The Role of "Condition": A Novel Scientific Knowledge Graph Representation and Construction Model

Tianwen Jiang[‡,†], Tong Zhao[†], Bing Qin[‡], Ting Liu[‡], Nitesh V. Chawla[†], Meng Jiang[†]

[‡]Research Center for Social Computing and Information Retrieval, Harbin Institute of Technology, China
[†]Department of Computer Science and Engineering, University of Notre Dame, Notre Dame, IN 46556, USA
twjiang@ir.hit.edu.cn,tzhao2@nd.edu,{bqin,tliu}@ir.hit.edu.cn,{nchawla,mjiang2}@nd.edu

ABSTRACT

Conditions play an essential role in scientific observations, hypotheses, and statements. Unfortunately, existing scientific knowledge graphs (SciKGs) represent factual knowledge as a flat relational network of concepts, as same as the KGs in general domain, without considering the *conditions* of the facts being valid, which loses important contexts for inference and exploration. In this work, we propose a novel representation of SciKG, which has three layers. The first layer has concept nodes, attribute nodes, as well as the attaching links from attribute to concept. The second layer represents both fact tuples and condition tuples. Each tuple is a node of the relation name, connecting to the subject and object that are concept or attribute nodes in the first layer. 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. We design a semi-supervised Multi-Input Multi-Output sequence labeling model that learns complex dependencies between the sequence tags from multiple signals and generates output sequences for fact and condition tuples. It has a self-training module of multiple strategies to leverage the massive scientific data for better performance when manual annotation is limited. Experiments on a data set of 141M sentences show that our model outperforms existing methods and the SciKGs we constructed provide a good understanding of the scientific statements.

CCS CONCEPTS

• **Information systems** → **Data mining**; Information extraction.

KEYWORDS

scientific knowledge graph, information extraction, conditional statement, sequence labeling

ACM Reference Format:
Tianwen Jiang, Tong Zhao, Bing Qin, Ting Liu, Nitesh V. Chawla, Meng Jiang. 2019. The Role of "Condition": A Novel Scientific Knowledge Graph Representation and Construction Model. In *The 25th ACM SIGKDD Conference on Knowledge Discovery & Data Mining (KDD'19), August 4–8, 2019, Anchorage, AK, USA.* ACM, NY, NY, USA, 9 pages. https://doi.org/10.1145/3292500.3330942

KDD '19, August 4–8, 2019, Anchorage, AK, USA

ACM ISBN 978-1-4503-6201-6/19/08…$15.00
https://doi.org/10.1145/3292500.3330942

1 INTRODUCTION

Since search engines could no longer satisfy the needs of scientist on exploring the literature [5], the idea of constructing scientific knowledge graphs (SciKGs) has been recently brought into attention [4, 12, 24]. The representation of KGs in the general domain is extended for sciences such as physics, chemistry, and biology. For example, disease-gene associations were represented as relational links between a disease node and a gene node [16]. The KG construction model extracts fact tuples in the format of (subject, relation, object) from massive corpora [26] and transforms them into links for reasoning [25, 29] and inference [2].

Conditions play an essential role in scientific statements: without the conditions that were precisely given by scientists, the facts might no longer be valid [14]. Unfortunately, existing SciKGs employ the same flat representation as general KGs and ignore the conditions when being constructed from text. For example, given a sentence below from a biomedical publication:

> *"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 SciKG should represent not only fact tuples but also condition tuples.*

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 scientific statements could be either concepts or concepts' attributes. For example, given the following sentence:

> *"We observed that ... alkaline pH increases the activity of TRPV5/V6 channels in Jurkat T cells."* [21]

existing information extraction systems would extract the following tuple as a unit of factual knowledge in SciKG [19]:

> (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 focus of the fact's object;
- the condition "TRPV5/V6 channels in Jurkat T cells" of the observation was not structured from the text.

