A Detailed Analysis and Improvement of Feature-based Named Entity Recognition for Turkish

Arda Akdemir and Tunga Güngör

Bogazici University, Istanbul, Turkey
{arda.akdemir,gungort}@boun.edu.tr

Abstract. Named Entity Recognition (NER) is an important task in Natural Language Processing (NLP) with a wide range of applications. Recently, word embedding based systems that does not rely on hand-crafted features dominate the task as in the case for many other sequence labeling tasks in NLP. However, we are also observing the emergence of hybrid models that make use of hand crafted features through data augmentation to improve performance of such NLP systems. Such hybrid systems are especially important for less resourced languages such as Turkish as deep learning models require a large dataset to achieve good performance. In this paper, we first give a detailed analysis of the effect of various syntactic, semantic and orthographic features on NER for Turkish. We also improve the performance of the best feature based models for Turkish using additional features. We believe that our results will guide the research in this area and help making use of the key features for data augmentation.

Keywords: Named Entity Recognition · Conditional Random Fields · Dependency Parsing · Turkish.

1 Introduction

Named Entity Recognition is first defined officially as an NLP task in the Message Understanding Conference (MUC) in 1995. According to its first formal definition [2], NER consists of two main subtasks: Detection of named entities and categorizing each detected entity into predefined categories. Identification is an important step which enables information extraction over large texts. The second step can be considered as a more refined task where the aim is to use any kind of contextual or word-level, sub-word level information to distinguish between subcategories of entities. Ratinov et al. [15] show that this step is more challenging compared to detection. Detecting and properly categorizing the named entities is an important first step for analyzing a given text and is shown to improve the performance of many other NLP tasks such as machine translation [10] and question answering on speech data [13]. Named entities are also used to select a better language model to enhance the performance of speech-to-text systems [1].

Dependency Parsing (DP) is an important research topic in NLP. It is demonstrated to be highly useful for various NLP tasks. Dependency parsing is shown to be useful for machine translation, question answering and named entity recognition [3, 18]. Following the previous work we used dependency parsing related features together with other features during our experiments to boost the NER performance in our feature based setting.

In this paper, we first give a detailed analysis of the effect of various morphological, syntactic and semantic level features on NER performance. Throughout our experiments we make use of a Conditional Random Fields (CRF) based model which makes use of hand crafted features. We also show improvements over the previous work on feature based NER for Turkish. Our final model which can be considered as an extension to the previous feature based models [19, 4], makes use of dependency parsing related features which are not tested extensively in this setting before to the best of our knowledge. The main contributions of this paper can be considered as follows:

- A detailed analysis of each hand crafted feature on the NER performance.
- Showing an improvement over the previous work on feature based NER models for Turkish by using dependency related features in addition.

The paper is organized as follows: We start by giving the previous work done on NER and feature based models . Then we describe the dataset we have used in Section 3. This will be followed by the Methodology Section which describes the CRF model and the feature sets we have used in detail. Finally we give the results we have obtained and compare our results with related work.

2 Previous Work

Early work in this area is dominated by feature based statistical models. McCallum et al. [12] give the first results for using a CRF based model together with hand crafted features for the NER task. A more detailed overview of the feature based statistical models used for NER until 2007 can be found in the work of Nadeau et al. [14].

Recent work on NER is dominated by deep learning models and these models are consistently shown to outperform the previous work in this area. Using Convolutional Neural Networks (CNN), Bidirectional Long-short Term Memory (BiLSTM) Recurrent Neural Networks (RNN) is frequent as in the case for many other NLP tasks that can be formulated as a sequence labeling task [8, 11]. The work done on less resourced languages is more limited but best results for NER are also obtained by using a similar deep learning architecture for the Turkish language [7].

Previous work on agglutinatively rich languages such as Turkish show that using morphological and syntactic features improve the performance of NER systems [4, 7, 19]. Using the surface form of the words causes the data sparsity problem as a single word can be extended in multiple ways in such agglutinative languages. Using stemming to solve this problem is often not a good idea as important semantic and syntactic information about the token is lost during this process. Specifically, morphological features are shown to be vital for such languages in several studies [22, 4].

