Proceedings of the 12th Conference on Language Resources and Evaluation (LREC 2020), pages 1–10
Marseille, 11–16 May 2020

Neural Mention Detection

Juntao Yu,[1] **Bernd Bohnet,**[2] **Massimo Poesio**[1]
[1]Queen Mary University of London, [2]Google Research
juntao.yu@qmul.ac.uk, bohnetbd@google.com, m.poesio@qmul.ac.uk

Abstract

Mention detection is an important preprocessing step for annotation and interpretation in applications such as NER and coreference resolution, but few stand-alone neural models have been proposed able to handle the full range of mentions. In this work, we propose and compare three neural network-based approaches to mention detection. The first approach is based on the mention detection part of a state of the art coreference resolution system; the second uses ELMO embeddings together with a bidirectional LSTM and a biaffine classifier; the third approach uses the recently introduced BERT model. Our best model (using a biaffine classifier) achieves gains of up to 1.8 percentage points on mention recall when compared with a strong baseline in a HIGH RECALL coreference annotation setting. The same model achieves improvements of up to 5.3 and 6.2 p.p. when compared with the best-reported mention detection F1 on the CONLL and CRAC coreference data sets respectively in a HIGH F1 annotation setting. We then evaluate our models for coreference resolution by using mentions predicted by our best model in start-of-the-art coreference systems. The enhanced model achieved absolute improvements of up to 1.7 and 0.7 p.p. when compared with our strong baseline systems (pipeline system and end-to-end system) respectively. For nested NER, the evaluation of our model on the GENIA corpora shows that our model matches or outperforms state-of-the-art models despite not being specifically designed for this task.

Keywords: Mention Detection, Coreference Resolution, Nested Named Entity Recognition, Deep Neural Networks

1. Introduction

Mention detection (MD) is the task of identifying mentions of entities in text. It is an important preprocessing step for downstream applications such as nested named entity recognition (Zheng et al., 2019) or coreference resolution (Poesio et al., 2016); thus, the quality of mention detection affects both the performance of models for such applications and the quality of annotated data used to train them (Chamberlain et al., 2016; Poesio et al., 2019).

Much mention detection research for NER has concentrated on a simplified version of MD that focuses on proper names only (i.e., it doesn't consider as mentions nominals such as *the protein* or pronouns such as *it*), and ignores the fact that mentions may nest (e.g., noun phrases such as *[[CCITA] mRNA]* in the GENIA corpus are mentions of two separate entities, *CCITA* and *CCITA and mRNA* (Alex et al., 2007)). However such simplified view of mentions is not sufficient for NER in domains such as biomedical, or for coreference, that requires full mention detection. Another limitation of typical mention detection systems is that they only predict mentions in a HIGH F1 fashion, whereas in coreference, for instance, mentions are usually predicted in a HIGH RECALL setting, since further pruning will be carried out at the coreference system (Clark and Manning, 2016b; Clark and Manning, 2016a; Lee et al., 2017; Lee et al., 2018; Kantor and Globerson, 2019).

There are only very few recent studies that attempt to apply neural network approaches to develop a standalone mention detector. Neural network approaches using context-sensitive embeddings such as ELMO (Peters et al., 2018) and BERT (Devlin et al., 2019) have resulted in substantial improvements for mention detectors in the NER benchmark CONLL 2003 data set. However, most coreference systems that appeared after Lee et al., (2017; 2018) carry out mention detection as a part of their end-to-end coreference system. Such systems do not output intermediate mentions, hence the mention detector cannot be directly used to extract mentions for an annotation project, or by other coreference systems. Thus the only standalone mention detectors that can be used as preprocessing for a coreference annotation are ones that do not take advantage of these advances and still heavily rely on parsing to identify all NPs as candidate mentions (Björkelund and Kuhn, 2014; Wiseman et al., 2015; Wiseman et al., 2016) or ones that use the rule-based mention detector from the Stanford deterministic system (Lee et al., 2013) to extract mentions from NPs, named entity mentions and pronouns (Clark and Manning, 2015; Clark and Manning, 2016b). To the best of our knowledge, Poesio et al. (2018) introduced the only standalone neural mention detector. By using a modified version of the NER system of Lample et al. (2016), they showed substantial performance gains at mention detection on the benchmark CONLL 2012 data set and on the CRAC 2018 data set when compared with the Stanford deterministic system.

In this paper, we compare three neural architectures for standalone MD. The first system is a slightly modified version of the mention detection part of the Lee et al. (2018) system. The second system employs a bi-directional LSTM on the sentence level and uses biaffine attention (Dozat and Manning, 2017) over the LSTM outputs to predict the mentions. The third system takes the outputs from BERT (Devlin et al., 2019) and feeds them into a feed-forward neural network to classify candidates into mentions and non mentions. All three systems have the options to output mentions in HIGH RECALL or HIGH F1 settings; the former is well suited for the coreference task, whereas the latter can be used as a standard mention detector for tasks like nested named entity recognition. We evaluate our models on the CONLL and the CRAC data sets for coreference mention detection, and on GENIA corpora for nested NER.

The contributions of this paper are therefore as follows. First, we show that mention detection performance improved by up to 1.5 percentage points[1] can be achieved

[1]This performance difference is measured on mention recall,

by training the mention detector alone. Second, our best system achieves improvements of 5.3 and 6.2 percentage points when compared with Poesio et al. (2018)'s neural MD system on CONLL and CRAC respectively. Third, by using better mentions from our mention detector, we can improve the end-to-end Lee et al. (2018) system and the Clark and Manning (2016a) pipeline system by up to 0.7% and 1.7% respectively. Fourth, we show that the state-of-the-art result on nested NER in the GENIA corpus can be achieved by our best model.