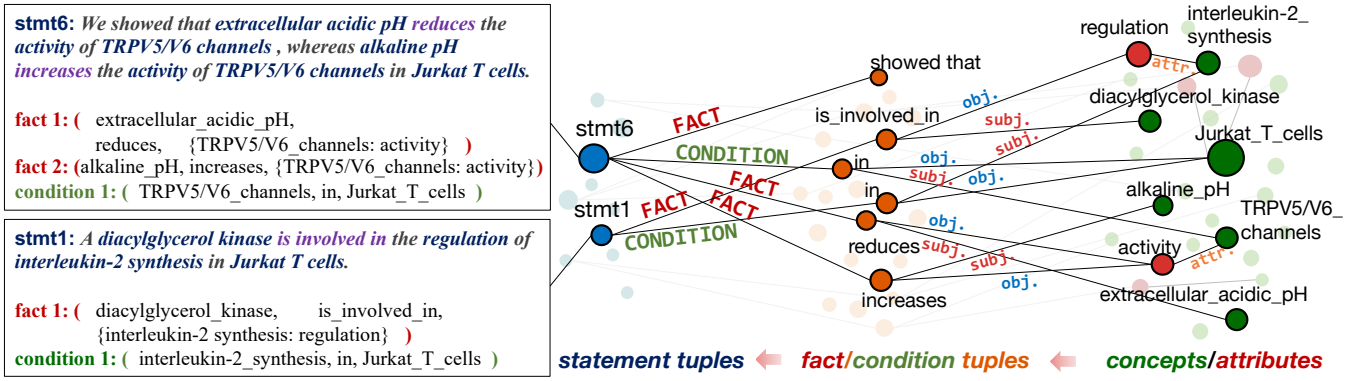


Figure 1: The proposed scientific knowledge graph representation has three layers: the bottom layer (right) has concept and attribute nodes; the middle layer has (subject, predicate, object)-tuples of facts and conditions; the top layer (left) is linking to the (conditional) statement sentences in the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

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 SciKG should represent not only concepts but also attributes.*

In this work, we propose a novel representation for structuring scientific 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 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 SciKG'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 green and red 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 SciKG, or say, extracting fact and/or condition tuples from scientific text, is non-trivial. Inspired by [19] 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 ∈ {fact, condition};
- Y ∈ {1: subject; 2: relation; 3: object};
- Z ∈ {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 three challenges:

C1: One token may have different tags in different tuples. For 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.

C2: Annotation is expensive and distant supervision is unavailable. It takes long for domain experts to annotate scientific 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, KBs are not widely available for sciences yet. It requires a design for training an effective model with a limited amount of labels and massive unlabelled documents.

C3: Noise exists in sequence tags when being transformed into tuple structures. Due to lack of training examples (C2), learning models can hardly secure that sequence tags are correctly positioned to composite tuple units such as concepts, attributes, subjects, objects, and relations. It is important to de-noise the predicted tags.

To address C1–3, we propose a **semi-supervised Multi-input Multi-output (MIMO) sequence labeling** model for scientific knowledge graph construction. It has the following novel design:

First, the model adopts a multi-task scheme that simultaneously generates tag sequences for fact tuples and condition tuples. The subtasks share the same encoder-decoder models [13, 27, 30] but use different linear-softmax layers for predictions.

Second, due to limited annotations, we seek for effective features. Thanks to high efficiency and satisfactory accuracy of fundamental NLP tasks such as Language Model (LM) [7], Part-of-Speech (POS) tagging [9], Concept detection, Attribute discovery, and Phrase mining (CAP) [8, 18, 24], results of each task are used as an input sequence along with the original sentence. The model has multi-input gates and ensembles to utilize complementary signals from the input sequences of the upstream tasks for learning complex dependencies between the 21 sequence tags.

