Improving Named Entity Recognition by Jointly Learning to Disambiguate Morphological Tags

Onur Güngör
Boğaziçi University, Istanbul, Turkey
onurgu@boun.edu.tr

Suzan Üsküdarlı
Boğaziçi University, Istanbul, Turkey
suzan.uskudarli@boun.edu.tr

Tunga Güngör
Boğaziçi University, Istanbul, Turkey
gungort@boun.edu.tr

Abstract

Previous studies have shown that linguistic features of a word such as possession, genitive or other grammatical cases can be employed in word representations of a named entity recognition (NER) tagger to improve the performance for morphologically rich languages. However, these taggers require external morphological disambiguation (MD) tools to function which are hard to obtain or non-existent for many languages. In this work, we propose a model which alleviates the need for such disambiguators by jointly learning NER and MD taggers in languages for which one can provide a list of candidate morphological analyses. We show that this can be done independent of the morphological annotation schemes, which differ among languages. Our experiments employing three different model architectures that join these two tasks show that joint learning improves NER performance. Furthermore, the morphological disambiguator's performance is shown to be competitive.

Title and Abstract in Turkish

Biçimbilimsel Etiketleri Ayrıştırmayı Birlikte Öğrenerek Varlık İsmi Tanıma Başarısını Artırmak

Daha önceki çalışmalar, biçimbilimsel olarak zengin dillerdeki varlık ismi tanıma (VAT) başarısını artırmak için sözcüklerin iyelik, genitif ve benzeri hâllerinin kullanılabileceğini göstermiştir. Ancak, bu türden varlık ismi tanıma işaretleyicilerinin çalışabilmesi için elde edilmesi zor veya bazı diller için imkansız olan dışsal biçimbilimsel ayrıştırıcılara (BA) ihtiyaç vardır. Bu çalışmada, bu tür ayrıştırıcılara olan ihtiyacı ortadan kaldırmak için VAT ve BA görevlerini aynı anda çözen ve aday biçimbilimsel çözümlemelerin sunulabildiği dillere uygulanabilen bir model önerilmektedir. Bunun dillere göre değişen biçimbilimsel işaretleme şemalarından bağımsız olarak yapılabildiği gösterilmiştir. Bu iki görevi aynı anda gerçekleştiren üç farklı model mimarisi kullanarak yaptığımız deneyler birlikte öğrenmenin VAT başarısını artırdığını göstermiştir. Buna ek olarak, biçimbilimsel ayrıştırıcının başarısının önceki çalışmalarla karşılaştırılabilir olduğu görülmüştür.

1 Introduction

Named entity recognition (NER) is the task of selecting the portions of text which refer to an entity that designate a person, location or organization. This makes it a basic natural language processing (NLP) task closely related to relation extraction, knowledge base population, and entity linking.

Works that represent the current state of the art in NER generally start by representing words with pretrained word embeddings, embeddings which rely on surface form characters (Lample et al., 2016; Ma and Hovy, 2016). These architectures feed the word representations to a bidirectional long short-term memory layer (Bi-LSTM) to represent the context where the disambiguation between the possible entities is undertaken by decoding on trellis provided by a conditional random field (CRF) model.

When these models are trained and evaluated for morphologically rich languages (MRLs), it has been shown that using embeddings based on characters or linguistic properties of the word such as morphological features indicating a grammatical case improves the performance compared to using only pretrained word embeddings (Gungor et al., 2017). Even though they provide a better approach for MRLs, they require an external morphological disambiguator for every language of interest, a requirement which can be hard or even impossible for some languages to satisfy. This is especially true for agglutinative languages where there can be many roots and morphological tag sequences for a single word. Although there is an effort to provide a tool for POS tagging and lemmatization for many languages in a single format (Straka and Straková, 2017), it has been shown that there is a better approach for morphological tagging in terms of performance which can utilize the information in the context of the target word (Shen et al., 2016).

In this paper, we propose a model to jointly learn the NER and morphological disambiguation (MD) tasks to offer a solution to this problem. We design our model so that any language with a mechanism which can provide a number of candidate morphological analyses for a word can utilize our joint model. This is easier compared to providing disambiguated morphological analyses because systems that disambiguate morphological analyses are harder to build. Furthermore, we do not require the labels of each task to be present in the same dataset. One can easily train the part of the model which is responsible for the MD task in another -preferably larger- dataset and start with the pretrained model. Our main contribution is to show that jointly disambiguating morphological tags and predicting the NER tags results in an equivalent level of performance compared to using externally provided morphological tags.

We give a survey of related work on the subject in Section 2. We explain our basic models and the proposed joint models in Section 3. In Section 4, we describe our dataset which is derived from a frequently used database in the literature. After running the experiments described in Section 5, we observe that jointly training our model for NER and MD results in an increase in the NER performance.

