Journal of Biomedical Informatics 70 (2017) 85–91

Contents lists available at ScienceDirect

Journal of Biomedical Informatics

journal homepage: www.elsevier.com/locate/yjbin

Character-level neural network for biomedical named entity recognition

Mourad Gridach

High Institute of Technology, Ibn Zohr University, Agadir, Morocco

ARTICLE INFO

Article history:
Received 10 December 2016
Revised 30 April 2017
Accepted 4 May 2017
Available online 11 May 2017

Keywords:
Deep neural networks
Natural language processing
Biomedical named entity recognition

ABSTRACT

Biomedical named entity recognition (BNER), which extracts important named entities such as genes and proteins, is a challenging task in automated systems that mine knowledge in biomedical texts. The previous state-of-the-art systems required large amounts of task-specific knowledge in the form of feature engineering, lexicons and data pre-processing to achieve high performance. In this paper, we introduce a novel neural network architecture that benefits from both word- and character-level representations automatically, by using a combination of bidirectional long short-term memory (LSTM) and conditional random field (CRF) eliminating the need for most feature engineering tasks. We evaluate our system on two datasets: JNLPBA corpus and the BioCreAtIvE II Gene Mention (GM) corpus. We obtained state-of-the-art performance by outperforming the previous systems. To the best of our knowledge, we are the first to investigate the combination of deep neural networks, CRF, word embeddings and character-level representation in recognizing biomedical named entities.

1. Introduction

Biomedical Named Entity Recognition (BNER) is a critical task for extracting patient information from biomedical texts to support biomedical and translational research. The main aim of BNER is to identify and extract important biomedical concepts such as genes and proteins or also semantic classes such as problems, treatments and lab tests. In recent years, there has been much work focused on extracting named entities from biomedical text, which contains interesting healthcare information. More importantly, biomedical systems that rely on structured data are unable to access directly such information locked in the biomedical text. Building a BNER is not an easy task because of the richness of biomedical text, which makes BNER a challenging task. The first challenge is the ambiguity problem: the same word or sentence can refer to more than one kind of named entities. For example, TNF alpha can refer to a protein or DNA [1]. In addition, various forms can describe the same biomedical named entities (e.g., HIV-1 enhancer versus HIV 1 enhancer).

Various biomedical NER approaches have been developed in general biomedical Natural Language Processing (NLP) systems. Generally, all the previous systems fall into three categories: rule-based approaches, dictionary-based approaches and more recently, machine learning approaches are more investigated in biomedical NER community. Rule-based approaches rely on existing biomedical vocabularies to identify entities. They were the dominant approaches in the early Biomedical NER systems [2–5] as well as some recent work [6–9]. The dictionary-based approaches were widely used because of their simplicity and their performance. A biomedical system based on dictionary approach can extract all the matched entities from a given biomedical text defined in a dictionary. This approach used lemmas to recognize a term by searching the most identical one in the dictionary [10–12]. The last approach uses Machine Learning methods; recently, it is widely used in the biomedical NER community with more annotated data is available. Most state-of-the-art systems used machine learning approach to extract useful features and feature combinations through feature engineering [13].

Machine learning based algorithms consider biomedical NER as a sequence labeling problem where the goal of each algorithm is to find the best label sequence (most of the time as BIO (Begin, Inside, Outside) format) for a given input sentence. Among them, Hidden Markov Models (HMMs) [14,15], Maximum Entropy Markov Models (MEMMs) [16,17], Conditional Random Fields (CRFs) [18–20], Support Vector Machines (SVMs) [21,22] and Structural Support Vector Machines (SSVMs) [23,24]. The best results were obtained by systems using CRFs because they are robust and representative algorithms for sequence labeling tasks such as biomedical NER [25]. In addition, systems based on SVMs obtained state-of-the-art results because they are powerful algorithms and showed better performance for classification tasks [25].

Most of these traditional approaches (HMMs, MEMMs, SVMs, CRFs and SSVMs) have shown significant improvement in term of coverage and robustness but their main shortcoming is that they

E-mail address: m.gridach@uiz.ac.ma

http://dx.doi.org/10.1016/j.jbi.2017.05.002
1532-0464/

rely heavily on a set of manually handcrafted engineering features. These systems are based on hand-crafted rules, orthographic features, external parsers (Part-of-speech (POS) taggers, chunkers, etc.) and external knowledge such as dictionaries. In this paper, we believe that feature engineering requires additional knowledge in biomedical NLP. However, the use of deep neural networks (DNNs) for biomedical named entity recognition has not been extensively evaluated yet. By using DNNs for biomedical NER, we eliminate the need for most feature engineering tasks.

In this paper, we investigate the use of DNNs because they showed better performance in wide areas such as speech recognition [26], image classification [27] and more recently in playing games by winning Go game [28]. Moreover, neural networks achieved state-of-the-art in various NLP applications such as sentiment analysis [29,30], language modeling [31], machine translation [32,33] and speech recognition [34]. Deep neural networks can use backpropagation algorithm for training [35].

We consider biomedical NER as a sequential labeling task, so the main architecture of our model is composed with a bidirectional Long Short-Term Memory (LSTM) network combined with a CRF on the top of the network. By using bidirectional LSTM, our network can capture infinite amount of context on both sides of any biomedical sentence [34] eliminating the main problem of limited context emerged from feed-forward neural networks. Our neural network takes advantage from character-level representation of words and also relies on unsupervised word representations learned from unannotated corpora, which demonstrate its effectiveness for many biomedical NER systems [36,1]. In addition, we used dropout training to encourage the model to learn to rely on both character-level and word embeddings. Dropout is a regularization method used to reduce overfitting. It prevents the neural network adaptation to the training data [37].

