A New Approach for Named Entity Recognition

Adlı Varlık Tanımada Yeni Bir Yaklaşım

Burak Ertopçu[1], Ali Buğra Kanburoğlu[1], Ozan Topsakal[1], Onur Açıkgöz[1]
Ali Tunca Gürkan[1], Berke Özenç[1], İlker Çam[1]
Begüm Avar[2], Gökhan Ercan[1], Olcay Taner Yıldız[1]
burak.ertopcu@isik.edu.tr, bugra.kanburoglu@isik.edu.tr, ozan.topsakal@isik.edu.tr, onur.acikgoz@isik.edu.tr
ali.gurkan@isik.edu.tr, berke.ozenc@isik.edu.tr, ilker.cam@isik.edu.tr
begum.avar@boun.edu.tr, gokhan.ercan@isik.edu.tr, olcaytaner@isikun.edu.tr
[1]Department of Computer Engineering, Işık University, İstanbul, Turkey
[2]Department of Linguistics, Boğaziçi University, İstanbul, Turkey

***Abstract*—Many sentences create certain impressions on people. These impressions help the reader to have an insight about the sentence via some entities. In NLP, this process corresponds to Named Entity Recognition (NER). NLP algorithms can trace a lot of entities in the sentence like person, location, date, time or money. One of the major problems in these operations are confusions about whether the word denotes the name of a person, a location or an organisation, or whether an integer stands for a date, time or money. In this study, we design a new model for NER algorithms. We train this model in our predefined dataset and compare the results with other models. In the end we get considerable outcomes in a dataset containing 1400 sentences.**

***Keywords*—Natural Language Processing, Information Extrac-tion, Named Entity Recognition**

***Özetçe*—Birçok cümlenin ilk bakışta insanlara verdiği izlenimler vardır. Bu izlenimler bir takım varlıklar sayesinde insanlara okudukları şeylerin ne hakkında olduğunu kavramada yardımcı olur. Bunun Doğal Dil Geliştirme'deki karşılığı Adlı Varlık Tanımadır (AVT). AVT algoritmaları temel olarak cümlede kişi, yer, zaman, tarih, saat veya para gibi bir çok varlığı tarayabilir. Bu işlemlerdeki en büyük problem bir ismin kişiye mi yoksa bir yere mi veya organizasyona mı, veya bir sayının tarihe mi yoksa paraya mı ait olduğu gibi sorulardır. Bu çalışmada Adlı Varlık Tanıma algoritmalarına yeni bir model tasarladık. Bunu, oluşturduğumuz veri setinde çalıştırıp elde edilen sonuçları diğer modellerinkilerle karşılaştırdık. Sonuç olarak ürettiğimiz 1400 cümlelik veri setinde kayda değer bulgular elde ettik.**

***Anahtar Sözcükler*—Doğal Dil Geliştirme, Veri Çekme, Adlı Varlık Tanıma**

I. INTRODUCTION

Natural language processing (NLP) is a field in computer science which investigates how a computer can understand and manipulate language in its written and spoken form. Language ability is one of the most important characteristics of human beings that sheds light on the functioning of the human brain. If human language can be modelled in computer environment, it can be used for advanced and effective communication tasks.

The foundations of NLP lie in a number of disciplines, namely, computer and information sciences, linguistics, mathematics, electrical and electronic engineering, artificial intelligence and robotics, and psychology. Some NLP applications include studies such as machine translation, natural language text processing and summarisation, user interfaces, multilingual and cross-language information retrieval (CLIR), speech recognition, artificial intelligence, and expert systems.

NLP studies, in general, focus on figuring out how to resolve the rules of natural languages for the machines. Through such resolution, many processes - such as translation, information extraction from structurally irregular texts, question answering, text summarisation - are aimed to be performed automatically by machines.

Named entity recognition (NER) is one of the subtasks in NLP. Using previously existing or published information, NER aims to recognise words such as person, institution, establishment, place names, time expressions, and currencies in a written text. In this study we create a new model for NER. We train this model in our predefined dataset, which contains 1400 sentences, and compare the results with other models.