2 Related Work

Early approaches to NER typically use several hand-crafted features such as capitalization, word length, gazetteer based features, and syntactic features (part-of-speech tags, chunk tags, etc.) (McCallum and Li, 2003; Finkel et al., 2005; Humphreys et al., 1998; Appelt et al., 1995). Some of them are data-driven approaches such as conditional random fields (CRF) (McCallum and Li, 2003; Finkel et al., 2005), maximum entropy (Borthwick, 1999), bootstrapping (Jiang and Zhai, 2007; Wu et al., 2009), latent semantic association (Guo et al., 2009), and decision trees (Szarvas et al., 2006).

Recently, RNN based sequence taggers have dominated the state of the art in NER (Lample et al., 2016; Ma and Hovy, 2016; Huang et al., 2015; Yang et al., 2016). These approaches model the words as fixed length vectors and employ Bi-LSTM or GRU layers to obtain a characterization of the relevant context of the word to be labeled. These context vectors are then transferred to a CRF module after transforming into score vectors. However, in these studies, the morphological information present in the surface form of the word is handled only through the use of character based embeddings. Although this is not a limiting factor for languages which are not morphologically rich, it has been shown that employing morphologically disambiguated tags when representing words in a neural architecture improves the NER performance (Gungor et al., 2017; Straková et al., 2016).

There has been other approaches to the NER task for morphologically rich languages (Demir and Özgür, 2014; Seker and Eryiğit, 2012; Yeniterzi, 2011; Tür et al., 2003; Hasan et al., 2009). A study which can be considered as one of the first attempts in tackling NER for morphologically rich languages uses a hidden Markov model and takes the morphological tag sequence as input along with others like the surface form, capitalization features and similar features (Tür et al., 2003). In a study which basically depends on handcrafted features given to a CRF-based sequence tagger system, the word morphology was captured using the first and last three characters of the word as a feature resulting in an improvement in the NER tagging performance for Bengali (Hasan et al., 2009). In another study (Yeniterzi, 2011), a similar approach is taken with features generated using the output of an external morphological disambiguator and also shown to improve the performance. Another study (Seker and Eryiğit, 2012) uses the same method but with a different approach for extracting morphological information, where they show an improvement over the previous state of the art results of Yeniterzi (2011). The first study focusing on morphologically rich languages to employ neural networks (Demir and Özgür, 2014) contains a regularized averaged perceptron (Freund and Schapire, 1999) and relies on handcrafted rules along with pretrained word embeddings. However, they refrain from using output from external morphological disambiguators and only rely on the first and last few characters of a word as features. Our work in this paper differs from these studies as it does not rely on handcrafted features. We represent words as fixed length vectors, employ morphological information to disambiguate the correct morphological analysis, and then combine them in such a way to obtain a context vector to label with NER tags.

In a recent study on morphological disambiguation (Yildiz et al., 2016), the authors propose a two-layer network for prediction. In the first layer, they process the candidate morphological analyses along with the correctly predicted analyses of previous words and obtain a vector to be processed in the second layer. The second layer takes all vectors propagated from the previous words and computes a `softmax` function over positive and negative classes. They predict the correct morphological analysis starting from the first word and use this prediction in the next word positions. The model is evaluated on a dataset manually labeled by the authors and considered as the state of the art for Turkish and competitive for French and German. Our proposed model for morphological disambiguation relies on scoring the candidate morphological analyses to predict the correct one for a word in a sentence. We borrow this idea from Shen et al. (2016). In their study, they feed the word representations to a Bi-LSTM and obtain context embeddings for each position. Using these embeddings, they score each morphological analysis by calculating a similarity function reaching the state of the art or competitive results for Turkish, Russian and Arabic.

Most of the work in morphological disambiguation or tagging strictly depend on their chosen specific output format for morphological analysis. This is due to the fragmented nature of computational approaches to morphological analysis for every language in the literature. However, we argue that our approach is immune to this problem as all of these output formats can be treated as a sequence. An example from Finnish is '`raha+[POS=NOUN]+[NUM=SG]+[CASE=ADE]`' (Silfverberg et al., 2016), another from Turkish is '`Ankara+Noun+Prop+A3sg+Pnon+Loc`' (Oflazer, 1994), and one for Hungarian is '`hír+NOUN+Case=Nom+Number=Plur`' (Trón et al., 2005). All of these can be split by the '+' symbol and transformed into a root and tag sequence. Moreover, there is an attempt in the area to unify the morphological annotation along with syntax annotation across many languages which will contribute more towards a solution (Nivre et al., 2016).

Many models targeting NLP tasks are designed to work independently although they usually employ linguistic information related with other tasks. Given that there are state of the art models which are similar in the sense that they all employ a sentence level Bi-LSTM, it is reasonable to hypothesize that jointly learning several tasks will improve the performance as shown in the literature (Hashimoto et al., 2017; Luong et al., 2015). In a recent study, it has been suggested that using separate layers for separate tasks is better rather than using the same (or usually top) layer for all the tasks (Søgaard and Goldberg, 2016).

