Bioinformatics
doi.10.1093/bioinformatics/xxxxxx
Advance Access Publication Date: Day Month Year
Manuscript Category

OXFORD

Subject Section

Cross-type Biomedical Named Entity Recognition with Deep Multi-Task Learning

Xuan Wang [1,*], Yu Zhang [1], Xiang Ren [2,*], Yuhao Zhang [3], Marinka Zitnik [4], Jingbo Shang [1], Curtis Langlotz [3] and Jiawei Han [1]

[1] Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, IL 61801, USA,
[2] Department of Computer Science, University of Southern California, Los Angeles, CA 90089, USA,
[3] School of Medicine, Stanford University, Stanford, CA 94305, USA, and
[4] Department of Computer Science, Stanford University, Stanford, CA 94305, USA

* To whom correspondence should be addressed.

Associate Editor: XXXXXXX

Received on XXXXX; revised on XXXXX; accepted on XXXXX

Abstract

Motivation: State-of-the-art biomedical named entity recognition (BioNER) systems often require handcrafted features specific to each entity type, such as genes, chemicals and diseases. Although recent studies explored using neural network models for BioNER to free experts from manual feature engineering, the performance remains limited by the available training data for each entity type.
Results: We propose a multi-task learning framework for BioNER to collectively use the training data of different types of entities and improve the performance on each of them. In experiments on 15 benchmark BioNER datasets, our multi-task model achieves substantially better performance compared with state-of-the-art BioNER systems and baseline neural sequence labeling models. Further analysis shows that the large performance gains come from sharing character- and word-level information among relevant biomedical entities across differently labeled corpora.
Availability: Our source code is available at https://github.com/yuzhimanhua/lm-lstm-crf.
Contact: xwang174@illinois.edu, xiangren@usc.edu.

1 Introduction

Biomedical named entity recognition (BioNER) is one of the most fundamental task in biomedical text mining that aims to automatically recognize and classify biomedical entities (e.g., genes, proteins, chemicals and diseases) from text. BioNER can be used to identify new gene names from text (Smith *et al.*, 2008). It also serves as a primitive step of many downstream applications, such as relation extraction (Cokol *et al.*, 2005) and knowledge base completion (Szklarczyk *et al.*, 2017; Wei *et al.*, 2013; Xie *et al.*, 2013; Szklarczyk *et al.*, 2015).

BioNER is typically formulated as a sequence labeling problem whose goal is to assign a label to each word in a sentence. State-of-the-art BioNER systems often require handcrafted features (e.g., capitalization, prefix and suffix) to be specifically designed for each entity type (Ando, 2007; Leaman and Lu, 2016; Zhou and Su, 2004; Lu *et al.*, 2015). This feature generation process takes the majority of time and cost in developing a BioNER system (Leser and Hakenberg, 2005), and leads to highly specialized systems that cannot be directly used to recognize new types of entities. The accuracy of the resulting BioNER tools remains a limiting factor in the performance of biomedical text mining pipelines (Huang and Lu, 2015).

Recent NER studies consider neural network models to automatically generate quality features (Chiu and Nichols, 2016; Ma and Hovy, 2016; Lample *et al.*, 2016; Liu *et al.*, 2018). Crichton *et al.* took each word token and its surrounding context words as input into a convolutional neural network (CNN). Habibi *et al.* adopted the model from Lample *et al.* and used word embeddings as input into a bidirectional long short-term memory-conditional random field (BiLSTM-CRF) model. These neural network models free experts from manual feature engineering. However, these models have millions of parameters and require very large datasets to reliably estimate the parameters. This poses a major challenge for biomedicine, where datasets of this scale are expensive and slow to create and thus neural network models cannot realize their potential

©

performance to the fullest (Camacho *et al.*, 2018). Although neural network models can outperform traditional sequence labeling models (e.g., CRF models (Lafferty *et al.*, 2001)), they are still outperfomed by handcrafted feature-based systems in multiple domains (Crichton *et al.*, 2017).

One direction to address the above challenge is to use the labeled data of different entity types to augment the training signals for each of them, as information like word semantics and grammatical structure may be shared across different datasets. However, simply combining all datasets and train one single model over multiple entity types can introduce many false negatives because each dataset is typically specifically annotated for one or only a few entity types. For example, combining dataset A for gene recogntion and dataset B for chemical recognition will result in missing chemical entity labels in dataset A and missing gene entity labels in dataset B. Multi-task learning (MTL) (Collobert and Weston, 2008; Søgaard and Goldberg, 2016) offers a solution to this issue by collectively training a model on several related tasks, so that each task benefits model learning in other tasks without introducing additional errors. MTL has been successfully applied in natural language processing (Collobert and Weston, 2008), speech recognition (Deng *et al.*, 2013), computer vision (Girshick, 2015) and drug discovery (Ramsundar *et al.*, 2015). But MTL is less commonly used and has seen limited success in BioNER so far. Crichton *et al.* explored MTL with a CNN model for BioNER. However, Crichton *et al.* only considers word-level features as input, ignoring character-level lexical information which are often crucial for modeling biomedical entities (e.g. -ase could be an important subword feature for gene/protein entity recognition). As a result, their best performing multi-task CNN model does not outperform state-of-the-art systems that use on handcrafted features (Crichton *et al.*, 2017).

In this paper, we propose a new multi-task learning framework using char-level neural models for BioNER. The proposed framework, despite being simple and not requiring any feature engineering, achieves excellent benchmark performance. Our multi-task model is built upon a single-task neural network model (Liu *et al.*, 2018). In particular, we consider a BiLSTM-CRF model with an additional context-dependent BiLSTM layer for modeling character sequences. A prominent advantage of our multi-task model is that inputs from different datasets can efficiently share both character- and word-level representations, by reusing parameters in the corresponding BiLSTM units. We compare the proposed multi-task model with state-of-the-art BioNER systems and baseline neural network models on 15 benchmark BioNER datasets and observe substantially better performance. We further show through detailed experimental analysis on 5 datasets that the proposed approach adds marginal computational overhead and outperforms strong baseline neural models that do not consider multi-task learning, suggesting that multi-task learning plays an important role in its success. Altogether, this work introduces a new text-mining approach that can help scientists exploit knowledge buried in biomedical literature in a systematic and unbiased way.

