electronics

MDPI

Article

A Multitask Learning Approach for Named Entity Recognition by Exploiting Sentence-Level Semantics Globally

Wenzhi Huang [1], Tao Qian [2,*], Chen Lyu [3], Junchi Zhang [1], Guonian Jin [2], Yongkui Li [2] and Yongrui Xu [2]

[1] School of Computer Science and Engineering, Wuhan Institute of Technology, Wuhan 430073, China
[2] School of Computer Science and Technology, Hubei University of Science and Technology, Xianning 437000, China
[3] School of Interpreting and Translation Studies, Guangdong University of Foreign Studies, Guangzhou 510420, China
* Correspondence: qtxcm2@gmail.com

Abstract: Named entity recognition (NER) is one fundamental task in natural language processing, which is usually viewed as a sequence labeling problem and typically addressed by neural conditional random field (CRF) models, such as BiLSTM-CRF. Intuitively, the entity types contain rich semantic information and the entity type sequence in a sentence can globally reflect the sentence-level semantics. However, most previous works recognize named entities based on the feature representation of each token in the input sentence, and the token-level features cannot capture the global-entity-type-related semantic information in the sentence. In this paper, we propose a joint model to exploit the global-type-related semantic information for NER. Concretely, we introduce a new auxiliary task, namely sentence-level entity type sequence prediction (TSP), to supervise and constrain the global feature representation learning process. Furthermore, a multitask learning method is used to integrate the global-type-related semantic information into the NER model. Experiments on the four datasets in different languages and domains show that our final model is highly effective, consistently outperforming the BiLSTM-CRF baseline and leading to competitive results on all datasets.

Keywords: named entity recognition; type sequence prediction; multitask learning

Citation: Huang, W.; Qian, T.; Lyu, C.; Zhang, J.; Jin, G.; Li, Y.; Xu, Y A Multitask Learning Approach for Named Entity Recognition by Exploiting Sentence-Level Semantics Globally. *Electronics* **2022**, *11*, 3048. https://doi.org/10.3390/electronics11193048

Academic Editor: Christos J. Bouras

Received: 31 July 2022
Accepted: 21 September 2022
Published: 24 September 2022

1. Introduction

Named entity recognition (NER) aims to seek and identify named entities in unstructured text into predefined entity types, such as person names, organizations, locations, etc. The extracted named entities can benefit various subsequent NLP tasks, including syntactic parsing [1], question answering [2] and relation extraction [3].

NER is usually regarded as a sequence labeling problem [4]. Neural-based conditional random fields (CRF) models, such as BiLSTM-CRF [5], have achieved state-of-the-art performance [6,7]. In particular, based on the word embeddings, we can exploit a long short-term memory network (LSTM) to capture implicit token-level global information, which has been demonstrated to be effective for NER in previous studies [6,8].

However, the token-level features cannot capture the global-type-related semantic information in the sentence, and this problem may limit the capacity for semantic information of the feature representation.

Intuitively, the entity types contain rich semantic information and the entity type sequence in a sentence can reflect the sentence-level semantics globally. For instance, as shown in Figure 1, the type sequence $(PERSON, ORG, LOC)$ can express the semantic information of the sentence. Determining whether a sentence contains a type sequence should enable us to learn rich type-related information for the representation of the token.

Figure 1. A sample with annotated entity types.

Inspired by the end-to-end neural translation method [9], we propose a new end-to-end entity type sequence prediction task (TSP) at the sentence level. The input of the task is the sentence and the output is the entity type sequence in the sentence. To incorporate the information learned from the TSP into the NER model, we employ a multitask learning strategy. The architecture of the joint model is shown in Figure 2. For NER, we use the standard BiLSTM-CRF model as a baseline [5], where the input could be either a word embedding or Bidirectional Encoder Representations from Transformers (BERT). For the TSP, an auxiliary task, we exploit the end-to-end translation model to generate possible type sequences. The two tasks share the same word representation. We combine the encoder representation of BiLSTM between NER and TSP as the final encoding representation for the NER model. In essence, the TSP enhances the semantic feature representation of the token and provides type constraint information for NER. Noticeably, the auxiliary TSP task does not require any extra training data, thus there is no additional annotation cost to achieve our final goal.

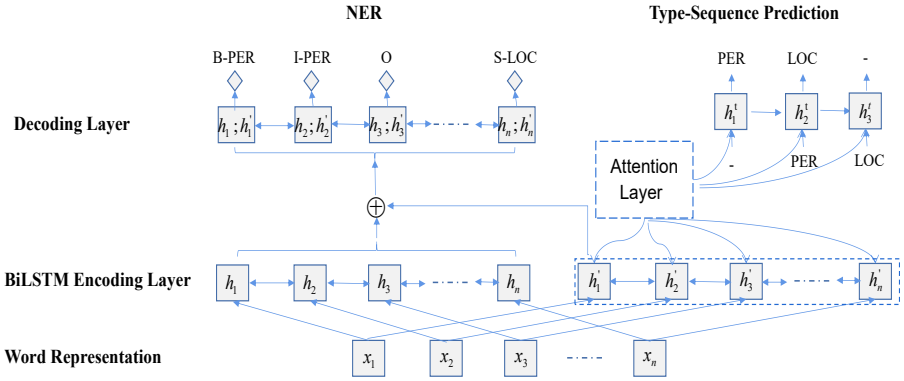


Figure 2. Main architecture of the joint model. The left part is the NER module and the right part is the TSP module. For NER, words tagged with O are outside of named entities and the B-XXX and I-XXX tag refer to the beginning and inside of a named entity of type XXX, respectively.

We conduct experiments on the four benchmark datasets in different languages (English, Chinese and Spanish) and different domains (general and biomedical) to show the effectiveness of our approach. Experimental results demonstrate our model is highly effective, resulting in improvements and leading to state-of-the-art performance on the CoNLL-2003 German NER and JNLPBA datasets without any extra cost. Furthermore, our model can almost achieve the same energy efficiency and latency as the baseline on embedded platforms. Our main contributions can be summarized as follows:

(1) We propose a multitask learning (MTL) approach for NER to exploit the global-type-related semantic information.

(2) We introduce a new auxiliary task, namely the sentence-level entity type sequence prediction (TSP), which enhances the semantic feature representation of tokens and provides type constraint information for NER.

(3) Experimental results demonstrate our MTL model is highly effective, resulting in consistent improvements and leading to state-of-the-art results on the CoNLL-2003 German NER and JNLPBA datasets without any external resource.

2. Related Work

2.1. Neural NER

Early studies on NER often used the hidden Markov model (HMM), maximum entropy Markov model (MEMM) and conditional random fields (CRF) [10] models based on manually crafted discrete features, which suffered from the feature sparse problem and required heavy feature engineering. Recently, neural network models have been successfully applied to sequence labeling [5,11–13]. Among these works, the model which uses BiLSTM for the feature extraction and CRF for decoding has achieved state-of-the-art performances [6,7].

To overcome the issue that neural NER usually requires huge quantities of annotated training data, a number of recent technologies have been proposed, such as pretrained embedding [14], multitask learning [15,16], transfer learning [17], multimodal learning [18,19] and knowledge distillation [20]. Multitask learning can leverage other annotation corpora. Refs. [21,22] studied multitask learning of sequence labeling with language models. Refs. [23,24] proposed multitask learning of NER with word segmentation. The method in [25] utilized multitask learning of NER with several NLP tasks such as part-of-speech (POS) tagging and parsing. Refs. [26,27] leveraged the performance of NER by multitask learning of several tasks of biomedical NLP. Different from the above multitask models, our model does not use external corpora and resources to enhance feature representation.

2.2. Type Constraints

The auxiliary task TSP can learn the type-related semantic information. In essence, our joint model proposes a way to incorporate type constraints for NER. Type constraints are a useful technique in natural language processing (NLP), and here we discuss related work in the literature.