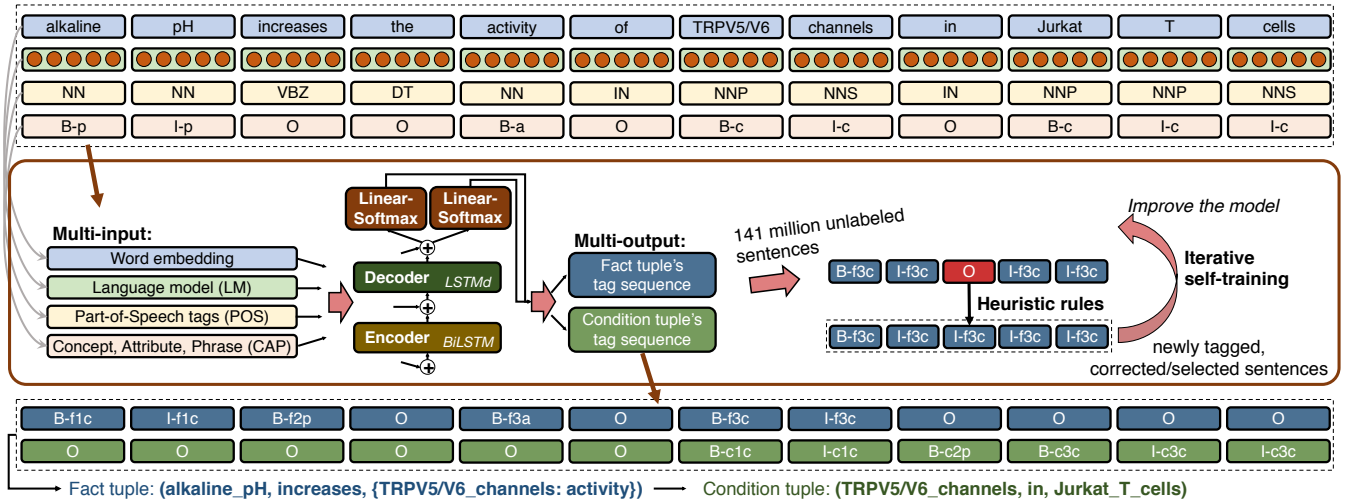


Figure 2: Our semi-supervised approach has two modules: one is a novel Multi-Input Multi-Output sequence labeling model, the other is an iterative self-training method based on heuristic rule correction. Multi-input sequences are given at the top; and the tag sequences are given at the bottom, which can be converted into the fact and condition tuples. (Best viewed in color.)

Third, our model has an iterative self-training module. For each iteration of training and prediction, the highly confident predicted tags are added into the training set to re-train the model. Multiple strategies are employed to secure the quality of the extended training sets via correcting or deleting commonly wrong patterns.

Experiments show that the proposed model improves the F1 score relatively by 33.3% and 12.5% over competitive baselines on tag prediction and tuple extraction, respectively. The SciKG we constructed has as many as 18.1M fact tuples, 7.5M condition tuples, 10.9M concept nodes, and 703K attribute nodes.

We highlight our contributions in this work as follows.

- **A novel SciKG representation:** The new structure represents facts and conditions in scientific statements. The SciKG has three layers: statement layer, fact/condition tuple layer and concept/attribute layer.
- **A novel SciKG construction model:** We propose a semi-supervised multi-input multi-output sequence labeling model for tag prediction and tuple extraction. It takes advantages of upstream NLP techniques and the big data volume.
- **Effectiveness, efficiency, and real use:** The proposed model outperforms baseline methods on a huge literature data. The constructed SciKG contains millions of facts and conditions.

The rest of this paper is organized as follows. Section 2 presents the problem definition. The proposed framework is given in Section 3. Experimental results are provided and analyzed in Section 4. Section 5 surveys the literature and Section 6 concludes the paper.

2 PROBLEM DEFINITION

In this section, we formally define fact tuple, condition tuple, structured statement, and the three-layer SciKG representation. Then we introduce how the SciKG 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 scientific 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 at the bottom of the first page in the introduction.

Definition 3 (Structured Statement). A scientific 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 statement 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 SciKG). It organizes concepts/attributes, facts/conditions, and statements in a bottom-up manner.
A SciKG 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 SciKG, especially when the nodes are mentioned with their colors and the edges are mentioned with their tags.