2 Background

In this section, we introduce basic neural network architectures that are relevant for our multi-task learning approach.

2.1 NER problem definition

Let Φ denote the set of labels indicating whether a word is part of a specific entity type or not. Given a sequence of words $\boldsymbol{w} = \{w_1, w_2, ..., w_n\}$, the output is a sequence of labels $\boldsymbol{y} = \{y_1, y_2, ..., y_n\}$, $y_i \in \Phi$.

2.2 Long Short-Term Memory (LSTM)

Long short-term memory neural network is a specific type of recurrent neural network that models dependencies between elements in a sequence

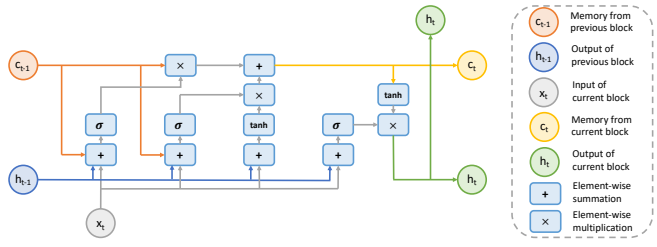

Fig. 1. Architecture of long short-term memory neural network.

through recurrent connections (Fig. 1). The input to an LSTM network is a sequence of vectors $\boldsymbol{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_T\}$, where vector $\boldsymbol{x}_i$ is a representation vector of a word in the input sentence. The output is a sequence of vectors $\boldsymbol{H} = \{\boldsymbol{h}_1, \boldsymbol{h}_2, ..., \boldsymbol{h}_T\}$, where $\boldsymbol{h}_i$ is a hidden state vector. At step t of the recurrent calculation, the network takes $\boldsymbol{x}_t, \boldsymbol{c}_{t-1}, \boldsymbol{h}_{t-1}$ as inputs and produces $\boldsymbol{c}_t, \boldsymbol{h}_t$ via the following intermediate calculations:

$$\boldsymbol{i}_t = \sigma(\boldsymbol{W}^i \boldsymbol{x}_t + \boldsymbol{U}^i \boldsymbol{h}_{t-1} + \boldsymbol{b}^i)$$

$$\boldsymbol{f}_t = \sigma(\boldsymbol{W}^f \boldsymbol{x}_t + \boldsymbol{U}^f \boldsymbol{h}_{t-1} + \boldsymbol{b}^f)$$

$$\boldsymbol{o}_t = \sigma(\boldsymbol{W}^o \boldsymbol{x}_t + \boldsymbol{U}^o \boldsymbol{h}_{t-1} + \boldsymbol{b}^o)$$

$$\boldsymbol{g}_t = tanh(\boldsymbol{W}^g \boldsymbol{x}_t + \boldsymbol{U}^g \boldsymbol{h}_{t-1} + \boldsymbol{b}^g)$$

$$\boldsymbol{c}_t = \boldsymbol{f}_t \odot \boldsymbol{c}_{t-1} + \boldsymbol{i}_t \odot \boldsymbol{g}_t$$

$$\boldsymbol{h}_t = \boldsymbol{o}_t \odot tanh(\boldsymbol{c}_t),$$

where $\sigma(\cdot)$ and $tanh(\cdot)$ denote element-wise sigmoid and hyperbolic tangent functions, respectively, and $\odot$ denotes element-wise multiplication. The $\boldsymbol{i}_t$, $\boldsymbol{f}_t$ and $\boldsymbol{o}_t$ are referred to as input, forget, and output gates, respectively. The $\boldsymbol{g}_t$ and $\boldsymbol{c}_t$ are intermediate calculation steps. At $t = 1$, $\boldsymbol{h}_0$ and $\boldsymbol{c}_0$ are initialized to zero vectors. The trainable parameters are $\boldsymbol{W}^j$,$\boldsymbol{U}^j$ and $\boldsymbol{b}^j$ for $j \in \{i, f, o, g\}$.

The LSTM architecture described above can only process the input in one direction. The bi-directional long short-term memory (BiLSTM) model improves the LSTM by feeding the input to the LSTM network twice, once in the original direction and once in the reversed direction. Outputs from both directions are concatenated to represent the final output. This design allows for detection of dependencies from both previous and subsequent words in a sequence.

2.3 Bi-directional Long Short-Term Memory-Conditional Random Field (BiLSTM-CRF)

A naive way of applying the BiLSTM network to sequence labeling is to use the output hidden state vectors to make independent tagging decisions. However, in many sequence labeling tasks such as BioNER, it is useful to also model the dependencies across output tags. The BiLSTM-CRF network adds a conditional random field (CRF) layer on top of a BiLSTM network. This BiLSTM-CRF network takes the input sequence $\boldsymbol{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_n\}$ to predict an output label sequence $\boldsymbol{y} = \{y_1, y_2, ..., y_n\}$. A score is defined as:

$$s(\boldsymbol{X}, \boldsymbol{y}) = \sum_{i=0}^{n} \boldsymbol{A}_{y_i, y_{i+1}} + \sum_{i=1}^{n} \boldsymbol{P}_{i, y_i},$$

where $\boldsymbol{P}$ is an $n \times k$ matrix of the output from the BiLSTM layer, n is the sequence length, k is the number of distinct labels, $\boldsymbol{A}$ is a $(k+2) \times (k+2)$ transition matrix and $\boldsymbol{A}_{i,j}$ represents the transition probability from the i-th label to the j-th label. Note that two additional labels <*start*> and <*end*> are used to represent the start and end of a sentence, respectively.

