NAMED ENTITY RECOGNITION IN TURKISH VIA INDUCTIVE LOGIC PROGRAMMING

Özlem Aydin, Tuğba Çalış
Trakya University, Department of Computer Engineering, 22030, Edirne, Turkey

Abstract

In this paper we concentrate on recognition of named entities in Turkish text using Inductive Logic Programming (ILP) algorithm. One of the main tasks of Information Extraction (IE) is the Named Entity Recognition (NER) which aims to locate and classify the named entities (i.e. person, location, organization names, date-time expressions) in text. ILP is a research area at the intersection of machine learning and logic programming. It uses techniques from both machine learning and logic programming. We have adopted a machine learning paradigm, namely ILP, to approach the task of NER in Turkish. This paper presents the performance results of ILP experiments we have conducted on Turkish data and a comparative evaluation of these results.

Key words: named entity recognition, inductive logic programming

1. INTRODUCTION

Along with the continuous increase of the data in the electronic environment and consequently the increasing complexity, it is a problem to reach the desired information from this large data set. At this point, Information Extraction (IE) is beneficial in obtaining the desired information. It is aimed to determine structurally the information in the text by analyzing the non-structural natural language texts with IE. Named Entity Recognition (NER) is one of the important tasks of IE. NER is a subcategory of IE and deals with identification and classification of named entities in text (Nadeau and Sekine, 2007). Classification of named entity is done according to predefined categories. Named entity categories defined in MUC and CoNLL conferences and often tried to be determined in works up to day are as follows: ENAMEX (person, location, organization), TIMEX (date, time), and NUMEX (money, percentage). Besides the named entities mentioned here, NER studies can be done for different named entities (for example; protein names, drug names, etc.) depending on the research area.

High performance results are achieved in rule-based NER. But formation of rules and change them when needed is not easy in most cases. If the rules are generated especially for a domain dependent NER work, these rules must to be reorganized when they applied to another domain. To overcome these problems, researchers have turned to machine learning methods, which can automatically generate rules with features derived from the text. Feature-based systems use a limited set of properties for named entities. Representation of set of features is important. In addition, the set of features may need to be contained features which represent contextual information.

Turkish is a morphologically agglutinative and a word can have different meanings with prefixes and suffixes added to it. Sometimes a word in Turkish can be a sentence in other languages with these attached forms. For example, consider the Turkish word “kutu-da-ki-ler-miş”. It corresponds to “They were in the box” in English. The root and suffixes are obtained for the Turkish word “kutudakilermiş” with morphological analysis are: kutu (box) + da (in) + ki (the) + ler (they) + miş (were). Tools for morphological analysis have been developed for many languages. There are several tools developed for Turkish: Morphological parser (Sak et al, 2008), Turkish Lexical Database Project (Oflazer, 2009), Zemberek (Akin and Akin, 2009).

It is possible to derive new words by adding different suffixes to the root of a word in all agglutinative languages. In the morphologically rich Turkish language, the number of new words derived from the root of a word can be hundreds. For example a Turkish noun “kutu” (box) may appear different surface forms with suffixes: kutum, kutusu, kutuda, kutular, kutuları, kutudakilerden,
This variety of words can cause data sparsity problem in Turkish NLP (Natural Language Processing) works. For solution of this sparsity problem, it is meaningful to use information that getting from sub-parts of word instead of its surface form.

The root of proper noun can easily be determined while it has an apostrophe. Because, the inflectional suffixes are separated from the lemma by an apostrophe for Turkish proper nouns. As a consequence, it is unnecessary to make morphological analysis for stemming of the proper noun.

Turkish is a free word order language. As a result of this property, the position of the word in a sentence doesn’t provide information about being a named entity or not (Şeker and Eryiğit, 2012). For example, "Ali will go to Edirne by bicycle" given in English sentence can be translate in different Turkish sentences as follows:

- “Ali bisiklet ile Edirne’ye gidecek.”
- “Bisiklet ile Edirne’ye Ali gidecek.”