We evaluate our model on two publicly available datasets: JNLPBA corpus and the BioCreAtIvE II Gene Mention (GM) corpus. Experimental results show that our model outperforms previous systems and we are able to obtain state-of-the-art performance on the previous datasets without using any large dictionaries and lots of handcrafted engineering features.

The main contributions of this paper are the following:

- As far as we know, we are the first to use deep neural networks combined with conditional random fields to extract biomedical named entities in biomedical texts;
- We get state-of-the-art results and outperform the existing systems on two publicly available datasets.
- Study the impact of bidirectional LSTM and CRF on biomedical Named Entity Recognition;
- The effectiveness of using character-level embeddings and word embeddings in extracting named entities from biomedical texts.

2. Models

In this section, we provide a brief description of the components (layers) of our neural network architecture. We introduce the neural layers in our neural network from the input layer to the output layer.

2.1. LSTM networks

Recurrent neural networks (RNNs) are an extension of a conventional feed-forward neural network. They are a powerful family of connectionist models that capture time dynamics via cycles in the graph. These models can handle sequences of variable length using a recurrent hidden unit whose activation at each time step is dependent on that of the previous one. However, in practice, they fail due to the gradient vanishing/exploding problems [38,39].

More recently, Hochreiter and Schmidhuber [40] propose "Long Short-Term Memory" (LSTM) networks, which are variants of RNNs, to cope with these gradient vanishing problems. Since then, the neural networks community came with some minor changes to the original LSTM unit and they are widely studied and used in both theory and practice. Basically, a LSTM unit computes a weighted sum of the input signal and applies a nonlinear activation function. Fig. 1 gives the basic structure of an LSMT unit. In general, to update an LSTM unit at each time t, the following formulas are used:

$$
\begin{aligned}
i_t &= \sigma(W_i h_{t-1} + U_i x_t + b_i) \\
f_t &= \sigma(W_f h_{t-1} + U_f x_t + b_f) \\
\widetilde{c}_t &= \tanh(W_c h_{t-1} + U_c x_t + b_c) \\
c_t &= f_t \odot c_{t-1} + i_t \odot \widetilde{c}_t \\
o_t &= \sigma(W_o h_{t-1} + U_o x_t + b_o) \\
h_t &= o_t \odot \tanh(c_t)
\end{aligned}
$$

where W_i, W_f, W_c, W_o are the weight matrices for hidden state h_t and U_i, U_f, U_c, U_o denote the weight matrices of different gates for input x_t. b_i, b_f, b_c, b_o denote the bias vectors. x_t is the input vector (e.g. word representation) at time t, and h_t is the hidden state (also called output) vector storing all the useful information at (and before) time t. σ is the element-wise sigmoid function and tanh is the hyperbolic tangent function. It should be noted that these LSTMs networks are trained using backpropagation through time (BPTT) [41].

2.2. Bidirectional LSTM networks

One shortcoming of standard LSTMs is that they are only able to make use of previous context which means that the LSTMs hidden state h_t takes information only from past, knowing nothing about the future. For Named Entity Recognition as a family of sequence labeling tasks, it is beneficial to have access to both left and right contexts. An efficient solution to cope with this problem is by using bidirectional LSTM (BLSTM). The main idea is to process each sequence forwards and backwards which results into two separate hidden states to capture past and future information, respectively. Then the two hidden states are concatenated to form the final output.

Fig. 2 shows a graphical illustration of bidirectional LSTMs with the biomedical sentence "Activation of the CD28 surface receptor" taken from JNLPBA corpus. For a given sentence $(x_1, x_2, \ldots, x_n)$ in the corpus containing n words, we compute two representations: the left context of the sentence at every word t denoted by $\overrightarrow{h}_t$ and the right context of the sentence denoted by $\overleftarrow{h}_t$ by using a second LSTM reading the same sentence in the opposite direction. The final output of any word will be the concatenation of the right and left context $h_t = [\overrightarrow{h}_t, \overleftarrow{h}_t]$.

To illustrate the efficient use of bidirectional LSTMs instead of LSTMs, we run experiments on JNLPBA corpus where the results are shown in Table 3 (Section 5.3). The experimental results showed that bidirectional LSTMs outperformed the LSTM networks which confirms the results reported by previous work on modeling sequences such as Part-of-Speech tagging [42], dependency parsing [43], phoneme classification [44], continuous speech recognition [45,46] and speech synthesis [47].

2.3. BLSTM CRF networks

For sequence labeling tasks, for a given sentence, it is useful to consider the correlations between labels in neighborhoods and jointly decode the best chain of them. We model label sequence

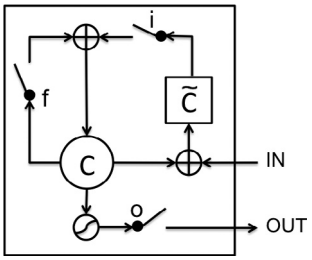

Fig. 1. Long short-term memory unit.

jointly using a conditional random field (CRF) [18], instead of decoding each label independently. Therefore, our final model combines bidirectional LSTM with a CRF model on the top. The rest of this section will describe the details about the architecture of our model. For CRF layer, we use a state transition matrix to predict the current tag. We denote this transition matrix by $T_{i,j}$ representing the transition score from the i-th tag to the j-th tag. For a given sentence X = $(x_1, x_2, \ldots, x_n)$, we denote $M([X]_1^T)_{i,t}$ to be the matrix of scores output by the bidirectional LSTM network for the sentence $[X]_1^T$ and the i-th tag at the t-th word. The sum of the scores from the bidirectional LSTM network along with the transition scores gives the final score for a sentence $[X]_1^T$ and a sequence of tags $[i]_1^T$. The following equation summarizes this final score:

$$s([X]_1^T, [i]_1^T) = \sum_{t=1}^{T} (T_{[i]_{t-1},[i]_t} + M([S]_1^T)_{[i]_t,t}) \quad (1)$$

3. Embeddings philosophy

In this section, we present our philosophy used to deal with word embeddings and character-level based embeddings in order to improve the performance of our model. On the one hand, we show that using character-based model of words is useful especially for biomedical texts. On the other hand, we argue that using word embeddings trained on large unlabeled data to initialize our word vectors improved the performance of our system.