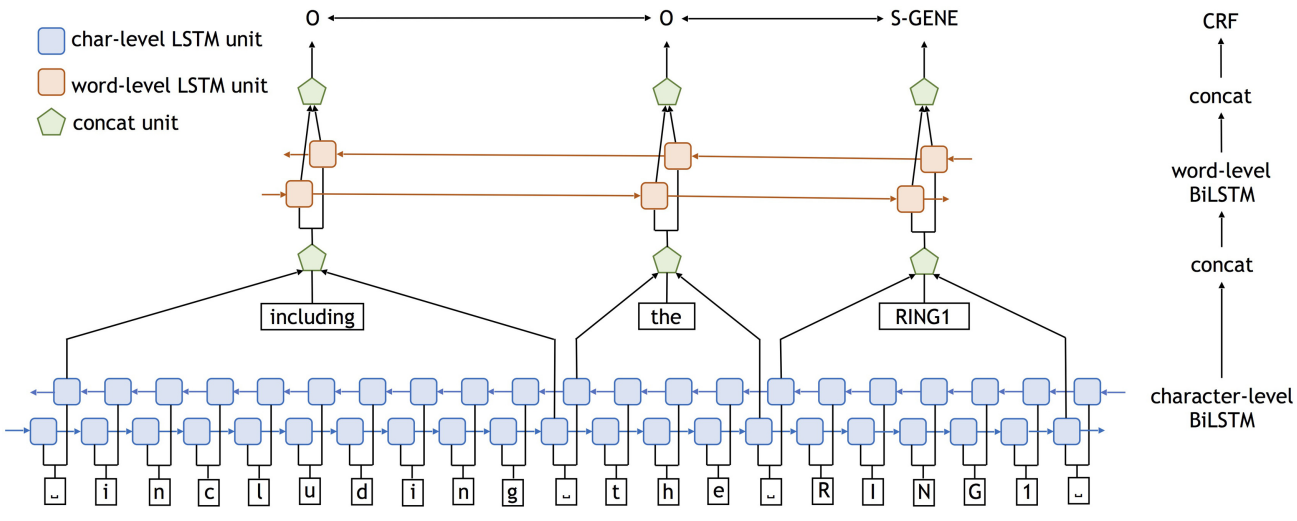

Fig. 2. Architecture of a single-task neural network. The input is a sentence from biomedical literature. White rectangles denote character and word embeddings; blue rectangles denote the first character-level BiLSTM; red rectangles represent the second word-level BiLSTM; green pentagons represent the concatenation units. The tags on the top, e.g., 'O', 'S-GENE', are the output of the final CRF layer, which are the entity labels we get for each word in the sentence.

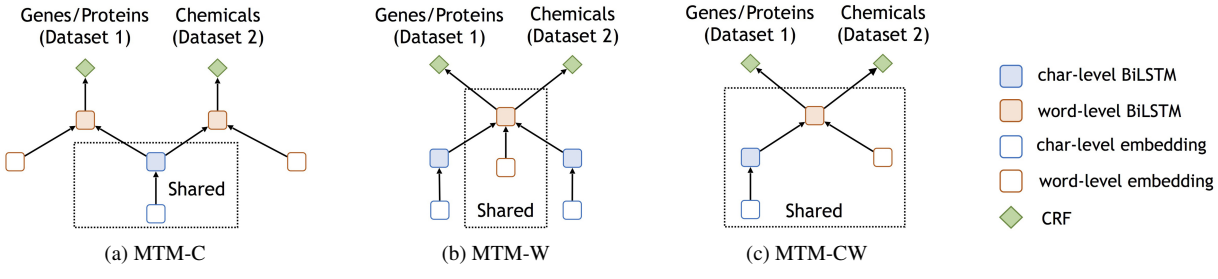

Fig. 3. Three multi-task learning neural network models. The blue empty rectangles represent the character embeddings. The blue filled rectangles represent the character-level BiLSTM. The red empty rectangles represent the word-level embeddings. The red filled rectangles represent the word-level BiLSTM. The green pentagons represent the CRF layer. (a) MTM-C: multi-task learning neural network with a shared character layer and a task-specific word layer, (b): MTM-W: multi-task learning neural network with a task-specific character layer and a shared word layer, (c) MTM-CW: multi-task learning neural network with shared character and word layers.

We further define $\boldsymbol{Y_X}$ as all possible sequence labels given the input sequence $\boldsymbol{X}$. The training process maximizes the log-probability of the label sequence $\boldsymbol{y}$ given the input sequence $\boldsymbol{X}$:

$$log(p(\boldsymbol{y}|\boldsymbol{X})) = log\frac{e^{s(\boldsymbol{X},\boldsymbol{y})}}{\sum_{\boldsymbol{y'}\in\boldsymbol{Y_X}} e^{s(\boldsymbol{X},\boldsymbol{y'})}}. \quad (1)$$

A three-layer BiLSTM-CRF architecture is employed by Lample *et al.* and Habibi *et al.* to jointly model the word and the character sequences in the input sentence. In this architecture, the first BiLSTM layer takes character embedding sequence of each word as input, and produces a character-level representation vector for this word as output. This character-level vector is then concatenated with a word embedding vector, and fed into a second BiLSTM layer. Lastly, a CRF layer takes the output vectors from the second BiLSTM layer, and outputs the best tag sequence by maximizing the log-probability in Equation 1.

In practice, the character embedding vectors are randomly initialized and co-trained during the model training process. The word embedding vectors are retrieved directly from a pre-trained word embedding lookup table. The classical Viterbi algorithm is used to infer the final labels for the CRF model. The three-layer BiLSTM-CRF model is a differentiable neural network architecture that can be trained by backpropagation.

3 Deep multi-task learning for BioNER

In this section, we first describe a single-task neural model that can effectively handle out-of-vocabulary words by modeling character sequences. We then introduce multi-task models that combine single-task models in three different ways and together represent a flexible deep multi-task learning framework for BioNER.