This paper is organised as follows: We define NER problem in Section II and give the previous work in Section III. In continuous models, we represent words with continuous vectors, namely word embeddings. A brief introduction to word embeddings is given in Section IV. The details of our dataset and how it is constructed are given in Section V. We give our experiment methodology in Section VI and results in Section VII. Lastly, we conclude in Section VIII.

II. NAMED ENTITY RECOGNITION

Anything that is denoted by a proper name, i. e., for instance, a person, a location, or an organization, is considered to be a named entity. In addition, named entities also include things like dates, times, or money. Here is a sample text with named entities marked:

[$_{ORG}$ Türk Hava Yolları] bu [$_{TIME}$ Pazartesi'den] itibaren [$_{LOC}$ İstanbul] [$_{LOC}$ Ankara] güzergahı için indirimli satışlarını [$_{MONEY}$ 90 TL'den] başlatacağını açıkladı.

This sentence contains 5 named entities including 3 words labeled as ORGANIZATION, 2 words labeled as LOCATION, 1 word labeled as TIME, and 1 word labeled as MONEY. Table I shows typical generic named entity types.

978-1-5386-0930-9/17/$31.00 ©2017 IEEE

Table I. LIST OF NAMED ENTITY TYPES WITH THE KINDS OF ENTITIES THEY BELONG TO

Tag	Sample Categories	Example
PER	people, characters	**Atatürk** yurdu düşmanlardan kurtardı.
ORG	companies, teams	**IMKB** günü 60 puan yükselerek kapattı.
LOC	regions, mountains, seas	Ülkemizin başkenti **Ankara'dır**.
TIME	time expressions	**Cuma günü** tatil yapacağım.
MONEY	monetarial expressions	Geçen gün **3000 TL** kazandık.

In named entity recognition, one tries to find the strings within a text that correspond to proper names (excluding TIME and MONEY) and classify the type of entity denoted by these strings. The problem is difficult partly due to the ambiguity in sentence segmentation; one needs to extract which words belong to a named entity, and which not. Another difficulty occurs when some word may be used as a name of either a person, an organization or a location. For example, *Deniz* may be used as the name of a person, or - within a compound - it can refer to a location *Marmara Denizi* "Marmara Sea", or an organization *Deniz Taşımacılık* "Deniz Transportation".

The standard approach for NER is a word-by-word classification, where the classifier is trained to label the words in the text with tags that indicate the presence of particular kinds of named entities. After giving the class labels (named entity tags) to our training data, the next step is to select a group of features to discriminate different named entities for each input word. Table II shows the sample text represented with tag labels and three possible features, namely the root form of the word, the part of speech (POS) tag of the word, and a boolean feature for checking the capital case.

Table II. NAMED ENTITY TAGGING AS CLASSIFICATION PROBLEM

Word	Features				Label
	Root	Pos	Capital	...	
Türk	Türk	Noun	True	...	ORGANIZATION
Hava	Hava	Noun	True	...	ORGANIZATION
Yolları	Yol	Noun	True	...	ORGANIZATION
bu	bu	Pronoun	False	...	NONE
Pazartesi'den	Pazartesi	Noun	True	...	TIME
itibaren	itibaren	Adverb	False	...	NONE
İstanbul	İstanbul	Noun	True	...	LOCATION
Ankara	Ankara	Noun	True	...	LOCATION
güzergahı	güzergah	Noun	False	...	NONE
için	için	Adverb	False	...	NONE
indirimli	indirimli	Adjective	False	...	NONE
satışlarını	sat	Noun	False	...	NONE
90	90	Number	False	...	MONEY
TL'den	TL	Noun	True	...	MONEY
başlatacağını	başlat	Noun	False	...	NONE
açıkladı	açıkla	Verb	False	...	NONE
.	.	Punctuation	False	...	NONE

Given such training data, a classifier like neural network or decision tree can be trained to label new sentences. Figure 1 shows the operation of such a classifier at the point where the word Ankara is next to be labeled. For this classifier, the window size is 2, that is, we assume a context window that includes two preceding words and two succeeding words.