3.1. Character-based models of words

Character-level representations of words has been used in various models in NLP applications and proved their effectiveness to improve the performance of such systems. Recently, it was used in Neural Machine Translation [48–50]. The authors showed that adding character-level features improve the translation performance of their systems. Other NLP applications such as parsing [43], language modeling [51,49] and document classification [52] also used character-level representations, they have shown that using this representation helped their systems to increase the performance. As far as we know, we are the first to use character-level representations to build a biomedical NER system.

Biomedical text is characterized by its richness and could contain more complex words (hyperbilirubinemia and oligonucleotide are a simple examples), which make the process of dealing with biomedical text harder than technical English text. It exhibits large vocabulary sizes and relatively high out-of-vocabulary (OOV) rates on the word level. It has been shown that word-level embeddings suffer from the out-of-vocabulary (OOV) problem, which makes the system unable to generalize on rare and unseen words. Other approaches used morphemes as the subword unit to improve the generalization [53]. The main advantage of using characters compared to morphemes is their direct availability from the original text and can be used without any pre-processing steps.

To add character-level representations to our model, we use the following philosophy: bidirectional LSTMs are used to compute character-based vector embeddings of words in the biomedical text. Hence, each character is represented with an LSTM cell. It should be noted that each character embedding is contained in a lookup table of characters that was initialized randomly. Then, we read words character by character from left to right to compute the first vector embedding (V_f). We compute the second vector embeddings (V_b) using the same method but by starting from the last character. We concatenate (adjoining the two vectors in a larger vector) the first and second vectors to get a vector of the final representation of the word based on its characters (V_r). Then, this vector is concatenated with the word embedding of the same word obtained from a lookup table. We initialized the lookup table using word embeddings (see the next section). A graphical illustration of the new architecture is depicted in Fig. 3.

3.2. Pretrained word embeddings

Word embeddings derived from unlabeled text have been applied to biomedical domain and showed significant improvement on entity recognition in biomedical literature [1,36]. This is due to the fact that in biomedical texts, names appear in regular contexts, which, in general, will be useful for sequence tagging task like biomedical NER. In addition, using character-level embeddings can capture useful morphological information and may provide additional information to the word embeddings [54].

In order to test the performance of pretrained word embeddings in biomedical NER, we carried out experiments with different sets of publicly available word embeddings and compare the results

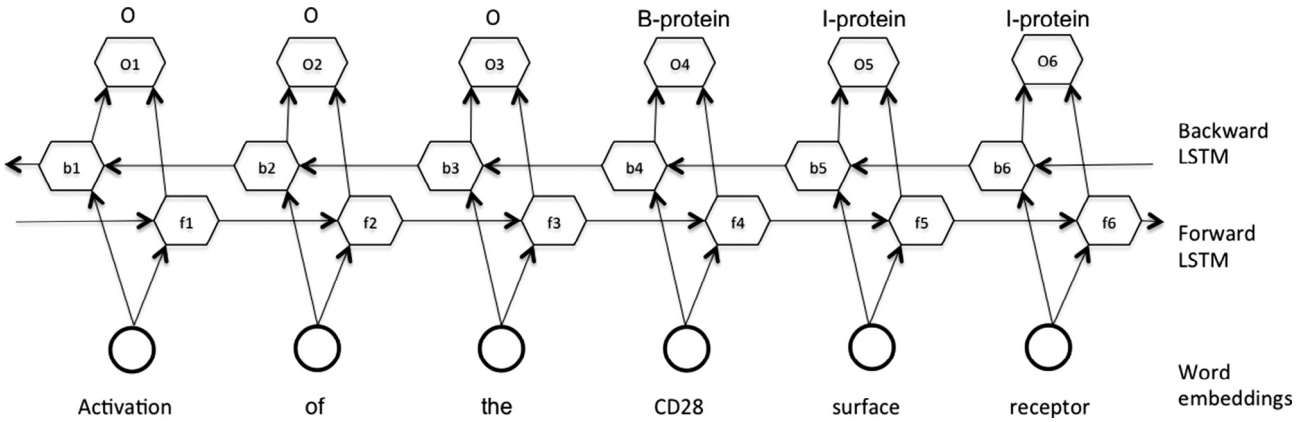

Fig. 2. Biomedical sentence represented by a bidirectional LSTM network.