3.1 Single-task model (STM)

The vanilla BiLSTM-CRF model can learn high-quality representations for words that appeared in the training dataset. However, it often fails to generalize to out-of-vocabulary (OOV) words (i.e., words that did not appear in the training dataset) because they don't have a pre-trained word embedding. These OOV words are common in biomedical text (67.21% OOV words of the datasets in Table 1). Therefore, for the baseline single-task BioNER model, we use a neural network architecture that better handles OOV words. As shown in Fig. 2, our single-task model consists of three layers. In the first layer, a BiLSTM network is used to model the character sequence of the input sentence. We use character embedding vectors as input to the network. Hidden state vectors at the word boundaries of this character-level BiLSTM are then selected and concatenated with word embedding vectors to form word representations. Next, these word representation vectors are fed into a word-level BiLSTM layer (i.e., the upper BiLSTM layer in Fig. 2). Lastly, output of this word-level BiLSTM is fed into the a CRF layer for label prediction. Compared to the vanilla BiLSTM-CRF model, a major advantage of this model is that it can infer the meaning of an out-of-vocabulary word from its character sequence and other characters around it. For example, the model is now able to infer that "RING2" likely represents a gene symbol, even though then network may have only seen the word "RING1" during training.

3.2 Multi-task models (MTMs)

An important characteristic of the BioNER task is the limited availability of supervised training data. We propose a multi-task learning approach to address this problem by training different BioNER models on datasets with different entity types while sharing parameters across these models. We hypothesize that the proposed approach can make more efficient use of the data and encourage the models to learn representations of words and characters (which are shared between multiple corpora) in a more effective and generalized way.

We give a formal definition of the multi-task setting as the following. Given m datasets, for $i \in \{1, ..., m\}$, each dataset D_i consists of n_i training samples, i.e., $D_i = \{\boldsymbol{w}_j^i, y_j^i\}_{j=1}^{n_i}$. We denote the training matrix for each dataset as $\boldsymbol{X}^i = \{\boldsymbol{x}_1^i, ..., \boldsymbol{x}_{n_i}^i\}$ ($\boldsymbol{X}^i$ is the feature representation of the input word sequence $\boldsymbol{w}_j^i$) and the labels for each dataset as $\boldsymbol{y}^i = \{y_1^i, ..., y_{n_i}^i\}$. A multi-task model therefore consists of m different models, each trained on a separate dataset, while sharing part of the model parameters across datasets. The loss function L of the multi-task model is:

$$L = \sum_{i=1}^{m} \lambda_i L_i = \sum_{i=1}^{m} \lambda_i \log(P_{\theta_i^w, \theta_i^c, \theta_i^o}(\boldsymbol{y}^i | \boldsymbol{X}^i)).$$

The log-likelihood term is shown in Equation 1 and λ_i is a positive hyper-parameter that controls the contribution of each dataset. We observed that our multi-task model is able to achieve very competitive performance with $\lambda_i = 1$ on all datasets that we evaluated on and therefore use this value in our experiments. However, we believe that the performance can be improved with further tuned λ_i values.

We propose three different multi-task models, as illustrated in Fig. 3. These three models differ in which part of the model parameters ($\theta_i^w, \theta_i^c, \theta_i^o$) are shared across multiple datasets:

MTM-C In this model, $\theta_i^c = \theta^c$ are shared among different tasks. All datasets are iteratively used to train the model. When a dataset is used, the parameters updated during the training are θ^c and θ_i^w. The detailed architecture of this multi-task model is shown in Fig. 3(a).

MTM-W In this model, $\theta_i^w = \theta^w$ are shared among different tasks. When a dataset is used, the parameters updated during the training are θ^w and θ_i^c. The detailed architecture of this multi-task model is shown in Fig. 3(b).

MTM-CW In this model, $\theta_i^c = \theta^c$ and $\theta_i^w = \theta^w$ are shared among different tasks. Each dataset has its specific θ_i^o for label prediction. MTM-CW shared the most information across tasks compared with the other two multi-task models. It enables sharing both character- and word-level information between different biomedical entities, while the other two models only enable sharing part of the information. The detailed architecture of this multi-task model is shown in Fig. 3(c).

4 Experimental setup

In this section, we describe the datasets, evaluation metrics, and provide an overview of experimental setup.

4.1 Datasets

We test our method on the same 15 datasets used by Crichton *et al.*, and find our model achieves substantially better performance on 14 of them compared with baseline neural network models. Due to space limit, here we report detailed results of the multi-task model on 5 main datasets (Table 1), which altogether cover major biomedical entity types (e.g., genes, proteins, chemicals, diseases). We also include full results on all the 15 datasets in *Supplementary Material: Performance comparison on 15 datasets*. The performance of the multi-task model is slightly different when trained on 5 datasets compared with trained on 15 datasets (shown in *Supplementary Material: Performance comparison on 15 datasets*), as the MTL model has access to more data. In our experiments, we follow the experiment setup of Crichton *et al.* and divide each dataset into training, development and test sets. We use training and development sets to train the final model. All datasets are publicly available.[1] As part of preprocessing, word labels are encoded using an IOBES scheme. In this scheme, for example, a word describing a gene entity is tagged with "B-Gene" if it is at the beginning of the entity, "I-Gene" if it is in the middle of the entity, and "E-Gene" if it is at the end of the entity. Single-word gene entities are tagged with "S-Gene". All other words not describing entities of interest are tagged as 'O'. Next, we briefly describe the 5 main datasets and their corresponding state-of-the-art BioNER systems.

Table 1. Biomedical NER datasets used in the experiments.

Dataset	Size	Entity types and counts
BC2GM	20,000 sentences	Gene/Protein (24,583)
BC4CHEMD	10,000 abstracts	Chemical (84,310)
BC5CDR	1,500 articles	Chemical (15,935), Disease (12,852)
NCBI-Disease	793 abstracts	Disease (6,881)
JNLPBA	2,404 abstracts	Gene/Protein (35,336), Cell Line (4,330), DNA (10,589), Cell Type (8,649), RNA (1,069)

BC2GM The state-of-the-art system reported for the BioCreative II gene mention recognition task adopts semi-supervised learning method with alternating structure optimization (Ando, 2007).