3 Models

We test our hypothesis by training a number of models where we choose to enable or disable the selected components and features[1]. We start by explaining two basic models for each task: (i) a Bi-LSTM based sequence tagger where we predict the correct NER tags with a CRF (Section 3.1), (ii) a Bi-LSTM tagger which is used to represent the context for selecting the correct morphological analysis at the given position (Section 3.2). The joint models are combinations of these two basic models in various ways (Section 3.3).

[1] The code to replicate the experiment environment and the actual source code is published at `https://github.com/onurgu/joint-ner-and-md-tagger`

3.1 NER Model

We formally define an input sentence as $X = (x_1, x_2, \ldots, x_n)$ where each x_i is a vector of size l and the corresponding NER labels as $y_{\text{NER}} = (y_{\text{NER},1}, y_{\text{NER},2}, \cdots, y_{\text{NER},n})$. x_i are then fed to a Bi-LSTM which is composed of two LSTMs (Hochreiter and Schmidhuber, 1997) treating the input forwards and backwards. The output of this Bi-LSTM at position i, h_i, is a vector of size $2p$ where p is the size of the LSTM cell. Further, we transform h_i through a fully connected layer $\texttt{FC}_{last}$ with $\texttt{tanh}$ activations at the output to combine the left and right contexts into a vector of size p. This is followed by another fully connected layer to obtain a vector s_i of size K, where K is the number of the NER tags.

We follow a conditional random field (CRF) based approach to model the dependencies between the consequent tokens (Lafferty et al., 2001). To do this, we take the vector s_i at each position i as the score vector of the corresponding word and aim to minimize the following loss function $\texttt{loss}_{\text{NER}}(X, y_{\text{NER}})$ for a single sample sentence X:

$$-\sum_{i=0}^{n} A_{y_i,y_{i+1}} - \sum_{i=1}^{n} s_{i,y_i} + logZ(X)$$

where $A_{i,j}$ represents the score of a transition from tag i to j, $Z(X) = \sum_{y' \in \mathbb{Y}} \texttt{exp}\left(\sum_{i=0}^{n} A_{y_i,y'_{i+1}} + \sum_{i=1}^{n} s_{i,y'_i}\right)$ where $\mathbb{Y}$ is the set of all possible label sequences. Using this model, we decode the most probable tagging sequence y^*_{NER} as $\text{argmax}_{\tilde{y}_{\text{NER}}} \texttt{loss}_{\text{NER}}(X, \tilde{y}_{\text{NER}})$. We call this basic model the NER model (Lample et al., 2016) (see Figure 2).

In the remaining part of the section, we explain the details of the word representations used in this study.

Representing words. As the default setting, we obtain word and character based embeddings as described below and combine them by concatenation. For the first component, we allocate a word embedding vector of size w_d for every word in our dataset. This can be loaded from a pretrained word embeddings database as is done frequently in the literature, but we chose to learn the word embeddings during training. The second component is generated from the surface forms. We feed the character sequence of the word into a Bi-LSTM as described at the beginning of this section. However, instead of using the outputs of LSTM cells at each position, we just take the last and the first cell outputs of the forward and backward LSTMs and concatenate them (Figure 1). The resulting representation is two times the length of one character embedding length, 2ch_d. This second component is in turn concatenated with the first component to obtain a word representation vector x_i of size $w_d + 2\text{ch}_d$.

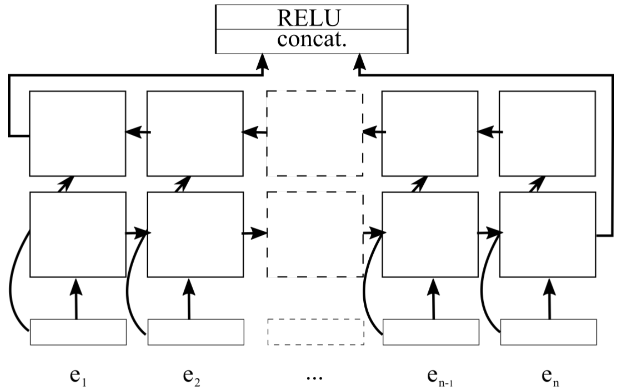


Figure 1: The basic model to generate representations for surface forms, roots, and morphological tag sequences. The input sequence $(e_1, e_2, \cdots, e_{n-1}, e_n)$ can either be the characters of the surface form, the characters of the root of the word, or the tags in the morphological tag sequence. RELU unit is active only for root and morphological tag sequences.