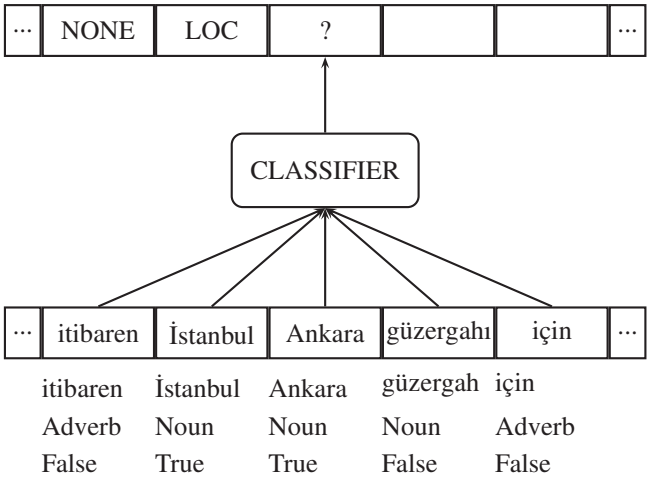

Figure 1. Classifier based approach to named entity recognition. The tagger slides a context window over the sentence, classifying words as it proceeds. At this point, the classifier is attempting to label Ankara. Features derived from the context typically include the words, part-of-speech tags, etc.

III. PREVIOUS WORK

A. Linguistic Background

While NER is a rather unproblematic task among NLP studies in well-studied languages like English, it faces certain challenges when dealing with a language like Turkish. One of the central foci of literature on Turkish linguistics has been the complexity of Turkish morphology. Turkish is a textbook example for an agglutinative language, i.e. words in their surface form may contain various morphemes, especially suffixes. The main problem posed by such languages is lexical sparsity [13], which, in general, can be considered as a challenge for earlier steps such as the Morphological Analysis and Disambiguation tasks which feed NER.

Not only the morphology but also the syntactic structure of Turkish is a fertile source for challenging issues. Turkish is typically treated as a subject-object-verb (SOV) language yet it has a rather free word order. In other words, the constituents of a sentence can occur in any order with slight modifications in the sentential meaning. A particular order is chosen mainly on pragmatic grounds [7]. Therefore, the position of a word within a sentence does not provide any clues about whether it is a Named Entity or not.

As an additional problem specific to NER, there exist many proper nouns in Turkish which are derived from common names (through suffixation such as the following names of cities in Turkey: *Denizli* derived from *deniz* "sea", compounding such as *Balıkesir* from *balık*+*esir* "fish+prisoner" or zero-derivation such as *Ordu* from *ordu* "army"). In speech, prosodic cues are helpful in distinguishing proper nouns, especially place names, from common nouns due to their idiosyncratic stress pattern [16]. Yet there is no orthographic correlates to stress. Instead, in formal texts at least, orthographic clues can be helpful in distinguishing a proper noun, which starts with a capital letter, from a common noun. In sentence-initial position, however, all words begin with a capital letter and hence this clue is not available.

In short, several linguistic features of Turkish, such as its rich morphology, free word-order, as well as derivation as a word-formation process frequently employed in forming proper names yield problems for NER tasks.

B. Computational Background

Study on NER has a big attention in the literature. However, most of the models are specific to the language of focus. A language independent method is proposed in [6] which is a bootstrapping algorithm based on iterative learning. The method relies on word interval and contextual clues. This work is tested on Turkish, Romanian, English, Greek and Hindi and can be considered as the first work on Turkish NER.