BC4CHEMD The state-of-the-art system reported for the BioCreative IV chemical entity mention recognition task is the *CHEMDNER* system (Lu *et al.*, 2015), which is based on mixed conditional random fields with Brown clustering of words.

BC5CDR The state-of-the-art system reported for the most recent BioCreative V chemical and disease mention recognition task is the *TaggerOne* system (Leaman and Lu, 2016), which uses a semi-Markov model for joint entity recognition and normalization.

NCBI-Disease The NCBI disease dataset was initially introduced for disease name recognition and normalization. It has been widely used for a lot of applications. The state-of-the-art system on this dataset is also the *TaggerOne* system (Leaman and Lu, 2016).

JNLPBA The state-of-the-art system (Zhou and Su, 2004) for the 2004 JNLPBA shared task on biomedical entity (gene/protein, DNA, RNA, cell line, cell type) recognition uses a hidden markov model (HMM). Although this task and the model is a bit old compared with the others, it still remains a competitive benchmark method for comparison.

4.2 Evaluation metrics

We report the performance of all the compared methods on the test set. We deem each predicted entity as correct only if both the entity boundary and entity types are the same as the ground-truth annotation (i.e., *exact match*). Then we calculate the precision, recall and F1 scores on all datasets and macro-averaged F1 scores on all entity types. For error analysis, we compare the ratios of false positive (FP) and false negative (FN) labels in the single-task and the multi-task models and include the results in *Supplementary Material: Error analysis*.

The test set of the BC2GM dataset is constructed slightly differently compared to the test sets of other datasets. BC2GM additionally provides

[1] All datasets can be downloaded from: https://github.com/cambridgeltl/MTL-Bioinformatics-2016.

Table 2. Performance and average training time of the baseline neural network models and the proposed MTM-CW model. Bold: best scores, *: significantly worse than the MTM-CW model ($p \leq 0.05$), **: significantly worse than the MTM-CW model ($p \leq 0.01$). The details of dataset benchmark systems and evaluation methods are described in Section 4.1 and 4.2, respectively.

		Dataset Benchmark	Crichton *et al.*	Lample *et al.* Habibi *et al.*	Ma and Hovy	STM	MTM-CW
BC2GM (Exact)	Precision	-	-	81.57±0.26*	79.09±0.63**	81.11±0.33*	**82.10±0.04**
	Recall	-	-	**79.48±0.27**	77.87±0.53**	78.91±0.40**	79.42±0.01
	F1	-	73.17**	80.51±0.09	78.48±0.31**	80.00±0.15*	**80.74±0.04**
BC2GM (Alternative)	Precision	88.48	-	87.27±0.41**	83.50±0.37**	88.21±0.28*	**89.45±0.32**
	Recall	85.97**	-	87.84±0.19	87.13±0.17*	87.43±0.18*	**88.67±0.37**
	F1	87.21**	84.41**	87.55±0.10*	85.27±0.11**	87.82±0.30*	**89.06±0.32**
BC4CHEMD	Precision	88.73**	-	89.68±0.22*	90.83±0.53	90.53±0.72*	**91.30±0.08**
	Recall	87.41	-	85.87±0.16*	83.19±0.20**	87.04±0.50	**87.53±0.11**
	F1	88.06*	83.02**	87.74±0.05**	86.84±0.07**	88.75±0.20	**89.37±0.07**
BC5CDR	Precision	**89.21**	-	87.60±0.08**	89.16±0.03	88.84±0.08	89.10±0.11
	Recall	84.45**	-	86.25±0.07**	84.28±0.02**	85.16±0.05**	**88.47±0.04**
	F1	86.76**	83.90**	86.92±0.06**	86.65±0.06**	86.96±0.00**	**88.78±0.12**
NCBI-Disease	Precision	85.10	-	86.11±0.33	**86.89±0.34**	84.95±0.41	85.86±0.90
	Recall	80.80**	-	85.49±0.93	78.75±0.16**	82.92±0.31*	**86.42±0.44**
	F1	82.90**	80.37**	85.80±0.16	82.62±0.29**	83.92±0.18*	**86.14±0.31**
JNLPBA	Precision	69.42**	-	**71.35±0.05**	70.28±0.03*	69.60±0.07**	70.91±0.02
	Recall	75.99	-	75.74±0.07	75.26±0.41	74.95±0.24*	**76.34±0.23**
	F1	72.55**	70.09**	73.48±0.03	72.68±0.21*	72.17±0.13**	**73.52±0.03**
Training time (s/sent.)		-	-	1.59	0.95	0.71	0.75

a list of alternative answers for each entity in the test set. A predicted entity is deemed correct as long as it matches the ground truth or one of the alternative answers. We refer to this measurement as *alternative match* and report scores under both *exact match* and *alternative match* for the BC2GM dataset.

4.3 Pre-trained word embeddings

We initialize the word embedding matrix with pre-trained word vectors from Pyysalo *et al.*, 2013 in all experiments.[2] These word embeddings are trained using a skip-gram model, as described in Mikolov *et al.*, 2013. These word vectors are trained on three different datasets: (1) abstracts from the PubMed database, (2) abstracts from the PubMed database together with full-text articles from the PubMed Central (PMC), and (3) the entire Pubmed database of abstracts and full-text articles together with the Wikipedia corpus. We found the third set of word vectors lead to best results on development set and therefore used it for the model development. We provide a full comparison of different word embeddings in *Supplementary Material: Performance of Word Embeddings*. In all experiments, we replace rare words (i.e., words with a frequency of less than 5) with a special *<UNK>* token, whose embedding is randomly initialized and fine-tuned during model training.