External morphological features. In order to compare our models with a previous method (Gungor et al., 2017), we utilize the golden morphological analysis provided with the dataset in addi-

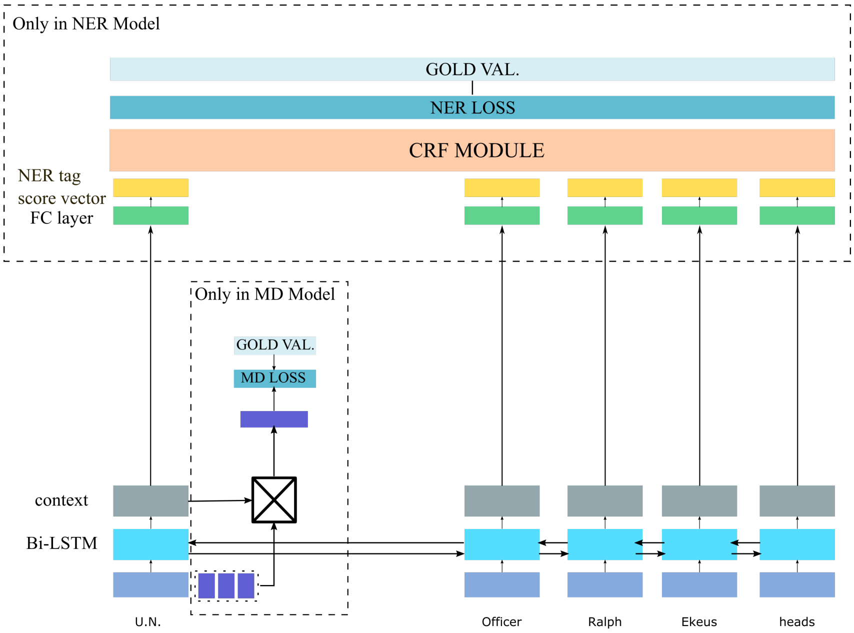


Figure 2: Our basic models: NER and MD. The portions of the model which are only active either for NER or MD models are indicated with dashed lines. The symbol ⊠ represents the selection of $\mathtt{ma}_{ij^*}$.

tion to the word and character based embeddings and call this model EXT_M_FEAT. The best approach reported by Gungor et al. (2017) treats the string form of a morphological analysis as a sequence of characters and apply the process depicted in Figure 1. For example, a morphological analysis in Hungarian is 'Magyar+PROPN+Case=Nom+Number=Sing' in string form and can be split into a list of characters as (M,a,g,y,a,r,+,P,R,O,P,N,+,C,a,s,e,=,N, o,m,+,N,u,m,b,e,r,=,S,i,n,g). Using the sequence of characters of the morphological analysis instead of the sequence of morphemes might seem counterintuitive at first glance. However it has been argued that a benefit of treating morphological analyses as sequences of characters is the information conveyed by the characters within the tags. For example, in Turkish, the tags 'A3sg' and 'A3pl' indicate third person singular and third person plural where the leading two characters 'A3' indicate third person agreement. This allows the model to represent the fragments of the tags which may improve the training performance. In this case, 'A3' would represent the third person agreement independent of the singular or plural case. The resulting vector representation is thus of length $2\mathtt{mt}_d$ which is added to word and character based embeddings to obtain a word representation of $w_d + 2\mathtt{ch}_d + 2\mathtt{mt}_d$.

3.2 MD Model

In this section, we describe our model for morphological disambiguation which is based on (Shen et al., 2016). In this model, we are given a sentence X in the same form as in the NER task, however we optimize the model to predict y_{MD} where $y_{\text{MD},i}$ represents the correct morphological analysis out of the candidate morphological analyses for word i. Like in the NER model, the MD model also employs a Bi-LSTM layer to obtain context representations when fed with the word representations x_i (Figure 2). We define the candidate morphological analyses for word i as $\mathtt{ma}_i = \{\mathtt{ma}_{i,1}, \mathtt{ma}_{i,2}, \cdots, \mathtt{ma}_{i,j}, \cdots, \mathtt{ma}_{i,K}\}$. To determine the correct morphological analysis, we examine each morphological analysis output form to extract the root surface form and the morpheme sequence and generate the representation $\mathtt{ma}_{ij}$ which we explain below.

We design this approach to be generalizable to many morphological analysis output forms described in Section 2. We give an example from Turkish here: the unique analysis of the Turkish word "Moda'da" is "Moda+Noun+Prop+A3sg+Pnon+Loc". The word literally means 'in Moda' (which is a district in Istanbul) and a common morpheme naming convention is used (Oflazer, 1994). So, we determine the root as 'Moda' and the morpheme sequence as '(Noun, Prop, A3sg, Pnon, Loc)'. The root and the morpheme sequence are used to generate a representation as depicted in Figure 1. Except in this case the RELU activation function (Nair and Hinton, 2010) is also applied to the concatenation of the root and morpheme sequence representations. We choose the resulting representations r_{ij} and $\mathtt{ms}_{ij}$ to be of two times the length of a morpheme embedding $\mathtt{mt}_d$. Furthermore, we add the root representation vector r_{ij} and the morpheme sequence representation vector $\mathtt{ms}_{ij}$ and apply hyperbolic tangent function ($\mathtt{tanh}$), thus the morphological analysis representation $\mathtt{ma}_{ij}$ is defined as follows $\mathtt{tanh}(r_{ij} + \mathtt{ms}_{ij})$.