Turkish-specific studies are relatively scarce when compared to languages that have a wider global distribution. First work on Turkish NER [18] presents a study based on information extraction. [5] focuses on conditional random fields using morphological and lexical features. Their baseline is influenced by a leading work however the dataset used is gathered from real natural language data. Taking Turkish NER studies into account, models based on Hidden Markov Models (HMMs) [18], Conditional Random Fields (CRFs) [15] and rule based [17] studies are presented on data gathered from news reports. Formal texts like news are written on certain rules of language. Authors of such texts follow the grammatical and orthographical rules of the language in question, and thus generate statistically less volatile data. Twitter is widely used in NLP studies as well, yet in tweets there is no need to follow spelling rules and words -and even letters in the case of emoticons and some informal abbreviations- can be used in different senses than usual. An experiment-based study on tweets [9] shows the difference between processing a formal syntax over a social media text.

IV. WORD EMBEDDINGS

Traditional representations of words (i.e., one-hot vectors) are based on word-word (W × W) co-occurrence sparse matrices where W is the number of distinct words in the corpus. On the other hand, distributed word representations (DRs) (i.e., word embeddings) are word-context (W × C) dense matrices where C < W and C is the number of context dimensions which are determined by underlying model assumptions. Dense representations are arguably better at capturing generalized information and more resistant to overfitting due to context vectors representing shared properties of words. DRs are real valued vectors where each context can be considered as a continuous feature of a word. Due to their ability to represent abstract features of a word, DRs are considered as reusable across higher level tasks in ease, even if they are trained with totally different datasets.

Prediction based DR models gained much attention after Mikolov et al.'s neural network based SkipGram model in 2013 [11]. The secret behind the prediction based models is simple: never build a sparse matrix at all. Prediction based models construct dense matrix representations directly instead of reducing sparse ones to dense ones. These models are trained like any other supervised learning task by giving lots of positive and negative samples without adding any human supervision costs. Aim of these models is to maximize the probability of each context c with the same distributional assumptions on word-context co-occurrences, similar to count based models [2] [10].

V. DATA

Our dataset is collected from Penn-Treebank corpus and each sentence of this dataset is translated into Turkish [19]. This dataset includes 1400 sentences, and 13194 words (including punctuation marks). Our data format is shown in Figure 2.

Figure 2. Data content as .train file

A. Morphological Disambiguation

Turkish is an agglutinative language in which words are formed by attaching derivational and inflectional suffixes to the roots. Morphemes added to a word can change its part of speech, i.e., for instance, convert a noun to a verb - or vice versa - or can create adverbs from adjectives. Moreover, during word formation, some letters can be changed or undergo deletion. Hence, without determining the lemma of a word from its surface form, based on its intended meaning, it is not possible to identify the word correctly and extract candidate senses from a dictionary.

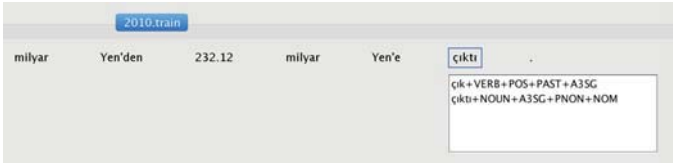

Figure 3. Morphological disambiguation tool

Following the translation, the corpus has been morphologically disambiguated. In that work, human annotators selected the correct morphological parsing from multiple possible analyses returned from the automatic parser (See Figure 3 for the morphological disambiguation tool used). The tag set and morphological representation was adopted from the study [12]. Each output of the parser comprises the root of the word, its part-of-speech tag and a set of morphemes, each separated with a '+' sign.

B. Ner Tagging

In the second stage, we had randomly selected 1400 sentences and annotated them manually by using the "NER Annotation Tool" of I¸sık University. The tool is shown in Figure 4.

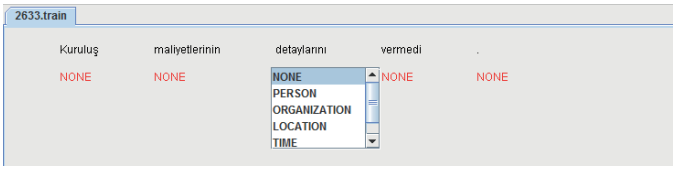

Figure 4. Annotation tool for NER

NER tags that are used as class labels in this study are: PERSON for person's names, LOCATION for place names,

ORGANIZATION for governmental or civil organizations, firms, associations etc., TIME for specific points in time such as dates, years, hours etc., MONEY for financial amounts or currencies, and NONE for everything else. By these categories of class labels, the distribution of the data is shown below in Table III. Annotated dataset and sources codes are freely available[1].

