Balancing Precision and Recall for Neural Biomedical Event Extraction

Fangfang Su, Yue Zhang, *Member, IEEE*, Fei Li, and Donghong Ji

***Abstract*—Biomedical event extraction is an essential task in the biomedical research. Existing models suffer from the issue of low recall due to the large proportion of unrecognized events and inflexible event argument combination. To address this issue, we propose an end-to-end multi-task approach for biomedical event extraction. Our model is able to achieve balanced precision and recall with several nichetargeting designs. First, neural encoders with rich lexical and syntactic features are used and shared by multiple subtasks such as event trigger recognition and argument relation extraction, in order to enhance the generalizability of the model. Second, a novel auxiliary subtask is added to identify the proteins that participate in the events, which helps decreasing the challenge of mining event-related proteins from the large candidate space. Third, event argument combination is performed using a strong neural network rather than inflexible rules or templates, to further increase the recall, especially for complex nested events. To demonstrate the effectiveness of our model, we evaluate it on two widely-used biomedical event extraction datasets used in the BioNLP 2011 and 2013 shared tasks. Our model achieves the state-of-the-art results (63.15% and 55.67% in F1 score) by significantly improving the recalls (compared with *DeepEvnetMine*$_{SciBERT}$, 4.65% and 5.0%) on the two datasets. Further experiments and analyses show the effectiveness of our proposed features and modules in the model.**

***Index Terms*—Argument combination, biomedical event extraction, dependency graph, graph convolutional network, multi-task approach.**

I. Introduction

BIOMEDICAL event extraction is the task of capturing important information from the unstructured biomedical text and presenting them in the form of logic and structure. It can help researchers gain a quick understanding of literature knowledge and facilitate computational biomedical research. While the task is essential and valuable, it is challenging to extract event information from biomedical text accurately.

Manuscript received September 8, 2021; revised December 16, 2021 and March 7, 2022; accepted March 7, 2022. Date of publication March 22, 2022; date of current version May 12, 2022. This work was supported in part by the National Natural Science Foundation of China under Grant 62176187, in part by the National Key Research and Development Program of China under Grant 2017YFC1200500, and in part by the Research Foundation of Ministry of Education of China under Grant 18JZD015. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Jing Huang. *(Corresponding author: Donghong Ji.)*

Fangfang Su, Fei Li, and Donghong Ji are with the Key Laboratory of Aerospace Information Security and Trusted Computing, Ministry of Education, School of Cyber Science and Engineering, Wuhan University, Wuhan, Hubei 430072, China (e-mail: fangsu@whu.edu.cn; foxlf823@wias.org.cn; dhji@whu.edu.cn).

Yue Zhang is with the Department of Engineering, Westlake University, Hangzhou 310024, China (e-mail: yue.zhang@wias.org.cn).

Digital Object Identifier 10.1109/TASLP.2022.3161146

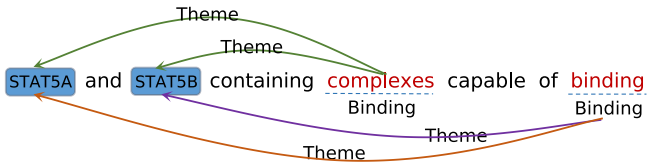
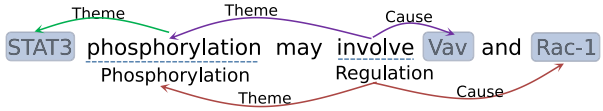


Fig. 1. A biomedical event example in BioNLP 2011.



(a) trigger recognition and argument assignment

Event ID	Event type	Trigger	Theme	Cause
E1	Phosphorylation	phosphorylation	STAT3	---
E2	Regulation	involve	E1	Vav
E3	Regulation	involve	E1	Rac-1

(b) event construction

Fig. 2. A biomedical event extraction example of nested structure.

As shown in Fig. 1, biomedical event extraction aims to discover event triggers with specific types and a set of their arguments in a given text. The whole task can be broken into several components, such as trigger recognition (underlined red font), relation extraction (the *Theme* role between the trigger "complexes" and the candidate argument "STAT5A"), and argument combination ("STAT5A" and "STAT5B" are arguments of the triggers "complexes" and "binding"; these two arguments and the trigger "complex" form a complete event, yet form two different events with the trigger "binding").

One challenge of this task is that an event could serve as an argument of another event, leading to a nested structure. In addition, a trigger with different arguments can form multiple events in a sentence. Fig. 2 shows an example of a nested event structure in the biomedical domain. In Fig. 2(a), the sentence is marked with three events: a *Phosphorylation* event and two *Regulation* events.

The entities *STAT3*, *Vav*, and *Rac-1* (in boxes) are proteins, and the words above dashed lines (i.e., *phosphorylation* and *involve*) are triggers. The words below dashed lines (i.e., *Phosphorylation* and *Regulation*) denote the types of triggers. An arc shows that one entity or event is an argument of another entity. The arc points from the trigger to the argument, and the arc label is the role of the argument. Fig. 2(b) gives the types and arguments of the three events in the sentence of Fig. 2(a). We can see that

2329-9290 © 2022 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See https://www.ieee.org/publications/rights/index.html for more information.

Authorized licensed use limited to: Westlake University. Downloaded on December 29,2022 at 04:55:22 UTC from IEEE Xplore. Restrictions apply.

E1 is a simple event, which is one argument of *E2* and *E3*, thus forming two nested events. In addition, the trigger, *involve*, generates two different events *E2* and *E3*. These examples show that nested events and overlapping events drastically increase the complexity of biomedical event extraction.

Earlier studies on biomedical event extraction are feature-based [1]–[4], adopting traditional methods such as support vector machine (SVM [5]). With the development of deep learning, biomedical event extraction has acquired considerable progress [6]–[9] by leveraging continuous representations of neural networks. However, these methods also show weaknesses. In particular, previous research results show that they obtained higher precision but lower recall (e.g., precision is 71.73% and recall is 53.21%). For a specific type of event, recall reflects the number of true examples found by the model compared with the total number of existing positive examples in corpora. Thus, recall is an essential evaluation criterion for event extraction. Existing methods give relatively low recall values for trigger recognition and relation extraction, thus suffering from significant proportions of undetected events. In addition, for the argument combination task, most previous research uses hand-crafted sentence templates for constructing events [3] or a traditional binary classification that relies on a manually designed set of features [4], where even joint methods [2], [10] also deal with the argument combination module separately. This leads to low performance, especially low recall for *Binding* events and *Complex* events.

To address the above issues, we aim to build a neural model that gives a stronger balance between precision and recall so that more useful events can be detected for downstream tasks. To this end, we consider an end-to-end neural model that includes the three sub-tasks. We consider three key factors for improving recall. First, neural encoders with multiple feature sources (e.g., syntax, sequence, and pre-training) are shared across tasks so that the chance of overfitting to certain task-specific spurious patterns on training data can be reduced [11]–[15]. Second, intermediate steps are added to shortlist structural candidates for further screening so that the challenge of identifying certain proteins from a large candidate space is reduced, and the learning challenge is reduced as well. Third, neural models are used instead of rules or templates to avoid extracting only a small fraction of specific targets, while omitting other instances. Fig. 3 shows the flow chart of our method involving trigger recognition, relation extraction, and argument combination. As shown in Fig. 3, for trigger recognition, we introduce an auxiliary task to identify the proteins that participate in the events as a preprocessing step, which we call **jprot**. This task can not only give valuable features for the subsequent trigger recognition task, but also reduce the difficulty of trigger recognition by reducing the candidate space. We adopt the widely-used GCN for information extraction task. Although the overall performance is not high after applying it to biomedical event extraction, this approach can balance precision and recall. Last but not least, a neural network method is designed for argument combination, which is jointly trained with trigger recognition and relation extraction. This gives our model more flexibility compared to post-processing rules [4], [16]–[18].

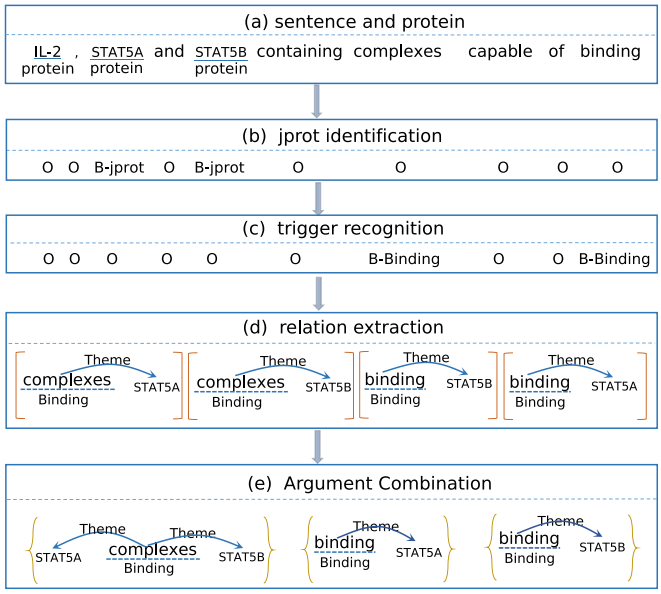


Fig. 3. Flow chart of biomedical event extraction.

We evaluate our model on the BioNLP 2011 (GE11), and BioNLP 2013 (GE13) shared task. Experimental results show the effectiveness of the neural method for improving recall. Our final model achieves the state-of-the-art on both datasets, achieving 63.15% and 55.67% F1 scores on the GE11 and GE13 test datasets, respectively, and obtaining up to 3.48% and 3.95% absolute gains in terms of recall compared with previous state-of-the-art methods on the GE11 and the GE13 test data, respectively. In addition, *jprot* gives much improvement for trigger recognition. By sharing two encoders with relation extraction, argument combination significantly improves *Binding* and *Complex* events. Finally, compared with most existing work [3], [8], [19], our end-to-end model for biomedical event extraction removes the cost for manual efforts. We release our code at *https://github.com/appleml/event_extraction*.