Type constraints have been widely used in NLP, especially for relation extraction and event detection. They have confirmed the fact that the entity types of two entities are important indicators for a specific relation. The authors of [28] explored fine-grained entity type constraints for distantly relation extraction. Ref. [29] used a multitask transfer learning method along with human guidance in the form of entity type constraints. The authors of [30] proposed a novel switchable LSTM, which was able to detect unseen entities using external knowledge. The authors of [31] explored how to use the glyph features of medical fonts to identify Chinese medical entities. The authors of [32] proposed a hierarchical modular event argument extraction model (HMEAE) based on hierarchical type correlation. However, as we do not know the entity boundary and its type in advance, integrating the type constraints into NER is a challenge. In NER, several recent studies proposed to exploit useful visual multimodal information to augment textual information. Some researchers have tried to use other auxiliary tasks to identify type patterns from examples. The authors of [33] proposed a joint model for NER with sentence-level named type prediction, which used an attention network to supervise the feature representation learning globally.

Our work is similar to [33] and both models employ other auxiliary tasks to capture type-related feature using a multitask learning framework. Different from the method in [33], which only predicts entity types, the auxiliary task in our work predicts the entity type sequence which reflects the order of types appearing in a sentence and can better represent the semantics of a sentence. In addition, we adopt a complex neural translation network instead of the attention network, which may limit the learning ability of the model in [33] due to a small number of parameters.

3. Methodology

In this section, we describe the proposed joint model in detail. Our model focuses on two tasks, NER and TSP, where the TSP is the secondary task, and it is exploited as an auxiliary for NER. First, we introduce the word representation used in our model, then we describe the NER and TSP. Finally, we introduce the multitask learning, which combines the NER and TSP by concatenating the hidden layer. Our final goal is to enhance the

performance of NER via the TSP. Figure 2 shows the overall architecture, where the left part is for NER and the right part is for the TSP.

3.1. Word Representation

For a given word sequence $\boldsymbol{s} = \{w_1 \cdots w_n\}$, where w_i is the ith word, the first step was to obtain the word representations of each token. Here, we exploited two kinds of word representations: embeddings and BERT.

Following [7], the embedding representation of one word was created by concatenating its word embedding and its character embeddings, as follows:

$$\boldsymbol{x}_i = [\boldsymbol{e}^w(w_i); \boldsymbol{x}_i^c] \tag{1}$$

where $\boldsymbol{e}^w$ denotes a word embedding lookup table and $\boldsymbol{x}_i^c$ denotes the character-level embeddings. We used BiLSTM to encode character-level embeddings. Assuming $w_i = \{c_{i,1}, \ldots, c_{i,j}, \ldots, c_{i,m_i}\}$ where $c_{i,j}$ denotes the jth character in the ith word ($j \in [1, m_i]$), each $c_{i,j}$ is represented using $\boldsymbol{x}_{i,j}^c = \boldsymbol{e}^c\{c_{i,j}\}$, where $\boldsymbol{e}^c$ denotes a character embedding lookup table. The word-level output $\boldsymbol{x}_i^c = \text{BiLSTM}(\boldsymbol{x}_{i,1}^c \cdots \boldsymbol{x}_{i,m_i}^c)$ denotes the output of the character-level encoding.

BERT has shown great potentials in NLP [14], which is one kind of contextualized word representations. Thus, we also exploited BERT to enhance word representations. We took the outputs of the last layer of a pretrained BERT model as word representations:

$$\boldsymbol{x}_1' \cdots \boldsymbol{x}_n' = \text{BERT}(w_1 \cdots w_n). \tag{2}$$

Finally, we concatenated the BERT representations and the embedding representations above together to obtain boosted word representations, $\boldsymbol{x}_i = [\boldsymbol{e}^w(w_i); \boldsymbol{x}_i^c; \boldsymbol{x}_i']$, as shown in Figure 3.

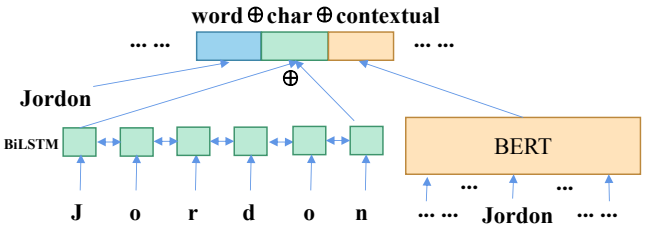


Figure 3. An example to illustrate our final word representation that concatenates its word embedding, char-level representation and contextual BERT representation.

3.2. NER

We employed the BiLSTM-CRF baseline [5] as an NER model in this work. We exploited a bidirectional LSTM network to capture high-level feature representations for NER, which was widely used in previous work [5]. Concretely, the word embedding sequence $(x_1, x_2, \ldots, x_n)$ of each input was taken as the input to a BiLSTM network in each step, then the output sequence of hidden states of the forward LSTM network and the corresponding output sequence of the backward LSTM network were combined according to the position to obtain the complete sequence of hidden states $(\boldsymbol{h}_1 \cdots \boldsymbol{h}_n)$:

$$\boldsymbol{h}_1 \cdots \boldsymbol{h}_n = \text{BiLSTM}(\boldsymbol{x}_1, \cdots, \boldsymbol{x}_n) \tag{3}$$

Finally, a standard CRF layer was used to score the candidate output sequences, which could capture the dependencies between successive labels. Formally, we took the above

sequence of hidden states $(\boldsymbol{h}_1 \cdots \boldsymbol{h}_n)$ as our input to the CRF layer, and its output was our final prediction tag sequence $\widehat{\boldsymbol{y}} = (\widehat{y}_1 \cdots \widehat{y}_n)$, which could be computed as follows:

$$\begin{aligned} \boldsymbol{o}_i &= \boldsymbol{W}\boldsymbol{h}_i \\ p(\widehat{\boldsymbol{y}}\boldsymbol{s}) &= \frac{e^{\sum_i \boldsymbol{o}_i[\widehat{y}_i] + \boldsymbol{T}[\widehat{y}_i, \widehat{y}_{i-1}]}}{\sum_{\boldsymbol{y}'} e^{\sum_i \boldsymbol{o}_i[y'_i] + \boldsymbol{T}[y'_i, y'_{i-1}]}}, \end{aligned} \tag{4}$$

where $\boldsymbol{y}' = y'_1 \cdots y'_n$ is any one of the candidate outputs, $\boldsymbol{W}$ (emission matrix) and $\boldsymbol{T}$ (transition matrix) are two model parameters of CRF and $[\cdot]$ indicates indexing.

The first-order Viterbi algorithm was used to find the highest scored label sequence during decoding. To train the model, the cross-entropy objective function was exploited. Assuming that the GOLD standard tag sequence for sentence $\boldsymbol{s} = w_1 \cdots w_n$ was $\boldsymbol{y} = y_1 \cdots y_n$, the loss function for this single training instance was defined as:

$$\mathcal{L}_{\text{ner}} = -\log p(\boldsymbol{g}\boldsymbol{s}), \tag{5}$$

where all model parameters were optimized to minimize the loss in an online manner.

3.3. TSP

We followed the neural machine translation architecture proposed by [9], which was constructed as a composite of an encoder network, an attention module and a decoder network. The word representation $(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m)$ was shared with the NER model as the input, and the output was a possible type sequence $\boldsymbol{t} = (t_1, \ldots, t_n)$.

