Effective Use of Bidirectional Language Modeling for Transfer Learning in Biomedical Named Entity Recognition

Devendra Singh Sachan♠ DEVENDRA.SINGH@PETUUM.COM
Pengtao Xie♠ PENGTAO.XIE@PETUUM.COM
Mrinmaya Sachan△ MRINMAYS@CS.CMU.COM
Eric P. Xing♠ ERIC.XING@PETUUM.COM
♠*Petuum Inc, Pittsburgh, PA, USA*
△*Machine Learning Department, CMU, Pittsburgh, PA, USA*

Editor: ...

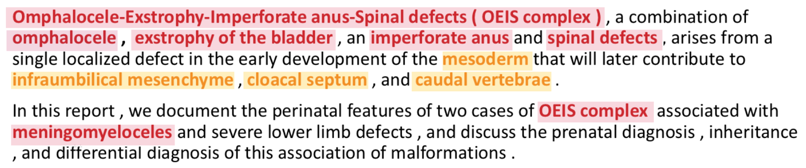
Abstract

Biomedical named entity recognition (NER) is a fundamental task in text mining of medical documents and has many applications. Deep learning based approaches to this task have been gaining increasing attention in recent years as their parameters can be learned end-to-end without the need for hand-engineered features. However, these approaches rely on high-quality labeled data, which is expensive to obtain. To address this issue, we investigate how to use unlabeled text data to improve the performance of NER models. Specifically, we train a bidirectional language model (BiLM) on unlabeled data and transfer its weights to "*pretrain*" an NER model with the same architecture as the BiLM, which results in a better parameter initialization of the NER model. We evaluate our approach on four benchmark datasets for biomedical NER and show that it leads to a substantial improvement in the F1 scores compared with the state-of-the-art approaches. We also show that BiLM weight transfer leads to a faster model training and the pretrained model requires fewer training examples to achieve a particular F1 score.

Keywords: biomedical NER, language modeling, pretraining, bidirectional LSTM, character CNN, CRF

1. Introduction

The field of biomedical text mining has received increased attention in recent years due to the rapid increase in the number of publications, scientific articles, reports, medical records, etc. that are available and readily accessible in electronic format. These biomedical data contains many mentions of biological and medical entities such as chemical ingredients, genes, proteins, medications, diseases, symptoms, etc. Figure 1 shows a medical text that contains seven disease entities (highlighted in red) and four anatomical entities (highlighted in yellow). The accurate identification of such entities in text collections is a very important subtask for information extraction systems in the field of biomedical text mining as it helps in transforming the unstructured information in texts into structured data. Search engines can index, organize, and link medical documents using such identified entities and this can improve medical information access as the users will be able to gather information from many pieces of text. The identification of entities can also be used to mine relations and extract associations from the medical research literature, which can be used in the


Omphalocele-Exstrophy-Imperforate anus-Spinal defects (OEIS complex) , a combination of omphalocele , exstrophy of the bladder , an imperforate anus and spinal defects , arises from a single localized defect in the early development of the mesoderm that will later contribute to infraumbilical mesenchyme , cloacal septum , and caudal vertebrae .

In this report , we document the perinatal features of two cases of OEIS complex associated with meningomyeloceles and severe lower limb defects , and discuss the prenatal diagnosis , inheritance , and differential diagnosis of this association of malformations .

Figure 1: Example of disease and anatomical entities in medical text. Disease entities are highlighted in red and anatomical entities are highlighted in yellow.

construction of medical knowledge graphs (Rotmensch et al., 2017). We refer to this task of identification and tagging of entities in a text as members of predefined categories such as diseases, chemicals, genes, etc. as named entity recognition (NER).

NER has been a widely studied task in the area of natural language processing (NLP) and a number of works have applied machine learning approaches to NER in the medical domain. Building NER systems with high precision and high recall for the medical domain is quite a challenging task due to high linguistic variation in data. First, a dictionary-based approach doing pattern matching will fail to correctly tag ambiguous abbreviations that can belong to different entity types. For example, the term CAT can refer to several phrases—"*chloramphenicol acetyl transferase*," "*computer-automated tomography*," "*choline acetyltransferase*," or "*computed axial tomography*" (Stevenson and Guo, 2010). Second, as the vocabulary of biomedical entities such as proteins is quite vast and is rapidly evolving, it makes the task of entity identification even more challenging and error-prone as it is difficult to create labeled training examples having a wide coverage. Also, in contrast to natural text, entities in the medical domain can have very long names as shown in Figure 1 that can lead an NER tagger to incorrectly predict the tags. Lastly, state-of-the-art machine learning approaches for NER task rely on high-quality labeled data, which is expensive to procure and is therefore available only in limited quantity. Therefore, there is a need for approaches that can use unlabeled data to improve the performance of NER systems.

NER can be devised as a supervised machine learning task in which the training data consists of labels or tags for each token in the text. A typical approach for NER task is to extract word-level features followed by training a linear model for tag classification. To extract features, our NER model makes use of pretrained word embeddings, learned character features, and word-level bidirectional long short-term memory (BiLSTM). The word embeddings are learned from a large collection of PubMed abstracts and it improves the F1 score on NER datasets compared with randomly initialized word vectors. The BiLSTM effectively models the left and right context information around the center word for every time step and this context based representation of a word can help in the disambiguation of abbreviations. The BiLSTM, when applied in combination with character features also maps similar terms like "*lymphoblastic leukemia*," "*null-cell leukemia*," and its varied forms in a latent space that captures the semantic meaning in the phrases. This powerful representation of terms in a latent semantic space can also help in the correct classification of unseen entities as entities with similar contexts are mapped closer together.