II. Task Definition

The task of biomedical event extraction was proposed in 2009 [20]. As shown in Fig. 3, a biomedical event has a structural representation, usually composed of a trigger with one or more arguments. The argument can be a protein or other events. Necessary subtasks for biomedical event extraction include protein identification involved in events, trigger recognition, relation extraction, and argument combination. Our method of biomedical event extraction consists of four stages. The overall process of biomedical event extraction for an example sentence is illustrated in Fig. 3

A. Jprot Identification

The proteins annotated in advance can be divided into two categories according to whether they participate in the events or not. We call the proteins involved in events **jprot**. As shown in Fig. 3(b), *IL-2*, *STAT5A*, and *STAT5B* are proteins, and *IL-2* is not a *jprot* because it is not involved in any events. We

Authorized licensed use limited to: Westlake University. Downloaded on December 29,2022 at 04:55:22 UTC from IEEE Xplore. Restrictions apply.

introduce *jport* identification as a separate sequence labeling task following named entity recognition [21], [22]. Considering that there are a large number of multi-word proteins, we apply the BIO scheme, where B, I and O indicate that a token is the *Beginning*(B), *Inside*(I), and *Outside*(O) of an entity mention, respectively. We apply the BiLSTM-CRF framework, effective in sequence labeling, to recognize *jprot*.

B. Trigger Recognition

Trigger identification is a primary task of biomedical event extraction. The types of triggers can be roughly divided into three categories: *Simple* type, *Binding* type, and *Complex* type. For the *Simple* type, a trigger has only one *Theme* argument, which must be a protein. The *Binding* triggers have one or more protein arguments. The *Complex* triggers have one or two arguments, which can be proteins or other events. The relation between a trigger and an argument can be *Theme* or *Cause*, which are also called **roles**.

Note that we ignore some special trigger cases (e.g., for a trigger with multiple types or the case of overlapping) but only keep simple types. As shown in Fig. 3(c), we recast trigger recognition as sequence labeling and adopt BiLSTM-CRF with different features compared to *jprot* identification.

C. Relation Extraction

Relation extraction aims to extract the relationship (a.k.a. role) between triggers and candidate arguments. There are only two role types in this task: *Theme* and *Cause*. The *Simple* and *Binding* triggers can have one or two *Theme* arguments, which are proteins. The *Complex* events have one obligatory *Theme* and one optional *Cause*, each of which can be either a protein or another event, thus resulting in nested events. We simplify nested events by judging the relationship between triggers and the triggers of argument events.

We combine syntactic information and sequence information for relation extraction. As shown in Fig. 3(d), a pair of orange brackets indicates the relation between a trigger and an argument that has been identified.

D. Argument Combination

Since triggers may be involved in more than one events with shared arguments, after relation extraction, we need further to generate events from the results of relation extraction. To this end, most existing researches manually define templates and rules [3], [16], [17] or use manual features with SVM to judge whether two arguments belong to the same event [4], [18]. In this paper, we use the neural network method to generate events. *Binding* and *Complex* events, which contain three complex types, namely *Regulation*, *Positive_Regulation*, and *Negative_Regulation*, account for a more significant proportion in the training set. The ability to effectively combine arguments has an enormous impact on the performance of the final result. This subtask can be solved as a binary classification problem. As shown in Fig. 3(e), yellow braces indicate the events extracted from the sentence.

E. Formal Definition

Given a sentence $x = [x_1, x_2, \ldots, x_n]$ of n tokens, where x_i denotes the i-th token in the sequence, we denote the protein set for the sentence x as Σ.

Jprot Identification: Jprot identification outputs are token-level labels $P = [p_1, p_2, ..., p_n]$, defined by a label set with the standard BIO labeling scheme [23], [24], namely $p_i \in [\textit{B-jprot}, \textit{I-jprot}, O]$.

Trigger Recognition: We use $\mathcal{E} \cup O$ to denote the trigger label alphabet, where $\mathcal{E}$ represents trigger types, and O indicates that the token is not a trigger. For trigger recognition, this subtask identifies the tag of each token with BIO tag scheme. We use $\mathcal{T} = [t_1, t_2, ..., t_m]$ $(m \ll n)$ to represent a trigger set for sentence x.

Relation Extraction: Let $\mathcal{R} \cup \varphi$ denote a set of pre-defined relation types describing the semantic relations among triggers and arguments. The task is to predict a relation r for every pair of trigger t_i and argument a_j $(t_i \in \mathcal{T}, a_j \in (\mathcal{T} \cup \Sigma), r(t_i, a_j) \in (\mathcal{R} \cup \varphi))$.

Argument Combination: Here a_i, a_j are two arguments $(a_i, a_j \in (\mathcal{T} \cup \Sigma))$. $r(a_i, a_j) = 1$ if and only if the two arguments a_i and a_j are in the same event, and 0 otherwise.

III. Our Approach

The overview of our model is given in Fig. 4, which is a multi-task approach for biomedical event extraction based on pre-trained language models. We describe our sequence labeling model in Section III-A, the relation extraction model in Section III-B, and the primary objective function for biomedical event extraction in Section III-C.

A. Entity Recognition Model

As mentioned earlier, we regard *jprot* identification and trigger recognition as sequence labeling tasks and employ BiLSTM-CRF models to solve them.

Feature Embedding: We adopt a pre-trained language model (e.g., SciBERT) to obtain contextualized representations emb_i^{word} for each input token x_i, which is tuned during training. On this base model, we introduce three different types of additional features, including character-level information, POS tags, and entity labels as common features for entity recognition. We use emb_i^{char}, emb_i^{pos} and emb_i^{prot} represent their embeddings, respectively. The character embedding emb_i^{char} is learned following prior work [22] by using convolutional neural networks [25] to encode character-level information of a word. The embeddings emb_i^{pos} and emb_i^{prot} are initialized randomly. We concatenate these embedding for the i-th word to obtain:

$$emb_i^{common} = [emb_i^{word}; emb_i^{pos}; emb_i^{char}; emb_i^{prot}]$$

In addition to the common features given above, we introduce a set of new features for *jprot* identification. If one protein is the argument of a trigger in the previous sentences of the same document (that is, it has been involved in events in piror sentence), the protein may also participate in other events in this sentence. These proteins are called **cjprot**. We check whether each protein in the current sentence participates in the events

Authorized licensed use limited to: Westlake University. Downloaded on December 29,2022 at 04:55:22 UTC from IEEE Xplore. Restrictions apply.

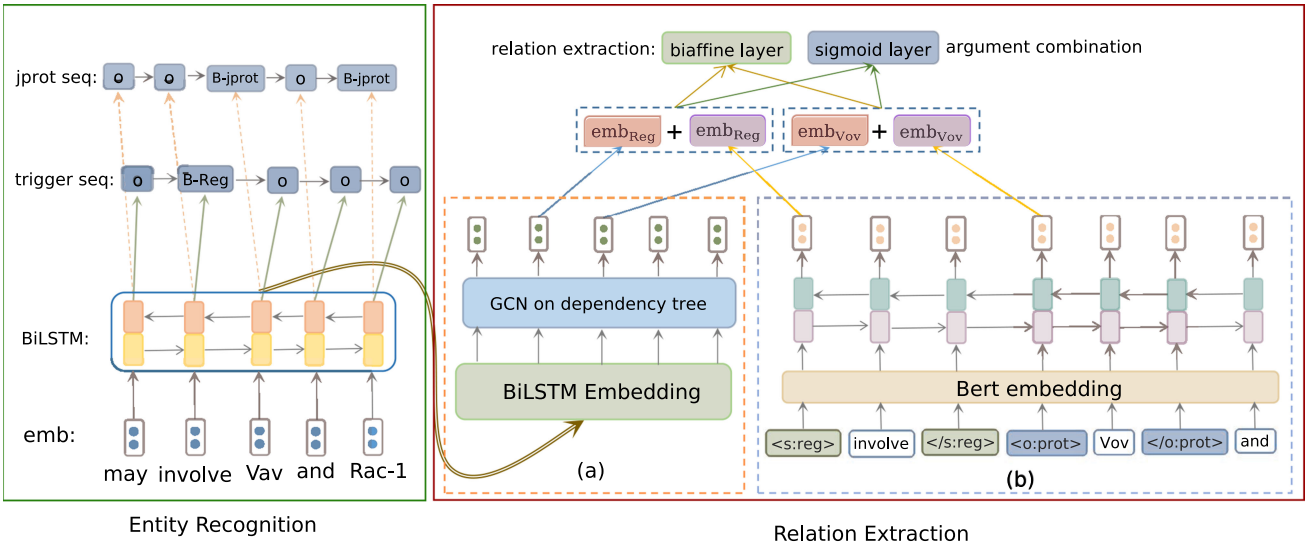


Fig. 4. Our end-to-end neural model architecture.

of previous sentence, and use such information as features to determine whether the protein is *jprot*.

$$emb_i^{jprot} = [emb_i^{common}; emb_i^{cjprot}]$$

These embeddings are concatenated as input for BiLSTM. The output hidden states $h = [h_1, h_2, ..., h_n]$ is further fed into the CRF layer to obtain the target sequence.

For trigger recognition, one unique feature is *jprot*, since in most cases, protein and trigger words are very close in the text. Besides, words that are the triggers in previous sentences of the same document may also be the triggers in the current sentence. We call them **ctrig**. These two extra features can boost the recall of biomedical event extraction. The final input embedding is:

$$emb_i^{trig} = [emb_i^{common}; emb_i^{jprot}; emb_i^{ctrig}]$$

BiLSTM Layer: We use one-layer bi-directional LSTMs [26] for sequence tagging. The forward computes a representation $\overrightarrow{h_i}$ of the left context of the sentence at the i-th word, and the backward LSTM reads the same sequence in reverse, generating a representation of the right context $\overleftarrow{h_i}$ at each step. The hidden state h at the position i of BiLSTM can be expressed as follows:

$$\overrightarrow{h_i} = LSTM_f(x_i, \overrightarrow{h}_{i-1}; \overrightarrow{\theta})$$
$$\overleftarrow{h_i} = LSTM_b(x_i, \overleftarrow{h}_{i+1}; \overleftarrow{\theta})$$

where $\overrightarrow{\theta}$ and $\overleftarrow{\theta}$ are trainable parameters. We concatenate the hidden state vectors of the two directions' LSTM units corresponding to each word as its output vector, $h_i = [\overrightarrow{h_i}; \overleftarrow{h_i}]$. The output of BiLSTM $h = [h_1, h_2, ..., h_n]$ is further fed into the subsequent layer to find the target sequence.