The encoder was also a bidirectional LSTM network that read the same input sequence $x = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m)$ as the NER model and calculated a forward sequence of hidden states and a backward sequence. Then, we concatenated the forward and backward hidden states to form a context set:

$$C = \boldsymbol{h}_1^{'} \cdots \boldsymbol{h}_n^{'} = \text{BiLSTM}(\boldsymbol{x}_1, \cdots, \boldsymbol{x}_n) \tag{6}$$

Then, the decoder computed the conditional distribution over all possible type sequences based on this context set. This was done by first rewriting the conditional probability of a predicted type sequence:

$$logp(\boldsymbol{t}\boldsymbol{x}) = \sum_{k'=1}^{K_t} logp(t_{k'} t_{<k'}, \boldsymbol{x}). \tag{7}$$

For each conditional term in the summation, the decoder BiLSTM network updated its hidden state by

$$h_{k'}^t = (e_t(t_{k'-1}), h_{k'-1}^t, c_{k'}), \tag{8}$$

where is the activation function and e_t is the continuous embedding of a target entity type. $c_{k'}$ is a context vector computed by an attention mechanism:

$$c_{t'} = f_{att}(e_t(t_{k'-1}), h_{k'-1}^t, C). \tag{9}$$

The attention mechanism f_{att} weighted each vector in the context set C according to its relevance to the entity type generated by the decoder. Note that the attention submodule was self-supervised, and the query of attention was the previous generated type. The weight of each vector $h_k^{'}$ was computed by

$$\alpha_{k,k'} = \frac{1}{Z} f_{score}(e_t(t_{k'-1}), h_{k'-1}^t, h_t^{'}), \tag{10}$$

where f_{score} is a parametric function returning an unnormalized score for $h_k^{'}$ given $h_{k'-1}$, $t_{k'-1}$. We used a feedforward network with a single hidden layer in this paper. Z was a normalization constant:

$$Z = \sum_{t=1}^{T_x} f_{score}(e_t(t_{k'-1}), h_{k'-1}^t, h_t^{'}). \tag{11}$$

This procedure can be understood as computing the potential relationship between the $k_{th}^{'}$ target type and t_{th} source word.

The hidden state $h_{k'}$, together with the previous target type $t_{k'-1}$ and the context vector $c_k^{'}$, was fed into a feedforward neural network to generate the conditional distribution:

$$p(t_{k'}t_{<k'}, \boldsymbol{x}) \propto e^{f_{out}(e_t(t_{k'-1}), h_{k'}, c_{k'})} \tag{12}$$

For training, the model parameters were learned by minimizing the negative log-likelihood, where the loss function for a single input sentence was defined as follows:

$$\mathcal{L}_{tsp}(\boldsymbol{t}) = -log(p(\boldsymbol{t}, \boldsymbol{x})). \tag{13}$$

3.4. Multitask Learning

In order to integrate the TSP task into our baseline model and create full interactions between the NER and auxiliary TSP task at the same time, we used the multitask learning method to integrate the type information into the NER model. We concatenated the hidden embedding of the two tasks in the BiLSTM encoder layers and obtained basic feature inputs for the decoding layer of the NER model.

For NER in the MTL model, we added a sequence of new features from the TSP task to enhance NER. Concretely, we combined the newly obtained hidden word representations from the TSP encoder with the original BiLSTM outputs to get the final feature representations for NER:

$$\boldsymbol{h}_i^{\text{joint}} = [\boldsymbol{h}_i; \boldsymbol{h}_i'], \tag{14}$$

where simple concatenations were adopted for the combination, and $\boldsymbol{h}_i^{\text{joint}}$ was used in our final joint model for NER.

The multitask learning model had two kinds of losses and we combine them by a dynamic weight λ as follows:

$$\mathcal{L}_{\text{joint}} = \lambda * \mathcal{L}_{\text{ner}} + (1 - \lambda) * \mathcal{L}_{\text{tsp}} \tag{15}$$

where the λ is a hyperparameter assigned to control the degree of importance for each task.

In our tests, we only predicted named entities and did not execute the TSP task. Therefore, there was only a very slight efficiency decrease during decoding compared with the BiLSTM-CRF baseline.

4. Experiments

4.1. Datasets

We conducted experiments on four benchmark datasets involving two domains and three languages to evaluate our model. The statistics of these datasets are described in Table 1. Detailed descriptions of the datasets are given in the following.

- **OntoNotes 5.0 (English)** is an English NER dataset with 18 entity types (PERSON, CARDINAL, LOC, PRODUCT, etc.), which consists of texts from the general domain [34]. We adopted the data splitting method in [35] to obtain the training, development and testing sets (http://conll.cemantix.org/2012/data.html, accessed on 25 December 2019).
- **CoNLL-2003 (German)** is a German NER dataset with four entity types: LOCATION, ORGANIZATION, PERSON and MISCELLANEOUS. We used the official splitting

method to divide the dataset into the training, development and testing sets (https://www.clips.uantwerpen.be/conll2003/ner/, accessed on 20 October 2021) [36].

- **MSRA (Chinese)** is a Chinese NER dataset in newswire domain [37], which includes three entity types: LOCATION, ORGANIZATION and PERSON. We used the official data splitting method to preprocess the dataset. During training, we randomly split the official training set into training and development sets with the ratio 9:1.
- **JNLPBA (medical)** is a biomedical NER dataset in the English language, including five entity types: PROTEIN, DNA, RNA, CELL_LINE and CELL_TYPE. We adopted the same dataset splitting method as [38] to preprocess the dataset (http://www.geniaproject.org/shared-tasks/bionlp-jnlpba-shared-task-2004, accessed on 20 October 2021).

Table 1. Statistics of the datasets, where #type denotes the number of types, #sent denotes the number of sentences and #entity denotes the number of entities.

Data		Training	Dev	Testing
OntoNotes 5.0 (English)	#type	18	18	18
	#sent	59,924	8528	8262
	#entity	81,828	11,066	11,257
CoNLL-2012 (German)	#type	4	4	4
	#sent	12,705	3068	3160
	#entity	11,851	4833	3673
MSRA (Chinese)	#type	3	-	3
	#sent	46,400	-	4400
	#entity	75,059	-	5334
JNLPBA (medical)	#type	5	5	5
	#sent	16,692	1854	3856
	#entity	46,390	4911	8662

4.2. Evaluation Metrics

Standard precision (P), recall (R) and F1-score (F_1) were used as the evaluation metrics for named entity recognition. Note that as an auxiliary task, we do not report the performance of the TSP task in the main results. All experiments were conducted 100 times with random seeds sampled from 1 to 100 and the average performance outputs were used for the result reporting and analysis. We converted the IOB boundary encoding to the BIOES as previous work found this encoding resulted in improved performance [39].

4.3. Settings

4.3.1. Hyperparameters

The hyperparameters were tuned on the corresponding development sets. Specifically, we selected the best hyperparameters by searching a combination of the hidden size, the learning rate and λ from the following range: the hidden size from $\{100, 200, \mathbf{400}\}$, the learning rate from $\{0.005, 0.010, \mathbf{0.015}, 0.020\}$ and λ from $\{0.5, 0.6, 0.7, \mathbf{0.8}, \mathbf{0.9}\}$. The bold parameters were selected as the hyperparameters in our experiments. We optimized our models with a stochastic gradient descent (SGD) following [7]. We applied a dropout rate of 0.5 to the embeddings and hidden states. The training procedure stopped when the results of the next five validations were not better than the previous best record. The other hyperparameter settings followed [7]. Table 2 shows all the hyperparameters used in our experiments.

Table 2. Hyperparameters.

Hyperparameters	Value	Hyperparameters	Value
char emb size	30	dropout	0.5
char hidden	50	batch size	10
word emb size (English)	100	optimizer	SGD
word emb size (Spanish)	300	momentum	0.9
word emb size (Chinese)	100	L_2 regularization	1×10^{-8}
word emb size (medical)	200	learning rate	0.015
word hidden	400	learning rate decay	0.05
CNN window	3	gradient clipping	5.0
word CNN layer	4	λ	0.8
number of attention heads	5		

4.3.2. Word Representations

We exploited two kinds of word representations. (1) Embeddings: For the OntoNotes 5.0 datasets, the 100-dimensional GloVe embeddings (https://nlp.stanford.edu/projects/glove/, accessed on 20 October 2021) [40] were used to initialize our word representation. For the CoNLL-2002 German dataset, we initialized the word embeddings using the 300-dimensional embeddings from Wikipedia2Vec (https://wikipedia2vec.github.io/wikipedia2vec/pretrained/, accessed on 20 October 2021) [41]. For JNLPAB, the 200-dimension embeddings (https://github.com/cambridgeltl/BioNLP-2016, accessed on 20 October 2021) from [42] were employed. For the MSRA dataset, we employed the 300-dimensional glove embeddings (https://fasttext.cc/docs/en/crawl-vectors.html, accessed on 20 October 2021). The word embeddings were fine-tuned during training.