3 Datasets

During all our experiments we have made use of a dataset extracted from Turkish newspapers [21]. It is one of the most frequently used datasets for NER for Turkish and considered as the most important benchmark in this setting. As the dataset is relatively old and reannotated and refined many times by different researchers, it is difficult to keep the consistency of the exact version of this dataset being used in each paper. Table 1 gives some statistics about the training and test sets we have used during this paper.

Table 1. A)Number of annotated entities in the Turkish NER dataset. B) Number of annotated tokens.

A	**LOC**	**ORG**	**PER**	**B**	**LOC**	**ORG**	**PER**
Training	9,800	9,117	14,693	Training	11,137	15,470	21,641
Test	1,116	865	1,597	Test	1,315	1,680	2,394

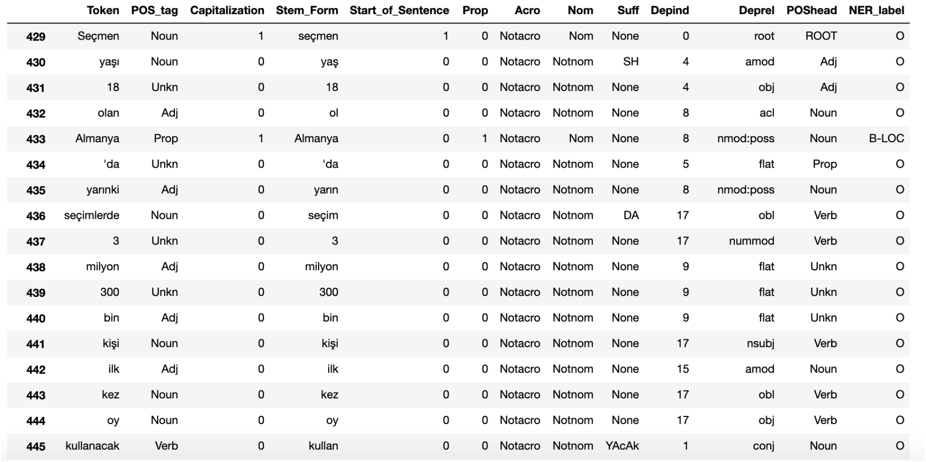
The dataset is annotated in BIO scheme. The initial token of each entity sequence is tagged with 'B' followed by its entity type and the remaining token tags start with 'I'. In our setting we have used the following three entity types: Location (LOC), Person (PER) and Organization (ORG). So an example annotation for a two-token entity of type 'Person' will be tagged as follows: Akira (B-PER) Kurosawa (I-PER). Figure 1 gives an example sentence from the dataset that we have made use of which is additionally annotated with many hand crafted features. The dataset is structured in a token-per-line format where each line contains a single token followed by its feature values. Each annotation following a token will be explained in detail in the following section.

4 Methodology

In this section we explain the undertaken methodology during this study to analyze and improve on using hand-crafted features for NER for Turkish. We begin by describing the model used during the experiments which will be followed by the explanation of each feature.

4.1 Model

During our experiments we have made use of the CRF based Wapiti toolkit implement by Lavergne et al. [9]. Wapiti is a sequence classifier toolkit which allows



	Token	POS_tag	Capitalization	Stem_Form	Start_of_Sentence	Prop	Acro	Nom	Suff	Depind	Deprel	POShead	NER_label
429	Seçmen	Noun	1	seçmen	1	0	Notacro	Nom	None	0	root	ROOT	O
430	yaşı	Noun	0	yaş	0	0	Notacro	Notnom	SH	4	amod	Adj	O
431	18	Unkn	0	18	0	0	Notacro	Notnom	None	4	obj	Adj	O
432	olan	Adj	0	ol	0	0	Notacro	Notnom	None	8	acl	Noun	O
433	Almanya	Prop	1	Almanya	0	1	Notacro	Nom	None	8	nmod:poss	Noun	B-LOC
434	'da	Unkn	0	'da	0	0	Notacro	Notnom	None	5	flat	Prop	O
435	yarınki	Adj	0	yarın	0	0	Notacro	Notnom	None	8	nmod:poss	Noun	O
436	seçimlerde	Noun	0	seçim	0	0	Notacro	Notnom	DA	17	obl	Verb	O
437	3	Unkn	0	3	0	0	Notacro	Notnom	None	17	nummod	Verb	O
438	milyon	Adj	0	milyon	0	0	Notacro	Notnom	None	9	flat	Unkn	O
439	300	Unkn	0	300	0	0	Notacro	Notnom	None	9	flat	Unkn	O
440	bin	Adj	0	bin	0	0	Notacro	Notnom	None	9	flat	Unkn	O
441	kişi	Noun	0	kişi	0	0	Notacro	Notnom	None	17	nsubj	Verb	O
442	ilk	Adj	0	ilk	0	0	Notacro	Notnom	None	15	amod	Noun	O
443	kez	Noun	0	kez	0	0	Notacro	Notnom	None	17	obl	Verb	O
444	oy	Noun	0	oy	0	0	Notacro	Notnom	None	17	obj	Verb	O
445	kullanacak	Verb	0	kullan	0	0	Notacro	Notnom	YAcAk	1	conj	Noun	O