2. Related Work

Mention detection.
Despite neural networks having shown high performance in many natural language processing tasks, the rule-based mention detector of the Stanford deterministic system (Lee et al., 2013) remained frequently used in top performing coreference systems that preceded the development of end-to-end architectures (Clark and Manning, 2015; Clark and Manning, 2016a; Clark and Manning, 2016b), including the best neural network coreference system based on a pipeline architecture, (Clark and Manning, 2016a). The Stanford Core mention detector uses a set of predefined heuristic rules to select mentions from NPs, pronouns and named entity mentions. Many other coreference systems simply use all the NPs as the candidate mentions (Björkelund and Kuhn, 2014; Wiseman et al., 2015; Wiseman et al., 2016).
Lee et al. (2017) first introduced a neural network based end-to-end coreference system in which the neural mention detection part is not separated. This strategy led to a greatly improved performance on the coreference task; however, the mention detection component of their system needs to be trained jointly with the coreference resolution part, hence can not be used separately for applications other than coreference. The Lee et al system has been later extended by Zhang et al. (2018), Lee et al. (2018) and Kantor and Globerson (2019). Zhang et al. (2018) added biaffine attention to the coreference part of the Lee et al. (2017) system, improving the system by 0.6%. (Biaffine attention is also used in one of our approaches (BIAFFINE MD), but in a totally different manner, i.e. we use biaffine attention for mention detection while in Zhang et al. (2018) biaffine attention was used for computing mention-pair scores.) In Lee et al. (2018) and Kantor and Globerson (2019), the Lee et al. (2017) model is substantially improved through the use of ELMO (Peters et al., 2018) and BERT (Kantor and Globerson, 2019) embeddings.
Other machine learning based mention detectors include Uryupina and Moschitti (2013) and Poesio et al. (2018). The Uryupina and Moschitti (2013) system takes all the NPs as candidates and trains a SVM-based binary classifier to select mentions from all the NPs. Poesio et al. (2018) briefly discuss a neural mention detector that they modified from the NER system of Lample et al. (2016). The system uses a bidirectional LSTM followed by a FFNN to select mentions from spans up to a maximum width. The system achieved substantial gains on mention F1 when compared with the (Lee et al., 2013) on CONLL and CRAC data sets.
Neural Named Entity Recognition. A subtask of mention detection that focuses only on detecting *named* entity mentions has been studied more frequently. However, most of the proposed approaches treat the NER task as a sequence labelling task, thus cannot be directly applied in tasks that require nested mentions, such as NER in the biomedical domain or coreference. The first neural network based NER model was introduced by Collobert et al. (2011), who used a CNN to encode the tokens and applied a CRF layer on top. After that, many other network architectures for NER MD have also been proposed, such as LSTM-CRF (Lample et al., 2016; Chiu and Nichols, 2016), LSTM-CRF + ELMO (Peters et al., 2018) and BERT (Devlin et al., 2019).
Recently, a number of NER systems based on neural network architectures have been introduced to solve nested NER. Ju et al. (2018) introduce a stacked LSTM-CRF approach to solve nested NER in multi-steps. Sohrab and Miwa (2018) use an exhaustive region classification model. Lin et al. (2019) solve the problem in two steps: they first detect the entity head, and then infer the entity boundaries and classes in the second step. Straková et al. (2019) infer the nested NER by a sequence-to-sequence model. Zheng et al. (2019) introduce a boundary aware network to train the boundary detection and the entity classification models in a multi-task learning setting. However, none of those systems can be directly used for coreference, due to the large difference between the settings used in NER and in coreference (e.g. for coreference the mention need to be predicted in a HIGH RECALL fashion). By contrast, our systems can be easily extended to do nested NER; we demonstrate this by evaluating our system on the GENIA corpus.

3. System architecture

Mention detection is the task of extracting candidate mentions from the document. For a given document D with T tokens, we define all possible spans in D as $N_{i=1}^{I}$ where $I = \frac{T(T+1)}{2}$, s_i, e_i are the start and the end indices of N_i where $1 \leq i \leq I$. The task for an MD system is to assign all the spans (N) a score (r_m) so that spans can be classified into two classes (mention or non mention), hence is a binary classification problem.
In this paper, we introduce three MD systems[2] that use the most recent neural network architectures. The first approach uses the mention detection part from a start-of-the-art coreference resolution system (Lee et al., 2018), which we refer to as LEE MD. We remove the coreference part of the system and change the loss function to sigmoid cross entropy, that is commonly used for binary classification problems. The second approach (BIAFFINE MD) uses a bidirectional LSTM to encode the sentences of the document, followed by a biaffine classifier (Dozat and Manning, 2017) to score the candidates. The third approach (BERT MD) uses BERT (Devlin et al., 2019) to encode the document in the sentence level; in addition, a feed-forward neural network (FFNN) to score the candidate mentions. The three

as we follow Lee et al. (2018) to use fixed mention/token ratio to compare the mentions selected by their joint system.