As you can see, the proper nouns “Ali” and “Edirne” can be found anywhere in the sentence and don’t change the meaning of it.

Our concern in the work presented in this paper is classification of named entities in Turkish texts. The system is developed to classify types of named entities which are person, location and organization. We have adopted a machine learning paradigm, namely ILP, to approach the task of NER in Turkish. This paper presents the performance results of ILP experiments we have conducted on Turkish data and a comparative evaluation of these results. It is the first study to recognizing the class of the named entities with ILP according to NER literature for Turkish language.

In the rest of this paper, we first summarize some previous related studies in Turkish and give information about existing resources in Turkish for NER (cf. Section 2). Afterwards, we introduce some basic concepts of ILP (cf. Section 3). This is followed by an explanation of how knowledge sources are used in our experiments along with a comparative evaluation of the results obtained (cf. Section 4). Finally, we offer a brief conclusion including a suggestion for future work (cf. Section 5).

2. PREVIOUS TURKISH NER STUDIES

The first paper on Turkish NER was developed as a language independent system and tested on English, Greek, Romanian, Turkish and Hindi by Cucerzan and Yarowsky (1999). In Tür et al (2003), they presented a statistical in formation extraction system which is related some other tasks: NER, sentence segmentation and topic segmentation. Bayraktar and Temizel (2008) performed their study for recognize Turkish person names on financial texts using local grammar method. Küçük and Yazıcı (2009) presented a rule-based NER system which employs several information sources such as a set of lexical resources and sets of rules. Özger and Dirı (2012) conducted a rule based NER system for different domain texts for classify named entities such as person, organization, location, date, time and money.

Conditional Random Fields (CRFs) was used some of NER studies in Turkish such as Yeniterzi (2011), Özkaya and Dirı (2011), Tatar and Çiçekli (2011), Küçük and Yazıcı (2012), Şeker and Eryiğit (2012) and Çelikkaya et al (2013). CRFs are undirected graphical models, a special case of which correspond to conditionally-trained finite state machines (Lafferty et al, 2001). When CRFs is used for NER problem, an observation sequence is the token sequence of a sentence and state sequence is its corresponding label sequence. CRF-based NER systems makes use of the different contextual information of the words along with the diversity of features that are beneficial in estimating the different named entities.
In recent years, NER systems are quite developed for formally written text documents. There are some valuable Turkish NER systems which gives high accuracy on well-formed texts. But with the increasing use of social media by people, it was necessary to extract information from User Generated Content (UGC) texts. However, grammatical and spelling mistakes can be found in social media texts. For this reason, it is necessary to apply original methods on UGC texts for NER studies. Çelikkaya et al. (2013) is the first Turkish NER system on UGC texts. This system can recognize named entities on three different domains, namely on datasets collected from Twitter, a Speech-to-Text Interface and a Hardware Forum. Küçük et al. (2014), Küçük and Steinberger (2014), Eken and Tantuğ (2015) developed NER system for Turkish tweets in their studies. Onal and Karagoz (2015) investigate the NER performance of the NLP from Stratch method on social media text. NLP from Stratch is a semi-supervised machine learning approach that has been shown to be successful for several NLP tasks. Demir and Özgür (2014) presented a NER system for Turkish and Czech languages by employing a semi-supervised learning approach based on neural networks. Küçük (2015) proposed an automatic approach to compile language for Turkish NER by utilizing Wikipedia article titles. Küçük and Arıcı (2016) developed a Wikipedia-based NER system which employs lists of person, location and organization that obtained from Turkish Wikipedia and a former rule-based named entity recognizer. The last work done in this area by the Şeker and Eryiğit (2017) introduce a CRF-based NER system which models the morphologically very rich nature of Turkish. This system recognize named entities which are in ENAMEX, TIMEX, and NUMEX and process extra challenging UGC coming with Web 2.0.