We then select the morphological analysis ma_{ij^*} by performing a dot product with the context vector h_i: $\mathtt{ma}_{ij^*} = \mathrm{argmax}_j\, h_i \cdot \mathtt{ma}_{ij}$ when decoding. During training, the loss $\mathtt{loss}_{\mathrm{MD}}(X, y_{\mathrm{MD}})$ is calculated as

$$-\sum_{i=1}^{n} \mathtt{log}\,\mathtt{softmax}(\mathtt{mscore}_i)$$

over a score vector $\mathtt{mscore}_i$ such that $\mathtt{mscore}_{ij} = \{h_i \cdot \mathtt{ma}_{ij}\}$.

3.3 Joint model for NER and MD

We have experimented with three approaches for jointly learning NER and MD tasks. In this section, we explain the details of each approach.

Integration mode 1 - In this scheme, we employ a Bi-LSTM layer which is fed with word representations as in the basic models, NER and MD. We then use the same context h_i to calculate the losses separately for NER and MD as explained in Sections 3.1 and 3.2. We call this joint model JOINT1 and show in Figure 3a. We then learn the model parameters to optimize $\mathtt{loss}_{\mathrm{JOINT1}}$

$$\mathtt{loss}_{\mathrm{NER}}(X, y_{\mathrm{NER}}) + \mathtt{loss}_{\mathrm{MD}}(X, y_{\mathrm{MD}}).$$

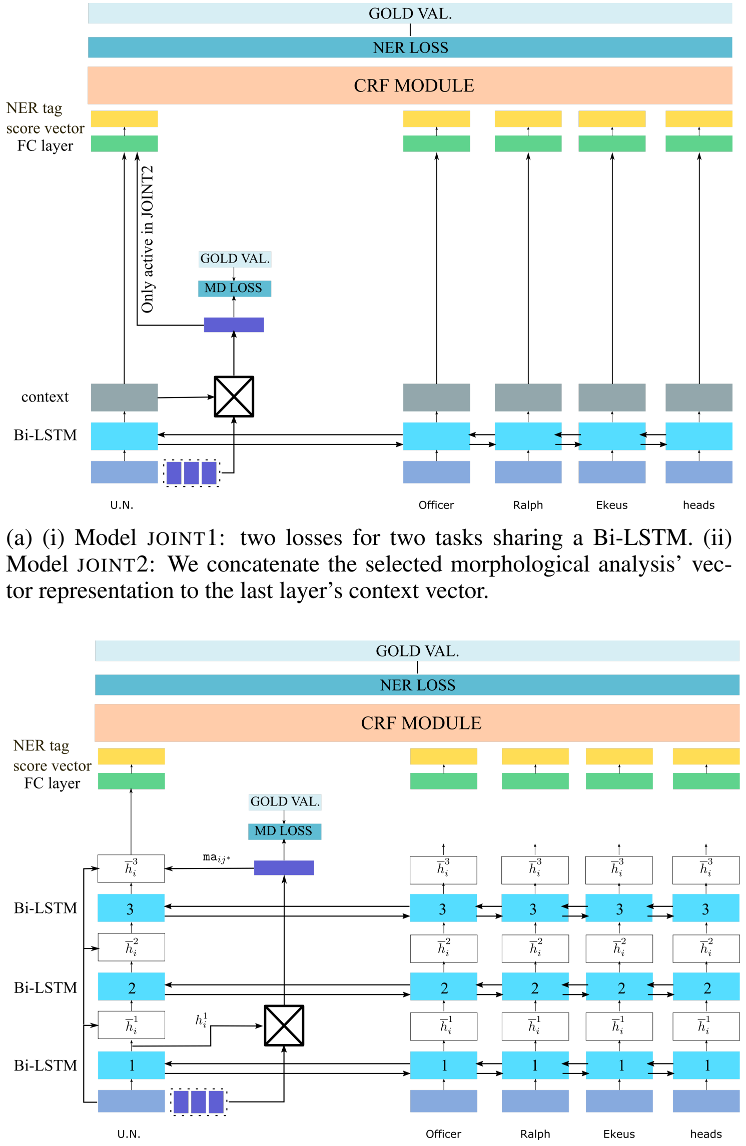
Integration mode 2 - As in the JOINT1 model, this model also calculates separate losses for each task and sums them to obtain a single loss to optimize. However, we additionally concatenate the selected morphological analysis representation $\mathtt{ma}_{ij^*}$ to h_i before feeding it into the fully connected network with $\mathtt{tanh}$ outputs as described in Section 3.1. The model is shown in Figure 3a. The rationale of this concatenation is to facilitate information flow from the disambiguated morphological analysis. We call this model JOINT2. The loss function $\mathtt{loss}_{\mathrm{JOINT2}}$ of this model is then calculated similar to $\mathtt{loss}_{\mathrm{JOINT1}}$.

