Biomedical Knowledge Graphs Construction from Conditional Statements

Tianwen Jiang, Qingkai Zeng, Tong Zhao, Bing Qin, Ting Liu, Nitesh V. Chawla, Meng Jiang

Abstract—Conditions play an essential role in biomedical statements. However, existing biomedical knowledge graphs (BioKGs) only focus on factual knowledge, organized as a flat relational network of biomedical concepts. These BioKGs ignore the conditions of the facts being valid, which loses essential contexts for knowledge exploration and inference. We consider both facts and their conditions in biomedical statements and proposed a three-layered information-lossless representation of BioKG. The first layer has biomedical concept nodes, attribute nodes. The second layer represents both biomedical fact and condition tuples by nodes of the relation phrases, connecting to the subject and object in the first layer. The third layer has nodes of statements connecting to a set of fact tuples and/or condition tuples in the second layer. We transform the BioKG construction problem into a sequence labeling problem based on a novel designed tag schema. We design a Multi-Input Multi-Output sequence labeling model (MIMO) that learns from *multiple input* signals and generates proper number of *multiple output* sequences for tuple extraction. Experiments on a newly constructed dataset show that MIMO outperforms the existing methods. Further case study demonstrates that the BioKGs constructed provide a good understanding of the biomedical statements.

Index Terms—biomedical knowledge graph, information extraction, conditional statement, sequence labeling

1 Introduction

Biomedical knowledge graphs (BioKG) has been recently brought into attention [1], [2], [3], [4]. For bio-scientists, BioKG provides a new support of exploring and analyzing relevant factual biomedical knowledge in massive biomedical corpora, which was limited in existing biomedical search engines, such as PubMed. Most existing BioKGs directly apply the representation of KGs in the general domain to their construction. For example, disease-gene associations were represented as relational links between a disease node and a gene node [1]. The KG construction model extracts fact tuples in the format of (subject, relation, object) from massive corpora [5] and transforms them into links for reasoning [6], [7] and inference [8]. However, such representation for BioKGs ignores an important role for factual knowledge–the *conditions*.

Conditions are essential in biomedical statements: without the conditions that were precisely given by bio-scientists, the facts might no longer be valid [9]. Biomedical statements are thus mostly conditional statements. Unfortunately, existing BioKGs employ the same flat representation as general KGs and ignore the conditions when being constructed from text. For example, given a biomedical statement below from a biomedical publication [10]:

> *"During T lymphocyte activation as well as production of cytokines, ..."*

the construction models would focus on the main clause and skip this subordinate clause that describes the specific, important conditions. Therefore, *a good BioKG should represent not only fact tuples but also condition tuples for information-lossless extraction.*

The subjects and/or objects of the tuples in general domains are usually named entities (e.g., "United States", "Donald Trump"); so the general KGs are often constructed through named entity detection (NER) and relation extraction. However, the subjects and/or objects in biomedical statements could be either concepts or concepts' attributes. Back to the first quoted sentence, both concepts, "T lymphocyte" and "cytokines" have their attributes ("activation" and "production") given in the conditional expression. So, *a good BioKG should represent not only concepts but also attributes.*

Given the following sentence:

> *"We observed that ... alkaline pH increases the activity of TRPV5/V6 channels in Jurkat T cells."* [10]

existing information extraction systems would extract the following tuple as a unit of factual knowledge in BioKG [11]:

> (alkaline pH, increases, activity of TRPV5/V6 channels in Jurkat T cells),

but this is not satisfactory, because

- the attribute "activity" of the concept "TRPV5/V6 channels" should be explicitly given as the structure of the fact's object;
- the condition "TRPV5/V6 channels in Jurkat T cells" of the observation was not structured from the text, and the concept "Jurkat T cells" should be explicitly given as the condition's object.

In this work, we propose a novel representation for structuring biomedical statements that maintains as much information as possible from the sentences. Each statement is represented as a set of fact tuple(s) and/or condition tuple(s). The subject or object of the tuple is formatted as {concept: attribute}, where the attribute can be null if it is a concept only. For a condition tuple, the subject can be

- *T. Jiang, B. Qin, and T. Liu are with the Research Center for Social Computing and Information Retrieval at Harbin Institute of Technology, Harbin, China. E-mail: {twjiang, bqin, tliu}@ir.hit.edu.cn*
- *Q. Zeng, T. Zhao, N. Chawla, and M. Jiang are with the Department of Computer Science and Engineering at University of Notre Dame, Notre Dame, IN 46556, USA. E-mail: {qzeng, tzhao2, mjiang2}@nd.edu*

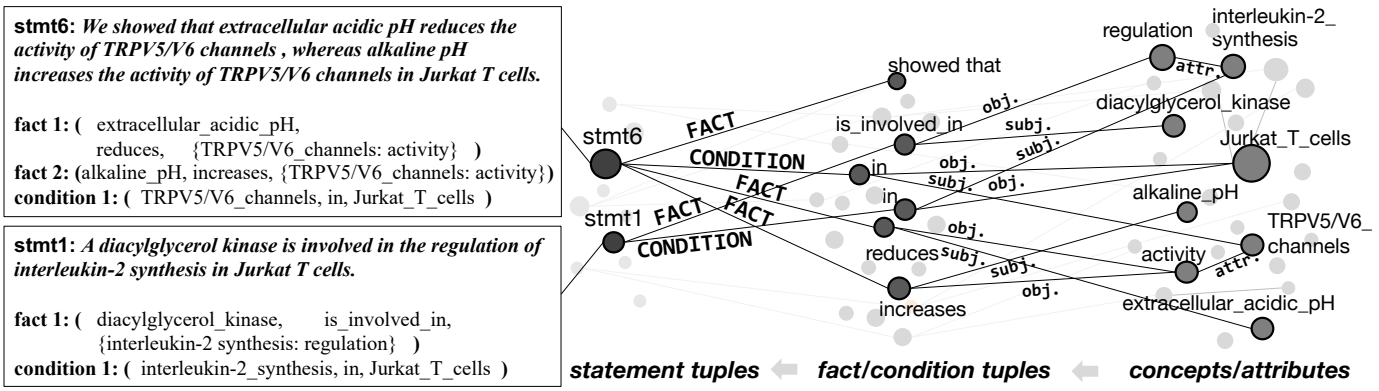

Fig. 1. The proposed biomedical knowledge graph representation has three layers: the bottom layer (right) has concept and attribute nodes; the middle layer has (subject, predicate, object)-tuples of facts and conditions; the top layer (left) is linking to the (conditional) statement sentences in the biomedical corpora.

null if it describes the mean or environment of observation rather than a specific setting of some concept/attribute in the facts; the object can be a concrete value in tuples such as (temperature, is, 63) and (pH, is, 3.4). Following the proposed idea, we expect to extract

- *Fact:* (alkaline pH, increases, {TRPV5/V6 channels: activity});
- *Condition:* (TRPV5/V6 channels, in, Jurkat T cells)

from the second example. And for the first example, we would have two condition tuples as follows:

- *Condition 1:* (null, during, {T lymphocyte: activation});
- *Condition 2:* (null, during, {cytokines: production}).

The novel tuple representation can be easily transformed into a graphical structure. We use Figure 1 to introduce a new BioKG's representation. There are three layers in the knowledge graph. The first layer consists of concept nodes, attribute nodes, as well as the attaching links from attribute to concept (see the nodes on the right-hand). The second layer represents both fact tuples and condition tuples. Each tuple is a node of the relation name (e.g., "reduces", "increases", "in"), connecting to the subject and object that are concept or attribute nodes in the first layer (see the orange nodes in the middle). The third layer has nodes of statement sentences traceable to the original paper and authors. Each statement node connects to a set of fact tuples and/or condition tuples in the second layer (see the blue nodes on the left-hand).

Constructing such a three-layer BioKG, or say, extracting fact and/or condition tuples from biomedical text, is non-trivial. Inspired by [11] which transforms open information extraction as a sequence labeling problem, we build a tag schema for our task:

Definition 1 (Tag Schema $\mathcal{Y}$). Given a sentence, each token will be assigned with a tag that represents the role of it in the tuple. The non-"O (outside)" tags are formatted as "B/I-XYZ", where

- B: beginning, I: inside;
- X $\in$ {fact, condition};
- Y $\in$ {1: subject; 2: relation; 3: object};
- Z $\in$ {concept, attribute, predicate}.

(Please refer to Figure 2 for concrete examples.) The number of unique tags is $|\mathcal{Y}| = 21$. Though not big, we have the following challenges:

Challenge 1: Annotation data is limited for modeling the complex tag schema and distant supervision is unavailable. It takes long for domain experts to annotate biomedical text. For news and tweets, knowledge bases (KBs) such as Wikipedia and Freebase are available for distantly labelling the named entities and specific relations. However, BioKGs considering both facts and conditions are still not available yet for distant supervision. It requires a design for training an effective model by making the most use of the only limited labeled data.

Challenge 2: A given biomedical statement has an uncertain number of tuples, and one token of the statement may have different tags in these different tuples. Actually, we find that 61%/52% statement sentences have more than one fact/condition tuples. These multiple tuples in a statement may often share the same concepts as subject and/or object. For example, in the above second example, the word "TRPV5/V6" was expected to be tagged as both (1) the object ("B-f3c") in the fact tuple and (2) the subject ("B-c1c") in the condition tuple. So, in order to generate one or multiple tuples for each input sentence, the construction model must be a *Multi-Output* sequence labeling model.

To address the two challenges, we propose a **Multi-input Multi-output (MIMO) sequence labeling** model for BioKG construction. It has the following novel designs:

Design 1: Multi-Input Module Features are rich, effective, thanks to high efficiency of fundamental NLP techniques. We evaluated state-of-the-art methods with our data set on upstream tasks such as Language Model (LM) [12], Part-of-Speech (POS) tagging [13], Concept detection, Attribute discovery, and Phrase mining (CAP) [14], [15], [3], [16], and found that most of them are achieving very high accuracy (e.g., 0.96 for POS) requiring no more training on local corpus. Multi-input module harnesses these multiple NLP techniques to process the text for input sequences and feeds them into a multi-head encoder-decoder model with multi-input gates. Such multi-head encoder-decoder model and multi-input gates can utilize complementary signals from the input sequences for learning complex dependencies between the 21 sequence tags.