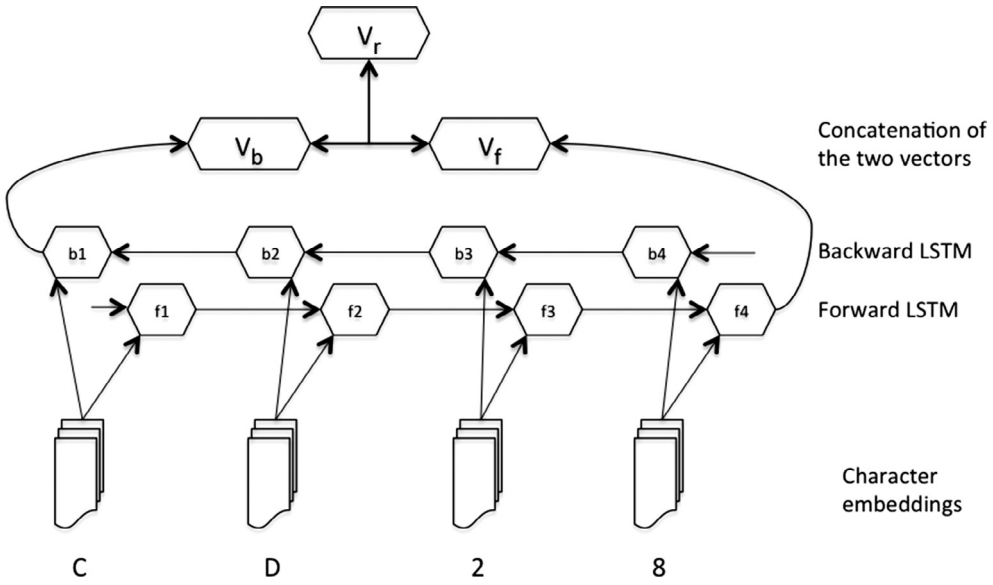

Fig. 3. The character embeddings of the word "CD28" using a bidirectional LSTMs. The final vector V_r is the result of concatenating the vector embedding V_f that represents the forward pass and the vector embedding V_b representing the backward pass.

with a randomly initialized embeddings. All the experiments were done using JNLPBA corpus. The performance of different word embeddings is discussed in coming sections.

4. Training procedure

In this section, we provide details about the procedure used to train our deep neural network. We begin by presenting the final architecture of our model and the implementation details. Then, we give the objective function used for training. Finally, we present the main algorithm used in this paper.

4.1. Final model architecture

The main architecture of our model is illustrated in Fig. 4. It should be noted that the two datasets are divided into sentences separated by a point (.). So, sentences will be the inputs for our neural networks. For each word in a sentence, we used its embeddings concatenated with its word vector obtained from the character-level embeddings as explained in the previous section. Hence, we feed these word embeddings of the sequence of words in a given sentence to the bidirectional LSTM network where it computes the right and the left representation of each word. The two vectors are concatenated and fed to the CRF layer to jointly decode the best label sequence and get predictions for each word in a given sentence. To implement our deep neural network model, we used the Theano library developed by [55].

4.2. Objective function and inference

The main model parameters can be deduced from Eq. (1): parameters obtained from the transition matrix $T_{i,j}$ of bigram scores represent the score of jumping from tag i to tag j. During training, we learned this transition matrix of parameters. In addition, there are also parameters that we learned from the matrix $M([X]_1^T)_{i,t}$ of scores output by the bidirectional LSTM network for the sentence $[X]_1^T$ and the i-th tag at the t-th word. In order to get probabilities for the sequence $[i]_1^T$ from Eq. (1), we use the standard softmax function over all possible tag sequences which results in the following equation:

$$p(y|[X]_1^T) = \frac{\exp(s([X]_1^T, [i]_1^T))}{\sum_{\tilde{c}\in I_X} \exp(s([X]_1^T, \tilde{c}))} \tag{2}$$

where I_x represents all possible tag sequences for a given sentence $[X]_1^T$ with length T. During training, we maximize the log-probability $\log(p(y|[X]_1^T)))$ of the correct tag sequence:

$$\log\left(p(y\Big|[X]_1^T)\right) = \log\left(\frac{\exp(s([X]_1^T, [i]_1^T))}{\sum_{\tilde{c}\in I_s} \exp(s([X]_1^T, \tilde{c}))}\right) \tag{3}$$

$$= s([X]_1^T, [i]_1^T) - \log\left(\sum_{\tilde{c}\in I_s} \exp(s([X]_1^T, \tilde{c}))\right) \tag{4}$$

4.3. The main algorithm

We used Stochastic Gradient Descent (SGD) to train our models. We fixed the learning rate in 0.01. In each epoch, we train one batch at time after dividing the training dataset into mini-batches. Each of these mini-batches contains a set of sentences in the training data with the same number of tokens. We trained our deep neural networks using the backpropagation through time (BBTT) [56] algorithm to update model parameters. We used a gradient clipping of 5.0 in order to reduce the exploding gradient problem [39].

We explored more sophisticated optimization algorithms such as momentum, RMSProp [57], Adam [58] and Adadelta [59], which are widely used in computer vision and have shown better results. The preliminary experiments demonstrate that these methods converge much faster than SGD, but they do not meaningfully improve upon SGD.

5. Experimental results

5.1. Effect of dropout

We apply dropout technique on the final embedding layer before inputting to the bidirectional LSTM. Experimental results showed that using this technique improves the model performance on the two datasets. Table 1 compares results with and without