NER studies for Turkish are very limited compared to English and some other languages. The reason of this is the lack of datasets labeled with named entities. One of the most known dataset for Turkish NER research is introduced by Tür et al (2003). This dataset annotated for ENAMEX types and consists of nearly 500K words collected from newspaper articles. Furthermore, Küçük et al. (2016) presented a dataset comprising news articles with named entity types of person, location and organization names. A total of 1425 named entities were labeled manually over 10 news articles which are randomly selected from METU (Middle East Technical University) Corpus. The other dataset for Turkish NER prepared from Sahin et al (2017). This dataset which is called Turkish Wikipedia Named Entity Recognition and Text Categorization (TWNERTC) is a collocation automatically categorized and annotated sentences obtained from Wikipedia. They used Turkish Wikipedia dumps and Freebase to form a large scale gazetteers to named entities.

3. INDUCTIVE LOGIC PROGRAMMING

ILP is a relatively new technique that inherits the techniques and theories from both machine learning and logic programming (Muggleton and De Raedt, 1994). The aim of this technique is to learn logic programs from examples. To this effect, tools and techniques are invented and improved to induce hypotheses from examples and synthesize new knowledge from experience. By using computational logic as the representation mechanism for hypotheses and observations, ILP can overcome the two main limitations of classical machine learning techniques:

1. the use of a limited knowledge representation formalism (essentially a propositional logic), and
2. difficulties in using substantial background knowledge in the learning process.

Definitions of the logical terms used in ILP are below:

- B is a finite set of clauses (background knowledge).
- E is a finite set of examples. \( E = E^+ \cup E^- \) consists of positive examples \( E^+ \) and negative examples \( E^- \).
- H is a hypothesis that the following conditions hold:
  1. Prior satisfiability. \( B \cup E^- \not\models \square \)
  2. Posterior satisfiability. \( B \cup H \cup E^- \not\models \)
3. Prior necessity. $B \not\models E^+$

4. Posterior sufficiency. $B \cup H \models e_1 \land e_2 \land ...$

Given a set of positive and negative examples and some background knowledge constraining these examples, ILP algorithms try to generate a hypothesis that covers all positive examples and none of the negative examples. Below is a simple example illustrating the logical terms used above.

$$B = \{\text{grandfather}(X,Y) \leftarrow \text{father}(X,Z), \text{parent}(Z,Y)\
\text{father}(\text{ali}, \text{ayşe}) \leftarrow \\
\text{mother}(\text{ayşe}, \text{ahmet}) \leftarrow \\
\text{mother}(\text{ayşe}, \text{suna}) \leftarrow \}$$

The background knowledge includes three facts:
- that Ali is Ayşe’s father
- that Ayşe is Ahmet’s mother
- that Ayşe is Suna’s mother

and a rule stating that the father of one’s parent is his/her grandfather.

The positive examples concern the relationships between particular grandfathers and their grandchildren:

$$E^+ = \{\text{grandfather}({\text{ali}, \text{ahmet}}) \leftarrow \\
\text{grandfather}({\text{ali}, \text{suna}}) \leftarrow \}$$

We also might be told that some negative examples do not hold:

$$E^- = \{\leftarrow \text{grandfather}({\text{ahmet}, \text{ali}}) \\
\leftarrow \text{grandfather}({\text{suna}, \text{ali}}) \}$$

Believing $B$ and faced with the new facts $E^+$ and $E^-$, one can be guess the following relationship:

$$H = \text{parent}(X,Y) \leftarrow \text{mother}(X,Y)$$

The obtained hypothesis $H$ states that one’s mother is also his/her parent.