Design 2: Multi-Output Module It has two layers: one is a relation name tagging layer that predicts the tags of relational phrases and determines the number of output

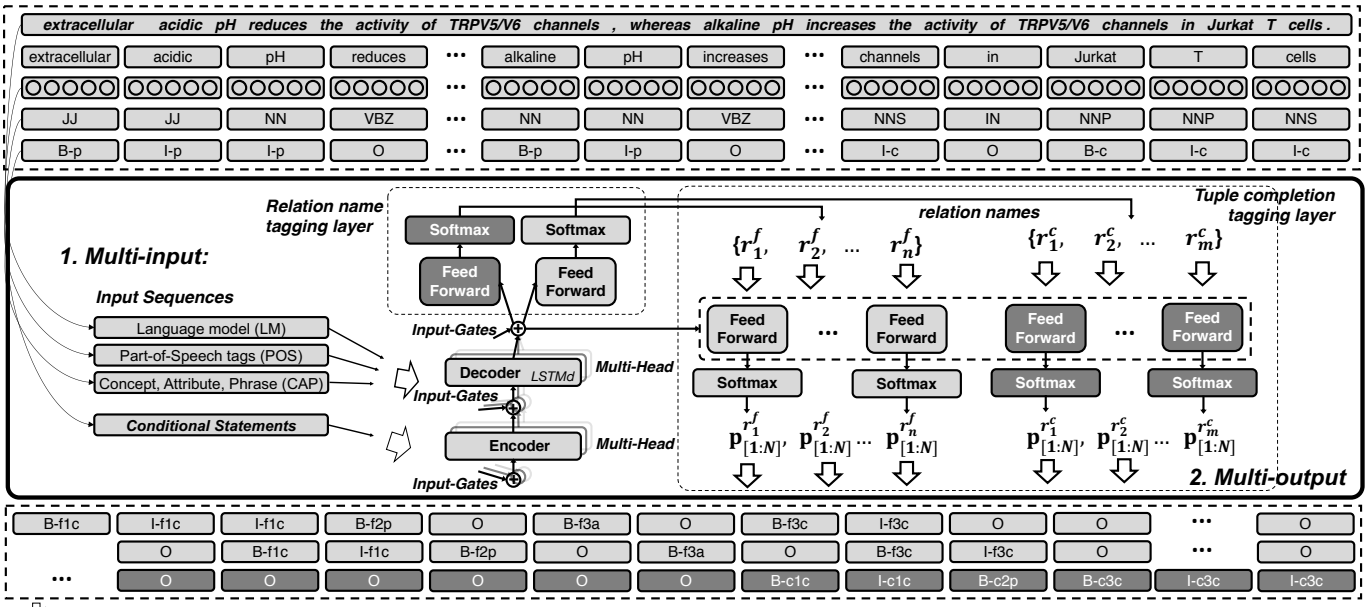

Fig. 2. Our proposed approach has two modules: (1) a Multi-Input module (left) based on a multi-head encoder-decoder model with multi-input gates; (2) a Multi-Output module (right) of a relation name tagging layer and a tuple completion tagging layer. Multi-input sequences are given at the top; and the tag sequences are given at the bottom, which can be converted into the fact and condition tuples.

sequences; the other is a tuple completion tagging layer that generates the tag sequences for completing the fact and condition tuples. Multiple tag sequences are generated, each of which represents a fact or condition tuple. The proposed multi-output sequence labeling model significantly outperforms the traditional single output sequence labeling model on our fact/condition tuple extraction tasks

Experiments demonstrate that MIMO improves F1 score relatively by 6.2% over state-of-the-art models for tuple extraction on a newly constructed biomedical text dataset we will introduce in the later section. We apply MIMO to a large set of 15.5M MEDLINE papers and the BioKG we constructed has as many as 18.1M fact tuples, 7.5M condition tuples, 10.9M concept nodes, and 703K attribute nodes. An example of the BioKG can be found in Figure 5.

We highlight our contributions in this work as follows.

- **A novel BioKG representation:** The new structure represents facts and conditions in biomedical statements. The BioKG has three layers: statement layer, fact/condition tuple layer and concept/attribute layer.
- **A novel BioKG construction model:** We propose a multi-input multi-output sequence labeling model for tag prediction and tuple extraction. It takes advantages of upstream NLP techniques, advanced encoder-decoder and only need small amount of training data.
- **Effectiveness and real use:** The proposed model outperforms baseline methods on a newly constructed dataset. The constructed BioKG contains millions of facts and conditions.

2 Problem Definition

In this section, we formally define fact tuple, condition tuple, structured statement, and the three-layer BioKG representation. Then we introduce how the BioKG construction problem can be transformed into a multi-output sequence labeling task.

Definition 2 (Fact Tuple and Condition Tuple). A (subject, relation, object)-tuple is used to describe the relation between the subject and the object [17]. The role of a tuple, *fact* or *condition*, is determined based on the tuples' semantic dependencies in the biomedical statement (which will be defined in Definition 3).

The subject/object can be either a concept or a concept's attribute and the relation is often a predicate. So we denote a tuple by

$$t = (\{c_1 : a_1\}, p, \{c_3 : a_3\}), \tag{1}$$

where $c_1, c_3 \in \{\text{“}null\text{”}\} + \mathcal{C}$, $p \in \mathcal{P}$ and $a_1, a_3 \in \{\text{“}null\text{”}\} + \mathcal{A}$. Here $\mathcal{C}$, $\mathcal{A}$, and $\mathcal{P}$ denote the set of concepts, attributes, and predicates. The number “1” is for subject and “3” is for object.

For fact and condition tuples, the attribute can be “null” if the subject/object is a concept only. For condition tuples, both the concept and attribute of the subject can be “null”, if the condition describes the mean or environment of observing/claiming the facts while neither concept or attribute is specified for the condition. Examples can be found in the introduction.

Definition 3 (Structured Statement). A biomedical statement sentence (e.g., observation, hypothesis) is structured as a list of fact tuples and/or condition tuples, which forms semantic dependencies that only when the conditions exist, the facts are valid (claimed by the source). For a structured statement $s \in \mathcal{S}$, where $\mathcal{S}$ is the set of structured statements, that has n fact tuples and m condition tuples, it can be

written as

$$s = [t_1^{(f)}, \ldots, t_n^{(f)};\ t_1^{(c)}, \ldots, t_m^{(c)}], \tag{2}$$

where $t_i^{(f)}$ ($i \in \{1, \ldots, n\}$) denotes the i-th fact tuple and $t_j^{(c)}$ ($j \in \{1, \ldots, m\}$) denotes the j-th condition tuple.

Definition 4 (Three-Layer BioKG). It organizes concepts/attributes, facts/conditions, and statements in a bottom-up manner.

A BioKG is formed as $G = \{L_1, L_2, L_3, E_{1,2}, E_{2,3}\}$, where L_i denotes the i-th layer and $E_{i,j}$ denotes the connections between L_i and L_j. Figure 1 can be referred to when we introduce the layout of the BioKG, especially when the nodes are mentioned with their text label and the edges are mentioned with their labels.

• The first layer is $L_1 = \{\mathcal{C}, \mathcal{A}, E_{\mathcal{C},\mathcal{A}}\}$. There is a link $e(c, a) \in E_{\mathcal{C},\mathcal{A}}$ (tagged as "attr.") from the concept node $c \in \mathcal{C}$ to the attribute node $a \in \mathcal{A}$, if a is c's attribute. Then $\{c : a\}$ might be spot as a subject or object in the fact/condition tuples.

• The second layer is $L_2 = \mathcal{T}$ where $\mathcal{T}$ is the set of all fact and condition tuples and each of the tuples is placed as an *orange* node. For a tuple t in Eq.(1), the node is tagged as p. A link $e(t, a_1) \in E_{1,2}$ is tagged as "subj." and a link $e(t, a_3) \in E_{1,2}$ is tagged as "obj." if a_1 and a_3 are not "null"; if one of them is "null", the link goes to the corresponding concept node c_1 or c_3.

• The third layer is $L_3 = \mathcal{S}$ where $\mathcal{S}$ is the set of structured statements and each is placed as a node. For a statement s in Eq.(2), a link $e(s, t_i^{(f)}) \in E_{2,3}$ is tagged as "FACT" and a link $e(s, t_i^{(c)}) \in E_{2,3}$ is tagged as "CONDITION".

Through the layout we know that the problem of *BioKG construction* is equivalent to *structuring a statement sentence into Eq.(2)*, and thus equivalent to *extracting every tuple as Eq.(1) from the input sentence*. Our idea is to transform the tuple extraction task into a *multi-output sequence labeling* problem where the output tag sequences will generate the tuples.

Definition 5 (Multi-Output Sequence Labeling). Given a token sequence $x = [w_1, \ldots, w_N]$ (i.e., a statement sentence), the outputs are multiple tag sequences, denoted by $y_t = [y_t^1, \ldots, y_t^N]$ for a tuple in the statement $t \in s$, where N is the length of the sequence and $y_t^i \in \mathcal{Y}$ belongs to the tag schema.

Here we provide a theorem (as well as an example) on the equivalence of the problem transformation.

Theorem. Given an input token sequence $\mathbf{w}$, extracting a tuple $t = (\{c_1 : a_1\}, p, \{c_3 : a_3\})$ is **equivalent** to tagging the tokens properly as an output sequence y_t, when (a) the tokens of each unit in t can be located in x as a consecutive sequence and (b) the tuple's units does not share any token.

Proof: Given constraint (a), each unit in t, $u \in \{c_1, a_1, p, c_3, a_3\}$ can be denoted as $x[j^u : k^u]$, where $1 \le j^u \le k^u \le N$ if u is not "null". When (b) is true, we have

$$\{j^{u_1}, j^{u_1}+1, \cdots, k^{u_1}\} \cap \{j^{u_2}, j^{u_2}+1, \cdots, k^{u_2}\} = \emptyset,$$
$$\forall u_1, u_2 \in \{c_1, a_1, p, c_3, a_3\} \wedge u_1 \neq u_2. \tag{3}$$

The strategy to generate an output tag sequence that corresponds to the tuple (if the tuple is a fact tuple) is as follows:

- $y_t^{j^{c_1}}$ = "B-f1c", $y_t^{j^{a_1}}$ = "B-f1a", $y_t^{j^{p}}$ = "B-f2p", $y_t^{j^{c_3}}$ = "B-f3c", $y_t^{j^{a_3}}$ = "B-f3a";
- y_t^i = "I-f1c" ($i \in [j^{c_1}+1, k^{c_1}]$), y_t^i = "I-f1a" ($i \in [j^{a_1}+1, k^{a_1}]$), y_t^i = "I-f2p" ($i \in [j^{p}+1, k^{p}]$), y_t^i = "I-f3c" ($i \in [j^{c_3}+1, k^{c_3}]$), y_t^i = "I-f3a" ($i \in [j^{a_3}+1, k^{a_3}]$);

and all other tags will be "O"; the strategy for a condition tuple is the same except the tag difference (e.g., "B-f1c" → "B-c1c").