Fig. 1. Example sentence from the NER dataset.

training models using various model types and optimization algorithms. The results achieved by this toolkit on the CoNLL-2003 English dataset is comparable to the state-of-the-art deep network based systems even though the training time is shorter and the memory requirement is significantly lower. The toolkit is chosen primarily because it enables fast configuration of various training models as well as fast configuration of the features that are being used by the model. Following subsections will describe the specific aspects of this toolkit.

The toolkit allows using various machine learning models for training as mentioned previously. The models and their brief description are as follows:

- **Maximum Entropy (MAXENT)**: Maximum Entropy models are very general probabilistic methods that pick the output with the highest entropy by considering the observations and the prior knowledge. These models are frequently used in NLP tasks that can be formulated as sequence labeling tasks. A Maximum Entropy based model is used in [16] for the POS tagging task.
- **Maximum Entropy Markov Models (MEMM)**: It is an extension of the Maximum Entropy models which consider the hidden features of Hiddden Markov Models. It is also frequently used in NLP, especially for the sequence labeling tasks such as POS tagging and NER [6].
- **Conditional Random Fields (CRF)**: This model calculates the transition probabilities from one prediction to another in addition to the Markovian assumption of the MEMM where the transition probabilities between tags are learned from the training dataset.

Our initial experiments showed that CRF based models consistently outperform others. Thus for the final models tested on the test set, the training is done using the CRF model. Sutton et al. [20] give a detailed formulation for CRF based models. Apart from the training model, we have trained the proposed models with several different optimization algorithms to be more confident about the results we have obtained. Below is the list of the optimization algorithms used together with a brief description:

- **l-bfgs**: Limited-memory Broyden-Fletcher-Goldfarb-Shanno algorithm [5]. It is a quasi-newton optimization algorithm with less memory requirements.
- **sgd-l1**: Stochastic gradient descent with l1 regularization. Our initial experiments showed that sgd-l1 is not suitable in our proposed setting so we have not included it in our grid search experiments.
- **rprop+/-**: Resilient backpropagation which only takes into account the sign of the partial derivative and acts independently on each weight. rprop- refers to the version of the algorithm without the backtracking step.

4.2 Features

We have analyzed many features in this study. Below we explain each feature briefly:

1. **Surface form** (Surf): The surface form of each word.
2. **Initial POS tag** (POS): The POS tag prediction for the stem form of the word by a third party morphological analyzer [17].
3. **Final POS tag**(POS) : The POS tag for the complete surface form of the word.
4. **Capitalization Feature** (Cap) : A four valued feature giving information about the orthographic structure of a token. 0 for alllowercase, 1 for Onlyfirstletter, 2 for ALLUPPER and 3 for miXeD. Capitalization feature is a fundamental feature for the NER task for Turkish as all named entities are expected to be capitalized. This feature significantly increases the performance in languages like Turkish.
5. **Stem of the word** (Stem) : This feature is important to tackle the out-of-vocabulary problem in agglutinative languages like Turkish.
6. **Start of sentence** (SS) : Binary feature to handle the ambiguity of capitalization at the beginning of each sentence.
7. **Proper noun** (Prop): This binary feature takes the value 1 if the morphological analyzer predicts the word to be a proper noun and 0 otherwise.
8. **Acronym feature** (Acro): Binary feature denoting whether the morphological analyzer predicts the word to be an acronym or not, e.g. ABD - Acro and Istanbul - Notacro.
9. **Nominal feature** (Nom) : This feature is a combination of three atomic features. Observing the morphological analyses of the labeled entities in the training set showed that, most of them share the following three features: They are capitalized, they are in their stem form and the analyzer predicts them to be Nominal. So we used a binary feature to check whether these three conditions are met or not.