(2) BERT was also exploited for a stronger baseline. For the OntoNotes 5.0 dataset, we used the BERT base released by Google as inputs (https://github.com/google-research/bert#pre-trained-models, accessed on 20 October 2021). For the German language, we exploited the pretrained German BERT (https://github.com/deepset-ai/FARM, accessed on 2021-10-20) as inputs. Furthermore, for the MSRA dataset, we used the pretrained Chinese-BERT [43]. For the JNLPBA dataset, we used BioBERT (https://github.com/dmis-lab/biobert/, accessed on 20 October 2021) [44]. In the BERT-based experiments, we froze the parameters in BERT and fine-tuned the accompanying word embeddings during training.

4.3.3. Implementation

The models were implemented using Pytorch 1.2.0. The number of model parameters was 17.6 M. All the experiments were conducted using a single NVIDIA 1080Ti GPU and Ubuntu 18.04. Furthermore, the training time on the OntoNotes 5.0, CoNLL-2003, MSRA and JNLPBA datasets were 18 h, 4 h, 8 h and 6 h, respectively. We released all codes publicly at https://github.com/qtxcm/JointNERwithTSP for research purpose under the Apache License 2.0.

4.3.4. Baselines

Besides the BiLSTM-CRF baseline, we also took the joint model proposed by [33] as another baseline to verify whether predicting type sequences was more effective than only predicting types.

For the convenience of description, we abbreviate the two baselines as BaseNER, BaseJoint, respectively, in the following experiments.

4.4. Development Experiments

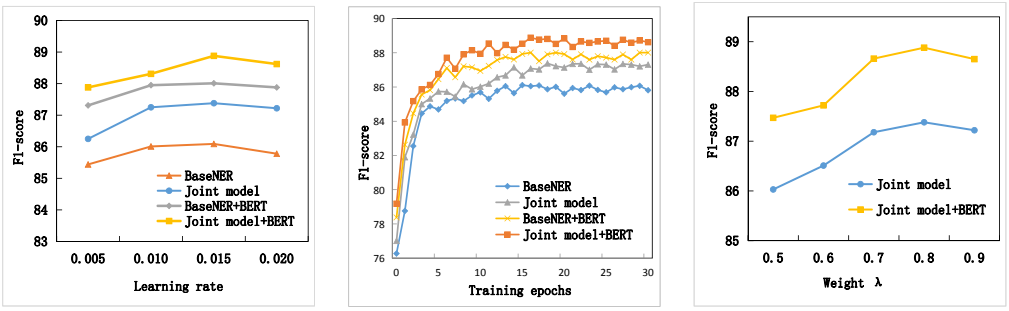
We conducted development experiments with various model configurations in order to select the best settings for the BaseNER and joint models without and with BERT.

4.4.1. The Influence of Hyperparameters

We studied the sensitivity of hyperparameters of model, i.e., the learning rate, the training epochs and the weights λ. All experiments were based on the development set of OntoNotes 5.0 (English). In Figure 4a, we compare the validation F1-score of our model against the BaseNER without and with BERT. We observed that the two models could achieve their best score at the learning rate of 0.015, while our model had a higher F1-score than the BaseNER when training with the same learning rate.

Figure 4b shows the F1-score curves for the BaseNER and joint models without and with BERT against the number of training iterations. As can be seen from the figure, BERT information was helpful for NER, improving the best development results on the BaseNER and our joint model, respectively. In addition, our joint models performed better than the BaseNER in all iterations.

As shown in Equation 15, λ was assigned to control the degree of importance for each task. Figure 4c shows the F1-score curves of NER for the joint models without and with BERT using different λ's. We set λ to 0.8 according to the model performance.



(**a**) (**b**) (**c**)

Figure 4. Comparisons between BaseNER and our method on the F1 sensitivity of the hyperparameters. All results are generated based on OntoNotes 5.0 (English). (**a**) F1-score vs. Learning rate; (**b**) F1-score vs. Training epochs; (**c**) F1-score vs. the Weight λ.

4.4.2. Development Results Against Baselines

Table 3 shows the development results. We observe that by adding the auxiliary tasks, the performance of our joint model and the BaseJoint model was higher than that of the BaseNER model on all four datasets. This shows that adding type constraints was effective for named entity recognition. Compared to the BaseNER model, the BaseJoint model had average F1-score improvements of 0.56% and 0.39% on word representations without and with BERT, respectively, while our method achieved the highest F1-scores in all cases with average improvements of 0.70% and 0.49%, respectively. This shows that our proposed objective was effective in different classification tasks and different word representations. On the other hand, our method achieved a better performance than the BaseJoint model in all cases. This demonstrates that predicting type sequence should be able to capture type-related features better than only predicting types.

Table 3. The P, R and F_1 of development sets on four datasets.

Model	OntoNotes 5.0 (English)			CoNLL-2003 (German)		
	P	**R**	**F_1**	**P**	**R**	**F_1**
BaseNER	85.94	86.25	86.09	79.48	64.35	71.12
BaseJoint	85.78	**87.58**	86.67	79.45	65.57	71.85
Joint model	**86.23**	87.38	**86.80**	**79.83**	**65.58**	**72.01**
BaseNER + BERT	87.63	88.01	87.82	87.25	87.06	87.15
BaseJoint + BERT	**88.34**	88.21	88.27	**88.47**	86.96	87.71
Joint model + BERT	87.85	**88.88**	**88.36**	87.83	**87.77**	**87.80**
Model	**MSRA(Chinese)**			**JNLPBA(Medical)**		
	P	**R**	**F_1**	**P**	**R**	**F_1**
BaseNER	90.92	88.17	89.52	79.56	77.31	78.42
BaseJoint	**91.86**	88.04	89.91	79.25	**78.67**	78.96
Joint model	91.03	**88.94**	**89.97**	**80.20**	78.13	**79.15**
BaseNER + BERT	93.98	94.37	94.17	81.06	79.44	80.24
BaseJoint + BERT	**95.16**	94.14	94.65	80.87	79.78	80.32
Joint model + BERT	94.18	**95.23**	**94.70**	**81.13**	**79.84**	**80.48**

4.5. Main Results

Tables 4–7 present the final results on the four benchmark test datasets when comparing with the baselines and existing state-of-the-art models. Similar to the development set results, the two auxiliary methods consistently improved the performance of the BaseNER model, and our joint model achieved better performance than the BaseJoint model in all cases. Specifically, by using BERT word representations, our final model obtained better performance than the BaseNER model, leading to improvements by 0.49%, 0.70%, 0.06% and 0.85% on the four datasets, respectively. By using other word embeddings as word representations, our final model could still bring F1-score improvements of 0.75%, 0.66%, 0.82% and 1.22% on the four datasets, respectively. We compare with previous works on the four datasets in the following.

Table 4. F_1-scores on the OntoNotes 5.0 test set. * denotes semisupervised and multitask learning.

Model	**F_1**
BiLSTM-CNN [45]	86.17
Dilated-CNN-CRF [46]	86.84
CNN-CNN-LSTM [47]	86.52
LS-BiLSTM-CRF [8]	87.95
Semisupervised CVT [25] *	**88.81**
Hierarchical BiLSTM-CRF [6]	87.98
GRN [48]	87.67
BiLSTM-LAN [7]	88.16
BaseNER	87.66
BaseJoint	88.37
Joint model	88.41
Bert-Tagger [14]	89.16
Hierarchical + BERT [6]	90.30
BERT-MRC [49]	**91.11**
BaseNER + BERT	89.07
BaseJoint + BERT	89.53
Joint model + BERT	89.56

Table 5. F_1-scores on CoNLL-2003 German.