[2]The code is available at https://github.com/juntaoy/dali-md

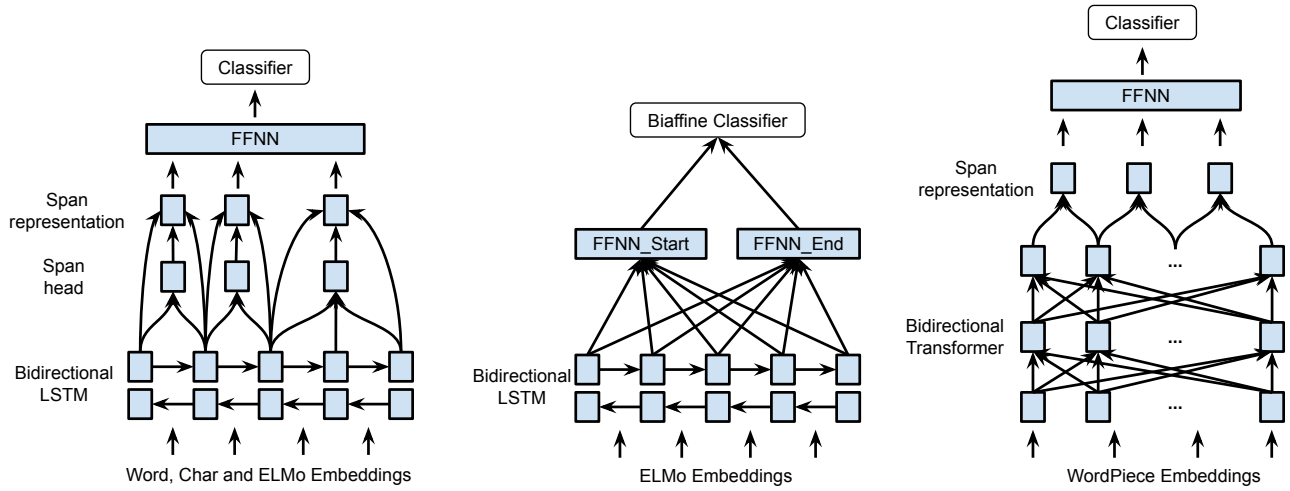

Figure 1: The overall network architectures of our approaches. (a) Our first approach that modified from Lee et al. (2018) coreference system. (b) Our second approach that uses biaffine classifier (Dozat and Manning, 2017). (c) Our third approach that uses BERT (Devlin et al., 2019) to encode the document.

architectures are summarized in Figure 1 and discussed in detail below.

All three architectures are available in two output modes: HIGH F1 and HIGH RECALL. The HIGH F1 mode is meant for applications that require highest accuracy, such as preprocessing for annotation or nested NER. The HIGH RECALL mode, on the other hand, predicts as many mentions as possible, which is more appropriate for preprocessing for a coreference system since mentions can be further filtered by the system during coreference resolution. In HIGH F1 mode we output mentions whose probability $p_m(i)$ is larger then a threshold β such as 0.5. In HIGH RECALL mode we output mentions based on a fixed mention/word ratio λ; this is the same method used by Lee et al. (2018).

3.1. LEE MD

Our first system is based on the mention detection part of the Lee et al. (2018) system. The system represents a candidate span with the outputs of a bi-directional LSTM. The sentences of a document are encoded bidirectional via the LSTMs to obtain forward/backward representations for each token in the sentence. The bi-directional LSTM takes as input the concatenated embeddings ($(x_t)_{t=1}^T$) of both word and character levels. For word embeddings, GloVe (Pennington et al., 2014) and ELMO (Peters et al., 2018) embeddings are used. Character embeddings are learned from convolution neural networks (CNN) during training. The tokens are represented by concatenated outputs from the forward and the backward LSTMs. The token representations $(x_t^*)_{t=1}^T$ are used together with head representations (h_i^*) to represent candidate spans (N_i^*). The h_i^* of a span is obtained by applying an attention over its token representations ($\{x_{s_i}^*, ..., x_{e_i}^*\}$), where s_i and e_i are the indices of the start and the end of the span respectively. Formally, we compute h_i^*, N_i^* as follows:

$$\alpha_t = \text{FFNN}_\alpha(x_t^*)$$

$$a_{i,t} = \frac{exp(\alpha_t)}{\sum_{k=s_i}^{e_i} exp(\alpha_k)}$$

$$h_i^* = \sum_{t=s_i}^{e_i} a_{i,t} \cdot x_t$$

$$N_i^* = [x_{s_i}^*, x_{e_i}^*, h_i^*, \phi(i)]$$

where $\phi(i)$ is the span width feature embeddings.
To make the task computationally tractable, the model only considers the spans up to a maximum length of l, i.e. $e_i - s_i < l$, $(s_i, e_i) \in N$. The span representations are passed to a FFNN to obtain the raw candidate scores (r_m). The raw scores are then used to create the probabilities (p_m) by applying a sigmoid function to the r_m:

$$r_m(i) = \text{FFNN}_m(N_i^*)$$

$$p_m(i) = \frac{1}{1 + e^{-r_m(i)}}$$

For the HIGH RECALL mode, the top ranked λT spans are selected from lT candidate spans ($\lambda < l$) by ranking the spans in a descending order by their probability (p_m). For the HIGH F1 mode, the spans that have a probability (p_m) larger than the threshold β are returned.

3.2. BIAFFINE MD

In our second model, the same bi-directional LSTM is used to encode the tokens of a document in the sentence level. However, instead of using the concatenations of multiple word/character embeddings, only ELMO embeddings are used, as we find in preliminary experiments that the additional GloVe embeddings and character-based embeddings do not improve the accuracy. After obtaining the token representations from the bidirectional LSTM, we apply two

separate FFNNs to create different representations (h_s/h_e) for the start/end of the spans. Using different representations for the start/end of the spans allows the system to learn important information to identify the start/end of the spans separately. This is an advantage when compared to the model directly using the output states of the LSTM, since the tokens that are likely to be the start of the mention and end of the mention are very different. Finally, we employ a biaffine attention (Dozat and Manning, 2017) over the sentence to create a $l_s \times l_s$ scoring metric (r_m), where l_s is the length of the sentence. More precisely, we compute the raw score for span i (N_i) by:

$$h_s(i) = \text{FFNN}_s(x^*_{s_i})$$
$$h_e(i) = \text{FFNN}_e(x^*_{e_i})$$
$$r_m(i) = h_s(i)^\top W_m h_e(i) + h_s(i) b_m$$

where s_i and e_i are the start and end indices of N_i, W_m is a $d \times d$ metric and b_m is a bias term which has a shape of $d \times 1$.

The computed raw score (r_m) covers all the span combinations in a sentence, to compute the probability scores (p_m) of the spans we further apply a simple constrain ($s_i \leq e_i$) such that the system only predict valid mentions. Formally:

$$p_m(i) = \begin{cases} \frac{1}{1+e^{-r_m(i)}} & s_i \leq e_i \\ 0 & s_i > e_i \end{cases}$$

The resulted p_m are then used to predict mentions by filtering out the spans according to different requirements (HIGH RECALL or HIGH F1).

3.3. BERT MD

Our third approach is based on the recently introduced BERT model (Devlin et al., 2019) in which sentences are encoded using deep bidirectional transformers. Our model uses a pre-trained BERT model to encode the documents in the sentence level to create token representations x^*_t. The pre-trained BERT model uses WordPiece embeddings (Wu et al., 2016), in which tokens are further split into smaller word pieces as the name suggested. For example in sentence:

We respect ##fully invite you to watch a special edition of Across China .