In this paper, we propose a transfer learning approach that makes use of unlabeled data to pretrain the weights of NER model using an auxiliary task. Specifically, we do language modeling in both the forward and backward directions to pretrain the weights of NER

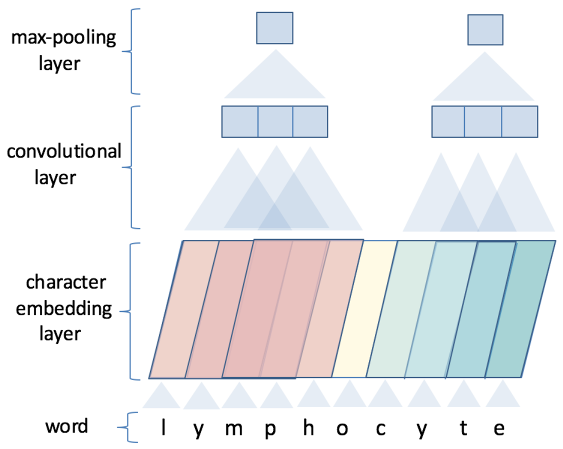


Figure 2: Character CNN block diagram

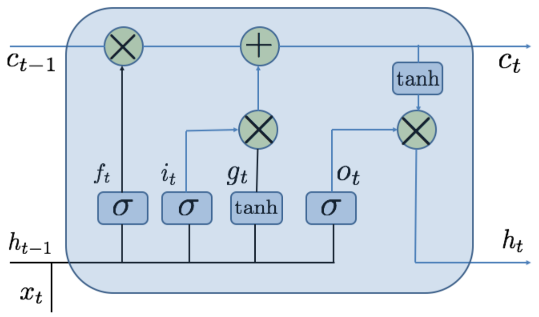


Figure 3: LSTM block diagram

model that is later fine-tuned using the supervised task-specific training data. We believe that such generative model pretraining can prevent overfitting, improve model training, and its convergence speed. We show that such pretraining of weights helps to substantially increase the F1 score on four benchmark datasets for biomedical NER compared with the state-of-the-art approaches. We also observe that BiLM weight transfer leads to faster convergence during the NER model fine-tuning step. As an unsupervised method, our transfer learning approach requires only unlabeled data and thus is generally applicable to different NER datasets compared with the supervised transfer learning approaches that rely on task-specific labeled data to pretrain the model parameters (Lee et al., 2018).

Following this Introduction, the remainder of this paper is organized as follows. Section 2 explains the NER model, its training methodology, and bidirectional language modeling. Section 3 describes the experimental setup such as datasets, model architecture, and the training process. Section 4 reports the results on these datasets and analyzes the performance of the pretrained NER model in detail. Section 5 reviews the related work for biomedical NER. The conclusion, in section 6, summarizes our methods, results, and discusses the future work.

2. Methods

The main building blocks of our neural network based NER model are: character-level convolutional neural network (CNN) layer, word embedding layer, word-level BiLSTM layer, decoder layer, and sentence-level label prediction layer (see Figure 4). During model training, all the layer are jointly trained. Before training, we also pretrain the parameters of the character-CNN, word embedding, and BiLSTM layers in the NER model using the learned parameters from a language model that has the same architecture. Specifically, we perform bidirectional language modeling (BiLM) to pretrain the weights of both the forward and backward LSTMs in the NER model. Next, we will describe these layers in detail.

2.1. Character-Level CNN

CNNs (LeCun et al., 1990) are widely used in computer vision tasks for visual feature extraction (Krizhevsky et al., 2012). In NLP, where the data is mostly sequential, successful applications of CNNs include tasks such as text classification (Kim, 2014) and sequence labeling (Collobert et al., 2011). In this paper, we use CNNs to extract features from characters (Kim et al., 2016) as they can encode morphological and lexical patterns observed in languages.

Similar to the concept of word embedding, each character is represented by an embedding vector. These character embeddings are stored in a lookup table $W_c \in \mathbb{R}^{V_c \times D_c}$, where V_c is the character vocabulary, D_c is the dimensionality of character embeddings. To compute character-level features, we perform 1D convolution along the temporal dimension.[1] Mathematically, this can be written as:

$$z_k[i] = f(W_k * X[:,\ i + s - 1] + b_k),$$

where $*$ is the dot product operator, b_k is the bias, $X \in \mathbb{R}^{D_c \times w_\ell}$ is the character-based embedding representation of a word, w_ℓ is the length of a word, W_k are filter weights, s is the convolution stride, f can be any nonlinear function such as tanh or rectified linear units ($f(x) = \max(0,\ x)$). To capture the important features of a word, multiple filters of different strides are used. Finally, the maximum value is computed over the time dimension also called *max-pooling* to get a single feature for every filter weight. All the features are concatenated to obtain character-based word representation v^w_{char}. A block diagram of character-level CNN is shown in Figure 2.

2.2. Word-Level Bidirectional LSTM

Recurrent neural network (Werbos, 1988) such as LSTM (Hochreiter and Schmidhuber, 1997) is widely used in NLP because it can model the long-range dependencies in language structure with their memory cells and explicit gating mechanism. The dynamics of an LSTM cell is controlled by an input vector (x_t), a forget gate (f_t), an input gate (i_t), an output gate (o_t), a cell state (c_t), and a hidden state (h_t), which are computed as:

$$\begin{aligned}
i_t &= \sigma(W_i * [h_{t-1}, x_t] + b_i) \\
f_t &= \sigma(W_f * [h_{t-1}, x_t] + b_f) \\
o_t &= \sigma(W_o * [h_{t-1}, x_t] + b_o) \\
g_t &= \tanh(W_g * [h_{t-1}, x_t] + b_g) \\
c_t &= f_t \odot c_{t-1} + i_t \odot g_t \\
h_t &= o_t \odot \tanh(c_t),
\end{aligned}$$

where c_{t-1} and h_{t-1} are the cell state and hidden state respectively from previous time step, σ is the sigmoid function ($\frac{1}{1+e^{-x}}$), tanh is the hyperbolic tangent function ($\frac{e^x - e^{-x}}{e^x + e^{-x}}$),

1. To have a uniform length, each word is right-padded with a special padding token so that the length of every word is the same as that of the longest word in every mini-batch. The embedding of the padding character is always a zero vector.

$\odot$ denotes element-wise multiplication. Figure 3 shows the block diagram of an LSTM cell. The parameters of the LSTM are shared for all the time steps.

In our NER model, the word embeddings (v^{w}_{emb}) and the character CNN features of a word (v^{w}_{char}) are concatenated and is given as input to the sequence encoder ($x_t = [v^{w}_{emb}, v^{w}_{char}]$). The sequence encoder consists of a forward LSTM and a backward LSTM, which is also known as bidirectional LSTM (BiLSTM) (Schuster and Paliwal, 1997). The input to the backward LSTM cell is the reversed order of words in the sequence.