Multilayer and Shortcut Connections. Our most complicated model is employing three Bi-LSTM layers instead of only one. We basically feed the output of the first layer h_i^1 to layer 2, the output of the second layer h_i^2 to layer 3. In addition to this, we transfer the word representation x_i to all layer inputs and concatenate with h_i^{level} to obtain $\overline{h}_i^{level}$. When processing to obtain the third layer's output $\overline{h}_i^3$, we also concatenate the selected morphological analysis representation $\mathtt{ma}_{ij^*}$ to h_i^3 in addition to x_i. This is done to propagate the information gained from the disambiguated morphological analysis to the last layer of the network. We use the first layer's output h_i^1 when calculating $\mathtt{mscore}_i$ as shown to be better for a variety of tasks (Hashimoto et al., 2017). We call this model J_MULTI and depict in Figure 3b.

4 Data

To test our proposed model, we derived a new dataset based on a dataset commonly used in the literature for the NER task for Turkish (Tür et al., 2003). This dataset contains sentences from the online edition of a Turkish national newspaper with NER labels. The creators of the dataset also provide a golden morphological analysis along with each word. However, golden morphological analyses in this dataset are sometimes erroneous. For example, words which are inflections of foreign words are usually problematic. An example is “Hillary’nin” which is the genitive case for the word “Hillary”. It has been incorrectly labeled as if it is in nominal case. Also, when the surface form is a number in some noun case, like “98’e” which is the dative case for the number ninety eight, the morphological analysis is almost always nominal. We believe the reason for this is the incorrect handling of the quote character when preparing the original version.

In our study, we have first divided the training portion of the original dataset into training and development sets. We then augmented these portions using candidate morphological analyses for each word



(a) (i) Model JOINT1: two losses for two tasks sharing a Bi-LSTM. (ii) Model JOINT2: We concatenate the selected morphological analysis' vector representation to the last layer's context vector.

(b) Model J_MULTI: We employ shortcut connections and two more Bi-LSTM layers.

Figure 3: Our joint models: (a) JOINT1 and JOINT2 models (b) J_MULTI model. The symbol ☒ represents the selection of $\mathtt{ma}_{ij*}$.

with a commonly used morphological analyzer (Oflazer, 1994). Unfortunately, the golden morphological analyses in about 5% of the word tokens were not found in these candidate analyses. To mitigate this issue, we listed the most frequent contexts where a specific mismatch happens, selected the most suitable morphological analysis out of the candidates for each context, thus providing a solution to the mismatch. We then automatically corrected all contexts with a mismatch which has a solution in our solution database. Although we tried to give the utmost attention to selecting the best solution, some of our solutions might be problematic. Thus, we share the data, the scripts and the tool which helps the user to select a solution as described for academic use and examination[2]. Unfortunately, there were still left a few hundred mismatches. As providing a solution for them required a lot of manual work and would only save 1-2 sentences for each, we just removed any sentence that contains any of these mismatches. This way, we have retained 25511 out of 28835 sentences in the original dataset for training, 2953 of 3336 for development and 2913 of 3328 for test. By this process, despite losing some of the sentences,

[2] The data can be found at `https://github.com/onurgu/joint-ner-and-md-tagger`

we have built a new dataset with both the NER labels and the candidate morphological analyses which have correct golden labels.

4.1 Training

We implemented the model using the DyNet Neural Network Toolkit in Python. The model parameters are basically the word embeddings, the parameters of Bi-LSTMs, the weights of the fully connected layer $\texttt{FC}_{last}$, and the CRF transition matrix A. We trained by calculating the gradients of the loss for a batch of five sentences consisting of surface forms and its associated NER and/or MD labels and updated the parameters with Adam (Kingma and Ba, 2014) for 50 epochs and reported the performance on test set of the model with the highest development set performance. We applied dropout (Srivastava et al., 2014) with probability 0.5 to the word representations x_i. To facilitate the reproducibility of our work, we also provide our system as a virtual environment[3] that provides the same environment on which we evaluated our system in an open manner.

5 Results

To test our approach, we train and evaluate every model for 10 times and report the mean F1-measure value for named entity recognition and accuracy for morphological disambiguation. This is done to decrease the potential negative effects of random initialization of model parameters as shown in the literature (Reimers and Gurevych, 2017). To accomplish this given our limited computing resources, we set the parameter dimension sizes to 10 and do not employ pretrained word embeddings.

The results are shown in Table 1. We see that the mean NER performance increases in joint models. We see that the JOINT2 model is performing better than just calculating two losses at the last layer as we did in the JOINT1 model. However, applying the Welch's t-test between the JOINT1 and JOINT2 runs does not strongly imply this difference ($p = .24$). Adding multiple Bi-LSTM layers to JOINT2 and obtaining J_MULTI also helped and achieved the best score among our joint models[4]. Employing Welch's t-test confirms the significance of this difference with other joint models, $p < .05$ for each pair.