The token "respectfully" is split into two pieces ("respect" and "fully"). If tokens have multiple representations (word pieces), we use the first representation of the token. An indicator list is created during the data preparation step to link the tokens to the correct word pieces. After obtaining the actual word representations, the model then creates candidate spans by considering spans up to a maximum span length (l). The spans are represented by the concatenated representations of the start/end tokens of the spans. This is followed by a FFNN and a sigmoid function to assign each span a probability score:

$$N^*_i = [x^*_{s_i}, x^*_{e_i}]$$
$$r_m(i) = \text{FFNN}_m(N^*_i)$$
$$p_m(i) = \frac{1}{1 + e^{-r_m(i)}}$$

We use the same methods we used for our first approach (LEE MD) to select mentions based on different settings (HIGH RECALL or HIGH F1) respectively.

3.4. Nested NER

In our evaluation on nested NER we assign each mention pairs C raw scores (r_m) ($C = 1+$ number of NER categories). The first score indicates the likelihood of a span is *not* a named mentions, the rest of the scores are corresponding to individual NER categories. The probability (p_m) is then calculated by the softmax function instead of the sigmoid function:

$$p_m(i_c) = \frac{e^{r_m(i_c)}}{\sum_{\hat{c}=1}^{C} e^{r_m(i_{\hat{c}})}}$$

3.5. Learning

The learning objective of our mention detectors is to learn to distinguish mentions form non-mentions. Hence it is a binary classification problem, we optimise our models on the sigmoid cross entropy.

$$-\sum_{i=1}^{N} y_i \log p_m(i) + (1 - y_i) \log(1 - p_m(i))$$

where y_i is the gold label ($y_i \in \{0, 1\}$) of i$_{th}$ spans.

For our further experiments on the nested NER we use the softmax cross entropy instead:

$$-\sum_{i=1}^{N} \sum_{c=1}^{C} y_{i_c} \log p_m(i_c)$$

4. Experiments

We ran three series of experiments. The first series of experiments focuses only on the mention detection task, and we evaluate the performance of the proposed mention detectors in isolation. The second series of experiments focuses on the effects of our model on coreference: i.e., we integrate the mentions extracted from our best system into state-of-the-art coreference systems (both end-to-end and the pipeline system). The third series of experiments focuses on the nested NER task. We evaluate our systems both on boundary detection and on the full NER tasks. The rest of this section introduces our experimental settings in detail.

4.1. Data Set

We evaluate our models on two different corpora for both the mention detection and the coreference tasks and one additional corpora for nested NER task, the CONLL 2012 English corpora (Pradhan et al., 2012), the CRAC 2018 corpora (Poesio et al., 2018) and the GENIA (Kim et al., 2003) corpora.

The CONLL data set is the standard reference corpora for coreference resolution. The English subset consists of 2802, 342, and 348 documents for the train, development and test sets respectively. The CONLL data set is not however ideal for mention detection, since not all mentions are annotated, but only mentions involved in coreference

chains of length > 1. This has a negative impact on learning since singleton mentions will always receive negative labels.

The CRAC corpus uses data from the ARRAU corpus (Uryupina et al., 2019). ARRAU consists of texts from four very distinct domains: news (the RST subcorpus), dialogue (the TRAINS subcorpus) and fiction (the PEAR stories). This corpus is more appropriate for studying mention detection as all mentions are annotated. As done in the CRAC shared task, we used the RST portion of the corpora, consisting of news texts (1/3 of the PENN Treebank). Since none of the state-of-the-art coreference systems predict singleton mentions, a version of the CRAC dataset with singleton mentions excluded was created for the coreference task evaluation.

The GENIA corpora is one of the main resources for studying nested NER. We use the GENIA v3.0.2 corpus and preprocess the dataset following the same settings of Finkel and Manning (2009) and Lu and Roth (2015). Historically, the dataset has been split into two different ways: the first approach splits the data into two sets (train and test) by 90:10 (GENIA90), whereas the second approach further creates a development set by splitting the data into 81:9:10 (GENIA81). We evaluate our model on both approaches to make the fair comparisons with previous work. For evaluation on GENIA90, since we do *not* have a development set, we train our model for 40K steps (20 epochs) and take evaluate on the final model.

4.2. Evaluation Metrics

For our experiments on the mention detection or nested NER, we report recall, precision and F1 scores for mentions. For our evaluation that involves the coreference system, we use the official CONLL 2012 scoring script to score our predictions. Following standard practice, we report recall, precision, and F1 scores for MUC, B^3 and $CEAF_{\phi_4}$ and the average F1 score of those three metrics.

4.3. Baseline System

For the mention detection evaluation we use the Lee et al. (2018) system as baseline. The baseline is trained end-to-end on the coreference task and we use as baseline the mentions predicted by the system before carrying out coreference resolution.

For the coreference evaluation we use the top performing Lee et al. (2018) system as our baseline for the end-to-end system, and the Clark and Manning (2016a) system as our baseline for the pipeline system. During the evaluation, we slightly modified the Lee et al. (2018) system to allow the system to take the mentions predicted by our model instead of its internal mention detector. Other than that we keep the system unchanged.

For the nested NER we compare our system with Zheng et al. (2019) and Sohrab and Miwa (2018) on GENIA81 and with Ju et al. (2018), Lin et al. (2019) an Straková et al. (2019) on GENIA90.

4.4. Hyperparameters

For our first model (LEE MD) we use the default settings of Lee et al. (2018). For word embeddings the system uses 300-dimensional GloVe embeddings (Pennington et al., 2014) and 1024-dimensional ELMO embeddings (Peters et al., 2018). The character-based embeddings are produced by a convolution neural network (CNN) which has a window sizes of 3, 4, and 5 characters (each has 50 filters). The characters embeddings (8-dimensional) are randomly initialised and learned during the training. The maximum span width is set to 30 tokens.