NLP is an area that can largely exploit the ILP technique. NLP requires an expressive knowledge representation language that includes relations and recursive and unbounded structural representations. ILP allow to naturally encode natural language statements representing both data and background knowledge and to learn or revise these logical representations (Dzeroski et al, 2000). The background knowledge and the examples, as well as the induced theory, can all be represented as formulas in a clausal language, which is a subset of predicate logic. In particular, due to this uniform representation, the use of background knowledge fits very naturally within a logical approach towards machine learning (Nienhuys and Wolf, 1997). Moreover, IE tasks can be naturally framed in the ILP relational setting where data have a relational structure and examples can be related each other (Junker et al,
In the literature, there are few works that take advantage of ILP principles to learn NER models from logical representations of texts (Patel et al., 2009) (Huong and Thien, 2010).

4. AN ILP-BASED NAMED ENTITY RECOGNIZER FOR TURKISH

4.1 Dataset

In this study, the dataset prepared by the Küçük et al. (2016) is used. We used only 62.5% of the train data and left the remaining for testing each news article in dataset. In this dataset each word tagged with named entity types of person, location and organization names. If the word is non-named entity, it is tagged with other. Statistical information on tagging named entities in the corpus is given in the Table 1.

<table>
<thead>
<tr>
<th>Corpus File Name</th>
<th>Person</th>
<th>Location</th>
<th>Organization</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>20010000</td>
<td>70</td>
<td>135</td>
<td>37</td>
<td>242</td>
</tr>
<tr>
<td>20020000</td>
<td>38</td>
<td>49</td>
<td>39</td>
<td>126</td>
</tr>
<tr>
<td>20030000</td>
<td>51</td>
<td>89</td>
<td>32</td>
<td>172</td>
</tr>
<tr>
<td>20040000</td>
<td>26</td>
<td>126</td>
<td>57</td>
<td>209</td>
</tr>
<tr>
<td>20050000</td>
<td>39</td>
<td>61</td>
<td>46</td>
<td>146</td>
</tr>
<tr>
<td>20060000</td>
<td>49</td>
<td>64</td>
<td>58</td>
<td>171</td>
</tr>
<tr>
<td>20070000</td>
<td>43</td>
<td>14</td>
<td>50</td>
<td>107</td>
</tr>
<tr>
<td>20080000</td>
<td>15</td>
<td>5</td>
<td>37</td>
<td>57</td>
</tr>
<tr>
<td>20090000</td>
<td>32</td>
<td>15</td>
<td>49</td>
<td>96</td>
</tr>
<tr>
<td>20100000</td>
<td>35</td>
<td>9</td>
<td>55</td>
<td>99</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>398</strong></td>
<td><strong>567</strong></td>
<td><strong>460</strong></td>
<td><strong>1425</strong></td>
</tr>
</tbody>
</table>

Table 1. Statistical information about dataset

4.2 The Selection of Features

The selection of features to be used for NER based machine learning is important. Because sufficient quantities are necessary to obtain a good classifier that will accurately predict the class of new encountered samples. A good feature should be able to distinguish samples from different classes and have the same value for the samples in the same class (Kononenko, 1994). We explore morphological and lexical features of the word for our NER task. In this study, morphological and lexical level features were defined and classified for both the target word and its contextual words.

In order to use the morphological features of the words in the text, the root and supplementary information of the words in the dataset must be determined. The automatic morphological processing is used for obtain morphological features which are listed below.

**Stem:** The stem of a word. For a proper noun that does not exist in the Zemberek dictionary, the stem detection has been automated for the condition that it contains an apostrophe.

**Part of speech:** A word can be labeled with the following part of speech categories: noun, proper noun, verb, pronoun, conjunction, preposition, adverb, interjection.
Noun case: Case for noun based words when it takes 0 (zero) value for non-noun based words, for noun based words can have the following properties: nominative, dative, locative, ablative, accusative. Lexical features that can be obtained from the words are also used in the classification. Here are the features of the each word:

Start of sentence: If the related word is at the beginning of the sentence, this feature takes the value 1 and not the value 0.

Including an apostrophe: If the related word includes an apostrophe, it takes value 1, if not, takes value 0.