CRF Layer: CRF [27] has been widely applied in sequence tagging [22], [28], [29], which explicitly models label dependencies by adding transition scores between neighboring labels and optimizing the model at the sequence level. Training and decoding can be solved efficiently by adopting the Viterbi algorithm [30]. The score of a named entity sequence for a given word sequence is defined as:

$$s(x, y) = \sum_{i=0}^{n-1} T_{y_{i-1}, y_i} + \sum_{i=1}^{n} P_{i, y_i} \quad (1)$$

where T_{y_{i-1}, y_i} represents the transmission score between the current tag y_i at time i and its predecessor y_{i-1}, P_{i,y_i} is the emission score of the current tag y_i on the word i-th word from BiLSTM encoder.

CRF defines a family of conditional probability $p(y|x)$ over all possible tag sequences y given x. We define the probability of the tag sequence as:

$$p(y|x) = \frac{exp(s(x, y))}{\sum_{\tilde{y} \in y} exp(s(x, \tilde{y}))} \quad (2)$$

During training, we maximize the log probability of the correct predictions. The objective of decoding is to find a tag sequence that maximizes the score for a given word sequence and model parameters:

$$y^* = \arg\max_{\hat{y} \in y} score(x, \hat{y})$$

B. Relation Extraction Model

As shown in Fig. 4, we integrate two neural encoders for relation extraction. The first is a graph convolutional network (GCN) [31] for modeling syntactic dependency structures. The second is built on top of deep pre-trained language models [32], following the work of [33] for introducing additional knowledge.

Graph Convolutional Networks: A graph convolutional network (GCN) [31] operates on graph-structured data. For every node in the graph (in our case, a word in a sentence), GCN encodes relevant information over its neighborhood as a real-valued feature vector. One layer GCN encodes information about immediate neighbors, and K layers are used to encode K-order neighborhoods.

Authorized licensed use limited to: Westlake University. Downloaded on December 29,2022 at 04:55:22 UTC from IEEE Xplore. Restrictions apply.

Formally, consider an undirected graph $G = (V, E)$, where $V(|V| = n)$ and E are sets of nodes and edges, respectively, and n is the length of sentence in our case. We can represent the graph structure with an $n \times n$ adjacency matrix A, where $A_{ij} = 1$ if there is an edge going from node i to node j (the opposite direction of a dependency arc is also included, $A_{ji} = 1$). In addition, we add a self-loop for each node in the graph, where $A_{ii} = 1$. The convolution computation for node i at the l-th layer takes the input feature representation $h^{(l-1)}$ as input and outputs the induced representation $h_i^{(l)}$:

$$h_i^{(l)} = \sigma\left(\sum_{j \in \mathcal{N}(i)} A_{ij} W^{(l)} h_j^{(l-1)} + b^{(l)}\right)$$

where $W^{(l)}$ is the weight matrix which is randomly initialized and optimized during training, $b^{(l)}$ is the bias vector, and $\mathcal{N}(i)$ is the set of one-hop neighbors of node i, Here $i \in \mathcal{N}(i)$ (because of self-loops) and σ is an activation function (e.g., RELU). $h_i^{(0)}$ is the output of BiLSTM for word x_i.

We additionally introduce the dependency type as a feature. The final entity representation with syntactic information is given by:

$$h_i^{(l)} = \sigma\left(\sum_{j \in \mathcal{N}(i)} A_{ij} W^{(l)} h_j^{(l-1)} + A_{ij} \mathcal{L}(L_{ij}) b^{(l)}\right)$$

where $\mathcal{L}$ is a linear mapping, L_{ij} is the dependency type embedding of the arc label of dependency edge between node i and node j. Similarly, self-loops have label *self*, and the type is *other* if there is no edge between i and j.

Through GCN module, we obtain the sequence encode $h^{(l)} = [h_1^{(l)}, h_2^{(l)}, \ldots, h_n^{(l)}]$, the span of the two entities have been defined above, namely $t = [t_1, \ldots, t_p](1 < t_1 < t_q < n)$ and $a = [a_1, \ldots, a_q](1 < a_1 < a_q < n)$. We apply average pooling over the final hidden $h^{(l)}$ by using t and a to get the embedding of trigger and candidate argument as follows:

$$e_t = avgpool[h_{t_1}^{(l)}, \ldots, h_{t_p}^{(l)}]$$
$$e_a = avgpool[h_{a_1}^{(l)}, \ldots, h_{a_q}^{(l)}]$$

Previous studies [21] show that the types of entities are beneficial for relation extraction. We concatenate these entity vectors and type vectors to derive the final representation:

$$\overline{h}_t = [e_t; e_t^{type}]$$
$$\overline{h}_a = [e_a; e_t^{type}]$$

where ";" is the concatenation operation, e_t^{type} and e_a^{type} are randomly initialized, which are tuned during training.

Sequence Encoder Model: Following the work on relation extraction [33], [34], we process each pair of subject-object spans e_i and e_j independently given an input sentence x, where t_{e_i}, t_{e_j} $(t_{e_i}, t_{e_j} \in \mathcal{E} \cup \phi)$ are types of subject-object spans respectively. We define text markers as $\langle s : t_{type} \rangle$, $\langle /s : t_{type} \rangle$, $\langle o : a_{type} \rangle$, and $\langle /o : a_{type} \rangle$ by using s and o to differentiate between the entities. Existing studies [35] also show that entity types are beneficial for relation extraction, and thus we add the types of two entities to the markers. Then, we augment sentence x with four text markers to denote the beginning and end of each entity mention to highlight two entities (Fig. 4(b)). Finally, this modified sequence $\widetilde{x}$ is computed as:

$$\widetilde{x} = \cdots, \langle s : t_{type} \rangle, x_{start(t)}, \ldots, x_{end(t)}, \langle /s : t_{type} \rangle,$$
$$\cdots, \langle o : a_{type} \rangle, x_{start(a)}, \ldots, x_{end(a)}, \langle /o : a_{type} \rangle$$

For a given sentence $\widetilde{x}$, we feed its token sequence into SciBERT [32] to calculate the word embedding. These embeddings are fed to a BiLSTM network to calculate hidden representations for words. The start markers vector $\tilde{h}_t$ and $\tilde{h}_a$ of the two entities $\langle s : t_{type} \rangle$ and $\langle o : a_{type} \rangle$ are extracted from the output $\tilde{h} = [\tilde{h}_1, \tilde{h}_2, \ldots, \tilde{h}_n]$ of BiLSTM according to the location of the start markers.

Biaffine Layer: The GCN encoder gives trigger and argument representations $\overline{h}_t$, $\overline{h}_a$. From the sequence encoder, we obtain the corresponding entities representations $\tilde{h}_t$ and $\tilde{h}_a$. The final representation h_t, h_a are obtained by adding them, namely:

$$h_t = \overline{h}_t + \widetilde{h}_t$$
$$h_a = \overline{h}_a + \widetilde{h}_a$$

Next, we use the biaffine classification [36] to determine the relation between the trigger and argument:

$$r = h_t U^{(1)} h_a + (h_t \oplus h_a)^{\mathsf{T}} U^{(2)} + b$$

where r is the relation between two entities, $U^{(1)}$ and $U^{(2)}$ are both weight matrices, which are randomly initialized and tuned during training, b is bias, and $\oplus$ is concatenation operation.

Sigmoid Layer: Now we have a GCN to encode syntactic dependency information and a sequence encoder with text markers to encode the positions of the entities in the sentence. We combine the information learned from the two encoders to represent arguments. The vectors of the first and second arguments are denoted as h_t and h_a:

Then to determine the propability whether two arguments belong to the same event, With the addition of the hidden states of two arguments, (i.e., $r = h_t + h_a$), we use a sigmoid layer with $r = h_t + h_a$ as features to obtain the probability P $(0 < P < 1)$. When the probability exceeds a certain threshold ρ (we set $\rho = 0.5$), we consider that two arguments belong to the same event, such process can be formalized as:

$$P = Sigmoid(r)$$

C. Training Objective

We update all parameters using Adam [37] with gradient clipping and L2-regularization, fine-tuning the pre-trained language models using task-specific losses. We also apply dropout [38] to the embedding layer and the hidden layers for entity recognition and relation extraction.

Entity: Let $\theta_e = (W_e, b_e)$ denote the set of model parameters associated with the entity recognition model, where W_e is the set of the weight matrices in the BiLSTM and the CRF layer, and b_e is the set of the bias vectors. To train the NER model, we minimize the negative log-likelihood of the correct label

Authorized licensed use limited to: Westlake University. Downloaded on December 29,2022 at 04:55:22 UTC from IEEE Xplore. Restrictions apply.

TABLE I
STATISTICS OF THE GENIA 2011 AND GENIA 2013 DATASET

Item	GE11			GE13		
	Train	Dev	Test	Train	Dev	Test
Documents	908	259	347	222	249	305
Sentences	8,664	2,888	3,363	6363	6462	6559
Tokens	230,737	74,334	90,091	54,938	57,907	75,144
Entities	11,625	4,690	5,301	3,571	4,138	4,359
Events	10,310	3,250	4,487	2,817	3,199	3,348

sequences over the training set. The objective for θ_e is defined as follows:

$$\mathcal{L}_e = -\frac{1}{N}\sum_{i=1}^{N} log(p(y_i|x_i, \theta_e))$$

where N is the number of training examples, x_i is input sequence and y_i is the ground truth tag sequence.

Relation: We denote $\theta_r = (W_r, b_r)$ as parameters for relation extraction, where W_r and b_r are the weight and bias parameters. We minimize a multi-class cross-entropy loss:

$$\mathcal{L}_r = -\frac{1}{N}\sum_{i=1}^{N}\sum_{(e_1,e_2)\in S_i} logP(r_{(e_1,e_2)}|(e_1, e_2), \theta_r)$$

where N is the number of training examples, and i denotes the sentence index. Here S_i is the entity pair set in i-th sentence, $r_{(e_1,e_2)}$ represents the gold relation type of span pair e_1, e_2 in the training data. For training the relation model, we only consider the gold entities S_r in the training set and use the gold entity labels as the input of the relation model.