Model	F_1
BiLSTM-CRF [12]	78.76
BiLSTM-CRF [50]	79.01
BaseNER	79.36
BaseJoint	79.94
Joint model	**80.02**
multilingual-BERT [51]	82.82
BaseNER + BERT	86.98
BaseJoint + BERT	87.37
Joint model + BERT	**87.68**

Table 6. F_1-scores on MSRA (Chinese), * indicates the models were trained with the use of external labeled data.

Model	F_1
Character-based LSTM-CRF [52]	90.95
Lattice LSTM [53] *	**93.18**
BaseNER	90.13
BaseJoint	90.86
Joint model	90.95
BERT-Tagger [14]	94.80
BERT-MRC [49] *	**95.75**
BaseNER + BERT	94.78
BaseJoint + BERT	95.35
Joint model + BERT	95.41

Table 7. F_1-scores on the JNLPBA test set.

Model	F_1
Char-attention BiLSTM-CRF [54]	72.70
nested BiLSTM-CRF [55]	75.44
Dictionary BiLSTM-CRF [56]	71.99
BiLSTM-softmax [57]	73.60
LMs-BiLSTM-CRF [58]	74.29
BaseNER	74.22
BaseJoint	75.01
Joint model	**75.44**
Bert-Tagger [14]	75.98
BaseNER + BERT	75.69
JointNER + BERT	76.38
Joint model + BERT	**76.54**

OntoNotes 5.0 (English) dataset For the OntoNotes 5.0 (English) dataset, there have been an extensive number of investigations, as shown in Table 4. We can see that both our embedding and BERT baselines were competitive, comparable with previous works. The state-of-the-art systems [25,49] on this dataset both utilized external resources. Ref. [25] presented a cross-view training for NER, with several auxiliary tasks such as chunking and dependency parsing. Ref. [49] reformalized NER as an MRC question-answering task and trained on the reconstructed data, which exploited expert knowledge to construct query questions. Actually, our MTL model with BERT could achieve the best performance when no external resource, such as syntax and any other human-supervision, was used.

CoNLL-2003 (German) dataset For the CoNLL-2003 German dataset, we also listed the state-of-the-art models in the literature. Two of them used the BiLSTM-CRF methods, similar to our baseline system, and the BERT-based model employed a multilanguage model. As shown in Table 5, our baseline models were comparable to the corresponding best-performance systems, and our final MTL models obtained the best results with both BERT and other word embedding-based representations.

MRSA (Chinese) dataset For the MSRA dataset, we listed the four state-of-art models. Our final model outperformed the state-of-the-art systems without external resources, except that [53] exploited an external lexicon and [49] used expert knowledge of entity types.

JNLPBA (Medical) dataset On the biomedical JNLPBA dataset, we compared our model with previous state-of-the-art models, including [54,56], etc. With a BERT-based representation, our MTL model consistently outperformed all previous models, achieving the best results.

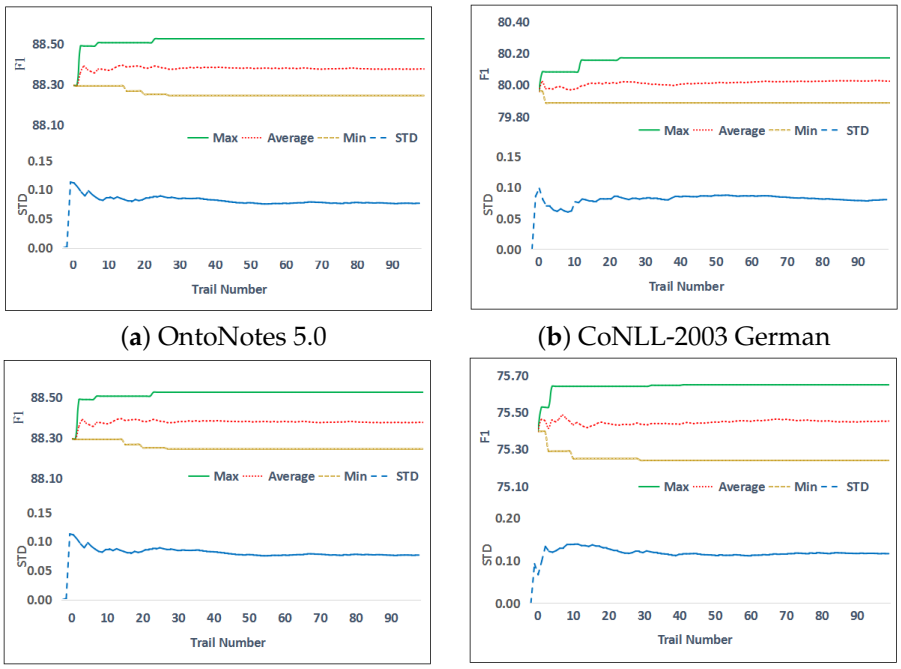
Overall, the results on four datasets indicated that our proposed model could achieve comparable NER performance on the four datasets. It demonstrated that our joint model could outperform the two baselines for NER by using the auxiliary TSP task. Moreover, we did not fine-tune any hyperparameters to fit the specific task. The results in Tables 4–7 are all with the same hyperparameters, which demonstrates the generalization ability of our joint framework.

4.6. Analysis

In this section, we conducted a detailed analysis to understand how our auxiliary TSP task helped to improve the performance of NER. The selected models here exploited embedding-based word representations.

4.6.1. Statistical Analysis

First, we investigated the statistical significance of the main experimental results and the effect of random initialization of weights in the networks. In the experiments, each dataset was tested 100 times with the same set of hyperparameters and random seeds sampled from 1 to 100. We introduce four metrics, (1) max, (2) average, (3) min and (4) standard deviation (STD), to measure the changes in F-scores for each trail. For example, max_i, $average_i$ and min_i are the maximum, average and minimum F-score of the first i trails, respectively, and STD_i represents the standard deviation of the first i F-scores . Figure 5 reports their curves on the four datasets. The corresponding values in the y-axis of index i are reported based on the experimental results of the first i trails. We can see that all curves became smooth after the first few trails. In addition, all STD values were also quite small, indicating that different random initialization of weights had no significant effect on the final results and the joint model was consistently stable.



(**a**) OntoNotes 5.0 (**b**) CoNLL-2003 German

(**c**) MSRA 5.0 (**d**) JNLPBA

Figure 5. Max, average, min and standard deviation (STD) of the F_1-scores of 100 random trails on the four datasets. X-axis denotes the trail time i ($i = 1, 2, ..., 100$). Y-axis denotes the corresponding values based on the experimental results of the first i trails.

4.6.2. Network Comparison

We compared the network structure of the joint model with the BaseNER model. Table 8 lists the parameter size of the two models and the training and inference time on the CoNLL-2003 German dataset. The joint model contained one additional TSP submodule and the parameter size of the joint model was 1.5 M larger than that of the baseline. Thus, the training time of the joint model was about 33% longer than that the baseline. However, we can see that the inference speeds of the two models were 91 st/s and 89 st/s, respectively, which were almost the same.

We also compared our joint model with the baseline on different computing platforms. As shown in Table 9, the experiments ran on three different computing platforms: the Intel Core i7-8700k, NVIDIA GTX1080 and ARM Cortex M7. Note that the power of the CPU was its thermal design power (https://ark.intel.com/content/www/us/en/ark/products/126684/intel-core-i7-8700k-processor-12m-cache-up-to-4-70-ghz.html, accessed on 2 August 2022), while the GPU power value was from the nvidia-smi program. Since we could not obtain the individual power values of each CPU component due to a lack of power sensors, we utilized a power measurement module to measure the whole board power consumption for the embedded ARM platform. We can see that our model almost performed the same as the baseline with different metrics on the same platform, although the performance varied widely across different platforms.

Table 8. Network comparison. Note M is short for million, s is short for second, st/s is the inference speed, which is the number of sentences inferred per second.