For our BIAFFINE MD model, we use the same LSTM settings and the hidden size of the FFNN as our first approach. For word embeddings, we only use the ELMO embeddings (Peters et al., 2018).

For our third model (BERT MD), we fine-tune on the pre-trained $\text{BERT}_{\text{BASE}}$ that consists of 12 layers of transformers. The transformers use 768-dimensional hidden states and 12 self-attention heads. The WordPiece embeddings (Wu et al., 2016) have a vocabulary of 30,000 tokens. We use the same maximum span width as in our first approach (30 tokens).

The detailed network settings can be found in Table 1.

Model	Parameter	Value
LEE, BIA	BiLSTM layers	3
LEE, BIA	BiLSTM size	200
LEE, BIA	BiLSTM dropout	0.4
BER	Transformer layers	12
BER	Transformer size	768
BER	Transformer dropout	0.1
LEE, BIA, BER	FFNN layers	2
LEE, BIA, BER	FFNN size	150
LEE, BIA, BER	FFNN dropout	0.2
LEE, BIA, BER	Embeddings dropout	0.5
LEE, BIA, BER	Optimiser	Adam
LEE, BIA	Learning rate	1e-3
BER	Learning rate	2e-5
LEE, BIA, BER	Training step	40K

Table 1: Major hyperparameters for our models. LEE, BIA, BER are used to indicate LEE MD, BIAFFINE MD, BERT MD respectively.

5. Results and Discussions

In this section, we first evaluate the proposed models in isolation on the mention detection task. We then integrate the mentions predicted by our system into coreference resolution systems to evaluate the effects of our MD systems on the downstream applications. Finally we evaluate our system on the nested NER task.

5.1. Mention Detection Task

Evaluation on the CONLL data set. For mention detection on the CONLL data set, we first take the best model from Lee et al. (2018) and use its default mention/token ratio ($\lambda = 0.4$) to output predicted mentions before coreference resolution. We use this as our baseline for the HIGH RECALL setting. We then evaluate all three proposed models with the same λ as that of the baseline. As a result, the number of mentions predicted by different systems is the same, which means mention precision will be similar. Thus, for

Data	Model	R	P	F1
CONLL	Lee et al. (2018)	96.6	28.2	43.7
	LEE MD	97.3	28.4	44.0
	BIAFFINE MD	**97.5**	**28.5**	**44.1**
	BERT MD	97.3	28.4	44.0
CRAC[3]	Lee et al. (2018)	95.4	34.4	50.6
	LEE MD	96.9	**35.0**	51.4
	BIAFFINE MD	**97.2**	**35.0**	**51.5**
	BERT MD	96.2	34.7	51.0

Table 2: Performance comparison between our mention detectors and the baseline (Lee et al. (2018) system) in a HIGH RECALL setting.

Data	Model	R	P	F1
CONLL	Lee et al. (2013)	89.5	40.4	55.7
	Poesio et al. (2018)	74.0	73.5	73.8
	HIGH RECALL	**97.5**	28.5	44.1
	HIGH F1	76.0	**82.6**	**79.1**
	BALANCE	90.6	53.0	66.9
CRAC[3]	Lee et al. (2013)	67.3	71.6	69.4
	Poesio et al. (2018)	86.2	79.3	82.6
	HIGH RECALL	**95.3**	74.2	83.5
	HIGH F1	87.9	**89.7**	**88.8**

Table 3: Comparison between our BIAFFINE MD and the top performing systems on the mention detection task using the CONLL and CRAC data sets.

Model	Num of Parameters	Infer Speed CONLL	Infer Speed CRAC
LEE MD	4 M	6.8	5.5
BIAFFINE MD	3.4 M	8.3	6.7
BERT MD	110 M	3.3	4.3

Table 4: Model complexity and inference speed (docs/s) comparison between our mention detectors

the HIGH RECALL setting we compare the systems by mention recall. As we can see from Table 2, the baseline system already achieved a reasonably good recall of 96.6%. But even when compared with such a strong baseline, by simply separately training the mention detection part of the baseline system, the stand-alone LEE MD achieved an improvement of 0.7 p.p. This indicates that mention detection task does not benefit from joint mention detection and coreference resolution. The BERT MD achieved the same recall as the LEE MD, but BERT MD uses a much deeper network and is more expensive to train. By contrast, the BIAFFINE MD uses the simplest network architecture among the three approaches, yet achieved the best results, outperforming the baseline by 0.9 p.p. (26.5% error reduction).

Evaluation on the CRAC data set[3] For the CRAC data set, we train the Lee et al. (2018) system end-to-end on the reduced corpus with singleton mentions removed and extract mentions from the system by set $\lambda = 0.4$. We then train our models with the same λ but on the full corpus, since our mention detectors naturally support both mention types (singleton and non-singleton mentions). Again, the baseline system has a decent recall of 95.4%. Benefiting from the singletons, our LEE MD and BIAFFINE MD models achieved larger improvements when compared with the gains achieved on the CONLL data set. The largest improvement (1.8 p.p.) is achieved by our BIAFFINE MD model with an error reduction rate of 39.1%. BERT MD achieved a relatively smaller gain (0.8 p.p.) when compared with the other models; this might be a result of the difference in corpus size between CRAC and CONLL data set. (The CRAC corpus is smaller than the CONLL data set.)

Model complexity and speed To give an idea of the difference in model complexity and inference speed, we listed the number of trainable parameters and the inference speed of our models on CONLL and CRAC in Table 4. The BIAFFINE MD model consists of the lowest number of trainable parameters among all three models, it uses 85% or 3% parameters when compared with the LEE MD and BERT MD respectively. In addition, the BIAFFINE MD is also the fastest model on both CONLL and CRAC datasets, which is able to process 8.3 CONLL or 6.7 CRAC documents per second[4].