Table III. DISTRIBUTION OF THE DATA

Label	Count
PERSON	606
LOCATION	235
ORGANIZATION	685
MONEY	387
TIME	299
NONE	10982

VI. METHODOLOGY

The seven known classification algorithms we use, are:

- Dummy: Decides based on the prior class probability without looking at the input. All test instances are assigned to the class with the maximum prior.
- C45: The archetypal decision tree method [14].
- Knn: K-Nearest Neighbor classification algorithm that uses the Euclidean distance.
- Lp: Linear perceptron with softmax outputs trained by gradient-descent to minimize cross-entropy [3].
- Mlp: Well-known multilayer perceptron classification algorithm [3].
- Nb: Classic Naive Bayes classifier where each feature is assumed to be Gaussian distributed [1] and each feature is independent from other features.
- Rf: Random Forest method improves bagging idea with randomizing features at each decision node [4] and called these random decision trees as weak learners. In the prediction time, these weak learners are combined using committee-based procedures.

A. Discrete Model

1) Features: We used the following features in our Discrete Models.

- CaseAttribute (C): This is a discrete attribute for a given word. If the last inflectional group of the word contains case information, the attribute will have that case value. Otherwise the attribute will have the value null.
- IsCapitalAttribute (IC): This is a binary attribute. It checks each word in a sentence. If the first character is uppercase, the attribute returns true, otherwise it returns false.
- IsDateAttribute (ID): This is a binary attribute that checks whether a word is written in date format. If the word is in date format, it returns true, otherwise it returns false.
- IsFractionAttribute (IF): This is a binary attribute that checks whether a word is a fractional number. If the word is a fractional number, it returns true, otherwise it returns false.
- IsHonorificAttribute (IH): This is a binary attribute. It checks each word in a sentence. If the word equals to one of the titles “bay”, “bayan”, “sayın”, “dr.”, “prof.” or “doç.”, it returns true, otherwise it returns false.
- IsMoneyAttribute (IM): This is a binary attribute. It checks whether a word in a sentence denotes a currency. If the word is “doları”, “lirası” or “avro”, the attribute returns true, otherwise it returns false.
- IsNumAttribute (IN): This attribute checks whether the word has num (number) tag.
- IsOrganizationAttribute (IO): This is a binary attribute. It checks a given word and if it equals to “corp”, “inc.”, or “co.”, it returns true, otherwise it returns false.
- IsPropAttribute (IP): This attribute checks whether the word has prop (proper name) tag.
- IsRealAttribute (IR): This is a binary attribute that checks whether a word is a real number or not. If the word is a real number, it returns true, otherwise it returns false.
- IsTimeAttribute (IT): This is a binary attribute that checks whether a word is written in time format. If the word is in time format, it returnstrue, otherwise it returns false.
- MainPosAttribute (MP): This is a discrete attribute. It returns the main part of speech of the word.
- LastIGContainsPossessiveAttribute (P): This is a binary attribute. If the last inflectional group of the word contains possessive information, the attribute will return true, otherwise it will return false.
- RootFormAttribute (RF): This is a discrete attribute. It returns the root form of a given word.
- RootPosAttribute (RP): This is a discrete attribute. It returns the part of speech of the root form of a given word.
- SurfaceFormAttribute (SF): This is a discrete attribute. It returns the surface form of the word.

2) Methods: First of all, we annotated 1400 sentences to generate our dataset. The annotation was done manually and it was a very significant point for the experiment because we needed an effective dataset to get reliable results.

We give the details of the methods, such as the attributes, the classifier and its parameters, in Table IV.