This work	
Model	Mean F1-measure
NER	81.07
JOINT1	81.28
JOINT2	81.84
J_MULTI	**83.21**
Previous work	
EXT_M_FEAT	**83.47**

Table 1: Evaluation of our models for NER performance with our dataset. We report F1-measure results over the test portion of our dataset averaged over 10 replications of the training with the same hyper parameters.

To make a comparison with a previous method (Gungor et al., 2017), we also evaluated a model where the golden morphological analysis in the corpus is represented as a vector and included in the word representation x_i, namely EXT_M_FEAT (see Section 3.1). As one can see from the table, it achieved the best results compared to our joint models. However, we cannot confirm the difference between EXT_M_FEAT and J_MULTI models as the calculated p is well above .05. Thus our best performing model J_MULTI is performing at a competitive level with an additional advantage of disambiguating the morphological tags while predicting the NER tags. This also serves as another confirmation to the

[3] You can obtain our implementation and find more information about how to use our virtual environment at `https://github.com/onurgu/joint-ner-and-md-tagger`.

[4] One can wonder whether this performance improvement could be due to an increase in the total number of parameters of the model. We saw that the increase is negligible as it only accounted for a 2% increase.

hypothesis that employing linguistic information such as morphological features leads to an increase in the NER performance.

This work	
Model	Mean Accuracy
MD	88.61
JOINT1	88.17
JOINT2	86.86
J_MULTI	88.05
Previous work	
Yuret and Türe (2006)	89.55
Shen et al. (2016)	**91.03**

Table 2: Evaluation of our models for MD performance. As in the NER evaluation, we report accuracies over the test dataset averaged over 10 replications of the training.

To evaluate the performance of morphological disambiguation, we have tested the MD performance of our models, which are trained with the training portion of our dataset, on the test portion of a frequently used dataset (Yuret and Türe, 2006). As can be seen from Table 2, we are very close to the state of the art MD performance even if we only trained with a low number of parameters as stated in the beginning of this section. We have to also note that in contrast with the NER task, the MD task did not enjoy a performance increase from joint learning. The mean morphological disambiguation accuracies for the test portion of our dataset also suggests the same, all hovering around 77% without much change. This might be due to the fact that NER can utilize the disambiguated morphological analysis of a word to predict the correct label, however a correctly predicted NER label does not contribute to the disambiguation of the word's morphology.

6 Conclusions

In this work, we propose a joint model of NER and MD tasks that removes the need for external morphological disambiguators. The method is applicable to every language given that one can provide the candidate morphological analyses for a word, making this approach portable to many languages. We have also shown that joint learning leads to an increase in the NER tagging performance. However, there is more work to do as we are still bound to language specific tools in obtaining the list of candidate morphological analyses. Generating the list of candidate analyses within the model, testing our hypothesis on other morphologically rich languages, and testing with models which have higher number of parameters are left for future work.

Acknowledgements

This study was supported by the Boğaziçi University Research Fund (BAP 13083).

References

Douglas E Appelt, Jerry R Hobbs, John Bear, David Israel, Megumi Kameyama, David Martin, Karen Myers, and Mabry Tyson. 1995. SRI International FASTUS system: MUC-6 test results and analysis. In *Proceedings of the 6th Conference on Message Understanding*, pages 237–248. ACL.

Andrew Eliot Borthwick. 1999. *A Maximum Entropy Approach to Named Entity Recognition*. Ph.D. thesis, New York University, New York, NY, USA.

Hakan Demir and Arzucan Özgür. 2014. Improving named entity recognition for morphologically rich languages using word embeddings. In *Machine Learning and Applications (ICMLA), 2014 13th International Conference on*, pages 117–122. IEEE.

Jenny Rose Finkel, Trond Grenager, and Christopher Manning. 2005. Incorporating non-local information into information extraction systems by Gibbs sampling. In *Proceedings of the 43rd Annual Meeting of ACL*, pages 363–370. ACL.

Yoav Freund and Robert E Schapire. 1999. Large margin classification using the perceptron algorithm. *Machine learning*, 37(3):277–296.

Onur Gungor, Eray Yildiz, Suzan Uskudarli, and Tunga Gungor. 2017. Morphological embeddings for named entity recognition in morphologically rich languages. *arXiv preprint arXiv:1706.00506*.

Honglei Guo, Huijia Zhu, Zhili Guo, Xiaoxun Zhang, Xian Wu, and Zhong Su. 2009. Domain adaptation with latent semantic association for named entity recognition. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the NAACL*, pages 281–289. ACL.

Kazi Saidul Hasan, Vincent Ng, et al. 2009. Learning-based named entity recognition for morphologically-rich, resource-scarce languages. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, pages 354–362. Association for Computational Linguistics.

Kazuma Hashimoto, Caiming Xiong, Yoshimasa Tsuruoka, and Richard Socher. 2017. A joint many-task model: Growing a neural network for multiple NLP tasks. In *EMNLP*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8).

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional LSTM-CRF models for sequence tagging. *arXiv preprint arXiv:1508.01991*.