Capitalization of the first letter: If the first letter of the related word is capital letter, it takes value 1, if not, it takes the value 0.

Uppercase all letters: If all the letters of the related word are uppercase, it takes value 1, if not, it takes value 0.

Lowercase all letters: If all the letters of the related word are lowercase, it takes value 1, if not, it takes value 0.

Numeric all characters: If all characters of the related word are numeric, it takes value 1, if not, it takes value 0.

Alphanumeric all characters: If all characters of the related word are alphanumerics, it takes value 1, if not, it takes value 0.

4.3 ALEPH System

There are many ILP systems which have been practiced and tested according to the literature. Some of these systems are FOIL, MFOIL, GOLEM, ALEPH, PROGOL, LINUS, MARKUS and MOBAL. We have developed our application using ALEPH system (Srinivasan, 1999). ALEPH requires 3 data files to construct theories:

- Background knowledge file which has .b file extension.
- A file consists of positive examples which have .p extension.
- A file consists of negative examples which have .n extension.

The data files prepared as described above are given as input to the ALEPH system. The names and meanings of the predicates, which are coded in the background knowledge of the features used in the following table are given.

<table>
<thead>
<tr>
<th>Predicate</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>part_of_speech</td>
<td>part of speech category of the word</td>
</tr>
<tr>
<td>noun_case</td>
<td>case information for noun based words</td>
</tr>
<tr>
<td>stem</td>
<td>the word’s stem</td>
</tr>
<tr>
<td>start_of_sentence</td>
<td>the word at the beginning of the sentence</td>
</tr>
<tr>
<td>apostrophe</td>
<td>the word including apostrophe</td>
</tr>
<tr>
<td>capitalize</td>
<td>capital letter of the word</td>
</tr>
<tr>
<td>all_uppercase</td>
<td>uppercase of all letters of the word</td>
</tr>
<tr>
<td>all_lowercase</td>
<td>lowercase of all letters of the word</td>
</tr>
<tr>
<td>numeric</td>
<td>numeric of all characters of the word</td>
</tr>
</tbody>
</table>
Table 2. Features are encoded in the background knowledge file for ALEPH

<table>
<thead>
<tr>
<th>Term</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>alphanumeric</td>
<td>alphanumeric of all characters of the word</td>
</tr>
<tr>
<td>left_token_stem</td>
<td>stem of the word which located in the word’s left</td>
</tr>
<tr>
<td>right_token_stem</td>
<td>stem of the word which located in the word’s right</td>
</tr>
<tr>
<td>left_token_pos</td>
<td>part of speech category of the word which located in the word’s left</td>
</tr>
<tr>
<td>right_token_pos</td>
<td>part of speech category of the word which located in the word’s right</td>
</tr>
<tr>
<td>left_token_nouncase</td>
<td>case for noun based words which located in the word’s left</td>
</tr>
<tr>
<td>right_token_nouncase</td>
<td>case for noun based words which located in the word’s right</td>
</tr>
<tr>
<td>left_token_apostrophe</td>
<td>word including apostrophe which located in the word’s left</td>
</tr>
<tr>
<td>right_token_apostrophe</td>
<td>word including apostrophe which located in the word’s right</td>
</tr>
<tr>
<td>left_token_capitalize</td>
<td>capital letter of the word which located in the word’s left</td>
</tr>
<tr>
<td>right_token_capitalize</td>
<td>capital letter of the word which located in the word’s right</td>
</tr>
<tr>
<td>left_token_all_uppercase</td>
<td>uppercase of all letters of the word which located in the word’s left</td>
</tr>
<tr>
<td>right_token_all_uppercase</td>
<td>uppercase of all letters of the word which located in the word’s right</td>
</tr>
<tr>
<td>left_token_all_lowercase</td>
<td>lowercase of all letters of the word which located in the word’s left</td>
</tr>
<tr>
<td>right_token_all_lowercase</td>
<td>lowercase of all letters of the word which located in the word’s right</td>
</tr>
</tbody>
</table>

