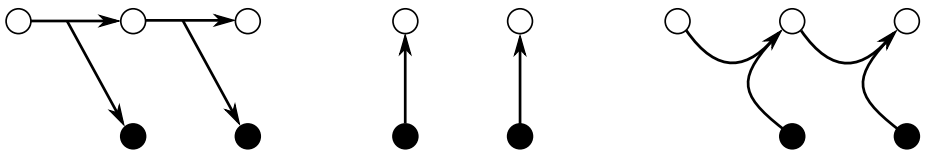
4.3.4 MEMM

A Maximum Entropy Markov Model [37] is a combination of Maximum Entropy and Markov Models. The motivation is to get the best of both methods. The HMM's ability to find sequences and ME's ability to use a lot of diverse features. In other words, we want to model probability distribution $p(y|x, y')$ using the Maximum Entropy principle. This can be achieved by splitting the problem into probability distributions $p_{y'}(x|y)$, creating one ME classifier (4.11) for each previous class.

$$p_{y'}(x|y) = \frac{1}{Z_{y'}(x)} \exp \sum_{i=1}^{n} \lambda_j f_j(x, y) \tag{4.11}$$

The transformation of dependencies can be seen on fig. 4.2. The black points are observations and the white ones are states or labels. The HMM uses a generative approach and models two distributions for state transition and observation emission. The ME uses a discriminative approach, but cannot exploit the Viterbi or Baum-Welch algorithm. The MEMM is a discriminative method that can use altered Viterbi and Baum-Welch method.

So far, MEMM seems to have only advantages. There is also one very important disadvantage that is data sparseness. While in ME data are used to train one classifier, in MEMM the data has to be split and used for $|y|$ classifiers, where $|y|$ is number of labels.



(a) (b) (c)

Figure 4.2: Dependency graphs for (a) HMM, (b) ME nad (c) MEMM.

4.3.5 Conditional random fields

Conditional Random Fields (CRF) were introduced in [38]. The idea of CRF is strongly based on ME. The difference is that ME classifies one instance after another while CRF classify the whole sequence at once. Mathematically written, ME estimates $p(y_i|x_i)$ for $i = 1, \ldots, n$ and CRF estimate $p(\mathbf{y}|\mathbf{x})$ where $\mathbf{y}$ and $\mathbf{x}$ are n-dimensional vectors. The probability $p(\mathbf{y}|\mathbf{x})$ can be computed using matrices and a variant of forward-backward algorithm.

$$p(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \exp\left(\sum_j \lambda_j f_j(\mathbf{y}, \mathbf{x})\right) \tag{4.12}$$

The features are extended and can use the previous state in contrast to ME. Two types of features are used, state s and transition t. The state features can be considered as a subset of transition features, where the previous state is not used, and a general feature definition (4.14) can be used.

$$\{f(y_{i-1}, y_i, \mathbf{x}, i)\}_1^{k+l} = \{s(y_i, \mathbf{x}, i)\}_1^k \cup \{t(y_{i-1}, y_i, \mathbf{x}, i)\}_1^l \tag{4.13}$$

Following the dependency graphs from 4.2 we can see the dependency graph for CRF on fig. 4.3. The parameters of this model are found using similar methods like ME, e.g. L-BFGS.

Initial tests on the NER task was done in [39]. Since their introduction, many systems used them with very good results[8][7]. CRF are considered to be the most successful classification method for NER.

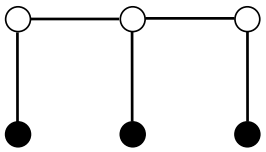


Figure 4.3: Dependency graph for CRF.

4.4 Semi- and unsupervised methods

Semi-supervised methods are usually based on a small set of training examples for each NE class. These examples are then used to find patterns or contexts around them. New NEs are then found using these patterns and contexts. This process is then iterated and the new NEs are always used as examples for the next step. This method is called *bootstrapping*.

Unsupervised methods are typically based on clustering. The clustering methods can use contexts, patterns etc. and rely on a large corpora. It is often difficult to find the boundary between these two classes.

One of the systems using bootstrapping is [22]. It uses small set of NEs for each class as a seed for automatic gazetteer generation and list-lookup strategy (extended by heuristics). Another system in this category is [40].

As the previous examples show, semi-supervised methods are often related with the problem of automatic gazetteer generation. There are many various ways how to deal with this task [41][42].