Comparison with the State-of-the-art. We compare our best system BIAFFINE MD with the rule-based mention detector of the Stanford deterministic system (Lee et al., 2013) and the neural mention detector of Poesio et al. (2018). For HIGH F1 setting we use the common threshold ($\beta = 0.5$) for binary classification problems without tuning. For evaluation on CONLL we create in addition a variant of the HIGH RECALL setting (BALANCE) by setting $\lambda = 0.2$; this is because we noticed that the score differences between the HIGH RECALL and HIGH F1 settings are relatively large (see Table 3). The score differences between our two settings on CRAC data set are smaller; this might because the CRAC data set annotated both singleton and non-singleton mentions, hence the models are trained in a more balanced way. Overall, when compared with the best-reported system (Poesio et al., 2018), our HIGH F1 settings outperforms their system by large margin of 5.3% and 6.2% on CONLL and CRAC data sets respectively.

5.2. Coreference Resolution Task

We then integrate the mentions predicted by our best system into the coreference resolution system to evaluate the effects of our better mention detectors on the downstream application.

Evaluation with an end-to-end system. We first evaluate our BIAFFINE MD in combination with the end-to-end Lee et al. (2018) system. We slightly modified the system to feed the system mentions predicted by our mention detector. As a result, the original mention selection function

[3]As the Lee et al. (2018) system does not predict singleton mentions, the results on CRAC data set in Table 2 are evaluated without singleton mentions, whereas the results reported in Table 3 are evaluated with singleton mentions included.

[4]All the speed test is conducted on a single GTX 1080ti GPU.

Data	Model	MUC			B^3			$CEAF_{\phi_4}$			Avg. F1
		P	R	F1	P	R	F1	P	R	F1	
CONLL	Lee et al. (2018)	**81.4**	**79.5**	**80.4**	**72.2**	**69.5**	**70.8**	**68.2**	**67.1**	**67.6**	**73.0**
	+ HIGH RECALL	80.0	**79.5**	79.7	70.5	**69.5**	70.0	67.3	66.9	67.1	72.3
	+ HIGH RECALL + joint	80.9	79.2	80.0	72.0	69.0	70.5	67.7	66.9	67.3	72.6
	Lee et al. (2017)	78.4	73.4	75.8	68.6	61.8	65.0	62.7	59.0	60.8	67.2
	+ HIGH RECALL	**78.6**	**74.0**	**76.2**	**68.9**	**62.2**	**65.4**	**63.2**	**59.6**	**61.4**	**67.7**
	Clark and Manning (2016a)	79.2	70.4	74.6	69.9	58.0	63.4	63.5	55.5	59.2	65.7
	+ HIGH RECALL	78.7	72.4	75.4	69.4	59.7	64.2	62.2	**57.7**	59.9	66.5
	+ BALANCE	**80.3**	**72.5**	**76.2**	**71.2**	**60.4**	**65.3**	**64.6**	57.1	**60.6**	**67.4**
CRAC	Lee et al. (2018)	**79.2**	71.9	75.3	**72.4**	63.5	67.7	66.2	58.6	62.2	68.4
	+ HIGH RECALL	76.2	73.1	74.6	68.4	**65.5**	66.9	65.1	61.8	63.4	68.3
	+ HIGH RECALL + joint	77.6	**73.4**	**75.4**	70.4	**65.5**	**67.9**	**66.4**	**61.9**	**64.1**	**69.1**

Table 5: Comparison between the baselines and the models enhanced by our BIAFFINE MD on the coreference resolution task.

is switched off, we keep all the other settings (include the mention scoring function) unchanged. We then train the modified system to obtain a new model. As illustrated in Table 5, the model trained using mentions supplied by our BIAFFINE MD achieved a F1 score slightly lower than the original end-to-end system, even though our mention detector has a better performance.

We think the performance drop might be the result of two factors. First, by replacing the original mention selection function, the system actually becomes a pipeline system, thus cannot benefit from joint learning. Second, the performance difference between our mention detector and the original mention selection function might not be large enough to deliver improvements on the final coreference results. To test our hypotheses, we evaluated our BIAFFINE MD with two additional experiments.

In the first experiment, we enabled the original mention selection function and fed the system slightly more mentions. More precisely, we configured our BIAFFINE MD to output 0.5 mention per token instead of 0.4 i.e. $\lambda = 0.5$. As a result, the coreference system has the freedom to select its own mentions from a candidate pool supplied by our BIAFFINE MD. After training the system with the new setting, we get an average F1 of 72.6% (see table 5), which narrows the performance gap between the end-to-end system and the model trained without the joint learning. This confirms our first hypothesis that by downgrading the system to a pipeline setting does harm the overall performance of the coreference resolution.

For our second experiment, we used the Lee et al. (2017) instead. The Lee et al. (2018) system is an extended version of the Lee et al. (2017) system, hence they share most of the network architecture. The Lee et al. (2017) has a lower performance on mention detection (93.5% recall when $\lambda = 0.4$), which creates a large (4%) difference when compared with the recall of our BIAFFINE MD. We train the system without the joint learning, and the newly trained model achieved an average F1 of 67.7% and this is 0.5 better than the original end-to-end Lee et al. (2017) system (see table 5). This confirms our second hypothesis that a larger gain on mention recall is needed in order to show improvement on the overall system.

We further evaluated the Lee et al. (2018) system on the CRAC data set. We first train the original Lee et al. (2018) on the reduced version (with singletons removed) of the CRAC data set to create a baseline. As we can see from Table 5, the baseline system has an average F1 score of 68.4%. We then evaluate the system with mentions predicted by our BIAFFINE MD, we experiment with both joint learning disabled and enabled. As shown in Table 5, the model without joint learning achieved an overall score 0.1% lower than the baseline, but the new model has clearly a better recall on all three metrics when compared with the baseline. The model trained with joint learning enabled achieved an average F1 of 69.1% which is 0.7% better than the baseline.