• 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 *green* concept node $c \in \mathcal{C}$ to the

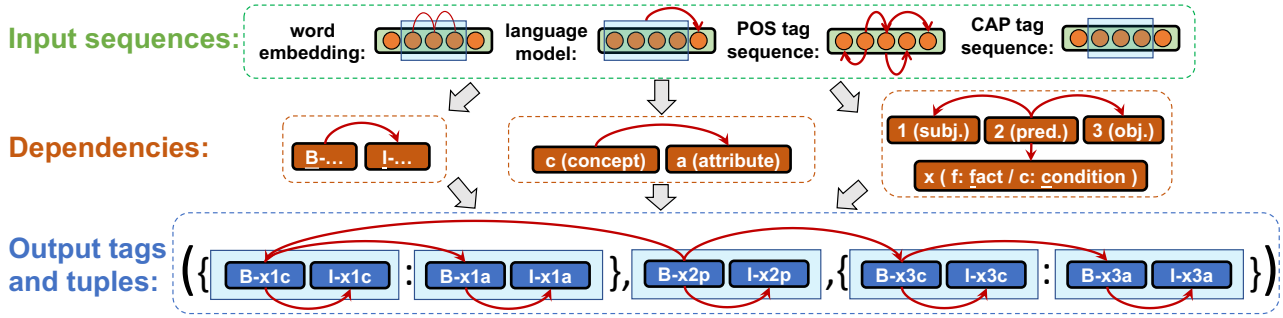


Figure 3: Dependencies between B-/I- tags, Concept/Attribute tags, Subject/Predicate/Object tags, and Fact/Condition tags can be learned from the four types of input signals, being combined to model complex dependencies between the expected tags.

red 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 *blue* 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 *SciKG 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 $\mathbf{w} = (w^1, \ldots, w^N)$ (i.e., a statement sentence), the outputs are multiple tag sequences, denoted by $\mathbf{y}_t = (y_t^1, \ldots, y_t^N)$ 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 $\mathbf{y}_t$, when (a) the tokens of each unit in t can be located in $\mathbf{w}$ as a consecutive subsequence 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 $= \mathbf{w}^{[j^u:k^u]}$, where $1 \le j^u \le k^u \le N$ if u is not "null". If constraint (b) is true, we have

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

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 blue original token sequence at the top, the fact/condition tuples at the bottom can be transformed into two tag sequences (dark blue and green).

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 SciKG.

3 THE PROPOSED APPROACH

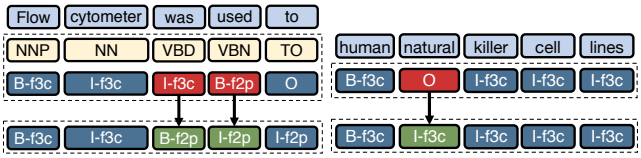
Figure 2 illustrates an overview of our proposed approach. It has two modules: one is a novel multi-input multi-output (MIMO) sequence labeling model (on the left-hand in the middle) and the other is an iterative self-training scheme for semi-supervised learning (on the right-hand). In this section, we first present the encoder-decoder model which is the core of the MIMO sequence labeling, and then introduce the multi-output and multi-input inventions. Finally, the self-training part will be presented in details.

3.1 Encoder-Decoder Model

We use the BiLSTM-LSTMd model that was proposed by Zheng *et al.* in 2017 [30]. Such a model in an end-to-end tagging scheme has been demonstrated to be effective on entity and relation extraction from New York Times corpus. This model has a bidirectional LSTM (BiLSTM) layer (the brown block in Figure 2) to encode the input sequence and an LSTMd block as decoder layer (the dark green block) for sequence tagging.

3.2 Multi-Output Model

We extend the well-accepted model design of sequence labeling, which is one-input one-output, to generate multiple outputs. Each output tag sequence corresponds to a fact/condition tuple. Here the fact tagging and condition tagging use the same encoder-decoder model to mutually enhance each other from the shared contexts. They use different linear-softmax layers to predict the concrete tags for facts and conditions, respectively. The linear layers perform a linear transformation (ω, $\mathbf{b}$) to the encoder-decoder model's output



(a) Association rule-based correction: ("VBD", "VBN") → ("B-f2p", "I-f2p")

(b) Tag consistency-based correction: ("B", "O", "I", "I") → ("B", "I", "I", "I")

Figure 4: Two tag correction strategies to secure the reliability of newly tagged data for iterative self-training.