10. **Final suffix** (Suf): The final suffix of the word is given in the morphological analysis format. If the word does not have any suffix 'None' value is given. In order to overcome the data sparsity of complete matching the surface form of the suffix is not used. For example the final suffix of the word 'kalitesinin' which means 'the quality of (something/someone)' is 'nin' but the feature value is 'NHn' where the uppercased letters denote the letters are subject to change in other words but the suffix itself is the same. By using this feature CRF based model can detect all the words that have the same suffix even though the surface form of them may differ as in the case of 'kalitesinin' - 'nin' and 'ormanın' - 'ın'.
11. **Regex Features**: Wapiti allows giving as input regular expressions which are converted either into binary features or the regex match itself is kept as the feature value. We used regular expressions to extract features such as all 1,2,3 and 4 character long suffixes and prefixes if they exist. We also used regular expression to create binary features to detect numericals and punctuations in a given token.
12. **Dependency Relation**(Deprel): The predicted relation between the word in question and its predicted head word by the dependency parser used.
13. **Dependency Index**(Depind): The index of the head word of the dependency relation. This can be considered as a positional feature.
14. **POS tag of the head word**(POShead): The POS tag of the head word of the dependency relation. In the case that the word itself is the root word a special POS tag "ROOT" is used.

All features are used with a window size two, i.e. two preceding and two succeeding words are taken into account for each token. Increasing the window size greatly increases the computational cost and we found that increasing the window size does not significantly improve the performance after two.

4.3 Evaluation Metrics

We used two evaluation metrics which are considered as the standard metrics for the NER task: F1 and MUC. For this task, F1 measures the systems performance of both detecting and categorizing an entity together. MUC metric considers detection and categorization as separate tasks and takes the average of the F1-measures obtained for each sub-task. Thus, MUC scores are higher compared to the F1 scores.

5 Results and Discussion

We performed various experiments with different subsets of the features given above. In this section we first give the results obtained on the 10% of the training set which is used as validation. We used the validation phase to find the best feature subset and then continued with a grid search over the learning algorithms and optimization methods explained in the previous section. Best performing model is tested on the test set to get the final results. We finish the section by comparing our results with the previous work on feature based NER for Turkish.

We started with analyzing the features by adding them cumulatively following the previous work [4]. We determined four core features as our baseline model (BM) and added the remaining features one-by-one. The core features are as follows: Surface form, POS tag, Capitalization and Stem form.

At each step we added each remaining feature, trained the model and observed the change in performance. According to the results we pick the feature that gave the highest improvement to be the next feature added to the current feature subset. The feature with the highest improvement can easily be referred from the order of appearance in Table 2 (A) given below.

Table 2. A) Initial results obtained for the first baseline (BM) together with the training times. **B)** Results with the updated baseline (BM2).

A	**MUC**	**F1-Measure**	**Training Time**	**B**	**F1-Measure**
BM	0.919	0.889	3,000s	**BM2**	0.894
+SS	0.921	0.889	3,300s	**+Cap+Stem+SS**	0.896
+Prop	0.924	0.896	3,400s	**+Prop+Acro+Nom**	0.899
+Acro	0.924	0.897	3,900s	**+Suf**	**0.900**
+Nom	0.925	0.896	4,800s	**+Depind+Deprel**	0.899

At the last step of the experiments addition of the Nominal feature caused a decrease in the performance which is counter-intuitive. Following this, we changed the core feature set and started experiments from the baseline again to analyze in detail the effect of each feature better. The core features for the new baseline model (BM2) are as follows: Surface form, POS tag and all regex features with a window size of 2. Regex features are described in the previous section and includes all orthographic features except for the capitalization feature. Then again we added features in a cumulative manner but this time analyzed the effect of adding these features in groups rather than one-by-one. Table 2 (B) gives the results obtained for these experiments.