2.3. Word-Level Likelihood

For every word, hidden state representations from BiLSTM are concatenated ($h_t = [\overrightarrow{h_t}, \overleftarrow{h_t}]$) and are fed to the decoder layer. The decoder layer computes an affine transformation of the hidden states

$$d_t = W_d h_t + b,$$

where H is the dimensionality of the BiLSTM hidden states, T is the total number of tags, $W_d \in \mathbb{R}^{T \times H}$ and b are learnable parameters. Decoder outputs are referred to as *logits* in the subsequent discussions. To compute the probability of a tag ($\hat{y}_t$) for a word, softmax function is used

$$p(\hat{y}_t = j \mid w_t) = \text{softmax}(d_t).$$

Let $\mathbf{y} = \{y_1, y_2, \ldots, y_N\}$ denote the sequence of tags in the training corpus, then the cross-entropy loss is calculated as:

$$\text{CE}_{ner}(\mathbf{y}, \hat{\mathbf{y}}) = -\sum_{t=1}^{N} \sum_{j=1}^{T} \mathbb{1}(y_t = \hat{y}_{t,j}) \log \hat{y}_{t,j}$$

Figure 4 shows the block diagram of our NER model. To learn the model parameters, average cross-entropy loss is minimized by backpropagation through time (BPTT) (Werbos, 1990). When the word-level likelihood is minimized to train the NER model, it is denoted as CNN-BiLSTM.

2.4. Sentence-Level Likelihood

A drawback of optimizing word-level likelihood is that it ignores the dependencies between other neighboring tags in the sentence. A better strategy is to model the entire sentence structure using a conditional random field (CRF). A CRF is a log-linear graphical model (Lafferty et al., 2001) that additionally considers the transition score from one tag to the next tag. This encourages valid transition paths among the tags based on the learned transition parameters ($W^{crf} \in \mathbb{R}^{T \times T}$). During training, we maximize the log-likelihood for the entire sentence.[2] Mathematically, this can be written as:[3]

$$\log p(\mathbf{y} \mid \mathbf{d}) = s(\mathbf{d}, \mathbf{y}) - \log \sum_{\mathbf{y}' \in S^m} e^{s(\mathbf{d}, \mathbf{y}')},$$

2. This can be done in polynomial time using the *forward-backward* algorithm (see Collins, 2013b).
3. For a detailed derivation of CRF, see Collins (2013a).

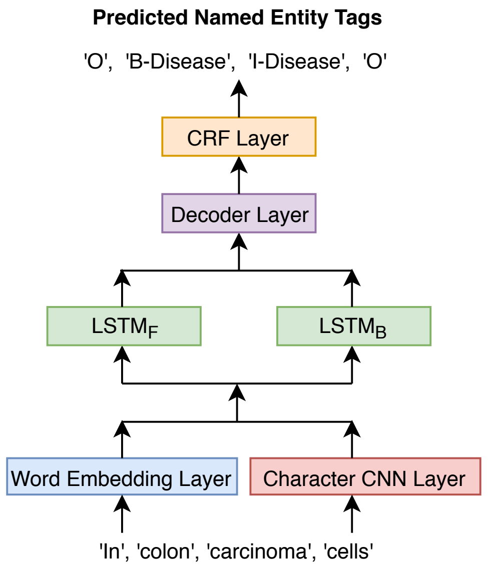

Predicted Named Entity Tags
'O', 'B-Disease', 'I-Disease', 'O'
CRF Layer
Decoder Layer
LSTM_F
LSTM_B
Word Embedding Layer
Character CNN Layer
'In', 'colon', 'carcinoma', 'cells'

Figure 4: NER model architecture.

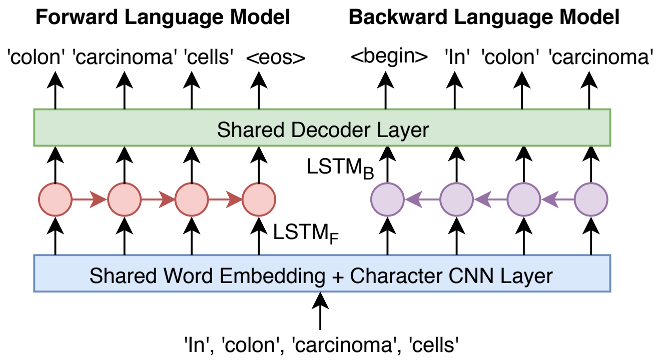

Forward Language Model
Backward Language Model
'colon' 'carcinoma' 'cells' <eos>
<begin> 'In' 'colon' 'carcinoma'
Shared Decoder Layer
LSTM_B
LSTM_F
Shared Word Embedding + Character CNN Layer
'In', 'colon', 'carcinoma', 'cells'

Figure 5: BiLM architecture.

where S^m is the set containing all possible tag combinations for a sentence, $s(\mathbf{d}, \mathbf{y})$ is a scoring function defined as:

$$s(\mathbf{d}, \mathbf{y}) = \sum_{t=1}^{N-1} W^{crf}_{y_t, y_{t+1}} + \sum_{t=1}^{N} d_{t,y_t}.$$

In this paper, when the logits d_{t,y_t} are fed to the CRF layer to optimize sentence likelihood, we call the NER model as CNN-BiLSTM-CRF. During inference, we use the Viterbi algorithm (Forney, 1973) to find the best tag sequence that maximizes the sentence likelihood.

2.5. Language Modeling

Here, we provide a short description of language modeling, as its parameters are used to initialize the NER model. In language modeling, the task is to train a model that maximizes the likelihood of a given sequence of words. At every step, a language model computes the probability of the next word in the sequence given all the previous words. If the sequence of words is $w_1, w_2, \dots, w_n$, its likelihood is given as

$$p_f(w_1, w_2, \dots, w_n) = \prod_{i=2}^{n+1} p(w_i \mid w_1, \dots, w_{i-1}),$$

where w_{n+1} is a special symbol for the end of a sequence. LSTM can be used to predict the probability of the next word given the current word and the previous sequence of words (Graves, 2013). This is done by applying an affine transformation to the hidden states of LSTM at every time step to obtain the logits for all the words in the vocabulary. We refer to this approach as the *forward language model* (LM_f).

We can also model the reversed sequence of words in a similar manner. In this, we compute the probability of the reversed sequence as:

$$p_b(w_n, w_{n-1}, \ldots, w_1) = \prod_{i=n-1}^{i=0} p(w_i \mid w_{i+1}, \ldots, w_n),$$

where w_0 is a special symbol for the start of the sequence. We refer to this approach as the *backward language model* (LM_b). The network architecture of both LM_f and LM_b is similar to the NER model (see Figure 5). While training, both LM_f and LM_b share the parameters of the word embedding layer, character embedding layer, character CNN filters, and the decoder layer. We refer to this as the *bidirectional language model* (BiLM). To learn the parameters of the BiLM, we perform joint training by minimizing the average cross-entropy losses of both the forward and backward language models

$$\mathrm{CE}_{\ell m} = -\lambda_{\ell m}(\log p_f(\mathbf{w}_{1:n}) + \log p_b(\mathbf{w}_{n:1})).$$

3. Experimental Setup

In this section, we will first describe the datasets, their preprocessing, and performance evaluation criteria. Next, we will discuss the architecture of NER model and language model followed by their training details.