Finally, The overall objective $\mathcal{J}(\theta)$ is defined as:

$$\mathcal{J}(\theta) = \mathcal{L}_e + \mathcal{L}_r$$

IV. EXPERIMENTS

A. Datasets and Evaluation Metric

We evaluate our model on the BioNLP 2011 and BioNLP 2013 benchmark datasets for biomedical event extraction: the Genia event task 2011 (GE11) [39] and the Genia event task 2013 (GE13) [40]. These tasks are toward fine-grained information extraction in the biomedical domain. These corpora comprise both abstracts and full-texts, and consist of the annotations for *protein* entities and 9 fine-grained event types. The detailed statistics of GE11 and GE13 are summarized in Table I.

Our model is validated on the development set and tested on the test set. All experiment results are measured using the primary micro *precision* (P), *recall* (R), and *F1 score* (F1) on biomedical event extraction, and we give the results of biomedical event extraction on the nine event types with online evaluation[1] using the test set. In line with previous work, we evaluate our approach according to the approximate span and recursive matching criteria [7], [41].

[1][Online]. Available: http://bionlp-st.dbcls.jp/GE/2011/eval-test/

B. Implementation Details

Each document is first processed by the sentence splitter[2] of Stanford CoreNLP. The GENIA tagger[3] is used to obtain fine-grained words and part-of-speech tags, chunk tags, and named-entity tags, which are specifically tuned for biomedical text such as MEDLINE abstracts. We convert a sentence into the corresponding dependency tree using a biaffine parser [36], which is trained on the Genia Corpora.[4]

We train the model end-to-end on a Transformer framework [42], adopting SciBERT [32] as the textual encoder, which has been pre-trained on large-scale scientific data. The entire framework is optimized using Adam [37]. The learning rate is tuned in $1e$-5 for SciBERT parameters and $1e$-4 for the model parameters, and the gradient clipping is set as 5. The max training epoch is set as 80.

We initialize all model parameters by Kaiming initialization [43], and randomly initialize *POS*-embedding and *NER*-embedding with 30 dimensions. We set *prot*-embedding, *jprot*-embedding and *cjprot*-embedding to be 2-dimensional respectively, *ctirg* are 10-dimensional. We choose two layer GCNs, and the GCN hidden size is set to 200 dimensions, where dependency type embedding is set to 30 dimensions. Dropout is employed to the input layer for jprot and trigger recognition, relation extraction and argument combination. All the above hyperparameters are tuned on the development dataset. The optimal hyperparameter settings are tuned by using grid search conducted on GE11 development set.

C. Developmental Results

We report some developmental experiments to determine the most critical factors of our final model. All the experiments are performed on the GE11 development set.

1) Impact of Jprot and Ctrig Features: We select fifteen different random seeds to do several rounds of experiments on the trigger recognition subtask to verify the effectiveness of *jprot* and *ctrig* features. Fig. 5 compares the averages and standard deviations of F1 scores for trigger recognition with regard to training epochs on the GE11 development dataset. Overall, the average F1 score using these two features is better than that without these two features. In addition, the standard deviation using these two features decreases as training epochs increase, while the standard deviation without these two features does not seem to change much. The observation indicates that the triggers are generally near the proteins and these two features are helpful and competently for trigger recognition.

2) Impact of Dependency Types: We examine the influence of the dependency types for event extraction. Fig. 6 shows the precision, recall, and F1 score curves of the GCN encoder with dependency types and without dependency types. First, the F1 scores are best using two layers of GCN with and without dependency types. We thus set the layers of the GCN encoder to 2. Second, dependency types effectively improve the F1 score

[2][Online]. Available: https://stanfordnlp.github.io/CoreNLP/ssplit.html
[3][Online]. Available: http://www.nactem.ac.uk /GENIA/tagger/
[4][Online]. Available: http://www.geniaproject.org/

Authorized licensed use limited to: Westlake University. Downloaded on December 29,2022 at 04:55:22 UTC from IEEE Xplore. Restrictions apply.

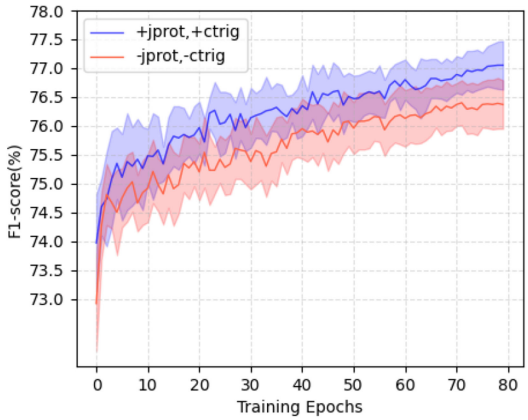


Fig. 5. The performance comparison of trigger recognition with or without *jprot* and *ctrig* features.

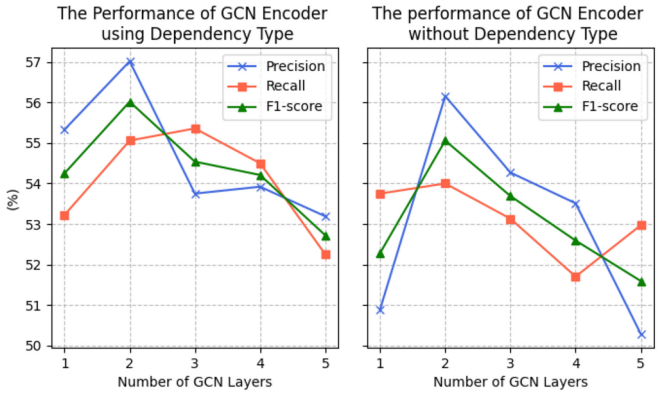


Fig. 6. The performance comparison of biomedical event extraction using dependency type information or not.

TABLE II
EVENT STATISTICS OF DIFFERENT ARGUMENT NUMBERS FOR THE BINDING AND COMPLEX EVENTS

Type	Train		Development	
	one argument	two arguments	one argument	two arguments
Binding	639	349	172	203
Regulation	739	374	158	138
Positive_Regulation	2,375	1,010	572	428
Negative_Regulation	1,024	267	321	150

for event extraction and narrow the precision and recall gap. The observation indicates that dependency types are essential for event extraction.

3) Impact of Argument Combination: Table II gives the statistics of the *Binding* and *Complex* types with one argument and two arguments in the corpora, which shows the necessity of argument combination. Table III compares the performance of the argument combination of pipeline training and joint training. *Argument combination*$_{pipeline}$ is the model where trigger recognition and relation extraction are jointly trained, but argument combination is processed separately. For this model, argument combination uses the same encoders as relation extraction but does not share with relation extraction. *Argument combination*$_{joint}$ refers to the joint training of three subtasks. Table III shows that the joint strategy is better than the pipeline for the Binding and Complex event types. *Argument combination*$_{joint}$ corrects the errors in the first two stages, showing the enormous advantage in finding events with two arguments for *Binding* and *Complex* event types.

TABLE III
EXPERIMENT RESULT OF JOINT LEARNING AND PIPELINE LEARNING FOR ARGUMENT COMBINATION

Type	argument combination$_{pipeline}$			argument combination$_{joint}$		
	P	R	F1	P	R	F1
Bind	$55.37_{\pm 1.9}$	$55.43_{\pm 2.7}$	$55.40_{\pm 2.2}$	$56.59_{\pm 2.3}$	$60.42_{\pm 1.1}$	$58.41_{\pm 1.3}$
Regu	$57.73_{\pm 2.5}$	$41.89_{\pm 3.0}$	$48.55_{\pm 1.7}$	$56.32_{\pm 3.2}$	$43.8_{\pm 2.3}$	$49.15_{\pm 1.0}$
Posi	$54.46_{\pm 0.5}$	$47.31_{\pm 1.8}$	$50.63_{\pm 0.8}$	$56.17_{\pm 2.6}$	$48.6_{\pm 0.5}$	$52.08_{\pm 1.1}$
Nega	$59.38_{\pm 3.1}$	$49.27_{\pm 2.6}$	$53.85_{\pm 2.9}$	$62.94_{\pm 3.3}$	$53.0_{\pm 1.5}$	$57.52_{\pm 2.0}$

· Bind refers to Binding;
· Regu refers to Regulation
· Posi refers to Positive_Regulation
· Nega refers to Negative_Regulation

In addition, as shown in Table III, we observe that compared with the performance of *argument combination*$_{joint}$, the F1 score of *argument combination*$_{pipeline}$ for the *Complex* events decreases slightly, while the performance of the *Binding* events decreases heavily. The reasons behind this may be (1) Events with two arguments account for 29.43% of the total number of the *Binding* and *Complex* events, which are not sufficient to train the neural network model with large parameters. (2) A particularity of the *Complex* events is that the roles of the two arguments are different. For *Complex* events, there are some hard constraints when generating events, that is, one *Theme* argument and one *Cause* argument form an event. By contrast the two arguments of *Binding* can be both the *Theme* roles, so the combination of two arguments can be more arbitrary. This leads to low performance in the *Binding* events. Given the above observations, we take the joint model design.

D. Main Results

We compare our method with the following SOTA models on the BioNLP 2011 shared task test sets.

1) **QA with BERT** adopts the multi-turn question answering approach introducing the notion of a question template that defines different types of questions.
2) **GEANet-SciBERT** proposes graph edge conditioned attention networks to integrate domain knowledge from a hierarchical knowledge base.
3) **HANN** presents stacked hypergraph aggregation neural network layers, in which a two-level model is used to model the interaction between local and global contexts for biomedical documents.
4) **DeepEventMine** is a representative neural model extracting nested events from a sentence in an end-to-end manner.

Authorized licensed use limited to: Westlake University. Downloaded on December 29,2022 at 04:55:22 UTC from IEEE Xplore. Restrictions apply.