At the final step we have observed a slight decrease in performance when we added the dependency related features together. Next we did a grid search over the training models and optimization algorithms, using the final feature set to be more confident about the results we have obtained. Table 3 gives the results for these experiments. After the grid search we have concluded that adding dependency relation and dependency index together does not improve the performance of the model, and the best model/optimization algorithm combination is the default combination of CRF/l-bfgs. This combination consistently outperforms all other combinations on both evaluation metrics (Overall F1 and MUC).

Next we trained a model by adding only the dependency relation feature and obtained the best results. The effect of this feature is given in Table 4. We

Table 3. Exploration of combinations of all training models and optimization algorithms.

Model	Optimization Algorithm	**PER**	**LOC**	**ORG**	**Overall F1**	**MUC**
crf	l-bfgs	0.909	0.898	0.883	0.899	0.919
	rprop-	0.904	0.892	0.860	0.887	0.904
	rprop+	0.903	0.892	0.860	0.887	0.905
maxent	l-bfgs	0.910	0.890	0.865	0.892	0.915
	rprop-	0.913	0.887	0.847	0.887	0.908
	rprop+	0.913	0.887	0.847	0.887	0.908
memm	l-bfgs	0.910	0.879	0.845	0.883	0.909
	rprop-	0.911	0.883	0.826	0.879	0.901
	rprop+	0.911	0.883	0.826	0.879	0.901

restate the previous best result we have achieved which we call ‘Previous Best’ for readability.

Table 4. Results for adding the dependency relation feature.

	PER	**LOC**	**ORG**	**Overall F1**
Previous Best	0.912	0.895	0.884	0.900
+Deprel	0.916	0.896	0.886	**0.902**

We have successfully shown on the validation set that the addition of the dependency relation feature in our setting slightly improves the performance. Next we evaluated the true performance of our final model by exploiting the dependency relation information on the test set. Table 5 gives the results obtained for these final experiments. ‘Previous Best’ denotes the feature combination with the highest F1 score without taking into account the dependency related features. The feature combination is as follows: Surface form, POS tag, Stem form, Capitalization, Start of Sentence, Proper Noun, Acronym, Nominal, Final Suffix and all regex features explained in the previous section.

We did not observe a significant difference when we take into account the POS tag of the head word of the dependency relation.

Finally we compare our results with the previous work on feature based NER for Turkish. Table 6 shows the comparison of our model with the related work. Yeniterzi et al. [22] exploits the morphological features and analyze the improvement obtained by using them. Following their work we have also made use of various morphological features as explained in the Methodology section. As we have no access to the gazetteers used by Seker et al. [19] and do not have access to the vector representations used by Demir et al. [4], we compare our

Table 5. Final results on the test set. "Previous Best" denotes the best combination observed during the experiments on the validation set

Model	Entity Type	**Precision**	**Recall**	**F1**
Previous Best+Deprel	PER	0.913	0.889	0.900
	LOC	0.921	0.899	0.910
	ORG	0.909	0.856	0.882
	Overall	0.915	**0.884**	**0.899**
+POShead	PER	0.917	0.880	0.898
	LOC	0.923	0.903	0.913
	ORG	0.917	0.850	0.882
	Overall	**0.919**	0.880	**0.899**

Table 6. Comparison with related work using F1 measure as the evaluation metric.

System	**PER**	**ORG**	**LOC**	**Overall**
Yeniterzi et al. [22]	89.32	83.50	92.15	88.94
Seker et al. [19] without using gazetteers	90.65	86.12	90.74	89.59
Demir et al. [4] without using vector representations	92.26	83.53	90.73	89.73
Our Model	90.07	88.15	90.98	**89.89**

results with their best versions that does not make use of gazetteers and vector representations.

6 Conclusion And Future Work

In this study we have given a detailed analysis of the effect of hand-crafted features on the performance of NER for Turkish language. We tried novel features such as dependency related features and analyzed their effect. We also compared our results with the previous work and showed improvement over them by using additional features. We hope that the findings stated in this work will guide the researchers working on this area. In future, we will be implementing deep learning models that make use of data augmentation by following the findings of this paper.