3.1. Dataset Preparation and Evaluation

We evaluate our proposed approach on four datasets: NCBI-disease (Doğan and Lu, 2012), BioCreative V Chemical Disease Relation Extraction (BC5CDR) task (Li et al., 2016), BC2GM (Ando, 2007), and JNLPBA (Kim et al., 2004). For each dataset, we use the training, development, and test set splits according to Crichton et al. (2017).[IV] An overall summary of these datasets such as the number of sentences, words, and entities is presented in Table 1. For each dataset, we use its training and development splits as unlabeled data for language modeling task.

We use a special token for the numbers and preserve case information. In all our experiments, we use IOBES tagging format (Borthwick, 1999) for the output tags. For evaluation, we report the precision, recall, and F1 scores for all the entities in the test set. We do exact matching of entity chunks to compute these metrics. For each dataset, we tune the hyperparameters of our model on the development set. Final training is done on both the training and development sets. We use PyTorch framework (Paszke et al., 2017) for all our experiments.

3.2. Model Architecture Details

As we use language model weights to initialize the parameters of the NER model, both the models have identical configurations except for the top decoder layer. Dimensions of character embeddings and word embeddings are set to 50 and 300 respectively. CNN filters

IV. For our experiments, we use the datasets publicly available at `https://github.com/cambridgeltl/MTL-Bioinformatics-2016`.

Properties	NCBI-disease	BC5CDR	BC2GM	JNLPBA
Entity type	Disease	Disease, Chemical	Gene/Protein	5 NEs
# Entity mentions	6,892	5,818	24,583	51,301
# Sentences	7,295	13,907	20,000	24,806
# Words	184,552	360,315	1,139,824	568,786
# Train documents	593	500	-	1,800
# Dev documents	100	500	-	200
# Test documents	100	500	-	404

Table 1: General statistics of the datasets used in this work. '#' symbol stands for the term 'number of'. Entity types in JNLPBA dataset consists of protein, DNA, RNA, cell-type, and cell-line.

have widths (w) in the range from 1 to 7. The number of filters are computed as a function of filter width as $\min(200,\ 50 * w)$. The hidden state of LSTM has 256 dimensions. As the decoder layer is not shared between NER model and language model, the dimensions of the decoder layer are different for each of them. For NER model, as it concatenates the hidden states of forward and backward LSTM to give input to the decoder layer, the dimensions of the decoder matrix are $W_d^{ner} \in \mathbb{R}^{512\times T}$. For language model, dimensions of the decoder matrix are $W_d^{\ell m} \in \mathbb{R}^{256\times V}$ where V is the vocabulary size.

3.3. Language Model Training

We initialize the weights of word embedding layer for the BiLM task using pretrained word vectors. These vectors were learned using skip-gram (Mikolov et al., 2013) method[V] applied to a large collection of PubMed abstracts.[VI]. The embeddings of out-of-vocabulary words are uniformly initialized. LSTM parameters are also uniformly initialized in the range $(-0.005, 0.005)$. For all the other model parameters, we use Xavier initialization (Glorot and Bengio, 2010).

For model training, we use mini-batch SGD with a dynamic batch size of 500 words. At the start of every mini-batch step, the LSTM starts from zero initial states. We do sentence-level language modeling and the network is trained using BPTT. We use Adam optimizer (Kingma and Ba, 2014) with default parameters settings and decay the learning rate by 0.5 when the model's performance plateaus. We train the model for 20 epochs and do early stopping if the perplexity doesn't improve for 3 consecutive epochs. To regularize the model, we apply dropout (Srivastava et al., 2014) with probability 0.5 on the word embeddings and LSTM's hidden states. To prevent the gradient explosion problem, we do gradient clipping by constraining its L_2 norm to be less than 1.0 (Pascanu et al., 2013).

3.4. NER Model Training

To pretrain the NER model, we remove the top decoder layer of the BiLM and transfer the remaining weights to the NER model with the same architecture. Next, we fine-tune

V. We learn word embeddings using the *word2vec* toolkit: `https://code.google.com/p/word2vec/`
VI. These PubMed abstracts are available from the BioASQ Task 4a challenge (Tsatsaronis et al., 2015).

Dataset	Metric	Benchmark	FFN	BiLSTM	MTM-CW	BiLM-NER
NCBI-disease	Precision	85.10	-	86.11	85.86	**86.41**
	Recall	80.80	-	85.49	86.42	**88.31**
	F1	82.90	80.46	85.80	86.14	**87.34**
BC5CDR	Precision	89.21	-	87.60	**89.10**	88.10
	Recall	84.45	-	86.25	88.47	**90.49**
	F1	86.76	83.90	86.92	88.78	**89.28**
BC2GM	Precision	-	-	81.57	**82.10**	81.81
	Recall	-	-	79.48	79.42	**81.57**
	F1	-	73.17	80.51	80.74	**81.69**
JNLPBA	Precision	69.42	-	71.35	70.91	**71.39**
	Recall	75.99	-	75.74	76.34	**79.06**
	F1	72.55	70.09	73.48	73.52	**75.03**

Table 2: Precision, recall, and F1 scores of our proposed BiLM pretrained NER model (last column) and recent state-of-the-art models. We use the CNN-BiLSTM-CRF architecture for our NER model in all the experiments. Source of benchmark performance scores of datasets are: NCBI-disease: Leaman and Lu (2016); BC5CDR: Li et al. (2015); JNLPBA: GuoDong and Jian (2004); **MTM-CW** was proposed in Wang et al. (2018a); **FFN (Feed-forward network)** was proposed in Crichton et al. (2017); **BiLSTM** was proposed in Habibi et al. (2017). The performance scores for these NER models are referred from Wang et al. (2018a).

the pretrained NER model using Adam optimizer (Kingma and Ba, 2014). In contrast to random initialization, during fine-tuning, the pretrained weights act as the starting point for the optimizer. We use mini-batch SGD with a dynamic batch size of $1,000$ words and train the model for 50 epochs. Other settings are similar to the language model training procedure as described above.

4. Results

In this section, we first evaluate the BiLM pretrained NER model on four biomedical datasets and compare the results with the state-of-the-art models. Next, we analyze different variations of NER model pretraining and also do three experiments to study the properties of pretrained NER model. Finally, in a case study on NCBI-disease dataset, we analyze the model's predictions on disease entities. We use the CNN-BiLSTM-CRF architecture for NER model in all our the experiments unless specified otherwise.