Model	Parameters Size			Time	
	Total	For Training	Random Weights	Training	Inference
BaseNER	4.8 M	4.8 M	0.8 M	586 s	91 st/s
Joint model	6.3 M	6.3 M	2.3 M	795 s	89 st/s

Table 9. Performance comparison between different platforms. st/s is used for the throughput metric, which is the number of sentences inferred per second.

Platform	**CPU (Intel Core i7-8700k) 3.7 GHz**		**GPU (NVIDIA GTX1080) 1607 MHz**		**ARM (ARM Cortex M7) 480 MHz, 32 M RAM**	
Model	**BaseNER**	**Joint Mode**	**BaseNER**	**Joint Mode**	**BaseNER**	**Joint Mode**
Power (W)	95.0	95.0	75.0	78.0	6.1	6.3
Latency (ms)	12	13	7.4	7.6	55	57
Energy (kJ)	12.1	12.5	6.8	7.2	2.1	2.2
Throughput (st/s)	65	63	91	89	25	24

4.6.3. The Correlation between NER and TSP

Our final model jointly performed NER and TSP. Although TSP was regarded as an auxiliary task, whose performance was not of concern, it was still interesting to check the performance correlation between the two tasks. In this section, we present a scatter figure, where the x-axis value denotes the TSP performance (i.e., BLUE-1) , and the y-axis value denotes the F_1-score of NER. For each point, the paired values were calculated by randomly selecting 50 sentences. We sampled 1000 times, arriving at 1000 points. Figure 6 shows the above results.

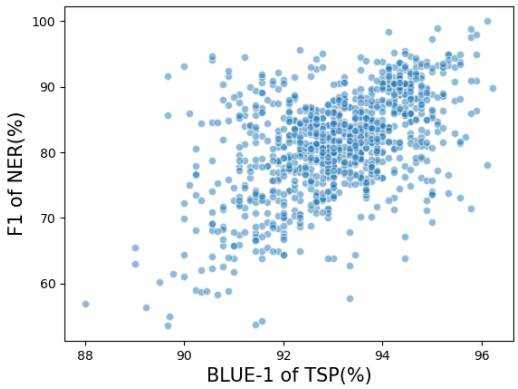


Figure 6. The scatter graph of F_1 of NER over accuracy of TSP on the OntoNotes test set.

We can see that a roughly positive correlation is shown by the picture, where the majority of points are around a line with a positive slope, which is consistent with the intuition that the TSP performance should demonstrate positive correlations with NER.

4.6.4. Fine-Grained Analysis for NER Performance

Further, we performed a fine-grained performance analysis for NER. We conducted the analysis on two aspects. First, we show the F1-score of NER with respect to the number of entities in sentence. We grouped the number into four categories, and the entity numbers above four (including four) were merged as one category. Figure 7a shows the results, where the performance of the BaseNER model, as well as that of the final model, is offered. We can see that, as the number of entities increased from one to four, the NER performance grew as well. The possible reason may be because the named entities in one sentence were closely related, thus each entity could help the recognition of the others. In addition, our final model outperformed the baseline in all categories, demonstrating the effectiveness of our method.

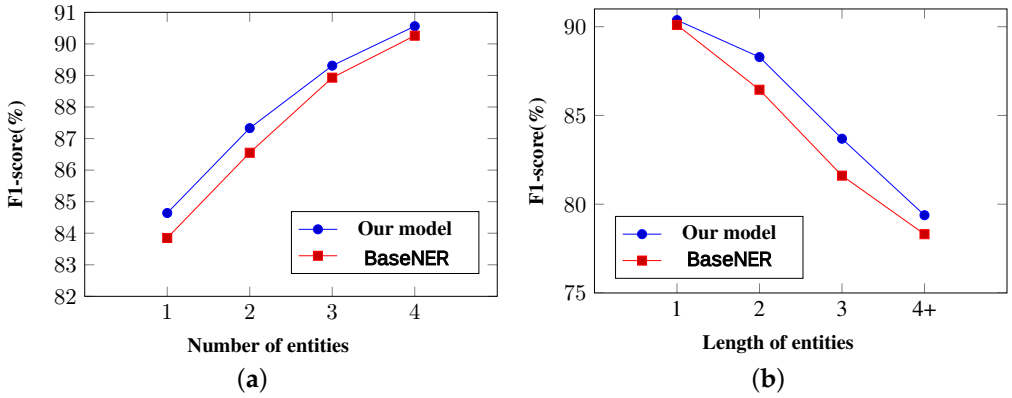


Figure 7. F_1-score against entity length and sentential entity number on the OntoNotes test set. (**a**) Sentential entity number. (**b**) Entity length.

Second, we show the NER performance in terms of NER length. We categorized the NER lengths into four categories: 1, 2, 3 and 4+ (i.e., $\geq$4). Figure 7b shows the results. Similarly, the F1-scores of both the BaseNER and the final systems are offered. As shown in Figure 7b, we can see that the NER performance decreased with the length increases, which was reasonable. Our final model was still better than the BaseNER model in each category, which further proved the benefit of TSP.

4.6.5. Case Study

We conducted a case study. Table 10 shows four cases with different numbers of entities on the OntoNotes dataset. In case 1, *Mt. Everest* is usually regarded as the type *LOCATION*, but it was *FACILITY* in sentence 1. Due to the limited contextual information, the BaseNER model made a mistake, while our MTL model correctly recognized it and was able to capture the type-related information by using type sequence prediction. For case 2, *Nine Eleven*, incorrectly identified as *TIME* by the BaseNER model, was correctly identified as *EVENT* by our model due to the integration of the type sequence information. In case 3, there were three entities in the sentence. We found that the BaseNER model made the same mistake identifying *the White House* as *ORGANIZATION*. However, the type of *the White House* is more likely *FACILITY* in this context. This type correlation information could be learned by the neural machine model in our model. In the last case, the BaseNER model did not recognize the entity *Western*, while our model could correctly identify it and its type. These cases demonstrated that the TSP could incorporate rich type-related information into NER and our model had a better capacity to deal with entity type ambiguity.

Further, we conducted another case study to compare the type sequence prediction performance of our model with the baselines on specific type sequences. The sentence-level type sequence prediction requires all entity types (i.e., type-sequence) in a sentence be recognized correctly. Table 11 shows the results of the three models on four specific type sequences from the OntoNotes dataset. We can see that the sentence-level entity type accuracy of our model was significantly improved, compared with that of the baselines. Furthermore, we also found that our model was more effective than the baselines when the sequence was longer. This showed that our model could better capture sentence-level type-related features by predicting type sequences.

Table 10. Examples of that our model can integrate type chaining structure to constrain entity types, while the BaseNER cannot.

	Case 1
Gold	They 're [Mt.Everest]_FACILITY to conquer.
BaseNER	They 're [Mt.Everest]_LOCATION to conquer.
Our model	They 're [Mt.Everest]_FACILITY to conquer.
	Case 2
Gold	[One]_CARDINAL was [Nine Eleven]_EVENT .
BaseNER	[One]_CARDINAL was [Nine Eleven]_TIME .
Our model	[One]_CARDINAL was [Nine Eleven]_EVENT .
	Case 3
Gold	[Republicans]_NORP have not held [the White House]_FACILITY for [eight years]_DATE.
BaseNER	[Republicans]_NORP have not held [the White House]_ORGANIZATION for [eight years]_DATE.
Our model	[Republicans]_NORP have not held [the White House]_FACILITY for [eight years]_DATE.
	Case 4
Gold	Later, as a result of [Sino-French]_NORP war of [1884]_DATE , [Tanshui]_GPE was again opened up to [Western]_NORP access.
BaseNER	Later, as a result of [Sino-French]_NORP war of [1884]_DATE , [Tanshui]_GPE was again opened up to Western access.
Our model	Later, as a result of [Sino-French]_NORP war of [1884]_DATE , [Tanshui]_GPE was again opened up to [Western]_NORP access.