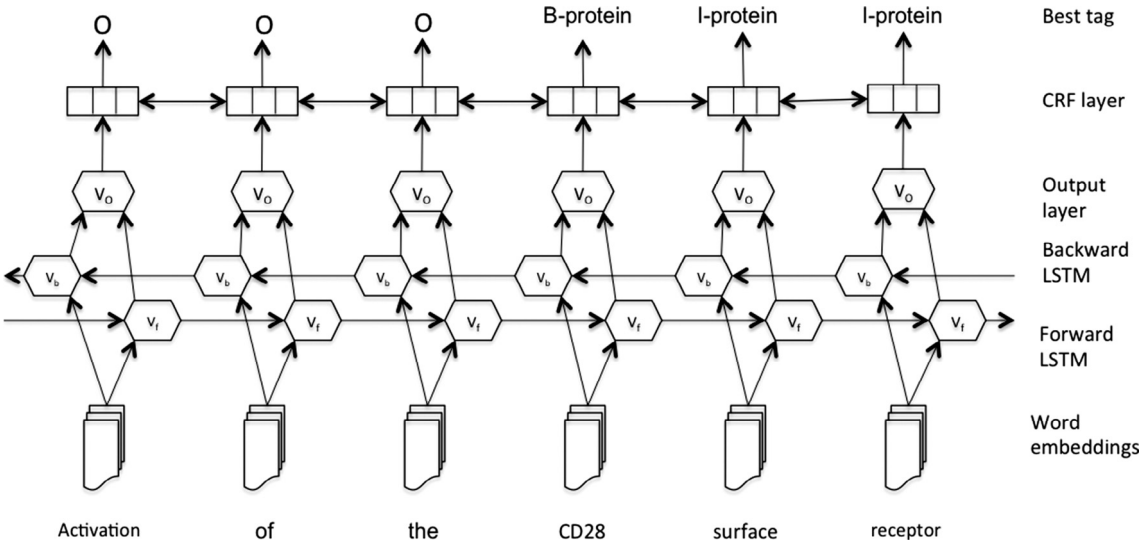

Fig. 4. Word embeddings are fed to a bidirectional LSTM where V_f, V_b respectively represent the forward and backward of a given word. V_o represents the concatenation of the previous two vectors resulting in a representation of a word in its context.

using dropout where all the other hyperparameters remain the same for the chosen best model. The main reason behind this result is by using dropout we encourage the model to belong to both word and character-level representations.

5.2. Word embeddings

In this section, we present the results obtained using different word embeddings implementation: Glove embeddings, two different word2vec embeddings and randomly sample embeddings. For Glove embeddings, we used Stanfords publicly available 100-dimensional embeddings[1] trained on 6 billion words from Wikipedia and web text [60]. We also run experiments on two other sets of published embeddings, namely Googles Word2Vec 300-dimensional embeddings[2] trained on 100 billion words from Google News [61] and word2vec 300-dimentional embeddings[3] developed in the context of biomedical natural language processing by [62] and they are publicly available for the community. These word embeddings were induced from PubMed and PMC texts. The last embeddings carried out in our experiments are the randomly sample embeddings used to test the effectiveness of the previous word embeddings. Random embeddings are uniformly sampled from range $\left[-\sqrt{\frac{3}{d}}, +\sqrt{\frac{3}{d}}\right]$ where d represents the embeddings dimension. Table 2 shows the different results obtained using different word embeddings.

According to the results in Table 2, we got a significant improvements using pretrained word embeddings developed in the biomedical natural language processing task as opposed to the Googles word2vec embeddings, Glove embeddings and randomly sample embeddings. Using word embeddings instead of randomly sample ones confirmed the results reported by previous biomedical named entity recognition systems [1,36].

5.3. Results

We first run experiments on JNLPBA dataset to compare the effectiveness of different models and architectures. The best model will be selected for testing on BioCreAtIvE II GM dataset and compare with other systems. The first part of the experiments was done on four baseline systems: LSTM with pretrained word embeddings, the bidirectional LSTM with pretrained word embeddings, the combination of BLSTM, pretrained word embeddings and character-level embeddings. The BLSTM, character-level embeddings, pretrained word embeddings and dropout will constitute the last combination.

The second part of the experiments concerns adding a CRF layer on the top of the previous network architecture. The results on the JNLPBA dataset are shown in Table 3. It should be noted that these experiments were done using pretrained word embeddings developed in the context of biomedical NLP and were induced from PubMed and PMC texts [62].

According to the results showed in Table 3, the last combination gives us the best performance by reaching an F1-score of 75.87. Using character-level embeddings improves our system by 0.79 points in F1-score and proved that using this representation is very important for biomedical named entity recognition. Adding dropout training also improves the system performance by 0.90 points in F1-score. Finally, adding a CRF layer on the top of the previous network architecture improves the system performance significantly by 1.48 points in F1-score. We select this last model and test it on the BioCreAtIvE II GM corpus. The results are shown in Table 5. For the last model, the computations are run on an Intel core i7 processor. Training takes around 1 day and half on both datasets.

Table 1
Results with and without dropout on both datasets.

	F1-score on JNLPBA corpus	F1-score on BioCreAtIvE II GM corpus
Without dropout	74.97	88.12
With dropout	75.87	89.46

Table 2
Results with different word embeddings choices.

Embeddings	Dimension	F1 Score
Random	100	71.13
Glove	100	74.70
Google word2vec	300	75.20
PubMed and PMC word2vec	300	**75.87**

Value in bold represents the best result.

[1] http://nlp.stanford.edu/projects/glove/.
[2] https://code.google.com/archive/p/word2vec/.
[3] http://bio.nlplab.org/.

Table 3
Results on the JNLPBA dataset.

Models	Precision	Recall	F1 Score
LSTM + WE	68.99	72.72	70.81
BLSTM + WE	72.55	72.87	72.70
BLSTM + WE + char	72.78	74.22	73.49
BLSTM + WE + char + dropout	75.22	73.58	74.39
BLSTM + WE + char + dropout + CRF	74.16	77.66	75.87

5.4. Comparison with the previous work and discussion

In this section, we compare our system with the previous systems well known in the literature. The comparison process will be done on two datasets: JNLPBA and BioCreAtIvE II GM. Table 4 lists the comparison between our system and other models on the JNLPBA corpus. Yao et al. [63] used a multilayer neural network to continuously learn the representation of features, achieving 71.01% F1-score. Tang et al. [1] used word embeddings combined with some features to build their biomedical NER system. Zhou and Su [64] method got 72.55% F1-score which was the best result in that competition. Lishuang Li et al. [20] adopted a bidirectional LSTM network to identify biomedical entities where twin word embeddings and sentence vector are added to rich input information and they achieved 72.76% F1-score. They got the state-of-the-art results on the JNLPBA corpus. By combining bidirectional LSTM, pretrained word embeddings, character-level embeddings and CRF on the top of the network, our model outperforms all the previous models and we obtain the state-of-the-art results by improving the F1-score by 3.11 points.