TABLE IV
EXPERIMENT RESULTS ON THE GENIA 2011 AND GENIA 2013 TEST SET

Dataset	Works	Methods	P	R	F1
Genia 2011	Riedel et al. (2011)	FAUST-Model Combination	64.75	49.41	56.04
	Miwa et al. (2012)	SVM pipeline(+coref)	63.48	53.35	57.98
	Venugopal et al. (2014)	SVM pipeline & MLN(joint)	63.61	53.42	58.07
	Majumder et al. (2016)	Stacked generalization	66.46	48.96	56.38
	Björne and Salakoski (2018)	TEES-CNN	69.45	49.94	58.10
	Li et al. (2019) $*$	KB-driven Tree-LSTM pipeline	67.01	52.14	58.65
	Wang et al. (2020)	QA with *BERT*	59.33	57.37	58.33
	Huang et al.(2020) $*$	GEANet$_{SciBERT}$	64.61	56.11	60.06
	Ramponi et al. (2020)	Multi-task BEESL$_{BioBERT}$	69.72	53.00	60.22
	Trieu et al. (2020)	DeepEventMine$_{SciBERT}$	71.71	56.20	63.02
	Zhao et al. (2021)	HANN(K=2)$_{GloVe}$	**71.73**	53.21	61.10
	Our methods$_{Glove}$	-	$64.37_{\pm 0.9}$	$57.33_{\pm 1.4}$	$60.64_{\pm 2.0}$
	Our methods$_{BioBERT}$	-	$66.78_{\pm 2.2}$	$58.69_{\pm 1.9}$	$62.47_{\pm 1.3}$
	Our methods$_{SciBERT}$	-	$65.66_{\pm 1.8}$	$\mathbf{60.85_{\pm 0.6}}$	$\mathbf{63.15_{\pm 0.8}}$
Genia 2013	Bui et al. (2013)	RulesAndPatterns	62.83	42.47	50.68
	Hakalal et al. (2013)	EVEX	58.03	45.44	50.97
	Venugopal et al. (2014)	SVM pipeline & MLN(joint)	59.24	48.95	53.61
	Björne et al. (2015)	TEES	56.32	46.17	50.74
	Rao et al. (2017)	DistantSupervisionBasedAMR	46.60	46.73	46.66
	Björne and Salakoski (2018)	TEES-CNN	**65.78**	44.38	53.00
	Ma and Lu et al. (2020)	ErrorDetectionLearning	63.20	47.83	54.45
	Trieu et al. (2020)	DeepEventMine$_{SciBERT}$	60.98	49.80	54.83
	Our method$_{Glove}$	-	$54.17_{\pm 2.4}$	$52.98_{\pm 1.8}$	$53.57_{\pm 2.0}$
	Our method$_{BioBERT}$	-	$56.31_{\pm 1.7}$	$\mathbf{54.80_{\pm 0.9}}$	$55.54_{\pm 1.1}$
	Our method$_{SciBERT}$	-	$56.58_{\pm 1.0}$	$54.80_{\pm 0.8}$	$\mathbf{55.67}_{\pm 0.7}$

"*" means that original paper leverage the external knowledge base.

We also compare with high-performance methods on the BioNLP 2013 shared task:

5) **SVM pipeline & MLN(joint)** combines the power of Markov logic networks and support vector machines and attempts to overcome error propagation using many high-dimensional sophisticated features.
6) **ErrorDetectionLearning** gives an error detection model based on representation for unlabeled data, where selected samples are added to enrich the training dataset and improve the classification performance.
7) **TEES-CNN** is a pipeline method with four classification stages implemented as multiclass classification tasks using a neural network with various vector space embeddings as input features.

As shown in Table IV, *SVM pipeline & MLN(joint)* [44] achieves the highest F1 score with a 58.07% and a 53.61% on the BioNLP 2011 and 2013 shared tasks and gains the best recall among all traditional methods. Compared with the above model, our approach significantly advances event performances, i.e., an absolute F1 score of +5.03% and +2.81% on the BioNLP 2011 and 2013 shared task, respectively.

With the application of neural network to this task, for instance, *TEES-CNN* [6] of early neural network work attains 58.10% and 53.00% F1 scores on the BioNLP 2011 and 2013 shared tasks. Compared with traditional methods, the neural network models improve the precision, but the recall decreases significantly, and the overall performance does not increase greatly. Compared to this baseline, our model gains 5% and 3.42% improvement on F1 on both datasets.

After introducing external resources based on neural networks, for instance, *GEANET*$_{SciBERT}$ [16] attains a 64.61% precision, a 56.11% recall, and a 60.06% F1 score on the BioNLP 2011 shared tasks. The recall and overall performance of this model improve significantly. It shows the effectiveness of external resources for the task. In contrast, without using external knowledge, our approach achieves a 2.84% better precision, a 3.17% better recall, and a 3.04% better F1 score than *GEANET*$_{SciBERT}$, respectively.

DeepEventMine$_{SciBERT}$ is the state-of-the-art model on the BioNLP 2011 and 2013 shared tasks. It achieves the highest F1 scores of 63.02% and 54.83% on the BioNLP 2011 and 2013 shared tasks. More specifically, it attains a 71.71% precision and a 56.20% recall on the BioNLP 2011 shared task. However, the precision and recall gap grows. Compared with this leading method, our approach achieves 0.13% and 0.84% better F1 scores, respectively. It is important to note that our model gains 3.48% and 3.95% improvement on the event recall. The precision and recall gap is 4.81% and 2.83% on the BioNLP 2011 and 2013 shared tasks, much lower than *DeepEventMine*$_{SciBERT}$'s 15.51% and 11.18%.

Authorized licensed use limited to: Westlake University. Downloaded on December 29,2022 at 04:55:22 UTC from IEEE Xplore. Restrictions apply.

TABLE V
PRECISION, RECALL AND THEIR GAP COMPARISION ON GE11 TEST SET ABOUT NIGH FINE-GRAINED EVENT TYPES

Types	Stacked Generalization				KBTL				BEESL				Our methods			
	P	R	F1	P-R	P	R	F1	P-R	P	R	F1	P-R	P	R	F1	P-R
Gene_expression	84.55	72.65	78.15	**11.90**	87.24	74.35	80.28	**12.89**	84.55	77.54	80.90	**7.01**	$81.07_{\pm 1.9}$	$82.46_{\pm 1.0}$	$81.74_{\pm 0.7}$	**-1.39**$_{\pm 2.6}$
Transcription	70.73	50.00	58.59	**20.23**	82.31	69.54	75.39	**12.77**	72.50	66.67	69.46	**5.53**	$72.22_{\pm 3.4}$	$67.73_{\pm 4.1}$	$69.90_{\pm 2.3}$	$4.49_{\pm 6.0}$
Protein_catabolism	77.78	46.67	58.33	**31.11**	87.50	46.67	60.87	**40.83**	83.33	66.67	74.07	**16.66**	$82.18_{\pm 9.5}$	$77.73_{\pm 7.1}$	$79.63_{\pm 7.2}$	**4.45**$_{\pm 9.0}$
Phosphorylation	83.70	83.24	83.47	**0.46**	87.28	81.62	84.36	**5.66**	94.05	85.41	89.52	**8.64**	$89.94_{\pm 1.1}$	$87.75_{\pm 2.7}$	$88.83_{\pm 1.7}$	$2.19_{\pm 3.0}$
Localization	85.34	51.83	64.50	**33.51**	80.28	59.69	68.47	**20.59**	83.21	59.69	69.51	**23.52**	$86.39_{\pm 2.3}$	$74.19_{\pm 4.9}$	$79.7_{\pm 2.6}$	**12.2**$_{\pm 6.1}$
Simple events	83.14	68.60	75.17	**14.54**	85.95	72.62	78.73	**13.33**	84.17	74.98	79.31	**9.19**	$80.15_{\pm 2.3}$	$77.33_{\pm 3.4}$	$78.71_{\pm 2.7}$	**2.82**$_{\pm 5.1}$
Binding event	57.35	47.66	52.06	**9.69**	53.16	37.68	44.10	**15.48**	65.36	40.73	50.19	**24.63**	$56.59_{\pm 2.3}$	$60.42_{\pm 1.1}$	$58.41_{\pm 1.3}$	**-3.84**$_{\pm 2.6}$
Regulation	54.30	31.17	39.60	**23.31**	53.61	36.62	43.52	**16.99**	62.22	36.36	45.90	**25.86**	$56.32_{\pm 3.2}$	$43.8_{\pm 2.3}$	$49.15_{\pm 1.0}$	**12.52**$_{\pm 5.2}$
Positive_regulation	56.02	38.05	45.32	**17.97**	57.90	41.37	48.26	**16.53**	60.14	41.93	49.41	**18.21**	$56.17_{\pm 2.6}$	$48.60_{\pm 0.5}$	$52.08_{\pm 1.1}$	**7.57**$_{\pm 2.8}$
Negative_regulation	53.54	35.73	42.86	**17.81**	52.39	46.06	49.02	**6.33**	53.19	42.38	47.17	**10.81**	$62.94_{\pm 3.3}$	$53.0_{\pm 1.5}$	$57.52_{\pm 2.0}$	**9.94**$_{\pm 2.7}$
Complex events	55.18	36.39	43.86	**18.79**	55.73	41.73	47.72	**14.00**	58.54	41.14	48.32	**17.40**	$58.13_{\pm 3.0}$	$47.40_{\pm 1.7}$	$52.22_{\pm 2.1}$	**10.73**$_{\pm 2.5}$
ALL-TOTAL	66.46	48.96	56.38	**17.5**	67.01	52.14	58.65	**14.87**	69.72	53.00	60.22	**16.72**	$65.66_{\pm 1.8}$	$60.85_{\pm 0.6}$	$63.15_{\pm 0.8}$	**4.81**$_{\pm 2.1}$

As shown in the Table IV, the model using *Glove* vectors achieves 60.64% and 53.57% F1 scores on the BioNLP 2011 and 2013 shared tasks, while the F1 scores of our approach using SciBERT are 2.51% and 2.1% better than those of the Glovebased model. In addition, the result using *SciBERT* [32] outperforms *BioBERT* [45], demonstrating the effectiveness of *SciBERT* on biomedical event extraction. Our model consistently outperforms the state-of-the-art models on the BioNLP 2011 and BioNLP 2013 shared tasks, achieving the best performance and yielding a new state-of-the-art with an F1 score of 63.15% and 55.67%, respectively.

E. Results on the Recall

Table V compares the precision and recall gap of our method and the previous best models.

1) **Stacked Generalization** proposes a stack model using SVM-based models with token and sentence-level handcrafted features.
2) **KB-driven Tree-LSTM** adopts an external knowledge base with type and sentence embeddings into a Tree-LSTM model.
3) **BEESL** recasts the task as sequence labeling and uses a multi-label aware decoder with BERT in a multi-task sequence labeling model.