B. Continuous Model

We used Mikolov et al.’s SkipGram and CBOW models for learning continuous representation of words. We used a news corpus consisting of 1 million (M) unannotated sentences for unsupervised training, collected from news and articles from

[1]http://haydut.isikun.edu.tr/nlptoolkit.html

Table IV. DETAILS OF METHODS USED IN DISCRETE MODEL

Method	Attributes	Window	Classifier	Parameters	Author
METHOD1	IC, ID, IR, IT, MP	4	Rf	attribute subset size = 4, ensemble size = 20	B.Ö.
METHOD2	IC, ID, IF, IT, MP, RF, SF	1	Mlp	learning rate = 0.1, number of hidden nodes = 30	B.E.
METHOD3	IH, IM, MP, RF, SF	0	Mlp	learning rate = 0.1, number of hidden nodes = 50	A.T.G.
METHOD4	C, IC, IH, IO, IR, MP, P, RF, RP, SF	2	Lp	learning rate = 0.1	A.B.K.
METHOD5	IC, IN, IP, MP, RF, SF	1	Mlp	learning rate = 0.1, number of hidden nodes = 5, 8	O.T.
METHOD6	IC, IP, MP, RF, SF	1	Mlp	learning rate = 0.1, number of hidden nodes = 10	O.A.

Table V. ERROR RATES OF METHODS USING STRATIFIED 10-FOLD CROSS-VALIDATION

Classifier	DUMMY	METHOD1	METHOD2	METHOD3	METHOD4	METHOD5	METHOD6
Error Rate	14.89	14.71	7.64	12.19	7.65	8.45	9.28

newspapers. We trained models with the same settings in three separate sizes (100K, 500K, 1M). DR models generally have tens of hyper-parameters. We listed the most important hyper-parameters only, along with default values we used in our experiments below:

- Context window (win): Size of a context window where the prediction model gathers co-occurrence information from. The reader should note that, even though concepts are similar, this parameter should not be confused with the context window size we used on executing classifiers. We used 5 as the default setting.
- Dimension (d): Size of a C context dimensions of vectors for word-context dense matrices (W ×C). We used 100 as the default setting.
- Deleting infrequent words (del): Most DR models ignore (delete) infrequent words in the corpus with some thresholding mechanism, assuming that infrequent words are informative. We did not ignore infrequent words due to the nature of the NER task requiring more samples of infrequent instances such as *58.97*, *Ohio*, *IBM*.

Unlike the DR text pre-processing conventions, we did not lowercase tokens and did not filter out any punctuations (e.g., commas, dots, hyphens) due to NER task requiring inclusion of proper words and punctuations. Therefore, distribution of punctuations are also learnt along with words. For example, as expected, our 1M sentenced model returns ")" as spatially closest token to "(". Another interesting example is that our model returns the tokens "oranında", "gerilemi¸s", "yüzde", "0.5", "oran" (words related to percentage in Turkish) as the most spatially closest ones to the token "%".

We trained our DR models in two kinds of word forms:

- Surface Form (SU): Natural form of a word which appeared in a text as it is. Ex: *Güzel gözlü turnalar, göçtüler.*
- Root Form (RO): Root of a word used in DR training based on morphological disambiguation of every sentence provided by the "Morphological Disambiguation Library" of I¸sık University [8]. Ex: *Güzel göz turna, göç.*

We used our trained DR outputs (W × C) to feed our classifiers as continuous features of words without doing any additional task-specific feature handcrafting. Therefore, we feed every word with C (*d*) continuous features by using C dimensional word vectors of our generated representations. If generated DRs don't have a word vector for a word instance requested by the classifier model, we provided zero-vector in C dimensions to represent a null feature value. DRs are real valued vectors normalized between -1 and 1. We also reported missing word requests with each experiment as out-of-vocabulary (OOV) rate due to the fact that classifier performances are highly dependent on OOV rates in experiments.