As far as we know, we are the first to explore the impact of character-level embeddings, pretrained word embeddings and contextual features (CRF) to develop a biomedical Named Entity Recognition system. Adding character-level embeddings allow our system to learn interesting morphological and orthographic features from rich biomedical texts instead of hand-engineering them.

Table 5 presents the comparison between our system and other previous work on the BioCreAtIvE II GM corpus. We compare our model with two other models: [1] achieved 80.96 points in F1-score by using word embeddings with additional features. Lishuang Li et al. [20] achieved 88.61 points in F1-score and outperforms the previous system by a good margin. They used combined a bidirectional LSTM network with word embeddings and sentence vector to identify biomedical entities. Our system outperforms the previous ones by achieving the state-of-the-art results on the BioCreAtIvE II GM corpus. We improved the state-of-the-art results by 0.85 points in F1-score.

Most of the previous biomedical named entity recognition systems rely heavily on hand engineering features and domain specific knowledge [24,1], which is time consuming and needs huge knowledge from linguists and experts in the domain. In addition, some models used large gazetteers or dictionaries [12], which is another drawback of these models. Our model is based on deep neural networks combined with additional sub-models: pretrained word embeddings, character-level embeddings and CRF layer. For unsupervised word embeddings learned from unannotated corpora, they give us a remarkable improvement in the system performance. This performance is consistent with the previous work [1,36]. Using character-level embeddings allow our model to reduce high out-of-vocabulary (OOV) rates on the word level because medical texts can contain complex words where some of them could have more than 15 characters (e.g., hyperbilirubinemia). Adding a CRF layer was useful for our model because it captures the correlations between labels. For a given sentence, it jointly decodes the best sequence of labels. Hence, our model does not rely on any dictionary or large gazetteers which is another advantage compared to the previous systems.

Table 4
Comparison between our model and previous systems on the JNLPBA corpus.

Systems	Precision	Recall	F1 Score
Yao et al. [63]	**76.13**	66.54	71.01
Tang et al. [1]	70.78	72.00	71.39
Zhou and Su [64]	75.99	69.42	72.55
Li et al. [20]	74.77	70.85	72.76
Our system	74.16	**77.66**	**75.87**

Values in bold represent the best result.

Table 5
Comparison between our model and previous systems on the BioCreAtIvE II GM corpus.

Systems	Precision	Recall	F1 Score
Tang et al. [1]	86.54	76.05	80.96
Li et al. [20]	89.54	87.69	88.61
Our system	**90.27**	**88.67**	**89.46**

Values in bold represent the best result.

6. Conclusion

In this paper, we proposed a neural network architecture for biomedical named entity recognition task. Our neural network is based on bidirectional LSTMs, character-level embeddings, pretrained word embeddings and CRF. We outperformed the previous best biomedical NER systems and achieved state-of-the-art performance on two datasets, namely JNLPBA and BioCreAtIvE II GM. Our model is based on powerful deep neural networks that gave us the ability to build a system for Biomedical NER without using any dictionary or gazetteers. Hence, we eliminate the need for most hand feature engineering tasks. Our model used pretrained word embeddings learned from unannotated corpora combined with the character-level embeddings, which make our system able to capture useful orthographic and morphological information and also reduce the Out-of-vocabulary (OOV) problem, which is crucial especially for medical texts containing complex words. Adding dropout training was useful for our model.

References

[1] B. Tang, H. Cao, X. Wang, Q. Chen, H. Xu, Evaluating word representation features in biomedical named entity recognition tasks, BioMed Res. Int. (2014).
[2] C. Friedman, P.O. Alderson, J.H. Austin, J.J. Cimino, S.B. Johnson, A general natural-language text processor for clinical radiology, J. Am. Med. Inform. Assoc. 1 (2) (1994) 161–174.
[3] P.J. Haug, S. Koehler, L.M. Lau, P. Wang, R. Rocha, S.M. Huff, Experience with a mixed semantic/syntactic parser, in: Proceedings of the Annual Symposium on Computer Application in Medical Care, American Medical Informatics Association, 1995, p. 284.
[4] K.-i. Fukuda, T. Tsunoda, A. Tamura, T. Takagi, et al., Toward information extraction: identifying protein names from biological papers, PAC Symp Biocomput, vol. 707, 1998, pp. 707–718.
[5] D. Proux, F. Rechenmann, L. Julliard, V. Pillet, B. Jacq, Detecting gene symbols and names in biological texts, Genome Inform. 9 (1998) 72–80.
[6] Q.T. Zeng, S. Goryachev, S. Weiss, M. Sordo, S.N. Murphy, R. Lazarus, Extracting principal diagnosis, co-morbidity and smoking status for asthma research: evaluation of a natural language processing system, BMC Med. Inform. Decis. Making 6 (1) (2006) 30.
[7] J.C. Denny, R.A. Miller, K.B. Johnson, A. Spickard III, Development and evaluation of a clinical note section header terminology, in: AMIA, 2008.
[8] A.R. Aronson, F.-M. Lang, An overview of metamap: historical perspective and recent advances, J. Am. Med. Inform. Assoc. 17 (3) (2010) 229–236.
[9] G.K. Savova, J.J. Masanz, P.V. Ogren, J. Zheng, S. Sohn, K.C. Kipper-Schuler, C.G. Chute, Mayo clinical text analysis and knowledge extraction system (cTAKES): architecture, component evaluation and applications, J. Am. Med. Inform. Assoc. 17 (5) (2010) 507–513.