4.5 Method combinations

In this section we will talk about how to combine different classification methods to improve the results of a solo classifier. In [13] a two passes algorithm is used to identify NEs. In both phases separate classifier is used for each class. The results of the first phase are used as features in the second phase.

Some authors have used a method similar to the one described earlier

[43][44]. The method is called *stacking* and its principle is that output of one classifier is taken as input to another one. Multiple classifier can be combined this way.

Another way to combine classifiers is *voting*. There are two types of voting. The first one [45] is called *majority voting* and uses the final results, i.e. one class of NE is assigned to each word. The second one [45] is called *weighted voting* and uses the raw output of classifiers, i.e. probabilities for all classes are assigned to each word. Both of these types can use weights α_i for classifiers $p_i(y|x)$. The α_i parameters can be found using some form of Expectation-Maximization algorithm.

$$p(y|x) = \alpha_1 p_1(y|x) + \ldots + \alpha_n p_n(y|x) \tag{4.14}$$

In [46] and [47] an ensemble of classifiers is used. All the classifiers uses the same classification method but different feature set. Genetic algorithms (GA) based classifier selection is used to find the best set of classifiers. GA are also used to choose between majority and weighted voting. Similar approach is also used in [29], but the problem is taken as multi-objective optimization.

In general, classifier combinations give better results then the individual classifiers. When dealing with NER task, it is advisable to use some combination of classifiers based on different approaches or feature sets.

5 Features

"What we observe is not nature itself, but nature exposed to our method of questioning."
Werner Heisenberg

If a machine learning approach to NER is used, the features are something like the senses for human. That is why choosing the right feature set has the highest importance. From the beginning of the NER task various features have been used. In the following paragraphs and sections we will try to describe some terms used in connection to features and the most important features.

At first we will focus on the context that the feature uses. Commonly, two categories are used, *local* and *global*. Local features use only a small neighbourhood of the classified word, while global features uses the whole document or corpus. Sometimes some meta-information about the document is also considered as global feature.

Another important property of the feature is a language dependency. The feature is *language independent*, if it can be used for another language without any changes. If we want to create a language independent NER, we obviously need to use only language independent features and methods.

Sometimes a term *external* is used to indicate a feature which uses an external system or information source (e.g. Wikipedia). Gazetteers and dictionaries are sometimes considered external, but more often have their own category, *dictionary* or *list-lookup* features.

Usually, the features are acquired for some small neighbourhood (window) of the classified word. This window is often $-2, \ldots, +2$ or $-3, \ldots, +3$.

5.1 Local features

5.1.1 Orthographic features

Orthographic features are based on the appearance of the word, e.g. the first letter is a capital letter, all letters are capital or the words consists of digits. These features are used very often [3], because they need only the word, are language independent and still very effective for many languages. Interesting approach for these features was presented in [16].

5.1.2 Affixes

Another language independent feature are affixes. Some types of NEs often share the same word ending or prefix. Only a small portion of affixes are meaningful and thus some kind of threshold or feature selection is needed to choose a reasonable number of affixes.

5.1.3 Word

A word itself can be used as a feature. In many cases all letters are converted to upper or lower case to capture a word at the start of a sentence as the same feature as in the middle[45].

5.1.4 Stemming and lemmatization

Stemming is a task which is trying to find the root of each word, resp. to remove all affixes. A stem can be used directly as a feature similarly to a simple word feature. Stemming can also improve the performance of other features, e.g. gazetteers.

Lemmatizetion is a task similar to stemming but the output is a lemma instead of a stem. Lemma is basic word form of a word often used as a dictionary entry.

The importance of stemming and lemmatization is influenced by the lan-

guage. For highly inflectional languages like Czech, stemming or lemmatization is almost a must because it is necessary to reduce the high number of different word forms.

5.1.5 N-grams

Two types of n-grams are used in NER. The less common type is a character n-gram. Character N-grams are used to capture the inner structure of a word. Affix feature described above can be considered a special case of character n-grams, where only n-grams at the beginning (resp. end) of the word are used.

The second type are word n-grams. Word n-grams are used to capture word sequences that are often NEs or that often have NEs in their neighbourhood. Unigrams are identical to word feature described above.