4.4 Training details

All the neural network models are trained on one GeForce GTX 1080 GPU. To train our neural models, we use a learning rate of 0.01 with a decay rate of 0.05 applied to every epoch of training. The dimensions of word and character embedding vectors are set to be 200 and 30, respectively (Liu *et al.*, 2018). We adopted 200 (best performance among 100, 200 and 300) for both character- and word-level BiLSTM layers. Note that Liu *et al.* considers advanced strategies, such as highway structures, to further improve performance. We did not observe any significant performance boost with these advanced strategies, thus do not adopt these strategies in this work. The performance of the model variations with these advanced strategies can be found in *Supplementary Material: Performance of Model Variations*. To train the baseline neural network models, we use the default parameter settings as used in their paper (Lample *et al.*, 2016; Habibi *et al.*, 2017; Ma and Hovy, 2016) because we found the default parameters also lead to almost optimal performance on the development set.

[2] The pre-trained word vectors can be download from: http://bio.nlplab.org/.

5 Results

5.1 Performance comparison on benchmark datasets

We compare the proposed single-task (Section 3.1) and multi-task models (Section 3.2) with state-of-the-art BioNER systems (reported for each dataset) and three neural network models from Crichton *et al.*, Lample *et al.*; Habibi *et al.*, and Ma and Hovy. The evaluation metrics include precision, recall and F1 score (Tsai *et al.*, 2006) (Table 2). We denote results of the best system priorly reported for each dataset as "Dataset Benchmark". For method proposed by Crichton *et al.*, we quote their experiment results directly. For other neural network models, we repeart each experiment three times with the mean and standard deviation reported (Table 2). To directly compare with the results in Crichton *et al.*, we measure statistical significance with the same t-test as used in their paper.

We observe that the MTM-CW model achieves significantly higher F1 scores than state-of-the-art benchmark systems (column Dataset Benchmark in Table 2) on all of the five datasets. Following established practice in the literature, we use exact match to compare benchmark performance on all the datasets except for the BC2GM, where we report benchmark performance based on alternative match. Furthermore, MTM-CW generally achieves significantly higher F1 scores than other neural network models. These results show that the proposed multi-task learning neural network significantly outperforms state-of-the-art systems and other neural networks. In particular, the MTM-CW model consistently achieves a better performance than the single task model, demonstrating that multi-task learning is able to successfully leverage information across different datasets and mutually enhance performance on each single task. We further investigate the performance of three multi-task models (MTM-C, MTM-W, and MTM-CW, Table 3). Results show that the best performing multi-task model is MTM-CW, indicating the importance of morphological

Table 3. F1 scores of three multi-task models proposed in this paper. Bold: best scores, *: significantly worse than the MTM-CW model ($p \leq 0.05$), **: significantly worse than the MTM-CW model ($p \leq 0.01$).

Dataset	MTM-C	MTM-W	MTM-CW
BC2GM	77.80±0.30**	79.42±0.08**	**80.74±0.04**
BC4CHEMD	88.16±0.07*	88.49±0.03*	**89.37±0.07**
BC5CDR	86.05±0.26**	88.26±0.05*	**88.78±0.12**
NCBI-Disease	82.94±0.31**	84.81±0.14	**86.14±0.31**
JNLPBA	71.79±0.41**	73.21±0.16	**73.52±0.03**

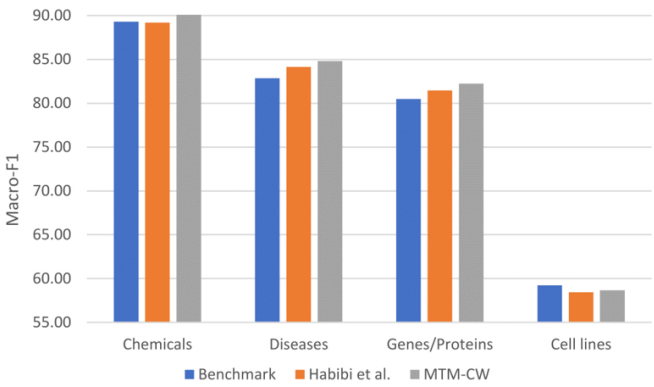

Fig. 4. Macro-averaged F1 scores of the proposed multi-task model compared with benchmark on different entities. Benchmark refers to the performance of state-of-the-art BioNER systems.

information captured by character-level BiLSTM as well as lexical and contextual information captured by word-level BiLSTM.

5.2 Performance on major biomedical entity types

We also conduct more fine-grained comparison of all models on four major biomedical entity types: genes/proteins, chemicals, diseases and cell lines since they are the most often annotated entity types (Fig. 4). Each entity type comes from multiple datasets: genes/proteins from BC2GM and JNLPBA, chemicals from BC4CHEMD and BC5CDR, diseases from BC5CDR and NCBI-Disease, and cell lines from JNLPBA.

The MTM-CW model performs consistently better than the neural network model (Habibi *et al.*, 2017) on all four entity types. It also outperforms the state-of-the-art systems (Benchmark in Fig. 4) on three entity types except for cell lines. These results further confirm that the multi-task neural network model achieves a significantly better performance compared with state-of-art systems and other neural network models for BioNER.

5.3 Integration of biomedical entity dictionaries

A biomedical entity dictionary is a manually-curated list of entity names that belong to a specific entity type. Traditional BioNER systems make heavy use of these dictionaries in addition to other data. To study whether our approach can benefit from the use of entity dictionaries, we retrieve biomedical entity dictionaries for three entity types (i.e., genes/proteins, chemicals and diseases) from the Comparative Toxicogenomics Database (CTD) (Davis *et al.*, 2017). We use these entity dictionaries in a neural network model in two different ways: (1) dictionary post-processing to match the 'O'-labeled entities with the dictionary to reduce the false negative rate, or (2) dictionary feature to provide additional information about words into the word-level BiLSTM. This dictionary feature indicates whether a word sequence consisting of a word and its neighbors is present in a dictionary. We consider word sequences of up to six words, which adds 21 additional dimensions for each entity type. We compare the performance of MTM-CW with and without adding dictionaries (Table 4).