[10] K.H.R. Gaizauskas, G. Demetriou, Term recognition and classification in biological science journal articles, in: Proceedings of the Computational Terminology for Medical and Biological Applications Workshop of the 2nd International Conference on NLP, 2000, pp. 37–44.
[11] W.J. Rindfleisch, C. Thomas, L. Tanabe, Edgar: extraction of drugs, genes and relations from the biomedical literature, in: Proc Pacific Symposium on Biocomputing, 2003.
[12] H.Y.M. Song, W.S. Han, Developing a hybrid dictionary-based bio-entity recognition technique, in: Proceedings of the ACM Eighth International Workshop on Data and Text Mining in Biomedical Informatics, Shanghai, China, 2014.
[13] N.C.C. Nobata, J. Tsujii, Automatic term identification and classification in biology texts, in: Proceedings of the 5th NLPRS, 1999, pp. 369–374.
[14] L.R. Rabiner, A tutorial on hidden markov models and selected applications in speech recognition, Proc. IEEE, vol. 77, 1989, pp. 257–285.
[15] S. Zhao, Named entity recognition in biomedical texts using an HMM model, in: Proceedings of the International Joint Workshop on Natural Language Processing in Biomedicine and its Applications, Stroudsburg, PA, USA, 2004, pp. 84–87.
[16] D.F.A. Mccallum, F. Pereira, Maximum entropy markov models for information extraction and segmentation, in: Proceedings of the 17th International Conference on Machine Learning, 2000, pp. 591–598.
[17] J. Finkel, S. Dingare, H. Nguyen, M. Nissim, C. Manning, G. Sinclair, Exploiting context for biomedical entity recognition: from syntax to the web, in: Proceedings of the International Joint Workshop on Natural Language Processing in Biomedicine and its Applications, Association for Computational Linguistics, 2004, pp. 88–91.
[18] J. Lafferty, A. McCallum, F. Pereira, et al., Conditional random fields: probabilistic models for segmenting and labeling sequence data, Proceedings of the Eighteenth International Conference on Machine Learning, ICML, vol. 1, 2001, pp. 282–289.
[19] B. Settles, Biomedical named entity recognition using conditional random fields and rich feature sets, in: Proceedings of the International Joint Workshop on Natural Language Processing in Biomedicine and its Applications, Stroudsburg, PA, USA, 2004, pp. 104–107.
[20] Y.J. Lishuang Li, Liuke Jin, D. Huang, Recognizing biomedical named entities based on the sentence vector/twin word embeddings conditioned bidirectional LSTM, in: Lecture Notes on Computer Science, 2016.
[21] C. Burges, A tutorial on support vector machines for pattern recognition, Data Min. Knowl. Disc. 2 (2) (1998) 21–167.
[22] Y.-C. Wu, T.-K. Fan, Y.-S. Lee, S.-J. Yen, Extracting named entities using support vector machines, in: International Workshop on Knowledge Discovery in Life Science LIterature, Springer, 2006, pp. 91–103.
[23] H.T. Tsochantaridis I, Joachims T, A. Y, Large margin methods for structured and interdependent output variables, J. Mach. Learn. Res. 6 (1) (2005) 1453–1484.
[24] B. Tang, H. Cao, Y. Wu, M. Jiang, H. Xu, Clinical entity recognition using structural support vector machines with rich features, in: Proceedings of the ACM Sixth International Workshop on Data and Text Mining in Biomedical Informatics, ACM, 2012, pp. 13–20.
[25] D. Li, K. Kipper-Schuler, G. Savova, Conditional random fields and support vector machines for disorder named entity recognition in clinical texts, in: Proceedings of the Workshop on Current Trends in Biomedical Natural Language Processing, Association for Computational Linguistics, 2008, pp. 94–95.
[26] G. Hinton, L. Deng, D. Yu, G.E. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T.N. Sainath, Deep neural networks for acoustic modeling in speech recognition: the shared views of four research groups, IEEE Signal Process. Mag. 29 (6) (2012) 82–97.
[27] S.R. Kaiming He, Xiangyu Zhang, J. Sun, Deep Residual Learning for Image Recognition, arXiv preprint.
[28] D. Silver, A. Huang, C.J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, Mastering the game of go with deep neural networks and tree search, Nature 529 (7587) (2016) 484–489.
[29] R. Socher, A. Perelygin, J.Y. Wu, J. Chuang, C.D. Manning, A.Y. Ng, C. Potts, et al., Recursive deep models for semantic compositionality over a sentiment treebank, Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), Citeseer, vol. 1631, 2013, p. 1642.
[30] C. dos Santos, M. Gatti, Deep convolutional neural networks for sentiment analysis of short texts, in: the 25 the International Conference on Computational Linguistics, Dublin, Ireland, 2014.
[31] T. Mikolov, S. Kombrink, L. Burget, J. Černockỳ, S. Khudanpur, Extensions of recurrent neural network language model, in: Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on, IEEE, 2011, pp. 5528–5531.
[32] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, Y. Bengio, Learning Phrase Representations Using RNN Encoder-Decoder for Statistical Machine Translation. Available from: <1406.1078>.
[33] I. Sutskever, O. Vinyals, Q.V. Le, Sequence to sequence learning with neural networks, in: Advances in Neural Information Processing Systems, 2014, pp. 3104–3112.
[34] A. Graves, Generating Sequences with Recurrent Neural Networks, arXiv preprint.