5.1.6 Part of speech and morphology

Morphological tags are a useful feature. Generally, NEs are most often nouns, adjectives and numbers. Other types like prepositions appear less frequently and some like verbs are rare. In inflective languages, morphological tags also give us a possibility to detect consecutive words in the same case which can improve the NE detection.

5.1.7 Patterns

Various types of patterns have been used in NER. The rule-based systems are based on patterns, but machine learning methods can also exploit patterns as features. For machine learning, it is good to use some method for automatic extraction of patterns [48][49]. Usage of more complex hand-made rules in machine learning systems tends to have a negative impact on the adaptability.

An interesting approach was presented in [50]. One of six categories is assigned for each word. Each category is represented by one character. The pattern is then a string of the category characters.

5.2 Global features

5.2.1 Previous appearance

Some authors use a previous appearance of an NE in the document. If NE is already marked in the text, new appearance of the same NE will be probably NE with the same class.

Sometimes this is not used as a feature but as a postprocessing step. After the classification step, all the found NEs are reviewed and the NE classes can be changed if they appeared in other class with higher probability.

5.2.2 Meta information

For some documents meta information is available. This meta information can consist of various things and some of them can be used as a features for NER. Good example of documents that can include meta information are news articles or emails. When dealing with news articles (resp. emails) it can be useful to use category of that article, its title (resp. subject) etc. However, usage of meta information has negative impact on adaptability, because the meta information will not be available or will be different for other domain or data source.

5.3 List-lookup features

5.3.1 Gazetteers

Gazetteers are lists of NEs. The opinions on gazetteers are mixed. Some authors stated that usage of gazetteers has not improved their results while other says it improved it significantly. The fact that gazetteers are used in many systems speaks for the second claim.

Systems with gazetteers are obviously loosing the possibility of direct use for another languages. There were attempts to mitigate this problem by unsupervised or semi-supervised gazetteer creation.

5.3.2 Trigger words

Trigger words are words which are not NEs, but are often in the neighbourhood of NEs. For example 'president' can be a trigger word for Person NE. List of trigger words can be automatically learned from corpora or can be made by hand (e.g. list of degrees).

5.4 External Features

5.4.1 Wikipedia

Wikipedia is a rich source on information, it is thus natural that some authors try to exploit it in NER. In [51][52] the authors use the categories of some word sequences as a feature for NER. The results are obviously dependent on the language of Wikipedia, because English has many times more articles than other languages.

Wikipedia have been used for another purposes then as a feature in NER. It was used as a automatically created corpus for NER [53] or for automatic creation of gazetteers [54]. Wikipedia is also often used as a source of information for Named Entity Disambiguation task, which is related to NER.

6 Future work

"Now this is not the end. It is not even the beginning of the end. But it is, perhaps, the end of the beginning."
Winston Churchill

Even though much effort was devoted to NER there are still challenging problems in this task. In this section we will try to enumerate them.

The first challenge is lower performance of Czech NER compared to English. The previous experiments show that Czech NER has significantly lower performance using the same methods and features like English. It is thus important to search new ways to improve it. We believe that solving the problem for Czech can also improve the results for other languages with similar problems.

Another challenge is multilingualism of NER. It is important to develop methods and features which works for all or at least many languages and do not decrease the performance of language dependent methods. All the main NER conferences addressed multilingualism. The proposed methods still have lower performance for some languages like Czech or German, this problem is thus connected with the one above.

It is also important to find some way to effectively adapt systems trained on one domain to other domains. This challenge can be extended if the new domain uses different set of classes than the original. Even though this is an important task, not many authors have addressed it.

Another area which needs further research are unsupervised and semi-supervised methods for NER.

One task is deeply connected to NER and it is Named Entity Disambiguation. This task follows the NER task and tries to connect all NEs which refer

to one object and to assign some standard form to it. It is significant help for other tasks which uses NER as preprocessing tool.

We also feel the importance of quality and reusable implementation of NER system, so it can be directly and effectively used for other projects and for commercial use. We will try to find or define a standard way or interface for NER.

6.1 Aims of Doctoral Thesis

- Develop new recognition methods and features to improve performance for Czech and other languages.
- Propose semi-supervised approaches to improve the adaptability of NER.
- Experiment with disambiguation on small subset of selected named entities.
- Create quality and reusable NER system, which will provide standard interfaces.