An example can be found in Figure 2. Given the light grey original token sequence at the top, the fact/condition tuples at the bottom can be transformed into three tag sequences (two light grey sequence for the fact tuples, and dark grey sequence for the condition tuple).

In the next section, we will introduce our proposed model for multi-output sequence labeling, while the ultimate goal is to construct the novel three-layer BioKG.

3 The Proposed Approach

Our approach has two modules: (1) a multi-input module that harnesses recent NLP development to process the text for input sequences from multiple tasks and feeds them into a multi-head encoder-decoder model with multi-input gates; (2) a multi-output module that generates multiple tuple tag sequences for fact and condition tuples, which consists of a relation name tagging layer and a tuple completion tagging layer, as shown in Figure 2. The model is named as Multi-Input Multi-Output sequence labeling.

3.1 The Multi-Input Module

Preprocessing for input sequences: Following fundamental NLP techniques have achieved high accuracy requiring no additional training with labeled data: Language Model (LM) [12], POS [13], CAP [18][14][15][3]. For any given input sentence x, we tokenize it to get a token sequence $[w_1, w_2, \cdots, w_N]$, and represent each token w_i by its word embedding $\mathbf{w}_i$ (pre-trained GloVe vector in this paper). Then we get another three input sequences by the input sentence and the above three fundamental NLP techniques. (1) A pre-trained LSTM-based language model takes the sentence as input and returns semantic embedding $\mathbf{w}_i^{\mathrm{LM}}$ for each token w_i in the sequence, where the dependencies between a token and its predecessors in distant contexts are preserved. (2) We employ NLTK tool to generate the POS tag sequence for the given sentence. We map the POS tag into a continuous space as a embedding $\mathbf{w}_i^{\mathrm{POS}}$ for each token w_i. The POS tag sequence indicates syntactic patterns of the words in a sentence, that is the dependencies between POS tags and output tags, like verbs (e.g., "VBD") and predicates (e.g., "B-f2p"). (3) Multiple complementary IE techniques are used to detect concepts, attributes and phrases from the given sentences, being merged and resulting a CAP sequence. We make tags in the format of "B/I - c/a/p" for the tokens of concepts, attributes, and phrases. We map the CAP tag into a continuous space as a embedding $\mathbf{w}_i^{\mathrm{CAP}}$ for each token w_i.

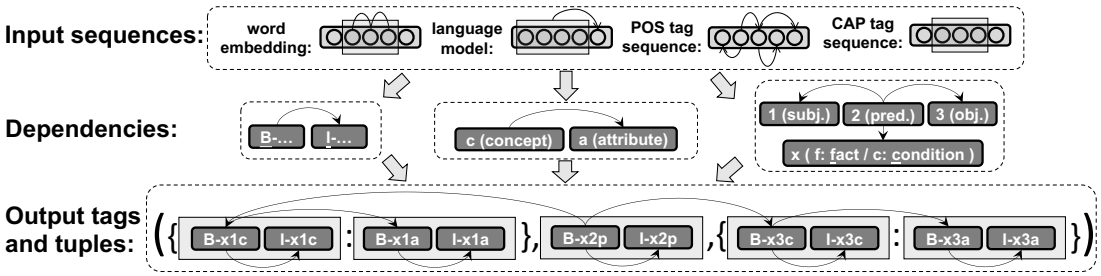

Fig. 3. Dependencies between B-/I- tags, Concept/Attribute tags, Subject/Predicate/Object tags, and Fact/Condition tags can be learned from the four types of input signals, being combined to model complex dependencies between the expected tags.

Why do we employ the four input sequences? Each input sequence encodes a specific type of dependencies, and the final, complex dependencies between the output tags are the combination of the dependencies of all the input signals. Figure 3 visualizes the idea of training a multi-input model, while we analyze the dependencies that each input sequence contributes to the learning process as follows.

- WE: It preserves the co-occurrence between two words in a semantic context window, which would model the dependencies between "B-" and "I-" output tags.
- LM: It preserves the dependencies between a token and its predecessors in distant contexts. So, dependencies between the subject/object and the predicate would be modeled.
- POS: It indicates syntactic patterns of the words in a sentence. Dependencies between POS tags and output tags, like verbs (e.g., "VBD") and predicates (e.g., "B-f2p"), would be modeled.
- CAP: There are high dependencies between the semantic roles and output tags. For example, the tokens of "B/I-c" and "B/I-a" tags would have high probability of being "B/I-XYc" and "B/I-XYa", respectively, where "X" is for "f" act or "c"ondition and "Y" is for "1" (subject) or "3" (object).

The input sequences include but not are not limited to the above. CNN-based character-level vectors and dependency parsing tags [19] can be easily incorporated into the multi-input model.

Multi-head Encoder-Decoder: We first investigate two neural models as encoder layer: one is bidirectional LSTM (BiLSTM) [20], the other is the renown, bidirectional encoder representations from Transformers (BERT) [21]. For the i-th token w_i and its embedding $\mathbf{w}_i$, we can get its encoder embedding as follow,

$$\mathbf{e}_i = \mathrm{Encoder}(\mathbf{w}_i) \tag{4}$$

where the $\mathrm{Encoder}(\cdot)$ is the encoder mapping function, which can be a BiLSTM encoder or a BERT encoder. More details of the implementations can be found in [20] and [21].

Then we adopt a variant LSTM structure , called LSTMd [22] as the decoder layer. LSTMd has a better ability to learn the tag dependence compared to the widely used LSTM in encoder layer, as [22] suggested. In decoder layer LSTMd, the input of the i-th token are the encoder embedding $\mathbf{e}_i$ given by encoder layer, previous output decoder embedding $\mathbf{d}_{i-1}$, former decoder hidden vector $\mathbf{h}_{i-1}$ and its corresponding cell vector $\mathbf{c}_{i-1}$. The detail operations of the memory block are defined as follows:

$$\mathbf{g}_i = \delta(\mathbf{W}^g\mathbf{e}_i + \mathbf{U}^g\mathbf{h}_{i-1} + \mathbf{V}^g\mathbf{d}_{i-1} + \mathbf{b}^g) \tag{5}$$

$$\mathbf{f}_i = \delta(\mathbf{W}^f\mathbf{e}_i + \mathbf{U}^f\mathbf{h}_{i-1} + \mathbf{V}^f\mathbf{d}_{i-1} + \mathbf{b}^f) \tag{6}$$

$$\mathbf{z}_i = \tanh(\mathbf{W}^c\mathbf{e}_i + \mathbf{U}^c\mathbf{h}_{i-1} + \mathbf{V}^c\mathbf{d}_{i-1} + \mathbf{b}^c) \tag{7}$$

$$\mathbf{c}_i = \mathbf{f}_i * \mathbf{c}_{i-1} + \mathbf{g}_i * \mathbf{z}_i \tag{8}$$

$$\mathbf{o}_i = \delta(\mathbf{W}^o\mathbf{e}_i + \mathbf{U}^o\mathbf{h}_{i-1} + \mathbf{V}^o\mathbf{t}_{i-1} + \mathbf{b}^o) \tag{9}$$

$$\mathbf{h}_i = \mathbf{o}_i * \tanh(\mathbf{c}_i) \tag{10}$$

The final output of the decoder layer for the i-th token is:

$$\mathbf{d}_i = \mathbf{W}^d\mathbf{h}_i + \mathbf{b}^d \tag{11}$$

where $\mathbf{g}$, $\mathbf{f}$ and $\mathbf{o}$ are the output embedding of input gate, forget gate and output gate respectively, $\mathbf{b}^{(\cdot)}$ is the bias term, $\mathbf{c}$ is the cell memory, and $\mathbf{W}^{(\cdot)}$, $\mathbf{U}^{(\cdot)}$ and $\mathbf{V}^{(\cdot)}$ are the parameters.

We observe that the input sequences may have different tag predictability on different sentences. For short sentences, POS and CAP are more useful (modeling local dependencies); for long sentences, LM is more effective (modeling distant dependencies). In order to secure the model's robustness on massive data, we apply a multi-head mechanism to the encoder-decoder model. Each head of the encoder-decoder is fed with one type of input sequence, and they are combined at the end of decoder layer. Thus, the tag prediction becomes more stable than using a simple encoder-decoder without the multi-head.

Multi-input gates: We adopt the multi-input gates in ResNet [23], i.e., the direct connecting, to take the most use of the multi-input sequences. By default, the input of BiLSTM or BERT encoder, the input of LSTMd decoder and the multi-output module are directly the outputs of the previous layers, that are: $\mathbf{w}_i$, $\mathbf{e}_i$ and $\mathbf{d}_i$. By applying the multi-input gates, we update the three inputs to $\hat{\mathbf{w}}_i$, $\hat{\mathbf{e}}_i$ and $\hat{\mathbf{d}}_i$ as follows:

$$\hat{\mathbf{w}}_i = \mathbf{w}_i + \alpha^{(1)}\mathbf{W}^{(1)}\mathbf{w}_i^{\mathrm{LM}} + \beta^{(1)}\mathbf{U}^{(1)}\mathbf{w}_i^{\mathrm{POS}} + \gamma^{(1)}\mathbf{V}^{(1)}\mathbf{w}_i^{\mathrm{CAP}} \tag{12}$$

$$\hat{\mathbf{e}}_i = \mathbf{e}_i + \alpha^{(2)}\mathbf{W}^{(2)}\mathbf{w}_i^{\mathrm{LM}} + \beta^{(2)}\mathbf{U}^{(2)}\mathbf{w}_i^{\mathrm{POS}} + \gamma^{(2)}\mathbf{V}^{(2)}\mathbf{w}_i^{\mathrm{CAP}} \tag{13}$$

$$\hat{\mathbf{d}}_i = \mathbf{d}_i + \alpha^{(3)}\mathbf{W}^{(3)}\mathbf{w}_i^{\mathrm{LM}} + \beta^{(3)}\mathbf{U}^{(3)}\mathbf{w}_i^{\mathrm{POS}} + \gamma^{(3)}\mathbf{V}^{(3)}\mathbf{w}_i^{\mathrm{CAP}} \tag{14}$$

where $\alpha^{(\cdot)}$, $\beta^{(\cdot)}$ and $\gamma^{(\cdot)}$ are the control value (1 for open and 0 for block) for the three type of inputs: $\mathbf{w}_i^{\mathrm{LM}}$, $\mathbf{w}_i^{\mathrm{POS}}$

and $\mathbf{w}_i^{\mathrm{CAP}}$ inputs respectively, and $\mathbf{W}^{(.)}$, $\mathbf{U}^{(.)}$ and $\mathbf{V}^{(.)}$ are the mapping matrix to the same input space for the three type of inputs respectively.