Our method significantly reduces the precision-and-recall gap among the nine fine-grained biomedical event types on the BioNLP 2011 shared task compared to the three approaches mentioned above. From a coarse-grained perspective, for *Simple events*, the precision-and-recall gaps of *Stacked Generalization*, *KBTL*, and *BEESL* are 14.54%, 13.33%, and 9.19%, respectively. In contrast, our model can strike a balance between precision and recall, with a gap of only 2.82%. For *Binding event*, *BEESL* (using the neural network) has an enormous gap of 24.63%. *KBTL* (with external knowledge) has a gap of 15.48%. In contrast, our model gives relatively little difference between precision and recall. For *Complex events*, the gaps for three baselines are 18.79%, 14.00%, and 17.40%, respectively. Our approach vastly reduces the precision-and-recall gap to 10.73%.

TABLE VI
RESULT OF NINE EVENT TYPES ON THE TEST DATA OF THE BIONLP 2013 SHARED TASK

Types	P	R	F1
Gene_expression	$76.75_{\pm 1.5}$	$82.42_{\pm 1.3}$	$79.46_{\pm 0.5}$
Transcription	$62.95_{\pm 4.3}$	$75.84_{\pm 0.5}$	$68.72_{\pm 2.5}$
Protein_catabolism	$44.97_{\pm 4.7}$	$90.0_{\pm 7.3}$	$59.87_{\pm 5.4}$
Localization	$76.2_{\pm 2.9}$	$48.89_{\pm 3.5}$	$59.47_{\pm 2.9}$
SIMPLE ALL	$73.71_{\pm 1.1}$	$77.77_{\pm 1.2}$	$75.67_{\pm 0.5}$
Binding	$37.11_{\pm 2.5}$	$53.87_{\pm 1.8}$	$43.92_{\pm 2.2}$
Phosphorylation	$80.1_{\pm 2.1}$	$89.75_{\pm 2.3}$	$84.62_{\pm 1.5}$
PROT MOD ALL	$80.1_{\pm 2.1}$	$75.18_{\pm 1.9}$	$77.54_{\pm 1.4}$
Regulation	$32.79_{\pm 2.9}$	$32.22_{\pm 1.9}$	$32.46_{\pm 2.1}$
Positive_regulation	$53.29_{\pm 3.9}$	$42.92_{\pm 0.9}$	$47.46_{\pm 1.3}$
Negative_regulation	$57.16_{\pm 1.5}$	$49.47_{\pm 1.4}$	$53.02_{\pm 1.3}$
REG-TOTAL	$50.75_{\pm 2.4}$	$43.11_{\pm 0.8}$	$46.6_{\pm 1.3}$
ALL-TOTAL	$56.58_{\pm 1.0}$	$54.80_{\pm 0.8}$	$55.67_{\pm 0.7}$

For the *Stacked Generalization* model, six out of the nine event types have precision and recall gap of more than 15%. *KBTL* and *BEESL* have five event types. In our result, in contrast, no type exceeds 15%, and only two types of events exceed 10%. As shown in Table IV, our method also outperforms other approaches by a large margin on *recall* (i.e., an absolute improvement of +3.48% over the *QA with Bert* model [17] and +4.65% in comparison with *DeepEventMine* [41] for biomedical event extraction on GE11 task). In addition, as shown in Table VI, the results of our model in the BioNLP 2013 shared task also verify that our model can effectively improve the recall.

As shown in Table V, the results of four models, including our approach, show that the recall of *Localization* among the *Simple* events is much lower than precision on the GE11 test set. The same phenomenon emerges on the GE13 test set from Table VI. The reasons behind this may be: first, the number of *Localization* event type labeled in the GE11 training set is too small and only has 281 events. However, *Phosphorylation* type has 3385 events. Furthermore, of 32% triggers marked as *Localization* type are also triggers of other event types. For example, the word

Authorized licensed use limited to: Westlake University. Downloaded on December 29,2022 at 04:55:22 UTC from IEEE Xplore. Restrictions apply.

TABLE VII
ABLATION EXPERIMENTS ON THE GE11 TEST DATA

Method	GE11		
	P	R	F1
BiLSTM-CRF$_{-jprot,-ctrig}$	**68.25**	56.92	62.07
BiLSTM-CRF$_{+jprot,-ctrig}$	66.80	58.02	62.10
BiLSTM-CRF$_{-jprot,+ctrig}$	64.72	60.30	62.44
BiLSTM-CRF$_{+jprot,+ctrig}$	65.66	**60.86**	**63.15**
Only GCN Enc$_{-dep\ type}$	52.75	55.89	54.28
Only GCN Enc$_{+dep\ type}$	55.79	56.45	56.12
Only Seq Enc$_{-bilstm}$	63.25	59.97	61.57
Only Seq Enc$_{+bilstm}$	65.43	59.17	62.14
GCN Enc$_{-dep\ type}$ + Seq Enc$_{-bilstm}$	66.85	56.02	61.07
GCN Enc$_{+dep\ type}$ + Seq Enc$_{-bilstm}$	67.25	58.38	62.50
GCN Enc$_{-dep\ type}$ + Seq Enc$_{+bilstm}$	**68.68**	56.63	62.08
GCN Enc$_{+dep\ type}$ + Seq Enc$_{+bilstm}$	65.66	**60.86**	**63.15**

"expressed" is a *Localization* event trigger, yet in most cases, this word is labeled as a *Gene_expression* type in the GE11 training set or labeled as a *Phosphorylation* type in rare cases. Finally, we analyze the experimental results of trigger recognition and relation extraction on the 2011 shared task development set. For trigger recognition, the recall is 10.24% lower than the precision for *Localization* and 8.33% for the relation extraction. Few examples of *Localization* types and enormous annotation ambiguity are the main reasons for the poor performance.

F. Ablation Study

We conduct ablation experiments to demonstrate the effectiveness of each component in our framework. The results for the verifing the effectiveness of each feature type and learning module are shown in Tables IV and VII. From the Table VII, we can see that the two features, *jprot* and *ctrig* contribute more significant improvements on recall, gaining an absolute improvement of +1.08% F1 score over the model. After introducing dependency types information, the GCN encoder has achieved 3.93% improvements on recall. Sequence encoder adding the BiLSTM model attains the best precision of 68.68%. Combining syntactic information and sequence information gives a new state-of-the-art with an F1 score of 63.15%.

1) Effect of Jprotand CtrigFeatures: As shown in the first four rows of Table VII, without the *jprot* and *ctrig* features, our approach obtains a 62.07% F1 score but only a 56.92% recall on the GE11 test set. The recall is largely improved by using independent *jprot* or *ctrig* features, and the precision-and-recall gap is reduced. We obtain the state-of-the-art F1 score on the GE11 and GE13 test sets by using these two features simultaneously. This is because, in most cases, the triggers are not too far away from their arguments, and therefore, using the recognized *jprot* as a feature can be helpful for the trigger model to identify the triggers effectively. The previous and subsequent sentences generally describe similar events. Thus the triggers in the previous sentence may still be triggers in the current sentence, which can improve the triggers' recall, thus is subsequently helpful to the promotion of the event recall.

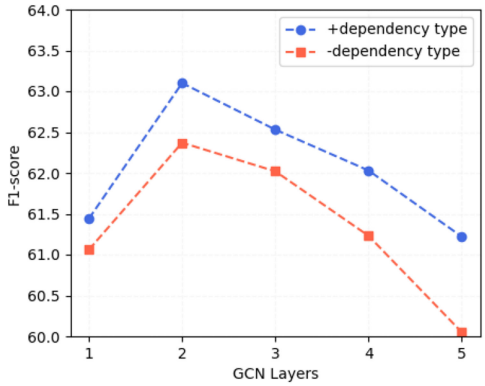


Fig. 7. Effect of GCN layers with or without dependency types.

2) Effect of Dependency Type Information: As shown in the fifth and sixth rows in Table VII, when the GCN encoder encodes only unlabeled trees, the event extraction F1 is only 54.28%. When we further add dependency types into the model, the precision and recall are balanced, and further improved. Finally, F1 score is increased from 54.28% to 56.12%. It shows that the dependency type is helpful for biomedical event extraction. Fig. 7 shows when using 2-layer GCN, the F1 scores are the best with dependency type or without on the GE11 test set. With increasing numbers of GCN layers, the performance decreases seriously. This can likely be that the model is overfitted. Furthermore, we can see that the performance of the GCN encoder with dependency type is better than that without dependency type.

c) Effect of the Sequence Encoder: Compared with the GCN encoder, the sequence encoder is more effective in biomedical event extraction. As shown in the seventh and eighth rows of Table VII, by inserting markers with entity types before and after entities, our model achieves a 61.57% F1 score. The recall of biomedical event extraction reaches 59.97%, which is the highest reported value. In summary, in addition to the two features, *jprot* and *ctrig*, another reason for the high recall of our model comes from the sequence encoder. After further adding a BiLSTM layer, there can be a slight improvement in the F1 score. It can be seen that the precision can be significantly improved, but it does not affect on the recall.

To highlight the importance of entity labels, we remove the GCN encoder (i.e., Fig. 4(a) of relation extraction). We replace two pairs markers $\langle S:type\rangle$, $\langle /S:type\rangle$ and $\langle O:type\rangle$, $\langle /O:type\rangle$ with $\langle S\rangle$, $\langle /S\rangle$ and $\langle O\rangle$, $\langle /O\rangle$ to check whether the entity type is essential. As shown in Table IX, both precision and recall decrease significantly, which shows that the entity type is vital for relation extraction and argument combination.

4) Effect of Combines Encoders: As shown in the bottom of Table VII, using either dependency types or BiLSTM alone, the results have around 1% improvement in F1 score compared with the model without them (i.e., the 9th row in Table VII). Consistent with the previous results, using BiLSTM alone does not improve the recall, but using dependency types can significantly improve the recall. When using the dependency types in the GCN encoder and BiLSTM layer in the sequence encoder simultaneously, the recall increased from 56.02% to 60.85%,

Authorized licensed use limited to: Westlake University. Downloaded on December 29,2022 at 04:55:22 UTC from IEEE Xplore. Restrictions apply.