Bibliography

[1] Ralph Grishman and Beth Sundheim. Message understanding conference-6: a brief history. In *Proceedings of the 16th conference on Computational linguistics - Volume 1*, pages 466–471, Morristown, NJ, USA, 1996. Association for Computational Linguistics.

[2] Erik F. Tjong Kim Sang. Introduction to the conll-2002 shared task: language-independent named entity recognition. In *proceedings of the 6th conference on Natural language learning - Volume 20*, COLING-02, pages 1–4, Stroudsburg, PA, USA, 2002. Association for Computational Linguistics.

[3] Erik F. Tjong Kim Sang and Fien De Meulder. Introduction to the CoNLL-2003 shared task: language-independent named entity recognition. In *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003 - Volume 4*, CONLL '03, pages 142–147, Stroudsburg, PA, USA, 2003. Association for Computational Linguistics.

[4] G. Doddington, A. Mitchell, M. Przybocki, L. Ramshaw, S. Strassel, and R. Weischedel. The Automatic Content Extraction (ACE) Program–Tasks, Data, and Evaluation. *Proceedings of LREC 2004*, pages 837–840, 2004.

[5] L. Ferro, L. Gerber, I. Mani, B. Sundheim, and G. Wilson. TIDES 2005 Standard for the Annotation of Temporal Expressions. Technical report, MITRE, September 2005.

[6] Hai Leong Chieu and Hwee Tou Ng. Named entity recognition with a maximum entropy approach. In Walter Daelemans and Miles Osborne, editors, *Proceedings of CoNLL-2003*, pages 160–163. Edmonton, Canada, 2003.

[7] Yassine Benajiba, Mona Diab, and Paolo Rosso. Arabic named entity recognition using optimized feature sets. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '08, pages 284–293, Stroudsburg, PA, USA, 2008. Association for Computational Linguistics.

[8] Georgi Georgiev, Preslav Nakov, Kuzman Ganchev, Petya Osenova, and Kiril Simov. Feature-rich named entity recognition for bulgarian using conditional random fields. In *Proceedings of the International Conference RANLP-2009*, pages 113–117, Borovets, Bulgaria, September 2009. Association for Computational Linguistics.

[9] *On Using Ensemble Methods for Chinese Named Entity Recognition*, Proceedings of the Fifth SIGHAN Workshop on Chinese Language Processing, 2006.

[10] Jana Kravalová and Zdeněk Žabokrtský. Czech named entity corpus and svm-based recognizer. In *Proceedings of the 2009 Named Entities Workshop: Shared Task on Transliteration*, NEWS '09, pages 194–201, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics.

[11] Xavier Carreras, Lluís Màrques, and Lluís Padró. Named entity extraction using adaboost. In *Proceedings of CoNLL-2002*, pages 167–170. Taipei, Taiwan, 2002.

[12] Radu Florian, Abe Ittycheriah, Hongyan Jing, and Tong Zhang. Named entity recognition through classifier combination. In Walter Daelemans and Miles Osborne, editors, *Proceedings of CoNLL-2003*, pages 168–171. Edmonton, Canada, 2003.

[13] Georgios Lucarelli, Xenofon Vasilakos, and Ion Androutsopoulos. Named entity recognition in greek texts with an ensemble of svms and active learning. *International Journal on Artificial Intelligence Tools*, 16(6):1015–1045, 2007.

[14] Richárd Farkas, György Szarvas, and András Kocsor. Named entity recognition for hungarian using various machine learning algorithms. *Acta Cybern.*, 17(3):633–646, January 2006.

[15] Magda Ševčíková, Zdeněk Žabokrtský, and Oldřich Krůza. Named entities in czech: annotating data and developing ne tagger. In *Proceedings of the 10th international conference on Text, speech and dialogue*, TSD'07, pages 188–195, Berlin, Heidelberg, 2007. Springer-Verlag.

[16] Massimiliano Ciaramita and Yasemin Altun. Named-entity recognition in novel domains with external lexical knowledge. *Proceedings of the NIPS Workshop on Advances in Structured Learning for Text and Speech Processing*, (Section 00):0–3, 2005.

[17] Thierry Poibeau and Leila Kosseim. Proper Name Extraction from Non-Journalistic Texts. *Language and Computers*, pages 144–157, December 2001.

[18] Jing Jiang and ChengXiang Zhai. Exploiting domain structure for named entity recognition. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, HLT-NAACL '06, pages 74–81, Stroudsburg, PA, USA, 2006. Association for Computational Linguistics.