given the i-th token w^i:

$$x^i_{t^{(f)}} = \omega^{(f)} \cdot Decoder(Encoder(w^i)) + \mathbf{b}^{(f)}, \quad (4)$$

$$x^i_{t^{(c)}} = \omega^{(c)} \cdot Decoder(Encoder(w^i)) + \mathbf{b}^{(c)}, \quad (5)$$

where "(f)" is for fact and "(c)" for condition. The softmax layers project x to the predict tag y with a probability:

$$p(y^i_{t^{(f)}} = y) = \frac{exp(x^{i,y}_{t^{(f)}})}{\sum_{y' \in \mathcal{Y}^{(f)}} exp(x^{i,y'}_{t^{(f)}})}, \quad (6)$$

$$p(y^i_{t^{(c)}} = y) = \frac{exp(x^{i,y}_{t^{(c)}})}{\sum_{y' \in \mathcal{Y}^{(c)}} exp(x^{i,y'}_{t^{(c)}})}, \quad (7)$$

where $\mathcal{Y}^{(f)}, \mathcal{Y}^{(c)} \subset \mathcal{Y}$ are subsets of tags that are related to fact or condition only, respectively, and each has 11 tags including "O".

3.3 Multi-Input Model

To make the model more effective on learning complex dependencies between the output tags (in $\mathcal{Y}$) from training sequence pairs, instead of fully relying on word embedding, we design a multi-input model fed with results from multiple fundamental NLP tasks. Thanks to the availability of the tools, satisfactory performance, as well as efficiency on large volume of text, each token has effective and complementary features for learning the tag dependencies. The top of Figure 2 presents four input feature sequences we use:

- *Word Embedding* (WE): It encodes the token's semantics into distributed representation by training on aggregated global word-word co-occurrence statistics [15].
- *Language Model* (LM): Neural LMs learn simultaneously the word's feature vector and the joint probability function of word sequences [7]. We employ the renown, bidirectional encoder representations from Transformers (BERT) [3].
- *Part-of-Speech tag* (POS): It assigns a special label to each token in a sentence to indicate grammatical categories [9].
- *Concept detection, Attribute extraction, and Phrase mining* (CAP): When concepts (e.g., "TRPV5/V6 channels") and attributes (e.g., "activity") are detected [8, 24], we assign tags such as "B-c", "I-c", "B-a", and "I-a" to their tokens and make an input sequence of the semantic units. For the rest of the word sequence, we use AutoPhrase [18] to find phrases that were not determined to be concept or attribute yet. We tag the tokens of each phrase with "B-p" and "I-p".

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. We analyze the dependencies that each input sequence contributes to the learning process as follows. Figure 3 visualizes the idea of training a multi-input model.

- *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 [1] can be easily incorporated into the multi-input model.

The model has two designs to effectively integrate the multiple inputs in the learning process.

(1) **Multi-input gates:** It is certainly desired of investigating how to feed the input sequences. Inspired by ResNet [6], we use a mechanism of multi-input gates to the encoder-decoder model. Specifically, we add input gates to the following three positions of the encoder-decoder model: the input of BiLSTM encoder, the input of LSTMd decoder, and the linear-softmax layers. These multiple input sequences flow through the shortcut connections to the model.

(2) **Multi-input ensembles:** We observe that for different sentences, the input sequences may have different predictability on the tags: For short sentences, POS and CAP are often more useful; for long sentences, LMs play a more important role. In order to secure the robustness of the model on any kind of statement sentences, we apply boosting-based ensembles to combine the decisions from each of the models trained by one of the input sequences. The final tag predictions become more stable than simple combinations.

3.4 Securing Iterative Self-Training

To leverage the large volume of unlabeled data, we adopt the iterative self-training scheme for semi-supervised learning. For each iteration, we expand the training set by adding predicted tags on unlabeled sentences. However, due to the noise on the newly tagged sentences, the self-training is vulnerable for error propagation. Therefore, we propose the following strategies to correct the predicted tags and secure the quality of training data expansion.

S1: Association rule-based correction (AR). We use association rule mining to derive interesting rules (of high support and confidence) from the training set. For example, we find that a sequential pattern of POS tags indicates a sequential pattern of output tags: [NNP NN VBD VBN] → [B-f1c, I-f1c, B-f2p, I-f2p]. Figure 4(a) presents how to correct the predicted tags with the rule. The words "flow sytometer was" matches the POS-tag pattern of the rule but the predicted tags do not match the rule's consequence. The tag "B-f2p" was assigned to the later word "used", so we use the rule to assign "B-f2p" to "was" and rectify the tag of "used" to be "I-f2p".