3.2 Multi-Output Module

We propose to generate multiple output sequences, observing that 93.8% statement sentences make multiple tuples. We cannot use only one single output tagging sequence from which parsing these multiple tuples. There are two main reasons for the proposal of multiple output sequence labeling model. First, it brings ambiguities when assigning tagged subjects/objects to the multiple tagged relation names. Second, a token may have different expected tags in these multiple tuples. Actually, 21.7% of the sentences have at least one token that appears in at least one fact tuple and at least one condition tuple, expecting tags "B/I-fYZ" as well as "B/I-cYZ"; 18.1% of the sentences have at least one token that appears in one condition tuple as a part of subject and in another condition tuple as a part of object, expecting tags "B/I-c1Z" as well as "B/I-c3Z". Therefore, we extend the typical single-output sequence labeling to a multi-output design.

Since the model will generate multiple output sequences, what is the number of the output sequences? We reveal the significant role of relation names in making tuples. If we firstly tag the relation names out, for each relation name (of tags beginning with "B-f2p" as a fact's and "B-c2p" as a condition's), the module would generate an output sequence, respectively. After that we can just extract all possible tuples, whose relation has been specified, from every output sequence. Two observations on the annotated data support this idea: We transform each of the 1,410 tuples into a tag sequence. For the same sentence, if the tuples' relation names are the same, we merge their tag sequences into one and then use the matching function in [11] to recover the tuples. First, 0 token has conflicting tags among the 240 merged sequences. Second, the recovery has 0 missing or wrong tuple. So, generating one output sequence and completing the tuples per relation name is practical.

We design two layers for multi-output module to output a proper number of tag sequences according the input statement: one is called the Relation Name Tagging (RNT) layer and the other is called the Tuple Completion Tagging (TCT) layer.

Relation name tagging (RNT) layer: It consists of feed-forward neural networks (FFNs) and softmax layers. Decoded vectors are fed into the FFNs and the softmax predict the probability distribution of tags on fact and condition, respectively:

$$\mathbf{p}_i^f = \mathrm{softmax}(\mathrm{FFN}_{\mathrm{RNT}}^f(\hat{\mathbf{d}}_i)), \quad (15)$$

$$\mathbf{p}_i^c = \mathrm{softmax}(\mathrm{FFN}_{\mathrm{RNT}}^c(\hat{\mathbf{d}}_i)). \quad (16)$$

where f is for fact and c for condition.

Now we have two tag sequences, one for fact and the other for condition. As we have argued with one-output, extracting tuples from the "two-output" sequences cannot resolve the tag conflicts, either. Fortunately, the relation name tags have no conflicts among multiple tuples, because of their significant role in making tuples as we have mentioned. Here in this layer we extract only the relation names: $\{r_1^f, r_2^f, \cdots, r_n^f\}$ denotes the n relation names (beginning with "B-f2p" tag) in fact tuples and $\{r_1^c, r_2^c, \cdots, r_m^c\}$ denotes the m relation names (beginning with "B-c2p" tag) in condition tuples.

Tuple completion tagging (TCT) Layer: This layer predicts n fact tag sequences and m condition tag sequences according to the number of relation names returned by previous RNT layer. Each sequence is generated by a FFN and a softmax layer. The FFN obtains the relation name from the RNT layer. The FFN's input also includes the token's vectors from the encoder-decoder model of the multi-input module.

Here we take condition sequences as an example to describe the details of the method. When predicting the j-th tag sequence, we obtain the position embedding of the i-th token as follows, according to the relative position to the j-th relation name's tag "B-c2p":

$$\mathbf{v}_{i,j}^c = \Phi(\big|i - \mathrm{position}(r_j^c)\big|). \quad (17)$$

where $\mathrm{position}(\cdot)$ is used to obtain the start position of the given extracted span (e.g., an extracted relation name), and $\Phi(\cdot)$ is to request the corresponding embedding of the given relative position. Thus, the tag probability distributions of the i-th token in the condition tag sequences are:

$$\begin{cases} \mathbf{p}_i^{(r_1^c)} = \mathrm{softmax}(\mathrm{FFN}_{\mathrm{TCT}}^c(\mathbf{W}^c\mathbf{v}_{i,1}^c + \hat{\mathbf{d}}_i)), \\ \mathbf{p}_i^{(r_2^c)} = \mathrm{softmax}(\mathrm{FFN}_{\mathrm{TCT}}^c(\mathbf{W}^c\mathbf{v}_{i,2}^c + \hat{\mathbf{d}}_i)), \\ \cdots, \\ \mathbf{p}_i^{(r_m^c)} = \mathrm{softmax}(\mathrm{FFN}_{\mathrm{TCT}}^c(\mathbf{W}^c\mathbf{v}_{i,m}^c + \hat{\mathbf{d}}_i)). \end{cases} \quad (18)$$

Similarly, we have the following tag distributions for the i-th token in the fact tag sequences:

$$\begin{cases} \mathbf{p}_i^{(r_1^f)} = \mathrm{softmax}(\mathrm{FFN}_{\mathrm{TCT}}^f(\mathbf{W}^f\mathbf{v}_{i,1}^f + \hat{\mathbf{d}}_i)), \\ \mathbf{p}_i^{(r_2^f)} = \mathrm{softmax}(\mathrm{FFN}_{\mathrm{TCT}}^f(\mathbf{W}^f\mathbf{v}_{i,2}^f + \hat{\mathbf{d}}_i)), \\ \cdots, \\ \mathbf{p}_i^{(r_n^f)} = \mathrm{softmax}(\mathrm{FFN}_{\mathrm{TCT}}^f(\mathbf{W}^f\mathbf{v}_{i,n}^f + \hat{\mathbf{d}}_i)), \end{cases} \quad (19)$$

where $\mathbf{v}_{i,j}^{(}\cdot)$ is the position embedding of the i-th token in the j-th fact sequence, representing the relative position to the j-th relation name's tag "B-f2p", and $\mathbf{W}^{(\cdot)}$ is a mapping matrix to map the position embedding to the same space of input embedding $\hat{\mathbf{d}}_i$

After that, we can directly obtain the final multiple fact and condition tag sequences from these tag distributions, by choosing the tag with highest probability as the predicted tag. Specifically, the predicted tags for the i-th token in multiple tag sequences are as follows,

$$\begin{cases} y_{(r_1^f)}^i = \mathrm{argmax}\ \mathbf{p}_i^{(r_1^f)}, \\ y_{(r_2^f)}^i = \mathrm{argmax}\ \mathbf{p}_i^{(r_2^f)}, \\ \cdots, \\ y_{(r_n^f)}^i = \mathrm{argmax}\ \mathbf{p}_i^{(r_n^f)}. \end{cases} \quad (20)$$

$$\begin{cases} y^i_{(r^c_1)} = \text{argmax } \mathbf{p}^{(r^c_1)}_i, \\ y^i_{(r^c_2)} = \text{argmax } \mathbf{p}^{(r^c_2)}_i, \\ \cdots, \\ y^i_{(r^c_m)} = \text{argmax } \mathbf{p}^{(r^c_m)}_i. \end{cases} \tag{21}$$

where $y^{(\cdot)}_{(\cdot)} \in \mathcal{Y}$. And the list of fact and condition tag sequences for the input statement sentence can be obtained as follows, respectively,

$$\begin{cases} y_{(r^f_1)} = [y^1_{(r^f_1)}, \cdots, y^i_{(r^f_1)}, \cdots, y^N_{(r^f_1)}], \\ y_{(r^f_2)} = [y^1_{(r^f_2)}, \cdots, y^i_{(r^f_2)}, \cdots, y^N_{(r^f_2)}], \\ \cdots, \\ y_{(r^f_n)} = [y^1_{(r^f_n)}, \cdots, y^i_{(r^f_n)}, \cdots, y^N_{(r^f_n)}]. \end{cases} \tag{22}$$

$$\begin{cases} y_{(r^c_1)} = [y^1_{(r^c_1)}, \cdots, y^i_{(r^c_1)}, \cdots, y^N_{(r^c_1)}], \\ y_{(r^c_2)} = [y^1_{(r^c_2)}, \cdots, y^i_{(r^c_2)}, \cdots, y^N_{(r^c_2)}], \\ \cdots, \\ y_{(r^c_m)} = [y^1_{(r^c_m)}, \cdots, y^i_{(r^c_m)}, \cdots, y^N_{(r^c_m)}]. \end{cases} \tag{23}$$

where N is the length of the input tokenized statement sentence $x = [w_1, w_2, \cdots, w_N]$.

Finally, we apply the matching function in [11] to complete and extract the tuples (i.e., the concepts and/or attributes in the subjects and objects) from each output tag sequence. Such transformation from tag sequences to tuples is executable because of the Theorem in Section 2. The relation name r^f_j specified tuple(s) $t^{(r^f_j)}$ was transformed from $y_{(r^f_j)}$ for instance: $t^{(r^f_j)} \leftarrow y_{(r^f_j)}$. The final structured statement by MIMO for the input statement sentence $x = [w_1, w_2, \cdots, w_N]$ is:

$$s = [t^{(r^f_1)}, \cdots, t^{(r^f_n)};\ t^{(r^c_1)}, \cdots, t^{(r^c_m)}] \tag{24}$$

3.3 Loss Function and Training

Loss in RNT layer: Given a sentence $x = [w_1, w_2, \cdots, w_N]$, the loss function of the relation name tagging layer can be written as below:

$$\ell^x_{\text{RNT}} = -\sum_{i=1}^{N} (\log(\mathbf{p}^f_{i,y^f_i}) + \log(\mathbf{p}^c_{i,y^c_i})), \tag{25}$$

where $\mathbf{p}^f_{i,y}$ and $\mathbf{p}^c_{i,y}$ are the probability of predicting y as the tag of the i-th token in the fact and condition tag sequences, respectively. y^f_i and y^c_i are the observed tag of the i-th token w_i in the fact and condition tuple, respectively.