[19] Ada Brunstein. Annotation guidelines for answer types, 2002.

[20] S. Sekine, K. Sudo, and C. Nobata. Extended named entity hierarchy. In M. Gonzáles Rodríguez and C. Paz Suárez Araujo, editors, *Proceedings of 3^{rd} International Conference on Language Resources and Evaluation (LREC'02)*, pages 1818–1824, Canary Islands, Spain, May 2002.

[21] Dan Shen, Jie Zhang, Guodong Zhou, Jian Su, and Chew lim Tan. Effective adaptation of a hidden markov model-based named entity recognizer for biomedical domain. In *In: Proceedings of NLP in Biomedicine, ACL*, pages 49–56, 2003.

[22] David Nadeau, Peter D. Turney, and Stan Matwin. Unsupervised named-entity recognition: generating gazetteers and resolving ambiguity. In *Proceedings of the 19th international conference on Advances in Artificial Intelligence: Canadian Society for Computational Studies of Intelligence*, AI'06, pages 266–277, Berlin, Heidelberg, 2006. Springer-Verlag.

[23] K. Humphreys, R. Gaizauskas, S. Azzam, C. Huyck, B. Mitchell, H. Cunningham, and Y. Wilks. University of sheffield: Description of the lasie-ii system as used for muc-7. In *In Proceedings of the Seventh Message Understanding Conferences (MUC-7*. Morgan, 1998.

[24] Andrei Mikheev, Claire Grover, and Marc Moens. Description of the ltg system used for muc-7. In *In Proceedings of 7th Message Understanding Conference (MUC-7*, 1998.

[25] Lance Ramshaw and Mitch Marcus. Text Chunking Using Transformation-Based Learning. In David Yarovsky and Kenneth Church, editors, *Proceedings of the Third Workshop on Very Large Corpora*, pages 82–94, Somerset, New Jersey, 1995. Association for Computational Linguistics.

[26] Silviu Cucerzan and David Yarowsky. Language independent ner using a unified model of internal and contextual evidence. In *Proceedings of CoNLL-2002*, pages 171–174. Taipei, Taiwan, 2002.

[27] GuoDong Zhou and Jian Su. Named entity recognition using an hmm-based chunk tagger. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 473–480, Stroudsburg, PA, USA, 2002. Association for Computational Linguistics.

[28] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Mach. Learn.*, 20(3):273–297, September 1995.

[29] Asif Ekbal and Sriparna Saha. Multiobjective optimization for classifier ensemble and feature selection: an application to named entity recognition. *International Journal on Document Analysis and Recognition*, pages 1–24. 10.1007/s10032-011-0155-7.

[30] Silviu Guiasu and Abe Shenitzer. The principle of maximum entropy. *The Mathematical Intelligencer*, 7:42–48, 1985. 10.1007/BF03023004.

[31] Adam L. Berger, Vincent J. Della Pietra, and Stephen A. Della Pietra. A maximum entropy approach to natural language processing. *Comput. Linguist.*, 22:39–71, March 1996.

[32] J. N. Darroch and D. Ratcliff. Generalized iterative scaling for log-linear models. *The Annals of Mathematical Statistics*, 43(5):pp. 1470–1480, 1972.

[33] D. C. Liu and J. Nocedal. On the limited memory bfgs method for large scale optimization. *Math. Program.*, 45:503–528, December 1989.

[34] Jorge Nocedal. Updating Quasi-Newton Matrices with Limited Storage. *Mathematics of Computation*, 35(151):773–782, 1980.

[35] Robert Malouf. A comparison of algorithms for maximum entropy parameter estimation. In *proceedings of the 6th conference on Natural language learning - Volume 20*, COLING-02, pages 1–7, Stroudsburg, PA, USA, 2002. Association for Computational Linguistics.

[36] Hai Leong Chieu and Hwee Tou Ng. Named entity recognition with a maximum entropy approach. In *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003 - Volume 4*, CONLL '03, pages 160–163, Stroudsburg, PA, USA, 2003. Association for Computational Linguistics.

[37] Andrew McCallum, Dayne Freitag, and Fernando C. N. Pereira. Maximum entropy markov models for information extraction and segmentation. In *Proceedings of the Seventeenth International Conference on Machine Learning*, ICML '00, pages 591–598, San Francisco, CA, USA, 2000. Morgan Kaufmann Publishers Inc.