Table 11. Sentence-level entity type accuracy on specific entity type sequences. N is the number of samples containing the specific entity type sequence.

Type Sequence	**N**	**Sentence-Level Entity Type Accuracy (%)**		
		BaseNER	**BaseJoint**	**Our Model**
PERSON	505	83.1	84.1	84.6
PERSON-ORG	53	71.7	77.4	79.3
ORG-PERSON	44	63.6	75.0	77.3
PERSON-ORG-GPE	19	31.6	52.6	57.9
ORG-GPE-DATE-PERSON	15	20.0	33.3	46.7

5. Conclusions

In this paper, we proposed a multitask learning framework that incorporated the entity type-related information for NER. To achieve this goal, we investigated the auxiliary TSP task. The TSP task had a positive correlation and could capture potential type-related information at the sentence level. Noticeably, the TSP task shared the same input with the NER, and the type labels for TSP could be directly obtained from the NER training corpus. Thus, the TSP task did not require any extra training data, and there was no additional annotation cost in our framework. Experimental results demonstrated our MTL model was

highly effective, resulting in consistent improvements and leading to state-of-the-art results on the CoNLL-2003 German NER and JNLPBA datasets without any external resource.

This article demonstrated the effectiveness of TSP on named entity recognition and our joint model did not depend on any additional annotation data. Therefore, we will explore the TSP task on other more challenging NLP tasks, such as relation extraction and reading comprehension, in the future.

Author Contributions: Conceptualization, T.Q. and W.H.; methodology, W.H.; software, C.L.; validation, J.Z., Y.X. and G.J.; writing—original draft preparation, W.H.; writing—review and editing, T.Q.; visualization, Y.L.; supervision, T.Q.; project administration, T.Q.; funding acquisition, T.Q. and G.J. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported in part by the Social Science Foundation of Ministry of Education of China (nos. 19YJAZH070, 19YJCZH114), the National Natural Science Foundation of China (nos. 61772378, 62006054, 62106179), the Doctoral Fund of Hubei University of Science and Technology (no. BK201812), the Science and Technology Project of Guangzhou (no. 202102020473), the National Key Research and Development Program of China (no. 2017YFC1200500), and the Scientific Research Program of Department of Education of Hubei Province (no. D20202801).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Informed consent was obtained from all subjects involved in the study.

Data Availability Statement: Data analyzed in this study were a re-analysis of existing data, which are openly available at locations cited in the reference section.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Koo, T.; Collins, M. Efficient third-order dependency parsers. In Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, Uppsala, Sweden, 11–16 July 2010; pp. 1–11.
2. Krishnamurthy, J.; Mitchell, T.M. Learning a compositional semantics for Freebase with an open predicate vocabulary. *Trans. Assoc. Comput. Linguist.* **2015**, *3*, 257–270. [CrossRef]
3. Lao, N.; Cohen, W.W. Relational retrieval using a combination of path-constrained random walks. *Mach. Learn.* **2010**, *81*, 53–67. [CrossRef]
4. McCallum, A.; Li, W. Early results for Named Entity Recognition with Conditional Random Fields, Feature Induction and Web-Enhanced Lexicons. In Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003, Edmonton, Canada, May 31–June 1 2003; pp. 188–191.
5. Huang, Z.; Xu, W.; Yu, K. Bidirectional LSTM-CRF models for sequence tagging. *arXiv* **2015**, arXiv:1508.01991.
6. Luo, Y.; Xiao, F.; Zhao, H. Hierarchical Contextualized Representation for Named Entity Recognition. *arXiv* **2019**, arXiv:1911.02257.
7. Cui, L.; Zhang, Y. Hierarchically-Refined Label Attention Network for Sequence Labeling. *arXiv* **2019**, arXiv:1908.08676.
8. Ghaddar, A.; Langlais, P. Robust Lexical Features for Improved Neural Network Named-Entity Recognition. In Proceedings of the 27th International Conference on Computational Linguistics, Santa Fe, NM, USA, 21–25 August 2018; pp. 1896–1907.
9. Luong, M.T.; Pham, H.; Manning, C.D. Effective Approaches to Attention-based Neural Machine Translation. In Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, Lisbon, Portugal, 17–21 September 2015; pp. 1412–1421.
10. Lafferty, J.D.; McCallum, A.; Pereira, F.C. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In Proceedings of the Eighteenth International Conference on Machine Learning, San Francisco, CA, USA, 28 June–1 July 2001; pp. 282–289.
11. Collobert, R.; Weston, J.; Bottou, L.; Karlen, M.; Kavukcuoglu, K.; Kuksa, P. Natural language processing (almost) from scratch. *J. Mach. Learn. Res.* **2011**, *12*, 2493–2537.
12. Lample, G.; Ballesteros, M.; Subramanian, S.; Kawakami, K.; Dyer, C. Neural Architectures for Named Entity Recognition. In Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego, CA, USA, 12–17 June 2016; pp. 260–270.
13. Ali, F.; Ali, A.; Imran, M.; Naqvi, R.A.; Siddiqi, M.H.; Kwak, K.S. Traffic accident detection and condition analysis based on social networking data. *Accid. Analy. Prevent.* **2021**, *151*, 105973. [CrossRef]
14. Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv* **2018**, arXiv:1810.04805.
15. Adi, Y.; Kermany, E.; Belinkov, Y.; Lavi, O.; Goldberg, Y. Fine-grained analysis of sentence embeddings using auxiliary prediction tasks. *arXiv* **2016**, arXiv:1608.04207.