4.1. Performance on Benchmark Datasets

We compare our proposed BiLM pretrained NER model with state-of-the-art NER systems such as the multi-task models of Crichton et al. (2017), Wang et al. (2018a), and pretrained embedding based method of Habibi et al. (2017). We show the precision, recall, and F1 scores of the models for all the above datasets in Table 2. From the results, we see that the approach of BiLM pretraining obtains the maximum F1 score for all the datasets. For NCBI-disease dataset, the F1 score of our model is 87.34%, which is an absolute improvement of

Dataset	Metric	No pretrain	LM_f pretrain	LM_b pretrain	BiLM pretrain
NCBI-disease	Precision	84.38	84.62	84.75	**86.41**
	Recall	87.37	87.89	88.00	**88.31**
	F1	85.35	86.22	86.34	**87.34**
BC5CDR	Precision	**88.95**	88.67	88.12	88.10
	Recall	88.64	89.28	89.41	**90.60**
	F1	88.79	88.97	88.76	**89.28**
BC2GM	Precision	81.40	**82.00**	81.04	81.81
	Recall	79.89	80.56	80.12	**81.57**
	F1	80.62	81.27	80.58	**81.69**
JNLPBA	Precision	71.23	70.51	71.00	**71.39**
	Recall	76.52	77.11	76.98	**79.06**
	F1	73.78	73.66	73.87	**75.03**

Table 3: Precision, recall, and F1 scores for different variations of our proposed model.

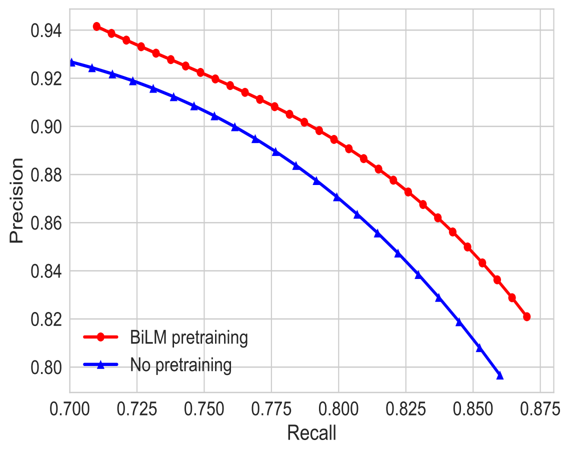
1.20% over the multi-task learning method of Wang et al. (2018a), in which they train the NER model jointly on all the datasets combined together. Similarly, for other datasets, we can see that our proposed approach outperforms other benchmark systems by a significant margin. We want to mention here that our model was trained only on the provided data for a particular dataset compared with the multi-task learning methods, which require a collection of labeled data to improve their performance. This also highlights the importance of doing pretraining of the model weights as this can improve their generalization ability on the test set.

4.2. Model Variations Based on Weights Pretraining

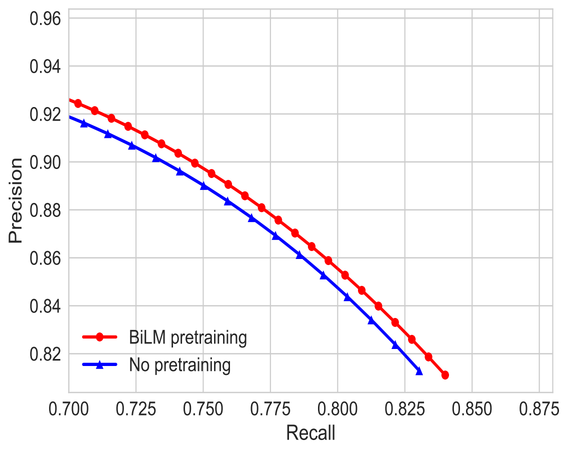
We also compare the performance of the following methods that are based on different parameter initialization strategies for the NER model.

- **No pretraining:** We randomly initialize the parameters of the NER model except word embeddings followed by supervised training.
- **LM_f pretraining**: We initialize the parameters of the NER model using the forward language model weights. The parameters of backward LSTM, decoder, and CRF are randomly initialized.
- **LM_b pretraining:** We initialize the parameters of the NER model using the backward language model weights. The parameters of forward LSTM, decoder, and CRF are randomly initialized.
- **$BiLM$ pretraining:** In this, the parameters of the NER model are initialized using the bidirectional language model weights. The parameters of the decoder and CRF are randomly initialized.

We show the results of the above variations in model pretraining in Table 3. Our model gives an absolute improvement of around 2% and 0.5% in F1 score on NCBI-disease



(a) NCBI-disease dataset



(b) BC5CDR dataset

Figure 6: Smoothed precision-recall curves for models with BiLM pretraining and no pretraining. Best viewed in color.

and BC5CDR dataset respectively over the model with no pretraining. We note that for all the datasets, LM_f pretraining and LM_b pretraining also gives an improvement over no pretraining. From the results, we also observe that the BiLM pretraining achieves better F1 score and precision in comparison to LM_f pretraining and LM_b pretraining, thus highlighting the importance of performing language modeling in both directions.

4.3. Model Studies

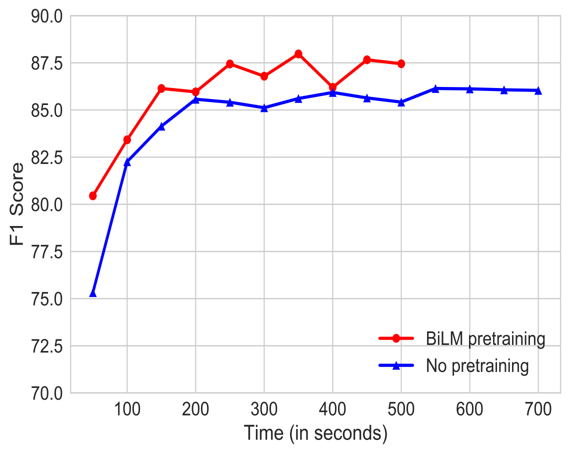
Next, we plot the precision-recall curve, convergence rate, and learning curve to gain additional insights about the NER model with BiLM pretraining and compare it with the randomly initialized model.