Kevin Humphreys, Robert Gaizauskas, Saliha Azzam, Chris Huyck, Brian Mitchell, Hamish Cunningham, and Yorick Wilks. 1998. University of Sheffield: Description of the LaSIE-II system as used for MUC-7. In *Proceedings of the Seventh Message Understanding Conferences (MUC-7)*. ACL.

Jing Jiang and Cheng Xiang Zhai. 2007. Instance weighting for domain adaptation in NLP. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics*, volume 7, pages 264–271, Prague. ACL.

Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

John Lafferty, Andrew McCallum, and Fernando Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the eighteenth international conference on machine learning, ICML*, volume 1, pages 282–289.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. *arXiv preprint arXiv:1603.01360*.

Minh-Thang Luong, Quoc V Le, Ilya Sutskever, Oriol Vinyals, and Lukasz Kaiser. 2015. Multi-task sequence to sequence learning. *arXiv preprint arXiv:1511.06114*.

Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF. *arXiv preprint arXiv:1603.01354*.

Andrew McCallum and Wei Li. 2003. Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons. In *Proceedings of the 7th Conference on Natural Language Learning at HLT-NAACL*, volume 4, pages 188–191. Association for Computational Linguistics.

Vinod Nair and Geoffrey E Hinton. 2010. Rectified linear units improve restricted Boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 807–814.

Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Yoav Goldberg, Jan Hajic, Christopher D Manning, Ryan T McDonald, Slav Petrov, Sampo Pyysalo, Natalia Silveira, et al. 2016. Universal dependencies v1: A multilingual treebank collection. In *LREC*.

Kemal Oflazer. 1994. Two-level description of Turkish morphology. *Literary and Linguistic Computing*, 9(2):137–148.

Nils Reimers and Iryna Gurevych. 2017. Optimal hyperparameters for deep LSTM-networks for sequence labeling tasks. *arXiv preprint arXiv:1707.06799*.

Gökhan Akın Seker and Gülşen Eryiğit. 2012. Initial explorations on using CRFs for Turkish named entity recognition. In *Proceedings of COLING 2012*, pages 2459–2474, Mumbai, India, December. The COLING 2012 Organizing Committee.

Qinlan Shen, Daniel Clothiaux, Emily Tagtow, Patrick Littell, and Chris Dyer. 2016. The role of context in neural morphological disambiguation. In *COLING*, pages 181–191.

Miikka Silfverberg, Teemu Ruokolainen, Krister Lindén, and Mikko Kurimo. 2016. FinnPos: an open-source morphological tagging and lemmatization toolkit for Finnish. *Language Resources and Evaluation*, 50(4):863–878.

Anders Søgaard and Yoav Goldberg. 2016. Deep multi-task learning with low level tasks supervised at lower layers. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, volume 2, pages 231–235.

Nitish Srivastava, Geoffrey E Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *Journal of machine learning research*, 15(1):1929–1958.

Milan Straka and Jana Straková. 2017. Tokenizing, POS tagging, lemmatizing and parsing UD 2.0 with UDPipe. *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 88–99.

Jana Straková, Milan Straka, and Jan Hajič. 2016. Neural networks for featureless named entity recognition in Czech. In *International Conference on Text, Speech, and Dialogue*, pages 173–181. Springer.

György Szarvas, Richárd Farkas, and András Kocsor. 2006. A multilingual named entity recognition system using boosting and C4.5 decision tree learning algorithms. In *International Conference on Discovery Science*, pages 267–278. Springer.

Viktor Trón, András Kornai, György Gyepesi, László Németh, Péter Halácsy, and Dániel Varga. 2005. Hunmorph: open source word analysis. In *Proceedings of the Workshop on Software*, pages 77–85. Association for Computational Linguistics.

Gökhan Tür, Dilek Hakkani-Tür, and Kemal Oflazer. 2003. A statistical information extraction system for Turkish. *Natural Language Engineering*, 9(2):181–210.

Dan Wu, Wee Sun Lee, Nan Ye, and Hai Leong Chieu. 2009. Domain adaptive bootstrapping for named entity recognition. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, volume 3, pages 1523–1532. ACL.

Zhilin Yang, Ruslan Salakhutdinov, and William W. Cohen. 2016. Multi-task cross-lingual sequence tagging from scratch. *CoRR*, abs/1603.06270.

Reyyan Yeniterzi. 2011. Exploiting morphology in Turkish named entity recognition system. In *Proceedings of the ACL 2011 Student Session*, HLT-SS '11, pages 105–110, Stroudsburg, PA, USA. Association for Computational Linguistics.

Eray Yildiz, Caglar Tirkaz, H Bahadir Sahin, Mustafa Tolga Eren, and Omer Ozan Sonmez. 2016. A morphology-aware network for morphological disambiguation. In *30th AAAI Conference on Artificial Intelligence*.

Deniz Yuret and Ferhan Türe. 2006. Learning morphological disambiguation rules for Turkish. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, pages 328–334. Association for Computational Linguistics.