TABLE VIII
PERFORMANCE COMPARISION OF THE BINDING AND COMPLEX EVENTS ON THE GE11 TEST SET

Method	Binding			Complex		
	P	R	F1	P	R	F1
Stacked Generalization [19]	57.35	47.66	52.06	55.18	36.39	43.86
TEES-CNN [6]	-	-	-	59.54	39.02	47.14
Tree-LSTM [46]	53.16	37.68	44.10	55.73	41.73	47.72
EDPR [47]	52.48	51.73	52.10	48.16	42.56	45.19
BEESL [7]	**65.36**	40.73	50.19	58.54	41.14	48.32
DeepEventMine [41]	-	-	-	**67.87**	45.35	**54.37**
Our method	56.59	**60.42**	**58.41**	58.13	**47.40**	52.22

"-" means that original paper do not give.

TABLE IX
SECOND ENCODER PERFORMANCE OF EVENT EXTRACTION ON THE GE11 TEST SET

Method	P	R	F1
Sequence Encoder$_{-type}$	62.65	54.61	58.36
Sequence Encoder$_{+type}$	65.43	59.17	62.14

leading to a significant F1 score improvement of 2.08%. It shows that the GCN encoder based on the dependency parsing tree can capture semantics between two entities far apart in the sentence, making up for the shortcomings of the sequence encoder. We combine syntactic information and sequence information for relation extraction, and argument combination is valid.

5) Effect of the Argument Combination: : To understand the effectiveness of the neural *argument combination* module, we present a detailed analysis from a coarse-grained perspective. Table VIII compares the F1 scores of our method to the previous best models on the *Binding* and *Complex* categories. Our method achieves the best performance on the *Binding* type and a comparable result on the *Complex* type on the GE11 test set. The result shows that the neural *argument combination* module benefits the four particular event types, which may have one or two arguments and occupy a large proportion of the dataset. Compared with manually defined features and sentence pattern templates [16], [17] as well as traditional binary classifiers [4], [18], our neural *argument combination* module is highly effective.

As shown in Table VIII, For *Binding* type, we achieve a 58.41% F1 score which is a 6.31% higher than **EDPR**. For*Complex* type, we obtain a 52.22% F1 score, lower than **DeepEventMine** (with an F1 score of 54.37%). The reason is that the latter focuses on nested events and builds a representative model that captures nested event structures. The emphasis of our model is solving argument combination after relation extraction. It is also worth noting that our approach obtains the highest recall than other methods in both the *Binding* type and the *Complex* type, achieving 8.69% and 2.05% recall improvements compared to the **EDPR** and **DeepEventMine**.

6) Cause Analysis of Low Performance of Complex Events: For the *Complex* events, complicating the situation, nested events and multi-level nesting are covering up to 37.17% of the total number of events, which is also the principal reason for the low F1 score of *Complex* event extraction. There are 4156 *Theme* role types and 1493 *Cause* role types in complex events. If an argument is both the *Theme* role of one event and the *Cause* role of another event, the *Cause* role is often incorrectly identified in the relation extraction process. This is also one of the reasons for low recall of *Complex* events because *Complex* events with two arguments are difficult to find. We also evaluate the extraction effect of the *Theme* argument and the *Cause* argument on the GE11 development set. As shown in Table X, it can be seen that the recognition effect of the *Cause* argument is much lower than that of the *Theme* argument. The experimental results also verify the above analysis.

TABLE X
TRIGGER AND ARGUMENT EXTRACTION PERFORMANCE OF COMPLEX EVENTS ON THE GE11 DEVELOPMENT SET

Type	Trigger Recognition			Role	Relation Extraction		
	P	R	F1		P	R	F1
Regu	63.95	62.27	63.09	Cause	69.31	69.07	69.19
				Theme	79.54	83.95	81.69
Posi	65.32	64.78	65.04	Cause	65.23	52.59	58.23
				Theme	80.83	88.27	84.39
Nega	72.69	68.18	70.37	Cause	64.20	50.81	56.73
				Theme	78.28	81.09	79.66

· Regu refers to Regulation
· Posi refers to Positive_regulation
· Neag refers to Negative_regulation

TABLE XI
CASE STUDY AND ERROR ANALYSIS

S1: thrombin and TRP enhanced CD69 expression and interleukin 2 production induced
e1: {enhanced : Positive_regulation, Theme : induced, Cause:thrombin }
e2: {enhanced : Positive_regulation, Theme : expression }
S2: Levels of bcl-2 mRNA and · · ·
e1: {Levels : Transcription, Theme : bcl-2}
e2: {Levels : Gene_expression, Theme : bcl-2}
S3: that binding of Sp1 and AP-2 to the LAL promoter
e1: {binding : Binding, Theme:Sp1 Theme2:LAL }
e2: {binding : Binding, Theme:LAL }

G. Case Study

We give three concrete examples in Table XI, highlighting the proteins and triggers in red and blue, respectively. Each brace pair represents a gold event. Sentence S1 contains two *Positive_regulation* events triggered by "enhanced". The event triggered by "induced" and the event triggered by "expression" are *Theme* argument of the trigger "enhance," the protein "thrombin" is a *Cause* argument of the trigger "enhanced". Previous methods adopt the *Theme* and *Cause* combination strategy for *Complex* events, and the events they identified are as follows:

1) {enhanced:Positive_regulation, Theme:induced, Cause: thrombin}
2) {enhanced:Positive_regulation, Theme: expression, Cause: thrombin}

However, the *Cause* argument "thrombin" and the *Theme* argument "expression" with the trigger "enhanced" should not form an event. In contrast, our model employs a neural argument combination strategy to judge whether two arguments are in the

Authorized licensed use limited to: Westlake University. Downloaded on December 29,2022 at 04:55:22 UTC from IEEE Xplore. Restrictions apply.

same event (for instance, the event triggered by"expression" as *Theme* role and "thrombin" as a *Cause* role of "enhanced" do not show in the same event). Our model can recognize such patterns, showing that the model can well capture this type of correlation among prediction results.

As a qualitative error analysis, the second example S2 shows a trigger recognition problem. In S2, the word "Level" is a trigger, which triggers two different event types. One event is a *Transcription* event and the other is a *Gene_expression* event. Our model adopts sequence labeling to simultaneously recognize triggers, failing to recognize the two types of one entity. One possible solution is to exhaustively consider all possible spans of a sentence with sizes less than or equal to the maximum span size and perform multi-label classification for each span.

The existing models cannot effectively solve the problem given in S3, the word "binding" generates two *Binding* events, and event **e2** is part of event **e1**. The two proteins "Sp1" and "LAL" appear in the same event through argument combination, but we cannot confirm which protein participates in another independent event with the trigger "binding," or the two proteins form two complete events or neither protein forms an independent event with the trigger "binding". A practical method is to integrate relation extraction and argument combination, generating the events in one step after trigger recognition.

V. Related Work

Previous work has heavily relied on manually designed features and domain expertise knowledge. For instance, Majumder *et al.* [19] propose a SVM-based stack model to process multi-subtask with hand-crafted lexical and syntactic features. More recently, with the rapid development of neural network methods, Li *et al.* [48] explore a dynamic extended tree containing two entities using the LSTM networks framework. Björne *et al.* [6] develop a convolutional neural network and use incoming and outgoing dependency vectors as input features to get better results. Zhu *et al.* [8] use a hybrid deep neural network to detect triggers, extract relations, and then construct the events by solving an optimization problem. To further enhance the neural network models for this task, extra knowledge has been proved effective [16], [46]. The experimental results show that these neural network models obtain superior performance compared to traditional shallow methods.

The aforementioned studies achieve good results, yet there are some defects: (1) Tradition methods heavily rely on manually designed features. (2) Many models focus on trigger recognition and relation extraction, ignoring argument combination, which is also solved by sentence templates or traditional binary classification. Our model adopts a neural network model for biomedical event extraction compared to the aforementioned studies. The model containing a neural argument combination obtain excellent performance.

Current approaches for event extraction fall into two main categories: pipeline methods and joint methods. Pipeline methods break down a task into multitiple tasks but ignore their interaction. Miwa. *et al.* [49] use a pipeline-based SVM model to extract events with the shortest paths features. Lately, most current works focus on joint methods to solve this problem. Liu *et al.* [18] propose a pairwise model that transforms event extraction into a simple multi-class problem of classifying pairs of text entities. Riedel *et al.* [2] apply a joint stacked model for biomedical event extraction, based on a discriminatively trained model that jointly predicts trigger labels, event arguments, and protein pairs in the *Binding* types. Vlachos *et al.* [3] decompose event extraction into a set of classification tasks that can be learned jointly using the search-based structured prediction framework. Ramponi *et al.* [7] introduce a joint end-to-end neural information extraction model, which recasts biomedical event extraction as sequence labeling, taking advantage of a multi-label aware encoding strategy and jointly modeling the intermediate tasks via multi-task learning.

Despite the success of existing methods, they generally suffer from some limitations: (1) Pipeline methods simplify the problem but ignore the interaction between the sub-tasks and make the process prone to accumulating errors. (2) The joint methods focus on predicting the triggers and arguments simultaneously but deal with argument combination separately. (3) Some models take advantage of syntactic information to extract relation yet abandon dependency type information, which contains rich clues for event extraction. Compared with the aforementioned methods, we adopt an end-to-end multi-task model to extract events, training trigger recognition, relation extraction, and argument combination jointly. In addition, we consider dependency types, which have not been exploited for the task, showing that they are helpful for biomedical event extraction.

VI. Conclusion

We proposed a neural biomedical event extraction model using the end-to-end multi-task learning method. We studied two features *jprot* and *ctrig*, which can not only effectively improve trigger recognition but also improve the overall result of biomedical event extraction, and in particular, the recall. We also combined the features of syntactic structure information and sequence information for relation extraction and argument combination via GCN and BERT-based encoders. Experiments on two public datasets demonstrate that our model outperforms previous competitive methods for biomedical event extraction. To our knowledge, we are the first to make a fully end-to-end neural model for complex event extraction, achieving the best recall over standard benchmarks.

References

[1] M. Miwa, S. Pyysalo, T. Hara, and J. Tsujii, "A comparative study of syntactic parsers for event extraction," in *Proc. Workshop Biomed. Natural Lang. Process.*, 2010, pp. 37–45.

[2] S. Riedel, D. McClosky, M. Surdeanu, A. McCallum, and C. D. Manning, "Model combination for event extraction in BioNLP 2011," in *Proc. BioNLP Shared Task Workshop*, 2011, pp. 51–55.