Table 4. F1 scores of the proposed multi-task model using the CTD entity dictionary. Bold: best scores, *: significantly worse than the MTM-CW model ($p \leq 0.05$), **: significantly worse than the MTM-CW model ($p \leq 0.01$).

Dataset	MTM-CW	+Dictionary Feature	+Dictionary Post-process
BC2GM	**80.74±0.04**	80.70±0.06	61.56±0.07**
BC4CHEMD	**89.37±0.07**	88.92±0.10	83.83±0.09**
BC5CDR	88.78±0.12	**88.82±0.17**	87.90±0.06**
NCBI-Disease	**86.14±0.31**	85.48±0.44	83.80±0.06*
JNLPBA	**73.52±0.03**	73.35±0.30	63.62±0.08**

We observe no significant performance improvement when biomedical entity dictionaries are included into the MTM-CW model at the pre-processing stage. Moreover, including dictionaries at the post-processing stage even hurts the performance. This is presumably due to a higher false positive rate introduced by the dictionaries, when some words share the surface name with dictionary entities but do not share the same meaning or entity types. These results indicate that our multi-task model, by sharing information at both the character and word levels, is able to learn effective data representations and generalize to new data without the use of external lexicon resources.

5.4 Comparison on training time

All of the neural network models are trained on one GeForce GTX 1080 GPU. We compare the average training time (seconds per sentence) of our method on the 5 main datasets with the baseline neural models in Table 2. Since our multi-task model requires training on the 5 datasets together, we calculate and compare the average training time on all datasets instead of on each individual one. We find that our single-task neural model STM is the most efficient among the neural models and almost halves the training time (0.71 s/sent.) when compared to Lample *et al.*; Habibi *et al.* (1.59 s/sent.). Compared to the single-task model STM, our multi-task model MTM-CW achieves 8.0% overall F1 improvements with only 5.1% additional training time. The reason that MTM-CW is slightly slower compared with STM is that it takes a few more epochs for MTM-CW to reach convergence when trained on 5 datasets together.

5.5 Case study

To investigate the major advantages of the multi-task models compared with the single task models, we examine some sentences with predicted labels (Table 5). The true labels and the predicted labels of each model are underlined in a sentence.

One major challenge of BioNER is to recognize a long entity with integrity. In Case 1, the true gene entity is "endo-beta-1,4-glucanase-encoding genes". The single-task model tends to break this whole entity into two parts separated by a comma, while the multi-task model can detect this gene entity as a whole. This result could due to the co-training of multiple datasets containing long entity training examples. Another challenge is to detect the correct boundaries of biomedical entities. In Case 2, the correct protein entity is "SMase" in the phrase "SMase - sphingomyelin complex structure". The single-task models recognize the whole phrase as a protein entity. Our multi-task model is able to detect the correct right boundary of the protein entity, probably also due to seeing more examples from other datasets which may contain "sphingomyelin" as a non-chemical entity. In Case 3, the adjective words "human" and "complement factor" in front of "H deficiency" should be included as part of the true entity. The single-task models missed the adjective words while the multi-task model is able to detect the correct right boundary of the disease entity. In summary, the multi-task model works better at dealing with two critical challenges for BioNER: (1) recognizing long entities with integrity and (2) detecting the correct left and right boundaries of

Table 5. Case study of the prediction results from different models. The true labels and the predicted labels of each model are underlined in the sentence. A brief summary of the error type is also included at the end of each example.

		Genes/Proteins
Case 1	True label	This fragment contains two complete endo - beta - 1, 4 - glucanase - encoding genes, designated celCCC and celCCG.
	Habibi	This fragment contains two complete endo - beta - 1, 4 - glucanase - encoding genes, designated celCCC and celCCG.
	STM	This fragment contains two complete endo - beta - 1, 4 - glucanase - encoding genes, designated celCCC and celCCG.
	MTM-CW	This fragment contains two complete endo - beta - 1, 4 - glucanase - encoding genes, designated celCCC and celCCG.
Error		Entity integrity: break a long entity into parts and lose the entity integrity.
Case 2	True label	A model for the SMase - sphingomyelin complex structure was built to investigate how the SMase specifically recognizes its substrate.
	Habibi	A model for the SMase - sphingomyelin complex structure was built to investigate how the SMase specifically recognizes its substrate.
	STM	A model for the SMase - sphingomyelin complex structure was built to investigate how the SMase specifically recognizes its substrate.
	MTM-CW	A model for the SMase - sphingomyelin complex structure was built to investigate how the SMase specifically recognizes its substrate.
Error		Right boundary error: false detection of non-entity tokens as part of the true entity.
		Diseases
Case 3	True label	... human complement factor H deficiency associated with hemolytic uremic syndrome.
	Habibi	...human complement factor H deficiency associated with hemolytic uremic syndrome.
	STM	... human complement factor H deficiency associated with hemolytic uremic syndrome.
	MTM-CW	... human complement factor H deficiency associated with hemolytic uremic syndrome.
Error		Left boundary error: fail to detect the correct left boundary of the true entity due to some adjective words in front.

biomedical entities. Both improvements come from collectively training multiple datasets with different entity types and sharing useful information between datasets.

6 Conclusion

We proposed an neural multi-task learning approach for biomedical named entity recognition. The proposed approach, despite being simple and not requiring manual feature engineering, outperformed state-of-the-art systems and several strong neural network models on benchmark BioNER datasets. We also showed through detailed analysis that the strong performance is achieved by the multi-task model with only marginally added training time, and confirmed that the large performance gains of our approach mainly come from sharing character- and word-level information between biomedical entity types.

Lastly, we highlight several future directions to improve the multi-task BioNER model. First, combining single-task and multi-task models might be a fruitful direction. Second, by further resolving the entity boundary and type conflict problem, we could build a unified system for recognizing multiple types of biomedical entities with high performance and efficiency.