4.3.1. Precision-Recall Curve

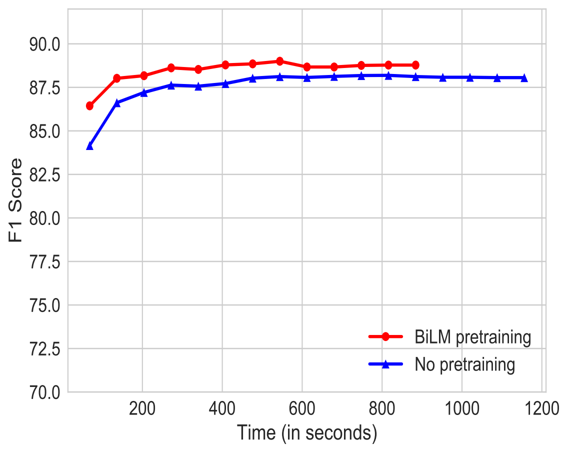
In Figure 6a and 6b, we plot the smoothed precision-recall curve for NCBI-disease and BC5CDR datasets. From both the plots, we see that the BiLM pretrained NER model is always optimal as its area under the precision-recall curve is always more than that of a randomly initialized NER model.

4.3.2. Rate of Convergence

We monitor the overall clock time and the time taken per epoch required for the two models to converge. We follow the same training process as outlined above. A typical run for both the models on NCBI-disease and BC5CDR dataset is shown in Figure 7a and 7b respectively. For NCBI-disease dataset, the model with BiLM pretraining converges in 10 epochs ($\approx$ 500s) compared with the model with no pretraining, which typically converges in 14 epochs ($\approx$ 700s). We observe a similar trend in the BC5CDR dataset where BiLM pretraining results in convergence in 11 epochs ($\approx$ 900s) whereas no pretraining takes around 17 epochs ($\approx$ 1150s). Thus, in terms of total time taken, we observe that pretraining



(a) NCBI-disease dataset



(b) BC5CDR dataset

Figure 7: F1 score versus time taken for training to converge for models with BiLM pretraining and no pretraining. Best viewed in color.

using BiLM weights results in faster convergence by about 28-35% compared with random parameter initialization setting. We also see that BiLM pretraining results in a better F1 score from first epoch onwards for both the datasets.

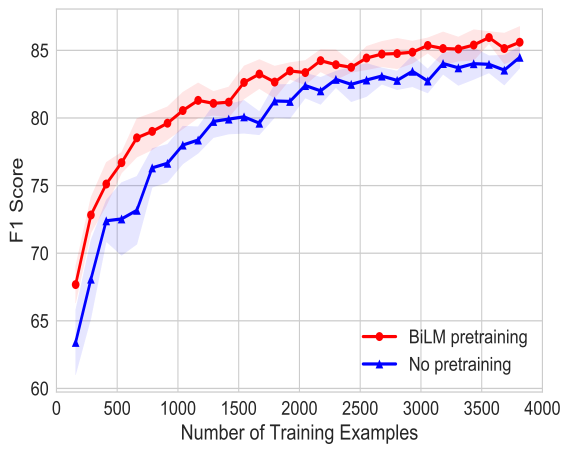
4.3.3. Learning Curve

In this setup, we analyze the F1 score of both the models by feeding them with an increasing number of examples during the training process (learning curve). The learning curve for both the models on NCBI-disease and BC5CDR datasets is shown in Figure 8a and 8b respectively. We can see that the BiLM pretrained model is always optimal (achieves higher F1 score) for any setting of the number of training examples.

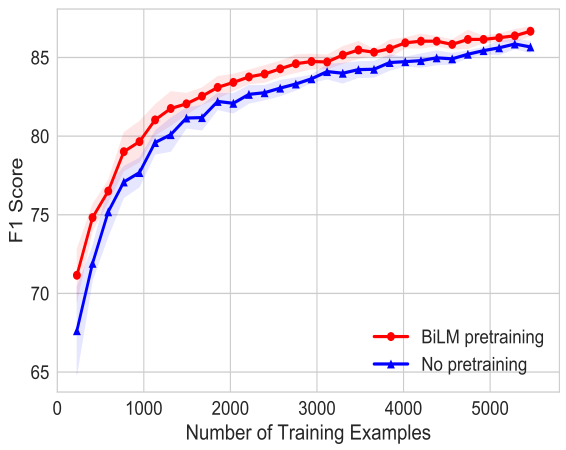
4.4. Case Study on NCBI-Disease Dataset

We will now discuss qualitative results of the BiLM pretrained NER model on NCBI-disease dataset. The NCBI-disease dataset consists of abstracts from medical research papers, which are written in a technical language and contains many complex entity names. In the NCBI-disease dataset, combined training and development set contains $1,902$ unique mentions of disease entities. In its test set, there are 423 unique occurrences of disease names and the BiLM pretrained NER model is able to correctly predict 365 such diseases. Some examples of the longer disease names that are hard to recognize but our approach is able to correctly predict are "*sporadic breast , brain , prostate and kidney cancer*," "*deficiency of the ninth component of human complement*," "*von hippel - lindau (vhl) tumor*," and "*deficiency of the lysosomal enzyme aspartylglucosaminidase*."

Among the 423 unique mentions of diseases in the test set, 232 of them are unseen in the combined training and development set. Our model was able to correctly predict around 120 unseen disease entities in the test set. Some examples of unseen disease entities that are correctly predicted are "*deficiency of the lysosomal enzyme aspartylglucosaminidase*,"



(a) NCBI-disease dataset



(b) BC5CDR dataset

Figure 8: F1 score versus increasing number of training examples for models with BiLM pretraining and no pretraining. Best viewed in color.

"*campomelic - metaphyseal skeletal dysplasia*," "*atrophic benign epidermolysis bullosa*," and "*ectopic intracranial retinoblastoma*." This can be attributed to the improved modeling of the relationship among context words during bidirectional language modeling pretraining step. Some examples of the disease entities where our model fails are "*bannayan - zonana (bzs) or ruvalcaba - riley - smith syndrome*," "*very - long - chain acyl - coenzyme a dehydrogenase deficiency*," "*vwf - deficient*," and "*diffuse mesangial sclerosis*." From these examples, we see that the model makes an incorrect prediction when the disease entities have longer names, which may also contain abbreviations.

5. Related Work

Traditionally, researchers have worked on carefully designing hand-engineered features to represent a word such as the use of parts-of-speech (POS) tags, capitalization information, use of rules such as regular expressions to identify numbers, use of gazetteers, etc. A combination of supervised classifiers using such features was used to achieve the best performance on CoNLL-2003 benchmark NER dataset (Florian et al., 2003). Lafferty et al. (2001) popularized the use of graphical models such as linear-chain conditional random fields (CRF) for NER tasks. Among the early approaches of NER systems in the biomedical domain include ABNER (Settles, 2004), BANNER (Leaman and Gonzalez, 2008), and GIMLI (Campos et al., 2013), which used a variety of lexical, contextual, and orthographic features as input to a linear-chain CRF.