[3] A. Vlachos and M. Craven, "Biomedical event extraction from abstracts and full papers using search-based structured prediction," *BMC Bioinf.*, vol. 13, no. 11, pp. 1–11, 2012.

[4] D. McClosky, S. Riedel, M. Surdeanu, A. McCallum, and C. D. Manning, "Combining joint models for biomedical event extraction," *BMC Bioinf.*, vol. 13, no. 11, pp. 1–12, 2012.

Authorized licensed use limited to: Westlake University. Downloaded on December 29,2022 at 04:55:22 UTC from IEEE Xplore. Restrictions apply.

[5] K. P. Bennett and C. Campbell, "Support vector machines: Hype or hallelujah?," *ACM SIGKDD Explorations Newslett.*, vol. 2, no. 2, pp. 1–13, 2000.

[6] J. Björne and T. Salakoski, "Biomedical event extraction using convolutional neural networks and dependency parsing," in *Proc. BioNLP Workshop*, 2018, pp. 98–108.

[7] A. Ramponi, R. van der Goot, R. Lombardo, and B. Plank, "Biomedical event extraction as sequence labeling," in *Proc. Conf. Empir. Methods Natural Lang. Process.*, 2020, pp. 5357–5367.

[8] L. Zhu and H. Zheng, "Biomedical event extraction with a novel combination strategy based on hybrid deep neural networks," *BMC Bioinf.*, vol. 21, no. 1, 2020, Art. no. 47.

[9] K. Espinosa, M. Miwa, and S. Ananiadou, "A search-based neural model for biomedical nested and overlapping event detection," in *Proc. Conf. Empirical Methods in Nat. Lang. Process. 9th Int. Joint Conf. Natural Lang. Process. (EMNLP-IJCNLP*, Nov. 2019, pp. 3679–3686, *arXiv:1910.10281*.

[10] S. Riedel and A. McCallum, "Robust biomedical event extraction with dual decomposition and minimal domain adaptation," in *Proc. BioNLP Shared Task Workshop*, 2011, pp. 46–50.

[11] L. Yang, J. Li, P. Cunningham, Y. Zhang, B. Smyth, and R. Dong, "Exploring the efficacy of automatically generated counterfactuals for sentiment analysis," in *Proc. 59th Annu. Meeting Assoc. Comput. Linguistics 11th Int. Joint Conf. Natural Lang. Process.*, 2021, pp. 306–316.

[12] D. Kaushik, E. Hovy, and Z. C. Lipton, "Learning the difference that makes a difference with counterfactually-augmented data," in *Proc. Int. Conf. Learn. Representations*, 2020, *arXiv:1909.12434*.

[13] S. Gururangan, S. Swayamdipta, O. Levy, R. Schwartz, S. R. Bowman, and N. A. Smith, "Annotation artifacts in natural language inference data," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics, Hum. Lang. Technol.*, vol. 2, 2018, pp. 107–112.

[14] X. Bai, Y. Chen, L. Song, and Y. Zhang, "Semantic representation for dialogue modeling," in *Proc. 59th Annu. Meeting Assoc. Comput. Linguistics 11th Int. Joint Conf. Natural Lang. Process.*, 2021, pp. 4430–4445. [Online]. Available: https://aclanthology.org/2021.acl-long.342

[15] X. Bai, P. Liu, and Y. Zhang, "Investigating typed syntactic dependencies for targeted sentiment classification using graph attention neural network," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 29, pp. 503–514, 2021.

[16] K.-H. Huang, M. Yang, and N. Peng, "Biomedical event extraction with hierarchical knowledge graphs," in *Proc. Findings Assoc. Comput. Linguistics, EMNLP*, Nov. 2020, pp. 1277–1285.

[17] X. D. Wang, L. Weber, and U. Leser, "Biomedical event extraction as multi-turn question answering," in *Proc. 11th Int. Workshop Health Text Mining Inf. Anal.*, 2020, pp. 88–96.

[18] X. Liu, A. Bordes, and Y. Grandvalet, "Fast recursive multi-class classification of pairs of text entities for biomedical event extraction," in *Proc. 14th Conf. Eur. Chapter Assoc. Comput. Linguistics*, 2014, pp. 692–701.

[19] A. Majumder, A. Ekbal, and S. K. Naskar, "Biomolecular event extraction using a stacked generalization based classifier," in *Proc. 13th Int. Conf. Natural Lang. Process.*, 2016, pp. 55–64.

[20] J.-D. Kim, T. Ohta, S. Pyysalo, Y. Kano, and J. Tsujii, "Overview of BioNLP'09 shared task on event extraction," in *Proc. BioNLP Workshop Companion Volume Shared Task*, 2009, pp. 1–9.

[21] Z. Huang, W. Xu, and K. Yu, "Bidirectional LSTM-CRF models for sequence tagging," 2015, *arXiv:1508.01991*.

[22] X. Ma and E. Hovy, "End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF," in *Proc. 54th Annu. Meeting Assoc. Comput. Linguistics*, Aug. 2016, pp. 1064–1074.

[23] L. A. Ramshaw and M. P. Marcus, "Text chunking using transformation-based learning," in *Natural Language Processing Using Very Large Corpora*, Berlin, Germany: Springer, 1999, pp. 157–176.

[24] L. Ratinov and D. Roth, "Design challenges and misconceptions in named entity recognition," in *Proc. 13th Conf. Comput. Natural Lang. Learn.*, 2009, pp. 147–155.

[25] Y. LeCun *et al.*, "Backpropagation applied to handwritten zip code recognition," *Neural Comput.*, vol. 1, no. 4, pp. 541–551, 1989.

[26] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[27] J. Lafferty, A. McCallum, and F. C. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," in *Proc. 18th Int. Conf. Mach. Learn.*, 2001, pp. 282–289.

[28] G. Lample, M. Ballesteros, S. Subramanian, K. Kawakami, and C. Dyer, "Neural architectures for named entity recognition," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics, Hum. Lang. Technol.*, San Diego, California, Assoc. Comput. Linguistics, Jun. 2016, pp. 260–270, doi: 10.18653/v1/N16-1030.

[29] Z. Yang, R. Salakhutdinov, and W. W. Cohen, "Transfer learning for sequence tagging with hierarchical recurrent networks," in *Proc. Submitted Blog Track ICLR*, 2017, *arXiv:1703.06345*.

[30] G. D. Forney, "The Viterbi algorithm," *Proc. IEEE*, vol. 61, no. 3, pp. 268–278, Mar. 1973.

[31] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *Proc. Submitted Blog Track ICLR*, 2017, *arXiv:1609.02907*.

[32] I. Beltagy, K. Lo, and A. Cohan, "SciBERT: A pretrained language model for scientific text," in *Proc. Conf. Empirical Methods in Nat. Lang. Process 9th Int. Joint Conf. Natural Lang.Process. (EMNLP-IJCNLP)*, Nov. 2019, pp. 3615–3620.

[33] Z. Zhong and D. Chen, "A frustratingly easy approach for joint entity and relation extraction," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics, Hum. Lang. Technol.*, Jun. 2021, pp. 50–61.

[34] L. B. Soares, N. FitzGerald, J. Ling, and T. Kwiatkowski, "Matching the blanks: Distributional similarity for relation learning," in *Proc. 57th Annu. Meeting Assoc. Comput. Linguistics*, Jul. 2019, pp. 2895–2905.

[35] H. Peng *et al.*, "Learning from context or names? An empirical study on neural relation extraction," in *Proc. Conf. EMNLP*, Nov. 2020, pp. 3661–3672.

[36] T. Dozat and C. D. Manning, "Deep biaffine attention for neural dependency parsing," 2017, *arXiv:1611.01734*.

[37] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. ICLR*, 2015, *arXiv:1412.6980*.

[38] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, 2014.

[39] J.-D. Kim, Y. Wang, T. Takagi, and A. Yonezawa, "Overview of genia event task in BioNLP shared task 2011," in *Proc. BioNLP Shared Task 2011 Workshop*, 2011, pp. 7–15.

[40] J.-D. Kim, Y. Wang, and Y. Yasunori, "The genia event extraction shared task, 2013 edition-overview," in *Proc. BioNLP Shared Task Workshop*, 2013, pp. 8–15.

[41] H.-L. Trieu, T. T. Tran, K. N. Duong, A. Nguyen, M. Miwa, and S. Ananiadou, "Deepeventmine: End-to-end neural nested event extraction from biomedical texts," *Bioinformatics*, vol. 36, no. 19, pp. 4910–4917, 2020.

[42] T. Wolf *et al.*, "Transformers: State-of-the-art natural language processing," in *Proc. Conf. Empir. Methods Natural Lang. Process.: Syst. Demonstrations*, 2020, pp. 38–45.

[43] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2015, pp. 1026–1034.

[44] D. Venugopal, C. Chen, V. Gogate, and V. Ng, "Relieving the computational bottleneck: Joint inference for event extraction with high-dimensional features," in *Proc. Conf. Empir. Methods Natural Lang. Process.*, 2014, pp. 831–843.

[45] J. Lee *et al.*, "BioBERT: A pre-trained biomedical language representation model for biomedical text mining," *Bioinformatics*, vol. 36, no. 4, pp. 1234–1240, 2020.

[46] D. Li, L. Huang, H. Ji, and J. Han, "Biomedical event extraction based on knowledge-driven tree-LSTM," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics: Hum. Lang. Technol.*, 2019, pp. 1421–1430.

[47] X. Ma, Y. Lu, Y. Lu, Z. Pei, and J. Liu, "Biomedical event extraction using a new error detection learning approach based on neural network," *Comput. Mater. Continua*, vol. 63, no. 2, pp. 923–941, 2020.

[48] L. Li, J. Zheng, J. Wan, D. Huang, and X. Lin, "Biomedical event extraction via long short term memory networks along dynamic extended tree," in *Proc. IEEE Int. Conf. Bioinf. Biomed.*, 2016, pp. 739–742.

[49] M. Miwa, P. Thompson, and S. Ananiadou, "Boosting automatic event extraction from the literature using domain adaptation and coreference resolution," *Bioinformatics*, vol. 28, no. 13, pp. 1759–1765, 2012.

Authorized licensed use limited to: Westlake University. Downloaded on December 29,2022 at 04:55:22 UTC from IEEE Xplore. Restrictions apply.