Evaluation with a pipeline system. We then evaluated our best model (BIAFFINE MD) with a pipeline system. We use the best-reported pipeline system by Clark and Manning (2016a) as our baseline. The original system used the rule-based mention detector from the Stanford deterministic coreference system (Lee et al., 2013) (a performance comparison between the Lee et al. (2013) and our BIAFFINE MD can be found in Table 3). We modified the preprocessing pipeline of the system to use mentions predicted by our BIAFFINE MD. We ran the system with both mentions from the HIGH RECALL and BALANCE settings, as both settings have reasonable good mention recall which is required to train a coreference system. After training the system with mentions from our BIAFFINE MD, the newly obtained models achieved large improvements of 0.8% and 1.7% for HIGH RECALL and BALANCE settings respectively. This suggests that the Clark and Manning (2016a) system works better on a smaller number of high-quality mentions than a larger number but lower quality mentions. We also noticed that the speed of the Clark and Manning (2016a) system is sensitive to the size of the predicted mentions, both training and testing finished much faster when tested on the BALANCE setting. We did not test the Clark and Manning (2016a) system on the CRAC data set, as a lot of effects are needed to fulfil the requirements of the

Model	R	P	F1
Zheng et al. (2019)	73.6	75.9	74.7
Sohrab and Miwa (2018)	64.0	**93.2**	77.1
Our model	**78.5**	77.8	**78.2**

Table 6: Overall performance comparison on GENIA81 test set.

Categories	R	P	F1	So F1	Zh F1
DNA	74.0	75.3	**74.6**	71.8	70.6
RNA	85.3	84.5	**84.9**	72.4	81.5
protein	82.6	78.5	80.5	**80.8**	76.4
cell line	70.1	77.4	**73.6**	67.9	71.3
cell type	72.1	78.3	75.1	**78.1**	72.5

Table 7: Individual category performance comparison on GENIA81 test set. Zh refers to Zheng et al. (2019) and So refers to Sohrab and Miwa (2018)

preprocessing pipeline, e.g. predicted parse trees, named entity tags. Overall our BIAFFINE MD showed its merit on enhancing the pipeline system.

Summary In summary, our results suggest that the picture regarding using our BIAFFINE MD for coreference *resolution* is more mixed than with coreference *annotation* as discussed in the previous Section (and with nested NER as shown in the following Section). Our model clearly improves the results of best current pipeline system; when used with a top performing end-to-end system, it improves the performance with the CRAC dataset but not with CONLL.

5.3. Nested Named Entity Recognition Task

We then extend our best system (BIAFFINE MD) to do nested NER task.

Evaluation on GENIA81 We first evaluate our system on the split (GENIA81) with the development set. We first run our BIAFFINE MD on boundary detection task which do not require any modification on the system. On boundary detection our system achieved 80.0%, 82.3% and 81.1% for recall precision and F1 score respectively. Our results out perform the previous state-of-the-art system (Zheng et al., 2019) by 2.8% (F1 score).

We then extend our system to predict full NER task, Table 6 and Table 7 show our overall and individual category results on the GENIA81 test set respectively. As we can see from Table 6 our system outperforms the previous state-of-the-art system by 1.1 percentage points. In addition, our system also achieved a much better results on three out of five categories (see Table 7). Overall our system achieved the new state-of-the-art on the GENIA81 data for both boundary detection and full nested NER tasks.

Evaluation on GENIA90 Next, we evaluate our system on the other split of the corpora (GENIA90), in this setting we have a larger training set but do *not* have a development set. After we train our model for 20 epochs, our final model outperforms the previous state-of-the-art by 0.3% (see Table 8). In Table 9 we present our detailed scores for individual categories, since both Lin et al. (2019) and Straková et al. (2019) did not report the detailed scores, we compare our system with Ju et al. (2018). Our system outperforms theirs on all five categories.

Model	R	P	F1
Ju et al. (2018)	71.3	78.5	74.7
Lin et al. (2019)	73.9	75.8	74.8
Straková et al. (2019)	-	-	78.3
Our model	**78.0**	**79.1**	**78.6**

Table 8: Overall performance comparison on GENIA90 test set.

Categories	R	P	F1	Ju F1
DNA	72.4	78.9	**75.5**	72.0
RNA	88.1	86.5	**87.3**	84.5
protein	82.2	79.5	**80.8**	76.7
cell line	67.4	80.0	**73.2**	71.2
cell type	74.4	75.4	**74.9**	72.0

Table 9: Individual category performance comparison on GENIA90 test set. Ju refers to Ju et al. (2018)

6. Conclusions

In this work, we compare three neural network based approaches for mention detection. The first model is a modified version of the mention detection part of a top performing coreference resolution system (Lee et al., 2018). The second model used ELMO embeddings together with a bidirectional LSTM, and with a biaffine classifier on top. The third model adapted the BERT model that based on the deep transformers and followed by a FFNN. We assessed the performance of our models in mention detection, coreference and nested NER tasks. In the evaluation of mention detection, our proposed models reduced up to 26% and 39% of the recall error when compared with the strong baseline on CONLL and CRAC data sets in a HIGH RECALL setting. The same model (BIAFFINE MD) outperforms the best performing system on the CONLL and CRAC by large 5-6% in a HIGH F1 setting. In term of the evaluation on coreference resolution task, by integrating our mention detector with the state-of-the-art coreference systems, we improved the end-to-end and pipeline systems by up to 0.7% and 1.7% respectively. The evaluation on the nested NER task showed that despite our model is not specifically designed for the task, we achieved the state-of-the-art on the GENIA corpora. Overall, we introduced three neural mention detectors and showed that the improvements achieved on the mention detection task can be transferred to the downstream coreference resolution task. In addition, our model is robust enough be used for nested NER task.

7. Acknowledgments

This research was supported in part by the DALI project, ERC Grant 695662.

8. Bibliographical References

Alex, B., Haddow, B., and Grover, C. (2007). Recognising nested named entities in biomedical text. In *Proc. of BioNLP*, pages 65–72.

Björkelund, A. and Kuhn, J. (2014). Learning structured perceptrons for coreference resolution with latent antecedents and non-local features. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 47–57.

Chamberlain, J., Poesio, M., and Kruschwitz, U. (2016). Phrase detectives corpus 1.0 crowdsourced anaphoric coreference. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, Paris, France, may. European Language Resources Association (ELRA).

Chiu, J. P. and Nichols, E. (2016). Named entity recognition with bidirectional lstm-cnns. *Transactions of the Association for Computational Linguistics*, 4:357–370.