To fully train the MIMO for a well performance on relation name prediction in RNT layer, we actually take a *pseudo* fact (or condition) tag sequence as ground-truth, as we cannot encode multiple tuples into one tag sequence. The *pseudo* tag sequence is made by merging all the multiple tag sequences together for each token, obeying the overwriting principle when a tag conflict happens.

Loss in TCT layer: The loss function of the tuple completion tagging layer is consisted of two parts, loss on fact tuples and loss on condition tuples:

$$\begin{cases} \ell^x_{\text{TCT}} = \ell^x_{fact} + \ell^x_{cond.}, \\ \ell^x_{fact} = -\sum_{i=1}^{N}\sum_{j=1}^{n} \log(\mathbf{p}^{(r^f_j)}_{i,y^f_{i,j}}), \\ \ell^x_{cond.} = -\sum_{i=1}^{N}\sum_{j=1}^{m} \log(\mathbf{p}^{(r^c_j)}_{i,y^c_{i,j}}), \end{cases} \tag{26}$$

where n and m are the number of fact and condition tag sequences for the sentence s, respectively. $\mathbf{p}^{(r^f_j)}_{i,y^f_{i,j}}$ and $\mathbf{p}^{(r^c_j)}_{i,y^c_{i,j}}$ are the probability of predicting $y^f_{i,j}$ and $y^c_{i,j}$ as the tag of the i-th token w_i in the j-th fact and condition ground-truth tag sequence, respectively.

As we know, for a given input statement, there are often multiple tag sequences as ground-truth and the predict relation name in RNT layer might be wrong, so that we do not known which predicted tag sequence corresponding to which real tag sequence. In practice, we take a greedy strategy to deal the assignment of predicted tag sequence to ground-truth tag sequence. First, for each ground-truth tag sequence, we assign it with the predicted tag sequence that have lowest loss to it. Then, for the each rest predicted tag sequences that have not been assigned to any ground-truth tag sequence, we assign it with the ground-truth tag sequence that have lowest loss to it.

The **overall loss** function for optimization is:

$$\ell = \ell_{\text{RNT}} + \ell_{\text{TCT}} = \sum_{x \in \mathcal{X}} (\ell^x_{\text{RNT}} + \ell^x_{\text{TCT}}), \tag{27}$$

where $\mathcal{X}$ is the set of statement sentences.

Training details: On one hand, Equations (25) and (26) show that the error signal can be propagated from the RNT/TCT layers to the encoder-decoder model. On the other hand, the RNT layer specifies the relation names, or say, the tokens that have tags "B/I-f2p" and "B/I-c2p" for each tag sequence in the TCT layer. So we cannot have smooth gradients for back propagation from the TCT layer to the RNT layer. So, in order to have good learning effectiveness, the quality of predicting relation names has been secured beforehand. We pre-train the RNT layer with the multi-input module till the relation name's tag prediction achieves a higher-than-0.8 F1 score. Then we plug the TCT layer onto the RNT layer and train the entire framework to generate the multi-output tag sequences.

4 Experiments

In the experiments, we construct a newly annotated dataset called BioCFE. To the best of our knowledge, BioCFE brings the first time that conditional information was carefully annotated on biomedical literature. We will report the experimental results of fact/condition tag prediction and tuple extraction by MIMO model, as well as its variants and the existing competitive methods, on the BioCFE. To the end of this section, we will give a case study to indicate the constructed BioKG providing a good understanding of the biomedical statements and a good use for bio-scientists.

4.1 Biomedical Text Datasets

We built a system with GUI (Figure 4) to collect a new dataset for the joint tuple extraction purpose, named Biomedical Conditional Fact Extraction (BioCFE). Three participants (experts in biomedical domain) manually annotated the fact and condition tuples from statement sentences from 31 paper abstracts in the MEDLINE database[1]. The

1. https://www.nlm.nih.gov/databases/download/pubmed_medline.html

TABLE 1
The proposed MIMO model outperforms existing methods in terms of Precision, Recall, and F1 scores on both sequence tag prediction and fact/condition tuple extraction. The results demonstrate the effectiveness of the multi-input signals including Language Models, POS tags, and Concept-Attribute-Phrase tags, and the effectiveness of Multi-Output module. Higher score performs better.

	Sequence Tag Prediction (%)			Fact/Condition Tuple Extraction (%)		
	P	**R**	**F / F_{Fact}, $F_{Condition}$**	**P**	**R**	**F / F_{Fact}, $F_{Condition}$**
Allennlp OpenIE [11]	-	-	-	42.60	38.22	40.29 / -, -
Stanford OpenIE [19]	-	-	-	47.11	41.62	44.19 / -, -
Structured SVM [24]	32.68	25.80	28.83 / 32.76, 24.71	47.62	46.15	46.87 / 45.01, 48.72
CRF [25]	60.07	41.92	49.37 / 56.23, 41.87	65.19	62.44	63.78 / 64.07, 63.44
BiLSTM-LSTMd [22]	61.00	56.26	58.53 / 65.16, 51.78	71.57	66.55	68.97 / 69.51, 68.41
BERT-LSTMd	70.07	70.19	70.13 / 74.30, 65.88	78.64	73.67	76.08 / 76.14, 75.99
MIMO (BiLSTM based)	67.80	58.24	62.66 / 66.67, 58.58	75.35	74.67	75.01 / 74.91, 75.10
MIMO (BERT based)	**75.91**	**71.08**	**73.41** / **76.01**, **70.75**	**81.06**	**80.53**	**80.79** / **79.94**, **81.64**
LM **POS** **CAP**	w/o **MO** (BiLSTM based, Tuple Extraction)			w/ **MO** (BERT based, Tuple Extraction)		
	78.64	73.67	76.08 (↑ 0.00%)/ 76.14, 75.99	77.38	79.19	78.27 (↑ 2.88%)/ 76.64, 79.89
✔	79.57	74.77	77.10 (↑ 1.34%)/ 77.47, 76.71	79.04	79.87	79.45 (↑ 4.43%)/ 79.09, 79.81
✔	79.66	74.59	77.04 (↑ 1.26%)/ 76.80, 77.27	79.40	79.50	79.45 (↑ 4.43%)/ 78.66, 80.24
✔	79.01	74.02	76.43 (↑ 0.46%)/ 77.43, 75.43	79.05	79.72	79.39 (↑ 4.35%)/ 78.41, 80.36
✔ ✔	80.66	75.52	78.01 (↑ 2.54%)/ 77.91, 78.09	79.67	**80.65**	80.16 (↑ 5.36%)/ 79.16, 81.14
✔ ✔	80.90	76.20	78.48 (↑ 3.15%)/ 78.35, 78.60	79.97	79.56	79.76 (↑ 4.84%)/ 79.06, 80.47
✔ ✔	81.13	76.20	78.59 (↑ 3.30%)/ 78.42, 78.73	79.41	79.98	79.70 (↑ 4.76%)/ 79.49, 79.90
✔ ✔ ✔	**81.74**	**76.29**	**78.92** (↑ 3.73%)/ **78.67**, **79.16**	**81.06**	80.53	**80.79** (↑ 6.19%)/ **79.94**, **81.64**

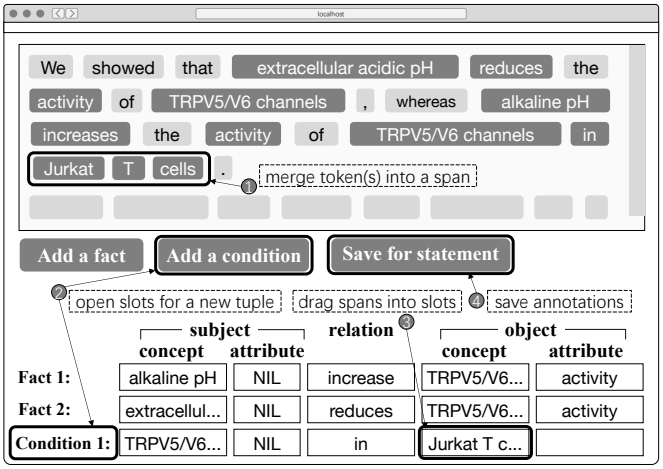

Fig. 4. Annotation by four steps: (1) merge token(s) into a span; (2) make slots for a new tuple; (3) drag spans into the slots; (4) save annotations.

annotation procedure took over 30 minutes on average for each paper. Here we present the brief guide to the system. First, the users merged the selected token(s) into a span. Second, they gave a proper number of fact and/or condition tuple(s), where the proper number is not fixed and depends on the concrete sentence. Each tuple has five slots (subject's concept, subject's attribute, relation phrase, object's concept, and object's attribute) to be filled. Third, they dragged the spans filling into the slots. If the three annotations are inconsistent, we filtered out the case. Eventually we get 756 fact tuples and 654 condition tuples from 336 consistently annotated sentences. It is common to see one sentence having multiple facts and/or conditions, and actually 61%/52% statement sentences have more than one fact/condition tuples in BioCFE.

After investigating the tag distributions of fact tuples and condition tuples, we have two observations: (1) The number of conditions is comparable with the number of facts, demonstrating the existence and significance of conditions in biomedical statements. (2) The attribute-related tags take 13.7% and 13.6% of non-"O" tags in fact and condition tuples, respectively. So, it is important to distinguish concept and concept's attribute in subjects/objects.

4.2 Experimental Settings

4.2.1 Validation Settings

The annotated sentences were randomly divided into a training set (60%, 201 sentences), a validation set (8%, 27 sentences), and an evaluation set (32%, 108 sentences). The evaluation contains 242 fact tuples and 209 condition tuples (on average) as ground truth. We repeat it 5 times, conduct experiments for each, and report the average results.

4.2.2 Baseline Methods

We conduct three lines of methods to be compared with our proposed MIMO. Since the BioCFE is a not large-scale dataset, it is worthy to firstly investigate the performance of the classical machine learning (ML) methods, in case that these methods can already handle the problem.

We choose two classical ML methods including: Structured Support Vector Machine (SVM) [24], which is an SVM-HMM sequence tagging algorithm that can deal with tagging problems with millions of words and millions of constructed features; Conditional random field (CRF) [25], a type of probabilistic graphical model that can be used to model sequential data, such as labels of words in a sentence

The second line is OpenIE systems that extract (subject, relation, object)-tuples without considering attributes or conditions: AllenNLP OpenIE [11], the state-of-the-art supervised OpenIE system which uses a deep sequence labeling model to extract a list of factual tuples; Stanford OpenIE [19], which splits each sentence into a set of entailed clauses. Each clause is then maximally shortened, producing a set of entailed shorter sentence fragments. These fragments are then segmented into OpenIE triples.