The next generation of methods involves neural networks as they can be trained end-to-end using only the available labeled data without the need of manual task-specific feature engineering. In their seminal work, Collobert et al. (2011) trained window-based and sentence-based models for several NLP tasks and demonstrated competitive performance. For NER task on newswire texts, Huang et al. (2015) uses word embeddings, spelling, and contextual features that are fed to a BiLSTM-CRF model. To incorporate character features, Lample et al. (2016) applies BiLSTM while Chiu and Nichols (2016); Ma and Hovy (2016) applies CNNs on character embeddings respectively. For biomedical NER task, Wei et al. (2016) combines the output of BiLSTM and traditional CRF-based model using an SVM classifier. Zeng et al. (2017) experiments with character-level and word-level BiLSTM for the task of drug NER. Habibi et al. (2017) investigates the effect of pretrained word embeddings on several biomedical NER datasets.

Pretraining the neural network model parameters using transfer learning has been widely studied and has shown to improve results in a variety of tasks such as deep autoencoders for dimensionality reduction (Hinton and Salakhutdinov, 2006), computer vision (Erhan et al., 2010), text classification (Dai and Le, 2015; Howard and Ruder, 2018), machine translation (Ramachandran et al., 2017; Qi et al., 2018), and question answering (Min et al., 2017).

For sequence tagging tasks, supervised transfer learning to pretrain the model from the weights of another model that was trained on a different labeled dataset has been applied to domain adaptation tasks (Qu et al., 2016), de-identification of patient notes (Lee et al., 2018), NER task in tweets (von Däniken and Cieliebak, 2017), and biomedical NER (Giorgi and Bader, 2018; Wang et al., 2018b). In contrast, we pretrain the weights of the NER model from a language model that is trained on unlabeled data and thus removing the hard dependency on the availability of larger labeled datasets for pretraining.

6. Conclusion

In this paper, we present a transfer learning approach for the task of biomedical NER. In our NER model, we use CNNs with different filters widths to extract character features and a word-level BiLSTM for sequence modeling which takes both word embeddings and character features as inputs. We pretrain the NER model weights using a BiLM such that the architectures of both the models are same except for the top decoder layer. The BiLM is trained in an unsupervised manner using only the unlabeled data.

We show that such pretraining of the NER model weights is a good initialization strategy for the optimizer as it leads to substantial improvements in the F1 scores for four benchmark datasets. Further, to achieve a particular F1 score, pretrained model requires less training data compared with a randomly initialized model. A pretrained model also converges faster during model fine-tuning. We also observe gains in the recall score for both seen and unseen disease entities.

For future work, we plan to train bigger sized language models on large collections of medical corpora and use it for providing additional features to the NER model so that it can incorporate wider context while training. We also plan to use external medical knowledge graphs to further improve the NER model's performance.

Acknowledgments

This work was supported by a generous research funding from CMU, MCDS students grant. We would also like to thank the anonymous reviewers for giving us their valuable feedback that helped to improve the paper.

References

Rie Ando. Biocreative II gene mention tagging system at IBM Watson. In *Second BioCreative Challenge Evaluation Workshop*, volume 23, pages 101–103. Centro Nacional de Investigaciones Oncologicas (CNIO) Madrid, Spain, 2007.

Andrew Borthwick. *A Maximum Entropy Approach to Named Entity Recognition.* PhD thesis, New York, NY, USA, 1999.

David Campos, Sérgio Matos, and José Oliveira. Gimli: open source and high-performance biomedical name recognition. *BMC Bioinformatics*, 14(1):54, Feb 2013.

Jason Chiu and Eric Nichols. Named entity recognition with bidirectional LSTM-CNNs. *Transactions of the Association for Computational Linguistics*, 4:357–370, 2016.

Michael Collins. Log-Linear Models, MEMMs, and CRFs. 2013a. URL http://www.cs.columbia.edu/~mcollins/crf.pdf.

Michael Collins. The forward-backward algorithm. 2013b. URL http://www.cs.columbia.edu/~mcollins/fb.pdf.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12:2493–2537, November 2011.

Gamal Crichton, Sampo Pyysalo, Billy Chiu, and Anna Korhonen. A neural network multi-task learning approach to biomedical named entity recognition. *BMC Bioinformatics*, 18 (1):368, 2017.

Andrew Dai and Quoc Le. Semi-supervised sequence learning. In *International Conference on Neural Information Processing Systems - Volume 2*, NIPS'15, pages 3079–3087, Cambridge, MA, USA, 2015.

Rezarta Doğan and Zhiyong Lu. An improved corpus of disease mentions in PubMed citations. In *Workshop on Biomedical Natural Language Processing*, BioNLP '12, pages 91–99, Stroudsburg, PA, USA, 2012.

Dumitru Erhan, Yoshua Bengio, Aaron Courville, Pierre-Antoine Manzagol, Pascal Vincent, and Samy Bengio. Why does unsupervised pre-training help deep learning? *Journal of Machine Learning Research*, 11:625–660, March 2010.

Radu Florian, Abe Ittycheriah, Hongyan Jing, and Tong Zhang. Named entity recognition through classifier combination. In *Conference on Natural Language Learning at HLT-NAACL 2003 - Volume 4*, CONLL '03, pages 168–171, Stroudsburg, PA, USA, 2003.

George David Forney. The Viterbi algorithm. *Proceedings of the IEEE*, 61(3):268–278, March 1973.

John M Giorgi and Gary D Bader. Transfer learning for biomedical named entity recognition with neural networks. *Bioinformatics*, page bty449, 2018.

Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *International Conference on Artificial Intelligence and Statistics*, volume 9 of *Machine Learning Research*, pages 249–256, Chia Laguna Resort, Sardinia, Italy, 13–15 May 2010.

Alex Graves. Generating sequences with recurrent neural networks. *Computing Research Repository*, arXiv:1308.0850, 2013.

Zhou GuoDong and Su Jian. Exploring deep knowledge resources in biomedical name recognition. In *International Joint Workshop on Natural Language Processing in Biomedicine and Its Applications*, JNLPBA '04, pages 96–99, Stroudsburg, PA, USA, 2004.

Maryam Habibi, Leon Weber, Mariana Neves, David Luis Wiegandt, and Ulf Leser. Deep learning with word embeddings improves biomedical named entity recognition. *Bioinformatics*, 33(14):i37–i48, 2017.

Geoffrey Hinton and Ruslan Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, November 1997.

Jeremy Howard and Sebastian Ruder. Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 328–339, 2018.