Clark, K. and Manning, C. D. (2015). Entity-centric coreference resolution with model stacking. In *Association for Computational Linguistics (ACL)*.

Clark, K. and Manning, C. D. (2016a). Deep reinforcement learning for mention-ranking coreference models. In *Empirical Methods on Natural Language Processing (EMNLP)*.

Clark, K. and Manning, C. D. (2016b). Improving coreference resolution by learning entity-level distributed representations. In *Association for Computational Linguistics (ACL)*.

Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., and Kuksa, P. (2011). Natural language processing (almost) from scratch. *Journal of machine learning research*, 12(Aug):2493–2537.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Annual Conference of the North American Chapter of the Association for Computational Linguistics*.

Dozat, T. and Manning, C. (2017). Deep biaffine attention for neural dependency parsing. In *Proceedings of 5th International Conference on Learning Representations (ICLR)*.

Finkel, J. R. and Manning, C. D. (2009). Nested named entity recognition. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 141–150, Singapore, August. Association for Computational Linguistics.

Ju, M., Miwa, M., and Ananiadou, S. (2018). A neural layered model for nested named entity recognition. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1446–1459, New Orleans, Louisiana, June. Association for Computational Linguistics.

Kantor, B. and Globerson, A. (2019). Coreference resolution with entity equalization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 673–677, Florence, Italy, July. Association for Computational Linguistics.

Kim, J.-D., Ohta, T., Tateisi, Y., and Tsujii, J. (2003). GENIA corpusa semantically annotated corpus for biotextmining. *Bioinformatics*, 19(suppl$_1$) : $i180 - -i182, 07$.

Lample, G., Ballesteros, M., Subramanian, S., Kawakami, K., and Dyer, C. (2016). Neural architectures for named entity recognition. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 260–270. Association for Computational Linguistics.

Lee, H., Chang, A., Peirsman, Y., Chambers, N., Surdeanu, M., and Jurafsky, D. (2013). Deterministic coreference resolution based on entity-centric, precision-ranked rules. *Computational Linguistics*, 39(4):885–916.

Lee, K., He, L., Lewis, M., and Zettlemoyer, L. (2017). End-to-end neural coreference resolution. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*.

Lee, K., He, L., and Zettlemoyer, L. S. (2018). Higher-order coreference resolution with coarse-to-fine inference. In *Proceedings of the 2018 Annual Conference of the North American Chapter of the Association for Computational Linguistics*.

Lin, H., Lu, Y., Han, X., and Sun, L. (2019). Sequence-to-nuggets: Nested entity mention detection via anchor-region networks. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5182–5192, Florence, Italy, July. Association for Computational Linguistics.

Lu, W. and Roth, D. (2015). Joint mention extraction and classification with mention hypergraphs. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 857–867, Lisbon, Portugal, September. Association for Computational Linguistics.

Pennington, J., Socher, R., and Manning, C. (2014). Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., and Zettlemoyer, L. S. (2018). Deep contextualized word representations. In *Proceedings of the 2018 Annual Conference of the North American Chapter of the Association for Computational Linguistics*.

Poesio, M., Stuckardt, R., and Versley, Y. (2016). *Anaphora Resolution: Algorithms, Resources and Applications*. Springer, Berlin.

Poesio, M., Grishina, Y., Kolhatkar, V., Moosavi, N., Roesiger, I., Roussel, A., Simonjetz, F., Uma, A., Uryupina, O., Yu, J., and Zinsmeister, H. (2018). Anaphora resolution with the arrau corpus. In *Proc. of the NAACL Worskhop on Computational Models of Reference, Anaphora and Coreference (CRAC)*, pages 11–22, New Orleans, June.

Poesio, M., Chamberlain, J., Paun, S., Yu, J., Uma, A., and Kruschwitz, U. (2019). A crowdsourced corpus of multiple judgments and disagreement on anaphoric interpretation. In *Proceedings of the 2019 Annual Conference of the North American Chapter of the Association for Computa-*

tional Linguistics.

Pradhan, S., Moschitti, A., Xue, N., Uryupina, O., and Zhang, Y. (2012). CoNLL-2012 shared task: Modeling multilingual unrestricted coreference in OntoNotes. In *Proceedings of the Sixteenth Conference on Computational Natural Language Learning (CoNLL 2012)*, Jeju, Korea.

Sohrab, M. G. and Miwa, M. (2018). Deep exhaustive model for nested named entity recognition. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2843–2849, Brussels, Belgium, October-November. Association for Computational Linguistics.

Straková, J., Straka, M., and Hajic, J. (2019). Neural architectures for nested NER through linearization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5326–5331, Florence, Italy, July. Association for Computational Linguistics.

Uryupina, O. and Moschitti, A. (2013). Multilingual mention detection for coreference resolution. In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, pages 100–108.

Uryupina, O., Artstein, R., Bristot, A., Cavicchio, F., Delogu, F., Rodriguez, K. J., and Poesio, M. (2019). Annotating a broad range of anaphoric phenomena, in a variety of genres: the ARRAU corpus. *Journal of Natural Language Engineering*.

Wiseman, S., Rush, A. M., Shieber, S., and Weston, J. (2015). Learning anaphoricity and antecedent ranking features for coreference resolution. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, volume 1, pages 1416–1426.

Wiseman, S., Rush, A. M., and Shieber, S. M. (2016). Learning global features for coreference resolution. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 994–1004.

Wu, Y., Schuster, M., Chen, Z., Le, Q. V., Norouzi, M., Macherey, W., Krikun, M., Cao, Y., Gao, Q., Macherey, K., et al. (2016). Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.

Zhang, R., Nogueira dos Santos, C., Yasunaga, M., Xiang, B., and Radev, D. (2018). Neural coreference resolution with deep biaffine attention by joint mention detection and mention clustering. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 102–107. Association for Computational Linguistics.

Zheng, C., Cai, Y., Xu, J., Leung, H.-f., and Xu, G. (2019). A boundary-aware neural model for nested named entity recognition. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 357–366, Hong Kong, China, November. Association for Computational Linguistics.