The last line is competitive neural sequence labeling methods for relation extraction: BiLSTM-LSTMd [22], containing a BiLSTM as encoder and a new LSTM-decoder (LSTMd) as decoder. BiLSTM-LSTMd outperformed previous models on entity and relation extraction tasks; We also employ BERT [21] as encoder to replace the BiLSTM in BiLSTM-LSTMd to make it a more competitive baseline, denoted as BERT-LSTMd.

We enhance statistical sequence labeling models (i.e., CRF and Structured SVM) with multi-input signals for fairness, and train them for fact tuple and condition tuple extraction separately. For OpenIE systems, we attach a condition/fact classification following the outputs of them, as these OpenIE systems do not distinguish the facts and conditions. In the neural baselines (BiLSTM-LSTMd and BERT-LSTMd), fact extraction and condition extraction share the encoder-decoder model and use different, proper parameters in the linear-softmax layers to recognize facts and conditions respectively.

4.2.3 Evaluation Methods

We evaluate the above methods on two tasks: (1) sequence tag prediction and (2) tuple extraction. The tag prediction task is actually 21-class classification (because the size of tag schema $\mathcal{Y} = 21$). We have similar observations on Micro F1 scores as Macro F1 scores, so we report Macro F1 only. For evaluating tuple extraction, we use pair-wise comparison to match the extracted and ground-truth tuples. Because it might be too difficult to capture the entire tuple, we evaluated the correctness at the level of the tuple slots (including concepts, attributes, and predicates). For both tasks, we use the standard metrics, precision (**P**), recall (**R**) and F1 score (**F**) to measure the performances.

4.2.4 Implementation Details

The multi-input module has a BiLSTM/BERT encoder and a LSTM decoder. The word embeddings were pre-trained over biomedical literature from MEDLINE database, using GloVe [26] with the dimension size d_{WE} = 50. The language model was pre-trained over the same corpus used in word embedding pre-training, where the dimension size is d_{LM} = 200. The size of POS tag embedding is d_{POS} = 6. The size of CAP tag embedding is d_{CAP} = 3. The number of LSTM units in the encoding layer is 300. The number of transformer units in the BERT encoding layer is 768.

4.3 Results on Tag Prediction and Fact/Condition Tuple Extraction

In this section, we first present the overall performance of MIMO and its baseline methods. After that we do the ablation study for MIMO to investigate the effectiveness of multi-input module and multi-output module. Then we present the error analysis of MIMO.

4.3.1 Overall Performance

Table 1 shows that the proposed MIMO equipped with a BERT encoder (BERT-based MIMO) consistently performs the best over all the baselines, including BiLSTM-based MIMO, on the tasks of tag prediction and tuple extraction. Compared to BiLSTM-LSTMd, BiLSTM-based MIMO improves F1 score relatively by 7.1% on tag prediction and by 8.8% on tuple extraction; compared to BERT-LSTMd, BERT-based MIMO improve F1 by **4.7%** and **6.2%** on the two tasks, respectively. Apparently the BERT encoder significantly improves the performance comparing with BiLSTM encoder based model (by 16.9–17.2% on tag prediction and 7.7–10.3% on tuple extraction). And the MIMO design can further improve it. Neural sequence labeling models perform better than OpenIE systems and classical ML methods. Neural sequence labeling models are more adaptive to learning structures with the new tag schema. Open IE methods are not effective, even we only evaluate their performance on tuple extraction ignoring fact/condition classification.

Compared to BERT-LSTMd, the BERT-based MIMO improves precision and recall relatively by 8.3% and 1.3% on tag prediction; and relatively by 3.1% and 9.3% on tuple extraction, respectively. When the tags were more precisely predicted, the tuple's five slots would be more accurately filled, and we would have more complete tuples.

Comparing the middle three columns (tag prediction task) and the right-hand three columns (tuple extraction task) of the competitive methods, the task of tuple extraction seems to be easier than the tag prediction, as the better performance consistently achieved by all the methods. However, the truth is that the corrupted tagged span will be discarded when parsing tuples from the tag sequences. These corrupted tags include: the tagged span without a "B-XYZ" at the beginning token, and the tagged span of which the distance to the nearest tagged span is larger than the pre-defined distance threshold. In this way, only when the tags were more precisely predicted, we will use the tags to complete the tuples, so that the improvement on tuple extraction is not as big as the one on tag predict among the baseline methods.

We observe that the improvements on condition's tags/tuples are consistently bigger than the improvements on fact's tag/tuples. It shows that the MIMO design recognizes the role of conditions in the statement sentences better.

4.3.2 Effectiveness of Modules

Table 1 also compares variants of the BERT-based MIMO to evaluate the effectiveness of the following modules: **multi-input module**, using such as none, or one (in LM, POS, and CAP), double combination, or triple combination of the input sequences;**multi-output module**, with the RNT layer only (generating one fact tag sequence and one condition tag sequence) or a combination of RNT and TCT layers (generating multiple sequences for each tuple type).

Multi-input sequences: When the choices of the encoder model and multi-output layers are specified, we observe that triple combination of input sequences performs better than double combinations and the double combinations win over the sole input. An additional input sequence makes a relative F1 improvement by 0.46–1.43%. The triple combination improves F1 relatively by 3.2–3.73%. We conclude that: the leveraging of any additional input sequence or all three input sequences brings the promising performance on tuple extraction; the three types of input sequences encode *complementary information* for learning dependencies in the

proposed tag schema. Here we attempt to investigate the usefulness of the three types of input sequences (i.e., LM, POS, and CAP) as follows.

Usefulness of LM: In Table 1, the models that have used POS, or CAP, or both, incorporating the LM sequence will consistently improve the F1 scores on tuple extraction task. The language model is designed to learn the dependencies between a token and its predecessors in distant contexts. Having the LM sequence we have more chance to correctly recognize the subject (or object) relative to its relation name, and thus reduce the false positives of tag "B/I-X1Z" (or "B/I-X3Z"). Given a statement sentence below:

> *"We demonstrated that a specific inhibitor of clathrin-dependent endocytosis, dynasore, blocked TRPV5/V6..."*

The MIMO models without LM input sequence all tag the attribute "specific inhibitor" as the object of "demonstrated" by "[B-f3a I-f3a]", where actually the attribute "specific inhibitor" is the subject of "blocked", even "demonstrated" is in a shorter distance. Fortunately, *LM encodes long dependencies between the tokens at the sentence-level*. The MIMO models with LM input sequence successfully tag "specific inhibitor" as "[B-f1a I-f1a]", which means that it is recognized as the subject of "blocked" instead of the nearby predicate word "demonstrated".

Usefulness of POS: The models that have used LM, or CAP, or both, incorporating the POS tag sequence consistently improves the F1 scores. The POS tag encodes the token's syntactic feature. For example, verbs and prepositions (e.g., "in", "during") often act as the relation name of facts and conditions, respectively; conjunction words (e.g., "that", "which") indicate subordinate clauses, so the noun phrase before the conjunction word is likely to be the subject of the tuple given by the clause. We conclude that POS tags are useful for recognizing long multi-word concepts in the subjects/objects as well as the condition roles of tuples. Here is an example that MIMO models without POS tag input sequence fail to tag, while it is perfectly recognized as a condition tuple when using POS. The statement sentence is as follow:

> *"...the role of oral epithelial cell-derived cytokines on T cell activation..."*

the POS tag sequence is

[...DT NN IN JJ JJ JJ NNS IN NNP NN NN...].

First, the patterns [JJ JJ JJ NNS] and [NNP NN NN] indicate the multi-word concept names: "oral epithelial cell-derived cytokines" and "T cell activation". *Second*, the pattern [DT...IN...] indicates that "role" is an attribute of "oral epithelial cell-derived cytokines". *Third*, the second "IN" indicates the condition role of a tuple ({oral epithelial cell-derived cytokines : role}, on, {T cell : activation}).

Usefulness of CAP: Table 1 shows the improvement contributed by CAP tag sequences. The formerly-detected concepts, attribute names, and phrases are absolutely useful for tagging the slots of subjects and objects. In other words, the tags "B/I-c" and "B/I-a" in the CAP sequence are strongly associated with the target tags "B/I-XYc" and "B/I-XYa", respectively. Given the above example, the CAP seq. is

[...O B-a O B-p I-p I-p I-p O B-c I-c B-a...].

First, the concept "T cell" and attribute "activation", expected to be tagged as "[B-c3c I-c3c]" and "[B-c3a]" eventually, have been tagged as "[B-c I-c]" and "[B-a]" by concept and attribute extraction techniques. *Second*, though "oral epithelial cell-derived cytokines" was not able to be tagged as a concept, it has been recognized as a multi-word phrase. And the sequence labeling model will learn the dependencies between phrases and potential concept-related tags.

Layers in the multi-output module: If the multi-output models have both RNT and TCT layers (the right three columns of the lower part in Table 1), the F1 score is relatively 1.41–3.87% higher than the models that have the RNT layer only (the middle three columns of the lower part in Table 1). Moreover, the recall is improved relatively by 4.41–7.49%. Single output model for fact/condition (e.g., RNT layer only) would bring ambiguities when assigning tagged subjects/objects to the multiple tagged relation names. Besides that, one token may have different expected tags in these multiple tuples, and this phenomenon is widely seen in condition tuple pairs (over 18%). The TCT layer, which generates multiple tag sequences for each type of tuple (i.e., fact and condition), *can precisely recall more tuples that were not well extracted by ambiguous tag assignment and tag conflicts problems in single output model*. For example, given the following statement sentence:

"*Immunohistochemical staining of the tumors demonstrated a decreased number of blood vessels in the treatment group versus the controls.*"

The proposed model is able to find one fact tuple and two condition tuples precisely:

-*Fact 1:* ({tumors:immunohistochemical_staining}, demonstrated, {blood_vessels:decreased_number})
-*Condition 1:* (blood_vessels,in,**treatment_group**)
-*Condition 2:* (**treatment_group**,versus,controls)

Note that the concept "treatment_group" acts as the *object* of Condition Tuple 1 (having tags "B/I-c3c") and the *subject* of Condition Tuple 2 (having tags "B/I-c1c"). The multi-output design tackled this issue while other models could not. The leveraging of multi-output module does leads a much better recall score as we excepted on tuple extraction, though it may sometimes slightly hurt precision score, as we have seen in Table 1.