References

Ando, R. K. (2007). BioCreative II gene mention tagging system at IBM Watson. In *Proc. Second BioCreative Chall. Eval. Work.*, volume 23, pages 101–103.

Camacho, D. M., Collins, K. M., Powers, R. K., Costello, J. C., and Collins, J. J. (2018). Next-Generation Machine Learning for Biological Networks. *Cell*, **173**(7), 1581–1592.

Chiu, J. P. C. and Nichols, E. (2016). Named Entity Recognition with Bidirectional LSTM-CNNs. *Trans. Assoc. Comput. Linguist.*, **4**, 357–370.

Cokol, M., Iossifov, I., Weinreb, C., and Rzhetsky, A. (2005). Emergent behavior of growing knowledge about molecular interactions. *Nat. Biotechnol.*, **23**(10), 1243–1247.

Collobert, R. and Weston, J. (2008). A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proc. 25th ICML*, pages 160–167.

Crichton, G., Pyysalo, S., Chiu, B., and Korhonen, A. (2017). A neural network multi-task learning approach to biomedical named entity recognition. *BMC Bioinf.*, **18**(1), 368.

Davis, A. P., Grondin, C. J., Johnson, R. J., Sciaky, D., King, B. L., McMorran, R., Wiegers, J., Wiegers, T. C., and Mattingly, C. J. (2017). The comparative toxicogenomics database: update 2017. *Nucleic Acids Res.*, **45**(D1), D972–D978.

Deng, L., Hinton, G., and Kingsbury, B. (2013). New types of deep neural network learning for speech recognition and related applications: An overview. In *Proc. IEEE ICASSP*, pages 8599–8603.

Girshick, R. (2015). Fast r-cnn. In *Proc. IEEE ICCV*, pages 1440–1448.

Habibi, M., Weber, L., Neves, M., Wiegandt, D. L., and Leser, U. (2017). Deep learning with word embeddings improves biomedical named entity recognition. *Bioinformatics*, **33**(14), i37–i48.

Huang, C.-C. and Lu, Z. (2015). Community challenges in biomedical text mining over 10 years: success, failure and the future. *Briefings Bioinf.*, **17**(1), 132–144.

Lafferty, J., McCallum, A., and Pereira, F. C. N. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. 17th ICML*, pages 282–289.

Lample, G., Ballesteros, M., Subramanian, S., Kawakami, K., and Dyer, C. (2016). Neural architectures for named entity recognition. In *Proc. Conf. NAACL-HLT*, pages 260–270.

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

Leser, U. and Hakenberg, J. (2005). What makes a gene name? Named entity recognition in the biomedical literature. *Briefings Bioinf.*, **6**(4), 357–369.

Liu, L., Shang, J., Xu, F., Ren, X., Gui, H., Peng, J., and Han, J. (2018). Empower Sequence Labeling with Task-Aware Neural Language Model. In *AAAI*, pages 5245–5253.

Lu, Y., Ji, D., Yao, X., Wei, X., and Liang, X. (2015). CHEMDNER system with mixed conditional random fields and multi-scale word clustering. *J. Cheminf.*, **7**(S1), S4.

Ma, X. and Hovy, E. (2016). End-to-end sequence labeling via bi-directional lstm-cnns-crf. In *Proc. 54th Annu. Meet. ACL*, pages 1064–1074.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013). Distributed Representations of Words and Phrases and their Compositionality. In *Adv. NIPS*, pages 3111–3119.

Pyysalo, S., Ginter, F., Moen, H., Salakoski, T., and Ananiadou, S. (2013). Distributional semantics resources for biomedical text processing. In *Proc. Lang. Biol. Med.*, pages 39–44.

Ramsundar, B., Kearnes, S. M., Riley, P., Webster, D., Konerding, D. E., and Pande, V. S. (2015). Massively multitask networks for drug discovery. *CoRR*, **abs/1502.02072**.

Smith, L., Tanabe, L. K., nee Ando, R. J., Kuo, C.-J., Chung, I.-F., Hsu, C.-N., Lin, Y.-S., Klinger, R., Friedrich, C. M., Ganchev, K., and Others (2008). Overview of BioCreative II gene mention recognition. *Genome Biol.*, **9**(S2), S2.

Søgaard, A. and Goldberg, Y. (2016). Deep multi-task learning with low level tasks supervised at lower layers. In *Proc. 54th Annu. Meet. ACL*, pages 231–235.

Szklarczyk, D., Santos, A., von Mering, C., Jensen, L. J., Bork, P., and Kuhn, M. (2015). Stitch 5: augmenting protein–chemical interaction networks with tissue and affinity data. *Nucleic Acids Res.*, **44**(D1), D380–D384.

Szklarczyk, D., Morris, J. H., Cook, H., Kuhn, M., Wyder, S., Simonovic, M., Santos, A., Doncheva, N. T., Roth, A., Bork, P., and Others (2017). The STRING database in 2017: quality-controlled protein–protein association networks, made broadly accessible. *Nucleic Acids Res.*, **45**(D1), D362–D368.

Tsai, R. T.-H., Wu, S.-H., Chou, W.-C., Lin, Y.-C., He, D., Hsiang, J., Sung, T.-Y., and Hsu, W.-L. (2006). Various criteria in the evaluation of biomedical named entity recognition. *BMC Bioinf.*, **7**(1), 92.

Wei, C. H., Kao, H. Y., and Lu, Z. (2013). PubTator: a web-based text mining tool for assisting biocuration. *Nucleic Acids Res.*, **41**, W518–22.

Xie, B., Ding, Q., Han, H., and Wu, D. (2013). miRCancer: a microRNA–cancer association database constructed by text mining on literature. *Bioinformatics*, **29**(5), 638–644.

Zhou, G. and Su, J. (2004). Exploring deep knowledge resources in biomedical name recognition. In *Proc. Int. Jt. Work. Nat. Lang. Process. Biomed. its Appl.*, pages 96–99.