VII. EXPERIMENTS

A. Inter-annotator Agreement

For evaluation of the annotated dataset, we used inter-annotator-agreement measure. Two different group of annotators annotated same sentences. Due to a lack of time, we could only re-annotate 100 of the total of 1400 sentences and got %97.5 of inter-annotator agreement. As a comparison, the expected inter-annotator agreement, assuming the annotators annotated completely randomly, is %16.7.

B. Discrete Model

We evaluated the proposed approach using a stratified 10-fold cross-validation on the data. The accuracies obtained from each classifier are shown in Table V.

C. Continuous Model

We fixed SkipGram (SG) model configuration as default with hyper-parameters win=5, d=100, del=0 to use as a baseline for all other configurations (SG-win5-d100-del0). We chose the default configuration based on our observations that it performs better on average compared to all other various configuration options we pre-experimented. We trained 6 models in total for 3 different corpus sizes (100K, 500K, 1M), and 2 different word forms (Surface, Root) using default configuration. We ran our models with Dummy, Lp with learning rate: 0.1, Mlp with learning rate 0.1, hidden nodes = 50, Knn with k = 3, C45 with pruning, and Nb classifiers with 10-fold cross-validation on data. We ran all classifiers with the window size of 1. Table VI shows classifier error rates and OOV percentages of our experiments:

Our continuous model experiments show that:

- Unsupervised DR models can provide valuable information as features of words for NER classification tasks and can perform as good as supervised, manually handcrafted discrete features. Our DR models outperformed discrete tasks on Mlp, Lp, Knn with a small margin (6.7, 6.56, 7.4 error rates).

Table VI. ERROR RATES OF DIFFERENT METHODS USING 10-FOLD STRATIFIED CROSS-VALIDATION OF CONTINUOUS MODEL EXPERIMENTS

	OOV%	Dummy	Lp	Mlp	Nb	Knn	C45
SURFACE (SU)							
100K	32.4	9.1	8.13	8.25	35.79	9.4	9.13
500K	23.05	10.75	8.06	7.81	35.18	10.24	10.61
1M	20.31	11.18	8.05	7.94	31.57	10.23	9.96
ROOT (RO)							
100K	20.77	9.79	6.87	6.74	15.22	9.3	7.85
500K	17.49	11.1	6.96	6.62	20.43	14.8	9.55
1M	16.33	11.56	6.7	6.56	9.52	16.82	9.87
OTHER CONF.							
1M-RO-d300	16.33	11.56	6.96	6.65	10.04	21.67	9.54
1M-RO-d20	16.33	11.56	7.47	6.76	11.49	7.92	9.33
1M-RO-d10	16.33	11.56	8.3	7.24	13.13	7.4	10.55
MIN	16.33	9.1	6.7	6.56	9.52	7.4	7.85

- Word embeddings with basic morphological root form enrichment can provide better results than simple surface form word embeddings (1.2pt reduction from 8 to 6.8 on average in Lp and Mlp). This improvement is linearly related with the reduction of OOV rates provided by the root form representation ability of models.
- Nb works significantly better with root forms compared to surface forms (reduced 31.57 to 9.52 error rate).
- Increasing corpus size did not increase the NER classification performances; it was even observed to decrease OOV rates.
- SkipGram performed better than CBOW almost on all experiments.
- Increasing vector dimensions (d = 300) for classification tasks did not increase NER performance. On the contrary, vectors in low dimensions (10, 20) performed significantly better with Knn.

VIII. CONCLUSION

Named entity recognition is the problem of detecting the type of named entities in sentences, such as person, organisation, time or money. The problem in NER lies in deciding whether a noun denotes a person, an organisation or a location and whether a number denotes time or money. As a new approach to NER, we produced six different discrete models with various features and tested these models on the tagged NER data created by us. The data consist of 1400 Turkish sentences which are manually tagged by 7 different annotators.

We also used Mikolov et al.'s SkipGram and CBOW models for learning continuous representation of words. Then we used our trained continuous representation of words to feed our classifiers as continuous features of words without doing any feature enhancement. Our results show that, these continuous models for NER classification tasks can perform as good as supervised, manually handcrafted discrete features.