4.4 Efficiency and Case Study on BioKG

Efficiency: All the experiments were conducted on 16 Graphics Cards (GeForce GTX 1080 Ti), where one individual model only used 1 GPU. Each model was trained for 1,000 epochs. For the BiLSTM-LSTMd MIMOs, the pre-training took 2.4 hours and the re-training (TCT layer) took 0.4 hour. For the BERT-LSTMd MIMOs of the best performance, the pre-training took 3.5 hours and the re-training took 0.9 hour. It took 5.7 hours to extract fact and condition tuples from 141 million sentences in the MEDLINE text data. It is comparable with existing approaches in terms of scalability.

Case study: Suppose we are interested in what *increased/decreased* "apoptosis". The BioKG provides us a snapshot in Figure 5. We conclude that (1) "OGD exposure" and the "RNAi-mediated knockdown" of "INHBB"

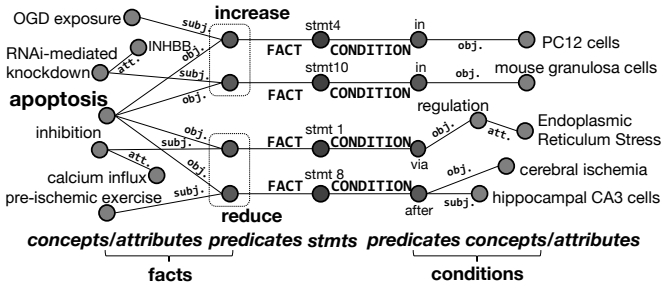

Fig. 5. Our knowledge graph provides a visualized, comprehensive understanding of what increased/reduced "apoptosis" under what kind of conditions.

increased apoptosis, and (2) the "inhibition" of "calcium influx" and "pre-ischemic exercise" reduced apoptosis with the left side of the figure. However, it is important to be aware of the condition for each factual claim on the right side of the figure. They describe either the methodology of the observation (e.g., "in", "via") or the context of it (e.g., "in" a specific cell or "via" a specific regulation). This BioKG will enable effective biomedical knowledge inference and reasoning.

The proposed BioKG has a good use for knowledge retrieval. Here we present some **Q**uestions about biomedical facts and their conditions, to investigate whether our BioKG can provide useful **A**nswers. We did not use any intelligent QA systems, which has been out of our topic, while we manually transform the questions into the simple retrieval queries. Here we list random selected returned results.

Q1. What facts *using flow cytometry*? (SELECT statement WHERE condition=[*, using, flow cytometry])

A1. ***facts:*** [{TRAIL specific death receptors: evaluation}, were performed on, tumours], [{TRAIL specific death receptors: evaluation}, were performed on, haematological malignant lines], ***conditions:*** [NIL, using, flow cytometry] (**Statement.** *Evaluation of TRAIL specific death receptors were performed on both tumours and haematological malignant lines using flow cytometry.*)

A2. ***facts:*** [HLA-DR(+)CD38(+) T cells, were measured in, blood], [HLA-DR(+)CD38(+) T cells, were measured in, cervical cells], ***conditions:*** [NIL, using, flow cytometry] (**Statement.** *HLA-DR(+)CD38(+) T cells were measured in blood and cervical cells using flow cytometry.*)

A3. ***facts:*** [CD4 Treg cells, were detected before, treatment], ***conditions:*** [NIL, using, flow cytometry], [CD4 Treg cells, in, peripheral blood] (**Statement.** *Using flow cytometry , the CD4 Treg cells in the peripheral blood of 22 MM patients were detected before and after treatment.*)

Q2. What experiments need to be done *by targeting certain proteins*? (SELECT statement WHERE condition=[*, by targeting, PTEN])

A1. ***facts:*** [miR-205, regulates, A549 cells], ***conditions:*** [NIL, by targeting, PTEN] (**Statement.** *miR-205 regulates A549 cells proliferation by targeting PTEN.*)

A2. ***facts:*** [miR-205, inhibits, DNA damage repair], ***conditions:*** [NIL, by targeting, ZEB1], [NIL, by targeting, ubiquitin-conjugating enzyme Ubc13] (**Statement.** *Moreover , miR-205 inhibits DNA damage repair by targeting ZEB1 and the ubiquitin-conjugating enzyme Ubc13.*)

A3. ***facts:*** [MIR137, regulates, starvation-induced autophagy] ***conditions:*** [NIL, by targeting, ATG7] (**Statement.** *MIR137 Regulates starvation-induced autophagy by Targeting ATG7.*)

Q3. What *increases or reduces cell proliferation (facts)*, and under what conditions? (SELECT statement WHERE fact=[*, increase, cell proliferation] or fact=[*, reduce, cell proliferation])

A1. ***facts:*** [VPA treatment, increased, cell proliferation], ***conditions:*** [NIL, using, in vitro pre-mature senescence model] (**Statement.** *Using an in vitro pre-mature senescence model , we found that VPA treatment increased cell proliferation and inhibited apoptosis through the suppression of the p16/p21 pathway.*)

A2. ***facts:*** [{HDLs: incubation}, increased, cell proliferation] ***conditions:*** [{HDLs: incubation}, from, des-fluoro-anacetrapib-treated animals] (**Statement.** *Incubation of HDLs from the des-fluoro-anacetrapib-treated animals with human coronary artery endothelial cells increased cell proliferation and migration relative to control.*)

A3. ***facts:*** [Chlorin e6-PDT, reduced, cell proliferation], ***conditions:*** [cell proliferation, in, different oral cancer], [cell proliferation, in, endothelial cells], [Chlorin e6-PDT, in combination with, nimotuzumab], [Chlorin e6-PDT, in combination with, cetuximab] (**Statement.** *Chlorin e6-PDT in combination with nimotuzumab and cetuximab reduced cell proliferation in different oral cancer and endothelial cells.*)

Q4. What *oxidative stress induces or causes (facts)*, and under what conditions? (SELECT statement WHERE fact=[Oxidative stress, induce, *] or fact=[Oxidative stress, cause, *])

A1. ***facts:*** [Oxidative stress, induces, {melanocyte growth: secretion}], ***conditions:*** [Oxidative stress, caused by, UVB] (**Statement.** *Oxidative stress caused by UVB induces the secretion of melanocyte growth and activating factors from keratinocytes , which results in the formation of cutaneous hyperpigmentation.*)

A2. ***facts:*** [Oxidative stress, can cause, {leukocytic DNA: damage}], [Oxidative stress, can cause, {homocysteine level: enhancement}], ***conditions:*** [homocysteine level, in, type 2 diabetic patients] (**Statement.** *Oxidative stress can cause damage to leukocytic DNA and enhancement of homocysteine level in sera of type 2 diabetic patients.*)

A3. ***facts:*** [Oxidative stress, can cause, injury], ***conditions:*** [injury, in, retinal endothelial cells] (**Statement.** *Oxidative stress can cause injury in retinal endothelial cells.*)

5 Related Work

In this section, we review the literature in four related topics: Scientific Knowledge Graphs, Scientific Information Extraction, OpenIE, and Sequence Labeling.

Scientific Knowledge Graphs. SciKGs have been constructed broadly in sciences such as biomedicine [1] and computer science [16][27]. Life-iNet [1] consists of life-science domain concepts (e.g. genes, diseases and drugs) and their relationships such as "may treat" between diseases and drugs. However, the edges of the graph represent co-occurrence of two concepts instead of concrete relation. Luan *et al.* [16] constructed a large-scale SciKG of computer

science in which the nodes are domain concepts (e.g. methods, metrics and datasets) and edges are their relations(e.g. "used for", "evaluated by"), requiring a pre-defined relation schema. However, these existing SciKGs employ the same flat representation as general KGs and ignore the conditions when being constructed from text. Our proposed a three-layer biomedical knowledge graph that contains facts as well as conditions of the facts being observed and valid.

Scientific Information Extraction. Information extraction in scientific literature, e.g., computer science, biology and chemistry, has been receiving much attention in recent years. ScienceIE in computer science focus on concept recognition and factual relation extraction [28][29][18]. ScienceIE in biological literature aims at identifying the relationships between biological concepts (i.e., proteins, diseases, drugs and genes) [30][31][32]. Rule-based approaches were used in early studies [33][30]. Recently, a wide line of neural network models have been proposed and outperformed traditional methods [34][31][32][35]. Wang *et al.* [34] investigated different kinds of word embeddings on different NLP tasks in the biological domain. Liu *et al.* [31] employed attention-based neural networks to extract chemical-protein relations. Xu *et al.* [32] used the BiLSTM model to recognize the drug interaction. In our work, we extract biological relational facts as well as their conditions. The condition tuples are essential to interpreting the factual claims.

OpenIE Systems. OpenIE systems are proposed to extend information extraction to open domains without requiring any relation-specific schema in advance [19][11][36][37]. Fact tuple, i.e. (subject, relation, object), is the main extraction unit of OpenIE systems. Distant supervision has been used in early systems due to the lack of standard benchmarks. Current systems prefer to apply rule-based techniques to extract fact tuples [19]. Stanovsky *et al.* [11] obtained labeled OpenIE data from semantic role labeling, making supervised neural sequence labeling possible for OpenIE. Our representation of biomedical statement sentences is different including modeling attributes and condition tuples.

Sequence Labeling. Neural networks have been applied to sequence labeling tasks with more promising performance than traditional statistical methods. Neural encoder-decoder model is one of the paradigms [38][39][22]. Ma *et al.* [38] combined CNNs-encoded character-level and word-level representations into BiLSTM to model contextual information of each word following a CRF to decode labels for the whole sentence. Yang *et al.* [39] leveraged continuous representations of KBs to enhance LSTM for sequence labeling. Zheng *et al.* [22] proposed BiLSTM-LSTMd which contains a BiLSTM as encoder and a new LSTM-decoder (LSTMd) as decoder. BiLSTM-LSTMd outperformed previous models on entity and relation extraction tasks [22]. We use BiLSTM-LSTMd for the sequence labeling module in our approach.

6 Conclusions

In this work, we proposed a novel representation of BioKG with the role of condition. The BioKG had three layers: concept/attribute nodes, fact/condition tuples and statement nodes. We proposed a Multi-Input Multi-Output sequence labeling model that learned complex dependencies between the sequence tags from multiple signals to generate output sequences for fact/condition tuples. Our model outperformed existing methods in experiments. Case study shows the BioKG is useful for knowledge discovery.

References

[1] X. Ren, J. Shen, M. Qu, X. Wang, Z. Wu, Q. Zhu, M. Jiang, F. Tao, S. Sinha, D. Liem *et al.*, "Life-inet: A structured network-based knowledge exploration and analytics system for life sciences," *ACL*, pp. 55–60, 2017.