After the induction process, ALEPH produces a model consisting of rules. The following are examples of the rules induced for the person, location and organization types:

\[
\text{named}_\text{entity}(A, B) \leftarrow \text{part}_\text{of}_\text{speech}(A, \text{proper}_\text{noun}), \text{right}_\text{token}_\text{pos}(A, 1, \text{proper}_\text{noun}), \\
\text{right}_\text{token}_\text{stem}(A, 1, \text{hoca}), B=\text{person}.
\]

\[
\text{named}_\text{entity}(A, B) \leftarrow \text{part}_\text{of}_\text{speech}(A, \text{proper}_\text{noun}), \text{not}(\text{apostrophe}(A, 0)), B=\text{location}.
\]

\[
\text{named}_\text{entity}(A, B) \leftarrow \text{part}_\text{of}_\text{speech}(A, \text{proper}_\text{noun}), \text{left}_\text{token}_\text{stem}(A, 1, \text{milli}), B=\text{organization}.
\]

For example, the third rule states that the named entity type of the A will be organization, if the part of speech category for A is proper noun, and stem of the first content word on the left is milli. This rule makes it clear that the models learned with ILP can be easily interpreted.

### 4.4 Training and Test Data Preparation

An application has been developed that automatically performs the retrieval of the value of specified features for each word from the dataset. The reason for the development of this application is that it is troublesome to manually generate the files supplied to the ALEPH system. The application receives the information needed from the dataset to create the contents of the data files supplied to the ALEPH system. The file containing the background knowledge, the positive sample file given the correct meanings of the samples and negative sample file given the correct meanings of the samples incorrectly are created separately for training and test samples.

For the morphological features, the stem and additional information of the Turkish words are examined by using the open source Zemberek library written in Java programming language. In some
cases, there may be more than one analytical result for a word in morphological analysis with Zemberek. In these cases, different stem and supplementary information options can be found in each case. It is important for the NER application to obtain the stem and supplemental information in the context. It is preferred to make the selection automatically in order to overcome his uncertainty situation. Among the possible results which were produced by Zemberek, the most accurate result selection was automated by choosing the longest stem chain. Situations that Zemberek could not produce any results were examined. It was found that numbers, dates, suffixes, characters were not able to produce any results. By analyzing the information of the stem and additional features, the special case was defined and most of these problems were solved with using own dictionaries. With the application, all possible words found in the stem and additional information are printed in a file. If the developed application can not produce the desired result, it is redefined. After redefining, the desired result is printed in the same file.

4.5 Experimental Results

The evaluation results for each test set are presented in Table 3 in terms of precision, recall, and F-measure calculated as follows:

**Precision:** The ratio of the number of named entities correctly recognized by the system to the total number of recognized named entities gives the precision.

\[
\text{precision} = \frac{\text{number of items correctly recognized by the system}}{\text{number of items recognized by the system}}
\]

**Recall:** A recall metric is used for the ratio of the number of named entities correctly recognized by the system to the total number of entity names that can be detected in the dataset.

\[
\text{recall} = \frac{\text{number of items correctly recognized by the system}}{\text{number of items in dataset}}
\]

**F-measure:** The F-measure metric is also calculated by the harmonic mean of the precision and recall metrics.

\[
F - \text{measure} = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}}
\]