[35] D.E. Rumelhart, G.E. Hinton, R.J. Williams, Learning Internal Representations by Error Propagation, Tech. Rep., DTIC Document, 1985.
[36] Y. Wu, J. Xu, M. Jiang, Y. Zhang, H. Xu, A study of neural word embeddings for named entity recognition in clinical text, AMIA Annual Symposium Proceedings, vol. 2015, American Medical Informatics Association, 2015, p. 1326.
[37] N. Srivastava, G.E. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, Dropout: a simple way to prevent neural networks from overfitting, J. Mach. Learn. Res. 15 (1) (2014) 1929–1958.
[38] Y. Bengio, P. Simard, P. Frasconi, Learning long-term dependencies with gradient descent is difficult, IEEE Trans. Neural Netw. 5 (2) (1994) 157–166.
[39] T.M. Razvan Pascanu, Y. Bengio, On the Difficulty of Training Recurrent Neural Networks. Available from: <1211.5063>.
[40] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural Comput. 9 (8) (1997) 1735–1780.
[41] M. Boden, A Guide to Recurrent Neural Networks and Backpropagation, Technical Report, 2002.
[42] B. Plank, A. Søgaard, Y. Goldberg, Multilingual Part-of-Speech Tagging with Bidirectional Long Short-term Memory Models and Auxiliary Loss. Available from: <1604.05529>.
[43] M. Ballesteros, C. Dyer, N.A. Smith, Improved Transition-based Parsing by Modeling Characters Instead of Words with LSTMs. Available from: <1508.00657>.
[44] A. Graves, S. Fernández, J. Schmidhuber, Bidirectional LSTM networks for improved phoneme classification and recognition, in: International Conference on Artificial Neural Networks, Springer, 2005, pp. 799–804.
[45] Z. Yu, V. Ramanarayanan, D. Suendermann-Oeft, X. Wang, K. Zechner, L. Chen, J. Tao, A. Ivanou, Y. Qian, Using bidirectional LSTM recurrent neural networks to learn high-level abstractions of sequential features for automated scoring of non-native spontaneous speech, in: Automatic Speech Recognition and Understanding (ASRU), 2015 IEEE Workshop on, IEEE, 2015, pp. 338–345.
[46] A. Graves, N. Jaitly, A.-r. Mohamed, Hybrid speech recognition with deep bidirectional LSTM, in: Automatic Speech Recognition and Understanding (ASRU), 2013 IEEE Workshop on, IEEE, 2013, pp. 273–278.
[47] Y. Fan, Y. Qian, F.-L. Xie, F.K. Soong, TTS synthesis with bidirectional LSTM based recurrent neural networks, in: Interspeech, 2014, pp. 1964–1968.
[48] M.-T. Luong, C.D. Manning, Achieving open vocabulary neural machine translation with hybrid word-character models, in: Proceedings of ACL, Berlin, Germany, 2016.
[49] W. Ling, T. Luís, L. Marujo, R.F. Astudillo, S. Amir, C. Dyer, A.W. Black, I. Trancoso, Finding Function in form: Compositional Character Models for Open Vocabulary Word Representation. Available from: <1508.02096>.
[50] K.C. Junyoung Chung, Y. Bengio, A character-level decoder without explicit segmentation for neural machine translation, in: Proceedings of ACL, Berlin, 2016.
[51] D.S. Yoon Kim, Yacine Jernite, A.M. Rush, Character-aware neural language models, CoRR. abs/1508.06615.
[52] Y. Xiao, K. Cho, Efficient Character-level Document Classification by Combining Convolution and Recurrent Layers. Available from: <1602.00367>.
[53] T. Luong, R. Socher, C.D. Manning, Better word representations with recursive neural networks for morphology, in: CoNLL, 2013, pp. 104–113.
[54] R. Soricut, F. Och, Unsupervised morphology induction using word embeddings, in: Proceedings of the NAACL-HLT, Denver, Colorado, 2015.
[55] B. James, B. Olivier, B. Frédéric, L. Pascal, P. Razvan, Theano: a CPU and GPU math expression compiler, in: Proceedings of the Python for Scientific Computing Conference (SciPy), 2010.
[56] R.J. Williams, D. Zipser, Gradient-based learning algorithms for recurrent networks and their computational complexity, Backpropagation: Theor. Architec. Appl. 1 (1995) 433–486.
[57] N.S. Geoffrey Hinton, K. Swersky, Coursera, lecture 6e: rmsprop: divide the gradient by a running average of its recent magnitude, in: Neural Networks for Machine Learning, 2012.
[58] D. Kingma, J. Ba, Adam: A Method for Stochastic Optimization. Available from: <1412.6980>.
[59] M.D. Zeiler, Adadelta: an adaptive learning rate method, CoRR. abs/1212.5701.
[60] R.S. Jeffrey Pennington, C. Manning, Glove: Global vectors for word representation, in: Proceedings of EMNLP, Doha, Qatar, 2014, pp. 1532–1543.
[61] T. Mikolov, I. Sutskever, K. Chen, G.S. Corrado, J. Dean, Distributed representations of words and phrases and their compositionality, in: Advances in Neural Information Processing Systems, 2013, pp. 3111–3119.
[62] S. Pyysalo, F. Ginter, H. Moen, T. Salakoski, S. Ananiadou, Distributional semantics resources for biomedical text processing, in: Proceedings of LBM 2013, 2013, pp. 39–44. URL <http://lbm2013.biopathway.org/lbm2013proceedings.pdf>.
[63] L. Yao, H. Liu, Y. Liu, X. Li, M.W. Anwar, Biomedical named entity recognition based on deep neutral network, Int. J. Hybrid Informat. Technol. 8 (8) (2015) 279–288.
[64] G. Zhou, J. Su, Exploring deep knowledge resources in biomedical name recognition, in: Proceedings of the Joint Workshop on Natural Language Processing in Biomedicine and its Applications, Geneva, Switzerland, 2004.