[2] P. Ernst, "Biomedical knowledge base construction from text and its applications in knowledge-based systems," 2017.

[3] X. Wang, Y. Zhang, Q. Li, Y. Chen, and J. Han, "Open information extraction with meta-pattern discovery in biomedical literature," in *ACM BCB*, 2018, pp. 291–300.

[4] G. Bakal, P. Talari, E. V. Kakani, and R. Kavuluru, "Exploiting semantic patterns over biomedical knowledge graphs for predicting treatment and causative relations," *Journal of biomedical informatics*, vol. 82, pp. 189–199, 2018.

[5] M. Yahya, S. Whang, R. Gupta, and A. Halevy, "Renoun: Fact extraction for nominal attributes," in *EMNLP*, 2014, pp. 325–335.

[6] W. Xiong, T. Hoang, and W. Y. Wang, "Deeppath: A reinforcement learning method for knowledge graph reasoning," in *EMNLP*, Sep. 2017, pp. 564–573.

[7] Y. Zhang, H. Dai, Z. Kozareva, A. J. Smola, and L. Song, "Variational reasoning for question answering with knowledge graph," in *AAAI*, 2018, pp. 6069–6076.

[8] T. Dettmers, P. Minervini, P. Stenetorp, and S. Riedel, "Convolutional 2d knowledge graph embeddings," in *AAAI*, 2018, pp. 1811–1818.

[9] D. L. Miller, "The nature of scientific statements," *Philosophy of Science*, vol. 14, no. 3, pp. 219–223, 1947.

[10] V. N. Tomilin, A. L. Cherezova, Y. A. Negulyaev, and S. B. Semenova, "Trpv5/v6 channels mediate ca2+ influx in jurkat t cells under the control of extracellular ph," *Journal of cellular biochemistry*, vol. 117, no. 1, pp. 197–206, 2016.

[11] G. Stanovsky, J. Michael, L. Zettlemoyer, and I. Dagan, "Supervised open information extraction," in *NAACL*, 2018, pp. 885–895.

[12] J. Howard and S. Ruder, "Universal language model fine-tuning for text classification," in *ACL*, 2018, pp. 328–339.

[13] M. Labeau, K. Löser, and A. Allauzen, "Non-lexical neural architecture for fine-grained pos tagging," in *EMNLP*, 2015, pp. 232–237.

[14] M. Jiang, J. Shang, T. Cassidy, X. Ren, L. M. Kaplan, T. P. Hanratty, and J. Han, "Metapad: Meta pattern discovery from massive text corpora," in *KDD*. ACM, 2017, pp. 877–886.

[15] J. Shang, J. Liu, M. Jiang, X. Ren, C. R. Voss, and J. Han, "Automated phrase mining from massive text corpora," *TKDE*, vol. 30, no. 10, pp. 1825–1837, 2018.

[16] Y. Luan, L. He, M. Ostendorf, and H. Hajishirzi, "Multi-task identification of entities, relations, and coreference for scientific knowledge graph construction," in *EMNLP*, 2018, pp. 3219–3232.

[17] M. Schmitz, R. Bart, S. Soderland, O. Etzioni *et al.*, "Open language learning for information extraction," in *EMNLP*, 2012, pp. 523–534.

[18] Y. Luan, L. He, M. Ostendorf, and H. Hajishirzi, "Multi-task identification of entities, relations, and coreferencefor scientific knowledge graph construction," in *EMNLP*, 2018.

[19] G. Angeli, M. J. Johnson Premkumar, and C. D. Manning, "Leveraging linguistic structure for open domain information extraction," in *ACL*, Jul. 2015, pp. 344–354.

[20] P. Zhou, W. Shi, J. Tian, Z. Qi, B. Li, H. Hao, and B. Xu, "Attention-based bidirectional long short-term memory networks for relation classification," in *ACL*, 2016, pp. 207–212.

[21] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," in *NAACL*, 2019.

[22] S. Zheng, F. Wang, H. Bao, Y. Hao, P. Zhou, and B. Xu, "Joint extraction of entities and relations based on a novel tagging scheme," in *ACL*, 2017, pp. 1227–1236.

[23] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *CVPR 2016*, 2016, pp. 770–778.

[24] I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun, "Large margin methods for structured and interdependent output variables," *Journal of machine learning research*, vol. 6, no. Sep, pp. 1453–1484, 2005.

[25] J. D. Lafferty, A. McCallum, and F. C. N. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," in *ICML*, 2001, pp. 282–289.
[26] J. Pennington, R. Socher, and C. Manning, "Glove: Global vectors for word representation," in *EMNLP*, 2014, pp. 1532–1543.
[27] W. Yu, Z. Li, Q. Zeng, and M. Jiang, "Tablepedia: Automating pdf table reading in an experimental evidence exploration and analytic system," in *The World Wide Web Conference*, 2019.
[28] Y. Luan, M. Ostendorf, and H. Hajishirzi, "Scientific information extraction with semi-supervised neural tagging," in *EMNLP*, Copenhagen, Denmark, Sep. 2017, pp. 2641–2651.
[29] K. Gábor, D. Buscaldi, A.-K. Schumann, B. QasemiZadeh, H. Zargayouna, and T. Charnois, "Semeval-2018 task 7: Semantic relation extraction and classification in scientific papers," in *SemEval*, 2018, pp. 679–688.
[30] N. Kang, B. Singh, Z. Afzal, E. M. van Mulligen, and J. A. Kors, "Using rule-based natural language processing to improve disease normalization in biomedical text," *Journal of the American Medical Informatics Association*, vol. 20, no. 5, pp. 876–881, 2012.
[31] S. Liu, F. Shen, R. Komandur Elayavilli, Y. Wang, M. Rastegar-Mojarad, V. Chaudhary, and H. Liu, "Extracting chemical–protein relations using attention-based neural networks," *Database*, vol. 2018, p. bay102, 2018.
[32] B. Xu, X. Shi, Z. Zhao, and W. Zheng, "Leveraging biomedical resources in bi-lstm for drug-drug interaction extraction," *IEEE Access*, vol. 6, pp. 33 432–33 439, 2018.
[33] T. C. Rindflesch and M. Fiszman, "The interaction of domain knowledge and linguistic structure in natural language processing: interpreting hypernymic propositions in biomedical text," *Journal of biomedical informatics*, vol. 36, no. 6, pp. 462–477, 2003.
[34] Y. Wang, S. Liu, N. Afzal, M. Rastegar-Mojarad, L. Wang, F. Shen, P. Kingsbury, and H. Liu, "A comparison of word embeddings for the biomedical natural language processing," *Journal of biomedical informatics*, vol. 87, pp. 12–20, 2018.
[35] T. Jiang, T. Zhao, B. Qin, T. Liu, N. V. Chawla, and M. Jiang, "The role of "condition": A novel scientific knowledge graph representation and construction model," in *KDD*. ACM, 2019, pp. 1634–1642.
[36] Q. Li, M. Jiang, X. Zhang, M. Qu, T. P. Hanratty, J. Gao, and J. Han, "Truepie: Discovering reliable patterns in pattern-based information extraction," in *KDD*. ACM, 2018, pp. 1675–1684.
[37] X. Wang, H. Zhang, Q. Li, Y. Shi, and M. Jiang, "A novel unsupervised approach for precise temporal slot filling from incomplete and noisy temporal contexts," in *The Web Conference*, 2019.
[38] X. Ma and E. Hovy, "End-to-end sequence labeling via bi-directional lstm-cnns-crf," in *ACL*, 2016, pp. 1064–1074.
[39] B. Yang and T. Mitchell, "Leveraging knowledge bases in lstms for improving machine reading," in *ACL*, 2017, pp. 1436–1446.

Tianwen Jiang is currently a Ph.D. candidate at Harbin Institute of Technology, Harbin, China. He received his B.S. degree from Harbin Institute of Technology in 2016. His research focuses on knowledge mining and structuring from massive text corpora.

Qingkai Zeng is currently a Ph.D. student at University of Notre Dame. He received his M.S. degree from University of Illinois at Urbana-Champaign in 2017. He received his B.S. degree from Sun Yat-sen University in 2015. His research focuses on scientific information extraction.

Tong Zhao is currently a Ph.D. student at University of Notre Dame. He received his B.S. degree from Case Western Reserve University in 2017. His research interests include computational behavior modeling, anomalous behavior detection, graph mining, and graph neural networks.

Bing Qin received her Ph.D Degree in 2005 from the Department of Computer Science, Harbin Institute of Technology, Harbin, China. She is a full professor of Department of Computer Science, and the deputy director of Research Center for Social Computing and Information Retrieval (HIT-SCIR) from Harbin Institute of Technology. Her research interests include natural language processing, information extraction, document-level discourse analysis and sentiment analysis.

Ting Liu received his Ph.D Degree in 1998 from the Department of Computer Science, Harbin Institute of Technology, Harbin, China. He is a full professor of Department of Computer Science, and the director of Research Center for Social Computing and Information Retrieval (HIT-SCIR) from Harbin Institute of Technology. His research interests include information retrieval, natural language processing and social media analysis.

Nitesh Chawla PhD is the Frank Freimann Professor of Computer Science and Engineering, Director of Data Inference Analysis and Learning Lab (DIAL), and Director of the Interdisciplinary Center for Network Science and Applications (iCeNSA). He started his tenure-track position at Notre Dame in 2007, and was promoted and tenured in 2011, and chaired full professor in 2015. His research is focused on machine learning, data science, and network science. He is the recipient of the 2015 IEEE CIS Outstanding Early Career Award; the IBM Watson Faculty Award, the IBM Big Data and Analytics Faculty Award, National Academy of Engineering New Faculty Fellowship, and his PhD dissertation also received the Outstanding Dissertation Award.

Meng Jiang received his B.E. degree and Ph.D. degree in 2010 and 2015 at the Department of Computer Science and Technology in Tsinghua University. He is now an assistant professor in the Department of Computer Science and Engineering at the University of Notre Dame. He worked as a postdoctoral research associate at University of Illinois at Urbana-Champaign from 2015 to 2017. He has published over 20 papers on behavior modeling and information extraction in top conferences and journals of the relevant field such as IEEE TKDE, ACM SIGKDD, AAAI, ACM CIKM and IEEE ICDM. He also has delivered six tutorials on the same topics in major conferences. He got the best paper finalist in ACM SIGKDD 2014.