As in many natural language processing subfields, words have been used as the atomic unit of language in distributional semantics (DS) modeling field for the sake of model simplicity. Even though word-based models yield good results for languages with limited vocabulary such as English, these models are pretty ineffective for morphologically rich languages with unlimited vocabularies such as Turkish.

ACKNOWLEDGEMENTS

This work was supported by Işık University BAP projects 14B206 and 15B201. All authors contributed equally to this work. O. A., İ. Ç., B. E., A. T. G., A. B. K., B. Ö., O. T. designed and implemented discrete model experiments. They also labeled the data and wrote the manuscript. G. E. designed and implemented continuous model experiments. B. A. wrote the manuscript. O. T. Y. supervised the project, gave conceptual advice and wrote the manuscript.

REFERENCES

[1] E. Alpaydın, *Introduction to Machine Learning*, 3rd ed. The MIT Press, 2010.

[2] M. Baroni, G. Dinu, and G. Kruszewski, "Don't count, predict! a systematic comparison of context-counting vs. context-predicting semantic vectors," in *ACLC*, 2014, pp. 238–247.

[3] C. M. Bishop, *Neural Networks for Pattern Recognition*. Oxford University Press, 1995.

[4] L. Breiman, "Random forests," *Machine Learning*, vol. 45, pp. 5–32, 2001.

[5] G. Celikkaya, D. Torunoglu, and G. Eryigit, "Named entity recognition on real data: a preliminary investigation for Turkish," in *7th International Conference on Application of Information and Communication Technologies*, 2013.

[6] S. Cucerzan and D. Yarowsky, "Language independent named entity recognition combining morphological and contextual evidence," in *Proceedings of the Joint SIGDAT Conference on EMNLP and VLC*, 1999.

[7] E. T. Erguvanlı, *The function of word order in Turkish Grammar*. Oxford: Oxford University Press, 1984.

[8] O. Gorgun and O. T. Yildiz, "A novel approach to morphological disambiguation for Turkish," in *International Conference on Computer and Information Science*, 2011, pp. 77–83.

[9] D. Küçük and R. Steinberger, "Experiments to improve named entity recognition on Turkish tweets," *arXiv:1410.8668*, 2014.

[10] O. Levy, Y. Goldberg, and I. Dagan, "Improving distributional similarity with lessons learned from word embeddings," *Transactions of the Association for Computational Linguistics*, vol. 3, p. 211–225, 2015.

[11] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.

[12] K. Oflazer, B. Say, D. Hakkani-Tür, and G. Tür, "Building a Turkish treebank," in *Building and exploiting syntactically-annotated corpora*. Dordrecht: Kluwer, 2003.

[13] E. Okur, *Named Entity Recognition for Turkish Microblog Texts Using Semi-Supervised Learning with Word Embeddings*, 2011.

[14] J. R. Quinlan, *C4.5: Programs for Machine Learning*. San Meteo, CA: Morgan Kaufmann, 1993.

[15] G. A. Seker and G. Eryigit, "Initial explorations on using crfs for Turkish named entity recognition," in *COLING*, 2012.

[16] E. Sezer, "On non-final stress in Turkish," *Journal of Turkish Studies*, vol. 5, pp. 61–69, 1981.

[17] S. Tatar and I. Cicekli, "Automatic rule learning exploiting morphological features for named entity recognition in Turkish," *Journal of Information Science*, vol. 37, pp. 137–151, 2011.

[18] G. Tür, D. Hakkani-Tür, and K. Oflazer, "A statistical information extraction system for Turkish," *Natural Language Engineering*, vol. 9, pp. 181–210, 2003.

[19] O. T. Yildiz, E. Solak, O. Gorgun, and R. Ehsani, "Constructing a Turkish-English parallel treebank," in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*. Baltimore, Maryland: Association for Computational Linguistics, June 2014, pp. 112–117.