[38] John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, pages 282–289, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc.

[39] Andrew McCallum and Wei Li. Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons. In *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003 - Volume 4*, pages 188–191, Morristown, NJ, USA, 2003. Association for Computational Linguistics.

[40] Oren Etzioni, Michael Cafarella, Doug Downey, Ana-Maria Popescu, Tal Shaked, Stephen Soderland, Daniel S. Weld, and Alexander Yates. Unsupervised named-entity extraction from the web: an experimental study. *Artif. Intell.*, 165(1):91–134, June 2005.

[41] Jun ichi Kazama and Kentaro Torisawa. Inducing gazetteers for named entity recognition by large-scale clustering of dependency relations. In *ACL 2008, Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics, June 15-20, 2008, Columbus, Ohio, USA*, pages 407–415. The Association for Computer Linguistics, 2008.

[42] Marius Pasca, Dekang Lin, Jeffrey Bigham, Andrei Lifchits, and Alpa Jain. Organizing and searching the world wide web of facts - step one: the one-million fact extraction challenge. In *proceedings of the 21st national conference on Artificial intelligence - Volume 2*, AAAI'06, pages 1400–1405. AAAI Press, 2006.

[43] Radu Florian. Named entity recognition as a house of cards: classifier stacking. In *proceedings of the 6th conference on Natural language learn-*

ing - Volume 20, COLING-02, pages 1–4, Stroudsburg, PA, USA, 2002. Association for Computational Linguistics.

[44] Koji Tsukamoto, Yutaka Mitsuishi, and Manabu Sassano. Learning with multiple stacking for named entity recognition. In *Proceedings of CoNLL-2002*, pages 191–194, 2002.

[45] Z. Kozareva, O. Ferrández, A. Montoyo, R. Muñoz, A. Suárez, and J. Gómez. Combining data-driven systems for improving named entity recognition. *Data & Knowledge Engineering*, 61(3):449 – 466, 2007. <ce:title>Advances on Natural Language Processing</ce:title> <ce:subtitle>NLDB 05</ce:subtitle>.

[46] Asif Ekbal and Sriparna Saha. Classifier ensemble selection using genetic algorithm for named entity recognition. *Research on Language & Computation*, 8:73–99, 2010. 10.1007/s11168-010-9071-0.

[47] Bart Desmet and Veronique Hoste. Dutch named entity recognition using ensemble classifiers. In Eline Westerhout, Thomas Markus, and Paola Monachesi, editors, *Computational Linguistics in the Netherlands 2010 : selected papers from the twentieth CLIN meeting (CLIN 2010)*, pages 29–41. Landelijke Onderzoeksschool Taalwetenschap (LOT), 2010.

[48] Sujan Kumar Saha, Sudeshna Sarkar, and Pabitra Mitra. A hybrid feature set based maximum entropy hindi named entity recognition. In *Proceedings of the Third International Joint Conference on Natural Language Processing*, 2008.

[49] Partha Pratim Talukdar, Thorsten Brants, Mark Liberman, and Fernando Pereira. A context pattern induction method for named entity extraction. In *Proceedings of the Tenth Conference on Computational Natural Language Learning*, CoNLL-X '06, pages 141–148, Stroudsburg, PA, USA, 2006. Association for Computational Linguistics.

[50] Xavier Carreras, Lluís Màrquez, and Lluís Padró. A simple named entity extractor using adaboost. In Walter Daelemans and Miles Osborne, editors, *Proceedings of CoNLL-2003*, pages 152–155. Edmonton, Canada, 2003.

[51] Jun'ichi Kazama and Kentaro Torisawa. Exploiting Wikipedia as External Knowledge for Named Entity Recognition. In *Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 698–707, 2007.

[52] Alexander E Richman, Patrick Schone, and Fort George G Meade. Mining wiki resources for multilingual named entity recognition. *Computational Linguistics*, (June):1–9, 2008.

[53] Joel Nothman, Nicky Ringland, Will Radford, Tara Murphy, and James R. Curran. Learning multilingual named entity recognition from wikipedia. *Artificial Intelligence*, (0):–, 2012.

[54] A. Toral and R. Munoz. A proposal to automatically build and maintain gazetteers for Named Entity Recognition by using Wikipedia. *EACL 2006*, 2006.