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

Jin-Dong Kim, Tomoko Ohta, Yoshimasa Tsuruoka, Yuka Tateisi, and Nigel Collier. Introduction to the bio-entity recognition task at JNLPBA. In *International Joint Workshop on Natural Language Processing in Biomedicine and Its Applications*, pages 70–75, 2004.

Yoon Kim. Convolutional neural networks for sentence classification. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, Doha, Qatar, October 2014.

Yoon Kim, Yacine Jernite, David Sontag, and Alexander Rush. Character-aware neural language models. In *AAAI Conference on Artificial Intelligence*, AAAI'16, pages 2741–2749. AAAI Press, 2016.

Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *Computing Research Repository*, arXiv:1412.6980, 2014.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton. ImageNet classification with deep convolutional neural networks. In *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'12, pages 1097–1105, USA, 2012.

John Lafferty, Andrew McCallum, and Fernando Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *International Conference on Machine Learning*, ICML '01, pages 282–289, San Francisco, CA, USA, 2001.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. Neural architectures for named entity recognition. In *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 260–270, San Diego, California, June 2016.

Robert Leaman and Graciela Gonzalez. BANNER: An executable survey of advances in biomedical named entity recognition. *Pacific Symposium on Biocomputing. Pacific Symposium on Biocomputing*, pages 652–63, 2008.

Robert Leaman and Zhiyong Lu. TaggerOne: joint named entity recognition and normalization with semi-Markov models. *Bioinformatics*, 32(18):2839–2846, 2016.

Yan LeCun, Bernhard Boser, John Denker, R. Howard, Wayne Habbard, Lawrence Jackel, and Donnie Henderson. Handwritten digit recognition with a back-propagation network. In *Advances in Neural Information Processing Systems 2*, pages 396–404. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1990.

Ji Young Lee, Franck Dernoncourt, and Peter Szolovits. Transfer learning for named-entity recognition with neural networks. In *Eleventh International Conference on Language Resources and Evaluation, LREC 2018, Miyazaki, Japan*, 2018.

Haodi Li, Qingcai Chen, Kai Chen, and Buzhou Tang. Hitsz cdr system for disease and chemical named entity recognition and relation extraction. 2015.

Jiao Li, Yueping Sun, Robin J Johnson, Daniela Sciaky, Chih-Hsuan Wei, Robert Leaman, Allan Peter Davis, Carolyn J Mattingly, Thomas C Wiegers, and Zhiyong Lu. BioCreative V CDR task corpus: a resource for chemical disease relation extraction. *Database*, 2016.

Xuezhe Ma and Eduard Hovy. End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF. In *Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1064–1074, Berlin, Germany, August 2016.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. In *International Conference on Neural Information Processing Systems - Volume 2*, NIPS'13, pages 3111–3119, USA, 2013.

Sewon Min, Minjoon Seo, and Hannaneh Hajishirzi. Question answering through transfer learning from large fine-grained supervision data. In *Association for Computational Linguistics (Volume 2: Short Papers)*, pages 510–517, 2017.

Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. On the difficulty of training recurrent neural networks. In *International Conference on Machine Learning - Volume 28*, ICML'13, pages 1310–1318. JMLR.org, 2013.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. In *NIPS-W*, 2017.

Ye Qi, Devendra Sachan, Matthieu Felix, Sarguna Padmanabhan, and Graham Neubig. When and why are pre-trained word embeddings useful for neural machine translation? In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 529–535, 2018.

Lizhen Qu, Gabriela Ferraro, Liyuan Zhou, Weiwei Hou, and Timothy Baldwin. Named entity recognition for novel types by transfer learning. In *Conference on Empirical Methods in Natural Language Processing*, pages 899–905, 2016.

Prajit Ramachandran, Peter Liu, and Quoc Le. Unsupervised pretraining for sequence to sequence learning. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 383–391, 2017.

Maya Rotmensch, Yoni Halpern, Abdulhakim Tlimat, Steven Horng, and David Sontag. Learning a health knowledge graph from electronic medical records. *Scientific Reports*, 2017.

Mike Schuster and Kuldip Paliwal. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681, November 1997.

Burr Settles. Biomedical named entity recognition using conditional random fields and rich feature sets. In *International Joint Workshop on Natural Language Processing in Biomedicine and Its Applications*, JNLPBA '04, pages 104–107, Stroudsburg, PA, USA, 2004.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958, January 2014.

Mark Stevenson and Yikun Guo. Disambiguation in the biomedical domain: The role of ambiguity type. *Journal of Biomedical Informatics*, 43(6):972–981, December 2010.

George Tsatsaronis, Georgios Balikas, Prodromos Malakasiotis, Ioannis Partalas, Matthias Zschunke, Michael Alvers, Dirk Weissenborn, Anastasia Krithara, Sergios Petridis, Dimitris Polychronopoulos, Yannis Almirantis, John Pavlopoulos, Nicolas Baskiotis, Patrick Gallinari, Thierry Artieres, Axel Ngonga, Norman Heino, Eric Gaussier, Liliana Barrio-Alvers, Michael Schroeder, Ion Androutsopoulos, and Georgios Paliouras. An overview of the BIOASQ large-scale biomedical semantic indexing and question answering competition. *BMC Bioinformatics*, 16:138, 2015.

Pius von Däniken and Mark Cieliebak. Transfer learning and sentence level features for named entity recognition on tweets. In *Workshop on Noisy User-generated Text*, pages 166–171, 2017.

Xuan Wang, Yu Zhang, Xiang Ren, Yuhao Zhang, Marinka Zitnik, Jingbo Shang, Curtis Langlotz, and Jiawei Han. Cross-type biomedical named entity recognition with deep multi-task learning. *Computing Research Repository*, arXiv:1801.09851, 2018a.

Zhenghui Wang, Yanru Qu, Liheng Chen, Jian Shen, Weinan Zhang, Shaodian Zhang, Yimei Gao, Gen Gu, Ken Chen, and Yong Yu. Label-aware double transfer learning for cross-specialty medical named entity recognition. In *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1–15, 2018b.

Qikang Wei, Tao Chen, Ruifeng Xu, Yulan He, and Lin Gui. Disease named entity recognition by combining conditional random fields and bidirectional recurrent neural networks. *Database*, 2016.

Paul Werbos. Generalization of backpropagation with application to a recurrent gas market model. *Neural networks*, 1(4):339–356, 1988.

Paul Werbos. Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, 78(10):1550–1560, Oct 1990.

Donghuo Zeng, Chengjie Sun, Lei Lin, and Bingquan Liu. LSTM-CRF for drug-named entity recognition. *Entropy*, 19(6), 2017.