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 Word Embeddings, Language Models, POS tags, and Concept-Attribute-Phrase tags. 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 [19]							-	42.60	38.22	40.29
Stanford OpenIE [1]					-	-	-	47.11	41.62	44.19
Structured SVM [22]					32.68	25.80	28.83 / 32.76, 24.71	47.62	46.15	46.87 / 45.01, 48.72
CRF [10]					60.07	41.92	49.37 / 56.23, 41.87	65.19	62.44	63.78 / 64.07, 63.44
	WE	**LM**	**POS**	**CAP**						
MO	✔				59.60	55.07	57.24 / 62.77, 51.69	65.39	65.92	65.65 / 66.52, 64.78
MIMO	✔	Bert-base [3]			61.74	53.48	57.31 / 62.97, 51.27	64.82	64.84	64.83 / 65.63, 64.02
	✔	Bert-large [3]			54.54	49.43	51.86 / 59.17, 44.18	63.50	62.85	63.17 / 64.69, 61.63
	✔	LSTM [20]			61.94	56.35	59.01 / 65.59, 52.36	67.74	67.92	67.83 / 68.61, 67.00
	✔		✔		63.20	58.37	60.69 / 64.53, 56.60	68.18	68.19	68.18 / 69.70, 66.64
	✔			✔	59.59	59.22	59.40 / 66.04, 52.68	67.07	67.32	67.19 / 68.90, 65.49
	✔	LSTM [20]	✔		63.68	58.94	61.22 / 66.70, 55.57	69.14	69.20	69.17 / **71.64**, 66.68
	✔	LSTM [20]		✔	63.53	58.81	61.08 / 66.93, 55.17	68.63	68.49	68.56 / 69.90, 67.22
	✔		✔	✔	64.35	**60.60**	62.42 / 67.11, **57.63**	68.58	68.81	68.69 / 69.91, 67.47
	✔	LSTM [20]	✔	✔	**65.09**	60.24	**62.57** / **68.23**, 56.83	**69.51**	**70.10**	**69.80** / 71.53, **68.06**

S2: Tag consistency correction (TC) and deletion (TCDEL). Figure 4(b) presents how to correct a tag based on the assumption of tag consistency. Given the predicted tag sequence [B-f3c, O, I-f3c, I-f3c, I-f3c], we are highly confident that the word "natural" should not be tagged as "O" but an "I-f3c" because the tag sequence will be valid to generate a concept in the object of a fact tuple. Another option is to delete the predicted example instead of correcting the tags when tag consistency is violated.

S3: Short sentence only (SH). This is a strategy of example selection, not correction. Predictions on short sentences are more reliable than those on long sentences, when being added into the training set for further model training.

S4: Deleting incomplete sequence (DEL). This is also for example selection. Some tag sequences suffer from incompleteness, for example, they may miss the subject, or object, or relation to formulate a complete tuple. We drop these examples for self-training.

4 EXPERIMENTS

In this section, we first introduce the dataset and experimental settings. Then we present experimental results of tag prediction and tuple extraction. Finally, we give case studies on the SciKG.

4.1 A Scientific Text Dataset

Our dataset has 140.9 million sentences from the abstracts of 15.5 million articles on MEDLINE[1] (a life science and biomedical literature database). We recruited domain experts to annotate facts and conditions on 31 randomly selected documents (336 sentences, 8,048 tokens). The final annotated dataset contains 756 fact tuples and 654 condition tuples.

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 Competitive Methods. We compare our proposed approach with two lines of methods: one is statistical methods for sequence labeling, including:

- *Structured Support Vector Machine (SVM)* [22]: It is an SVM-HMM sequence tagging algorithm that can handle tagging problems with millions of words and millions of features;
- *Conditional random field (CRF)* [10]: It is a type of probabilistic graphical model that can be used to model sequential data, such as labels of words in a sentence;

the other line is OpenIE systems that extract (subject, relation, object)-tuples without considering attributes or conditions:

- *AllenNLP OpenIE* [19]: It is the state-of-the-art supervised OpenIE system which uses a deep sequence labeling model to extract a list of propositions, each composed of a single predicate and arguments.
- *Stanford OpenIE* [1]: It 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.