16. Beryozkin, G.; Drori, Y.; Gilon, O.; Hartman, T.; Szpektor, I. A Joint Named-Entity Recognizer for Heterogeneous Tag-setsUsing a Tag Hierarchy. *arXiv* **2019**, arXiv:1905.09135.
17. Lee, J.Y.; Dernoncourt, F.; Szolovits, P. Transfer Learning for Named-Entity Recognition with Neural Networks. *arXiv* **2017**, arXiv:1705.06273.
18. Moon, S.; Neves, L.; Carvalho, V. Multimodal Named Entity Recognition for Short Social Media Posts. In Proceedings of the Proceedings of NAACL-HLT, New Orleans, LA, USA, 1–6 June 2018; pp. 852–860.
19. Zhang, D.; Wei, S.; Li, S.; Wu, H.; Zhu, Q.; Zhou, G. Multi-modal graph fusion for named entity recognition with targeted visual guidance. In Proceedings of the AAAI Conference on Artificial Intelligence, Virtual, 22 February–1 March 2022; Volume 35, pp. 14347–14355.
20. Rashid, A.; Lioutas, V.; Ghaddar, A.; Rezagholizadeh, M. Towards Zero-Shot Knowledge Distillation for Natural Language Processing. In Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, Punta Cana, Dominican Republic, 7–11 November 2021; pp. 6551–6561.
21. Liu, P.; Qiu, X.; Huang, X. Adversarial multi-task learning for text classification. *arXiv* **2017**, arXiv:1704.05742.
22. Rei, M. Semi-supervised multitask learning for sequence labeling. *arXiv* **2017**, arXiv:1704.07156.
23. Aguilar, G.; Maharjan, S.; López-Monroy, A.P.; Solorio, T. A multi-task approach for named entity recognition in social media data. *arXiv* **2019**, arXiv:1906.04135.
24. Cao, L.; Li, L.; Zheng, J.; Fan, X.; Yin, F.; Shen, H.; Zhang, J. Multi-task neural networks for joint hippocampus segmentation and clinical score regression. *Multimed. Tools Appl.* **2018**, *77*, 29669–29686. [CrossRef]
25. Clark, K.; Luong, M.T.; Manning, C.D.; Le, Q. Semi-Supervised Sequence Modeling with Cross-View Training. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, 31 October–4 November 2018; pp. 1914–1925.
26. Crichton, G.; Pyysalo, S.; Chiu, B.; Korhonen, A. A neural network multi-task learning approach to biomedical named entity recognition. *BMC Bioinform.* **2017**, *18*, 368. [CrossRef]
27. Wang, X.; Zhang, Y.; Ren, X.; Zhang, Y.; Zitnik, M.; Shang, J.; Langlotz, C.; Han, J. Cross-type biomedical named entity recognition with deep multi-task learning. *Bioinformatics* **2019**, *35*, 1745–1752. [CrossRef]
28. Liu, Y.; Liu, K.; Xu, L.; Zhao, J. Exploring fine-grained entity type constraints for distantly supervised relation extraction. In Proceedings of the Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, Dublin, Ireland, 23–29 August 2014; pp. 2107–2116.
29. Jiang, J. Multi-Task Transfer Learning for Weakly-Supervised Relation Extraction. In Proceedings of the 47th Annual Meeting of the Association for Computational Linguistics and the 4th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing (ACL-IJCNLP'09), Suntec, Singapore, 2–7 August 2009; Volume 1012.
30. Wu, Y.; Jiang, L.; Yang, Y. Switchable Novel Object Captioner. *IEEE Trans. Pattern Anal. Mach. Intell.* **2022**. [CrossRef]
31. Li, J.; Liu, R.; Chen, C.; Zhou, S.; Shang, X.; Wang, Y. An RG-FLAT-CRF Model for Named Entity Recognition of Chinese Electronic Clinical Records. *Electronics* **2022**, *11*, 1282. [CrossRef]
32. Wang, X.; Wang, Z.; Han, X.; Liu, Z.; Li, J.; Li, P.; Sun, M.; Zhou, J.; Ren, X. HMEAE: Hierarchical modular event argument extraction. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), Hong Kong, China, 3–7 November 2019; pp. 5781–5787.
33. Qian, T.; Zhang, M.; Lou, Y.; Hua, D. A joint model for named entity recognition with sentence-level entity type Attentions. *IEEE/ACM Trans. Audio Speech Lang. Process.* **2021**, *29*, 1438–1448. [CrossRef]
34. Pradhan, S.; Moschitti, A.; Xue, N.; Ng, H.T.; Björkelund, A.; Uryupina, O.; Zhang, Y.; Zhong, Z. Towards robust linguistic analysis using ontonotes. In Proceedings of the Seventeenth Conference on Computational Natural Language Learning, Sofia, Bulgaria, 8–9 August 2013; pp. 143–152.
35. Pradhan, S.; Moschitti, A.; Xue, N.; Uryupina, O.; Zhang, Y. CoNLL-2012 shared task: Modeling multilingual unrestricted coreference in OntoNotes. In Proceedings of the Joint Conference on EMNLP and CoNLL-Shared Task. Association for Computational Linguistics, Jeju, Korea, 13 July 2012; pp. 1–40.
36. Tjong Kim Sang, E.F. Introduction to the CoNLL-2002 Shared Task: Language-Independent Named Entity Recognition. In Proceedings of the Proceedings of CoNLL-2002, Taipei, Taiwan, 31 August–1 September 2002; pp. 155–158.
37. Levow, G.A. The third international Chinese language processing bakeoff: Word segmentation and named entity recognition. In Proceedings of the Fifth SIGHAN Workshop on Chinese Language Processing, Sydney, Australia, 22–23 July 2006; pp. 108–117.
38. Kim, J.D.; Ohta, T.; Tsuruoka, Y.; Tateisi, Y.; Collier, N. Introduction to the bio-entity recognition task at JNLPBA. In Proceedings of the In bioNLP, Geneva, Switzerland, 28–29 August 2004; pp. 70–75.
39. Ratinov, L.; Roth, D. Design challenges and misconceptions in named entity recognition. In Proceedings of the CoNLL-2009, Boulder, Colorado, 4 June 2009; pp. 147–155.
40. Pennington, J.; Socher, R.; Manning, C.D. Glove: Global vectors for word representation. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Doha, Qatar, 25–29 October 2014; pp. 1532–1543.
41. Yamada, I.; Asai, A.; Sakuma, J.; Shindo, H.; Takeda, H.; Takefuji, Y.; Matsumoto, Y. Wikipedia2Vec: An Efficient Toolkit for Learning and Visualizing the Embeddings of Words and Entities from Wikipedia. *arXiv* **2020**, arXiv:1812.06280v3.
42. Chiu, B.; Crichton, G.; Korhonen, A.; Pyysalo, S. How to train good word embeddings for biomedical NLP. In Proceedings of the 15th Workshop on Biomedical Natural Language Processing, Berlin, Germany, 12 August 2016; pp. 166–174.

43. Cui, Y.; Che, W.; Liu, T.; Qin, B.; Wang, S.; Hu, G. Revisiting Pre-Trained Models for Chinese Natural Language Processing. In Proceedings of the Findings of EMNLP, Online, 16–20 November 2020.
44. Lee, J.; Yoon, W.; Kim, S.; Kim, D.; Kim, S.; So, C.H.; Kang, J. BioBERT: A pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics* **2020**, *36*, 1234–1240. [CrossRef] [PubMed]
45. Chiu, J.P.; Nichols, E. Named entity recognition with bidirectional LSTM-CNNs. *Trans. Assoc. Comput. Linguist.* **2016**, *4*, 357–370. [CrossRef]
46. Strubell, E.; Verga, P.; Belanger, D.; McCallum, A. Fast and accurate entity recognition with iterated dilated convolutions. *arXiv* **2017**, arXiv:1702.02098.
47. Shen, Y.; Yun, H.; Lipton, Z.C.; Kronrod, Y.; Anandkumar, A. Deep active learning for named entity recognition. *arXiv* **2017**, arXiv:1707.05928.
48. Chen, H.; Lin, Z.; Ding, G.; Lou, J.; Zhang, Y.; Karlsson, B. GRN: Gated relation network to enhance convolutional neural network for named entity recognition. In Proceedings of the AAAI Conference on Artificial Intelligence, Honolulu, HI, USA, 27 January–1 February 2019; Volume 33, pp. 6236–6243.
49. Li, X.; Feng, J.; Meng, Y.; Han, Q.; Wu, F.; Li, J. A Unified MRC Framework for Named Entity Recognition. *arXiv* **2019**, arXiv:1910.11476.
50. Yadav, V.; Sharp, R.; Bethard, S. Deep affix features improve neural named entity recognizers. In Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics, New Orleans, LA, USA, 5–6 June 2018; pp. 167–172.
51. Wu, S.; Dredze, M. Beto, bentz, becas: The surprising cross-lingual effectiveness of bert. *arXiv* **2019**, arXiv:1904.09077.
52. Dong, C.; Zhang, J.; Zong, C.; Hattori, M.; Di, H. Character-based LSTM-CRF with radical-level features for Chinese named entity recognition. In *Natural Language Understanding and Intelligent Applications*; Springer: Berlin, Germany, 2016; pp. 239–250.
53. Zhang, Y.; Yang, J. Chinese NER Using Lattice LSTM. In Proceedings of the ACL, Melbourne, Australia, 15–20 July 2018; pp. 1554–1564.
54. Rei, M.; Crichton, G.; Pyysalo, S. Attending to Characters in Neural Sequence Labeling Models. In Proceedings of the Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers, Osaka, Japan, 11–16 December 2016; pp. 309–318.
55. Ju, M.; Miwa, M.; Ananiadou, S. A neural layered model for nested named entity recognition. In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers), New Orleans, LA, USA, 1–6 June 2018; pp. 1446–1459.
56. Nayel, H.A. Integrating Dictionary Feature into A Deep Learning Model for Disease Named Entity Recognition. *arXiv* **2019**, arXiv:1911.01600.
57. Zheng, C.; Cai, Y.; Xu, J.; Leung, H.f.; Xu, G. A Boundary-aware Neural Model for Nested Named Entity Recognition. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), Hong Kong, China, 3–7 November 2019; pp. 357–366.
58. Dai, X.; Karimi, S.; Hachey, B.; Paris, C. Using Similarity Measures to Select Pretraining Data for NER. *arXiv* **2019**, arXiv:1904.00585.