<table>
<thead>
<tr>
<th>Corpus File Name</th>
<th>Precision</th>
<th>Recall</th>
<th>F-Measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>20010000</td>
<td>0.98</td>
<td>0.90</td>
<td>0.94</td>
</tr>
<tr>
<td>20020000</td>
<td>0.97</td>
<td>0.94</td>
<td>0.95</td>
</tr>
<tr>
<td>20030000</td>
<td>0.98</td>
<td>0.93</td>
<td>0.95</td>
</tr>
<tr>
<td>20040000</td>
<td>0.96</td>
<td>0.91</td>
<td>0.93</td>
</tr>
<tr>
<td>20050000</td>
<td>0.96</td>
<td>0.92</td>
<td>0.94</td>
</tr>
<tr>
<td>20060000</td>
<td>0.97</td>
<td>0.92</td>
<td>0.94</td>
</tr>
<tr>
<td>20070000</td>
<td>0.99</td>
<td>0.97</td>
<td>0.98</td>
</tr>
<tr>
<td>20080000</td>
<td>0.99</td>
<td>0.98</td>
<td>0.98</td>
</tr>
</tbody>
</table>
Below is the results obtained for a rule-based NER system by (Küçük et al, 2016) on the dataset which is used for this study.

<table>
<thead>
<tr>
<th>Corpus File Name</th>
<th>Precision</th>
<th>Recall</th>
<th>F-Measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>20010000</td>
<td>0.92</td>
<td>0.91</td>
<td>0.92</td>
</tr>
<tr>
<td>20020000</td>
<td>0.88</td>
<td>0.81</td>
<td>0.84</td>
</tr>
<tr>
<td>20030000</td>
<td>0.96</td>
<td>0.94</td>
<td>0.95</td>
</tr>
<tr>
<td>20040000</td>
<td>0.95</td>
<td>0.88</td>
<td>0.91</td>
</tr>
<tr>
<td>20050000</td>
<td>0.86</td>
<td>0.84</td>
<td>0.85</td>
</tr>
<tr>
<td>20060000</td>
<td>0.95</td>
<td>0.93</td>
<td>0.94</td>
</tr>
<tr>
<td>20070000</td>
<td>0.83</td>
<td>0.71</td>
<td>0.76</td>
</tr>
<tr>
<td>20080000</td>
<td>0.90</td>
<td>0.88</td>
<td>0.89</td>
</tr>
<tr>
<td>20090000</td>
<td>0.86</td>
<td>0.78</td>
<td>0.82</td>
</tr>
<tr>
<td>20100000</td>
<td>0.72</td>
<td>0.63</td>
<td>0.67</td>
</tr>
</tbody>
</table>

Table 4. Results of the experiments with a rule-based NER system by (Küçük et al, 2016)

We see that the ILP technique significantly outperforms the rule-based technique. Otherwise, the number of named entity in our training data is small. For this reason, the rules for all named entity types could not be defined with the classification for each document. For example, the rules which are generated by our NER system for “20060000” corpus file, as follows:

**[Rule 1]** [Pos cover = 1095 Neg cover = 4]

named_entity(A, B) :- all_uppercase(A, 0), apostrophe(A, 0), B=other.

**[Rule 2]** [Pos cover = 36 Neg cover = 15]

named_entity(A, B) :- part_of_speech(A, proper_noun), not(apostrophe(A, 0)), B=location.

**[Rule 3]** [Pos cover = 13 Neg cover = 1]

named_entity(A, B) :- part_of_speech(A, proper_noun), not(all_uppercase(A, 0)), B=organization.

**[Rule 4]** [Pos cover = 1090 Neg cover = 0]

named_entity(A, B) :- not(part_of_speech(A, proper_noun)), B=other
As you can see, while the system could generate rules for location, organization and non-named entity, it could not generate rule for person. Because of the tagged dataset limitations, we could not test the performance of ILP-based NER system with a large training data.

5. CONCLUSIONS

In this study we presented a Turkish NER system using ILP. We have investigated the use of ILP as a technique for incorporating knowledge sources into the construction of NER models. ILP systems provide an appropriate framework for dealing with data. The experimental results indicate that our system’s performance comparable to other NER systems in Turkish. We tried to compare the results and the evaluation metrics of a rule-based NER work in Turkish with our ILP-based NER work. We aim to implement different large datasets that can be fed into our application in future. The other goal is adding NUMEX and TIMEX types to our system.

REFERENCES