References

1. Bharadwaj, S.S., Medapati, S.B.: Named-entity based speech recognition (Mar 26 2015), uS Patent App. 14/035,845
2. Chinchor, N., Robinson, P.: Muc-7 named entity task definition. In: Proceedings of the 7th Conference on Message Understanding. vol. 29 (1997)
3. Chiu, J.P., Nichols, E.: Named entity recognition with bidirectional lstm-cnns. arXiv preprint arXiv:1511.08308 (2015)

4. Demir, H., Ozgur, A.: Improving named entity recognition for morphologically rich languages using word embeddings. In: ICMLA. pp. 117–122 (2014)
5. Fletcher, R.: Practical methods of optimization. John Wiley & Sons (2013)
6. Fresko, M., Rosenfeld, B., Feldman, R.: A hybrid approach to ner by memm and manual rules. In: Proceedings of the 14th ACM international conference on Information and knowledge management. pp. 361–362. ACM (2005)
7. Güngör, O., Üsküdarlı, S., Güngör, T.: Improving named entity recognition by jointly learning to disambiguate morphological tags. arXiv preprint arXiv:1807.06683 (2018)
8. Lample, G., Ballesteros, M., Subramanian, S., Kawakami, K., Dyer, C.: Neural architectures for named entity recognition. arXiv preprint arXiv:1603.01360 (2016)
9. Lavergne, T., Cappé, O., Yvon, F.: Practical very large scale CRFs. In: Proceedings the 48th Annual Meeting of the Association for Computational Linguistics (ACL). pp. 504–513. Association for Computational Linguistics (July 2010), http://www.aclweb.org/anthology/P10-1052
10. Li, Z., Wang, X., Aw, A., Chng, E.S., Li, H.: Named-entity tagging and domain adaptation for better customized translation. In: Proceedings of the Seventh Named Entities Workshop. pp. 41–46 (2018)
11. Ma, X., Hovy, E.: End-to-end sequence labeling via bi-directional lstm-cnns-crf. arXiv preprint arXiv:1603.01354 (2016)
12. McCallum, A., Li, W.: 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. pp. 188–191. CONLL '03, Association for Computational Linguistics, Stroudsburg, PA, USA (2003). https://doi.org/10.3115/1119176.1119206, https://doi.org/10.3115/1119176.1119206
13. Mollá, D., Van Zaanen, M., Cassidy, S., et al.: Named entity recognition in question answering of speech data (2007)
14. Nadeau, D., Sekine, S.: A survey of named entity recognition and classification. Lingvisticae Investigationes **30**(1), 3–26 (2007)
15. Ratinov, L., Roth, D.: Design challenges and misconceptions in named entity recognition. In: Proceedings of the Thirteenth Conference on Computational Natural Language Learning. pp. 147–155. Association for Computational Linguistics (2009)
16. Ratnaparkhi, A.: A maximum entropy model for part-of-speech tagging. In: Conference on Empirical Methods in Natural Language Processing (1996)
17. Sak, H., Güngör, T., Saraçlar, M.: Morphological disambiguation of turkish text with perceptron algorithm. In: International Conference on Intelligent Text Processing and Computational Linguistics. pp. 107–118. Springer (2007)
18. Sasano, R., Kurohashi, S.: Japanese named entity recognition using structural natural language processing. In: Proceedings of the Third International Joint Conference on Natural Language Processing: Volume-II (2008)
19. Şeker, G.A., Eryiğit, G.: Initial explorations on using crfs for turkish named entity recognition. Proceedings of COLING 2012 pp. 2459–2474 (2012)
20. Sutton, C., McCallum, A., et al.: An introduction to conditional random fields. Foundations and Trends® in Machine Learning **4**(4), 267–373 (2012)
21. Tür, G., Hakkani-Tür, D., Oflazer, K.: A statistical information extraction system for turkish. Natural Language Engineering **9**(2), 181–210 (2003)
22. Yeniterzi, R.: Exploiting morphology in turkish named entity recognition system. In: Proceedings of the ACL 2011 Student Session. pp. 105–110. Association for Computational Linguistics (2011)