We implement the proposed approach with different settings to evaluate the effectiveness:

- *MO* vs *MIMO*: Multi-output sequence labeling is the essential feature of the task for fact/condition tuple extraction. Using one or multiple input sequences is optional. *MO* uses only the word sequence (with embeddings) and *MIMO* has multiple.
- Four input sequences: *WE*, *LM*, *POS*, and *CAP*. We compare models that use two, three, or all the sequences as input to demonstrate the effectiveness of the multi-input idea.
- For each model setting, we investigated the settings of multi-input gates and reported the best for sake of space.

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

Table 2: The semi-supervised MIMO achieves higher scores (on Precision, Recall, and F1) than the model w/o self-training. Results are given at the best combinations of heuristic rules applied for each multi-input setting. We find that the association rules (AR) and confident short sentence only (SH) are the most effective strategies. Higher score means better performance.

	Multi-input signals				Self-training strategies					Sequence Tag Prediction (%)			Fact/Condition Tuple Extraction (%)		
	WE	LM	POS	CAP	AR	TC	TCDEL	SH	DEL	P	R	F / F_{Fact}, $F_{Condition}$	P	R	F / F_{Fact}, $F_{Condition}$
MO	✔				✗	✔	✗	✔	✗	61.01	58.05	59.49 / 64.76, 54.19	65.99	66.34	66.16 / 66.75, 65.57
MIMO	✔	✔			✔	✗	✗	✔	✗	63.07	61.61	62.33 / 67.49, 57.16	69.05	69.16	69.11 / 70.41, 67.80
	✔		✔		✔	✔	✗	✔	✗	64.47	63.38	63.92 / 68.65, 59.17	70.23	71.77	70.99 / 71.89, **70.08**
	✔			✔	✔	✗	✔	✔	✗	64.32	62.27	63.27 / 67.59, 58.92	66.95	68.72	67.82 / 69.01, 66.62
	✔	✔	✔		✔	✔	✗	✔	✗	62.58	**65.98**	64.23 / 69.37, 59.08	69.05	70.95	69.99 / 71.75, 68.22
	✔	✔		✔	✔	✗	✔	✔	✔	63.10	65.60	64.32 / 69.79, 58.84	68.38	70.38	69.37 / 71.23, 67.50
	✔		✔	✔	✔	✗	✔	✔	✗	64.07	64.82	64.44 / 69.28, 59.56	68.81	71.31	70.04 / 71.91, 68.16
	✔	✔	✔	✔	✔	✔	✗	✔	✔	**67.37**	64.36	**65.82** / **70.23**, **61.40**	**71.42**	**72.08**	**71.75** / **73.56**, 69.94

- *Self-training* and strategies (*AR*, *TC*, *TCDEL*, *SH*, and *DEL*): Note that the strategies can be applied at the same time. So we investigate all the combinations of the strategies for each multi-input setting. We report which strategies are widely applied and demonstrated to be effective (see Table 2).

4.2.3 Evaluation Methods. We evaluate the above methods on two tasks: (1) sequence tag prediction and (2) tuple extraction. The first task is actually 21-class classification (because the size of tag schema $\mathcal{Y} = 21$). For the second, because it might be too difficult to capture the entire tuple, we evaluated the accuracy at the level of the tuple units (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.

4.3 Results on Tag Prediction and Fact/Condition Tuple Extraction

In this section, we first compare the MIMO model (w/o self-training) of different input settings with baseline methods to demonstrate the effectiveness of (a) the multi-output design and (b) the multi-input design. Table 1 presents the results on the tasks of tag prediction and tuple extraction. Second, we compare the MIMO models that have or do not have the iterative self-training module to demonstrate the effectiveness of (c) the semi-supervised training idea. Table 2 presents the corresponding results. Figure 5 summarizes the comparisons using P-R curve and bar charts.

4.3.1 Effectiveness of Multi-Output Encoder-Decoder Model. As shown in Table 1, the multi-output model (MO) performs better than Structured SVM, CRF, and OpenIE systems. We have three observations. *First*, CRF is the best baseline in the tasks because it models distant dependencies among the tags. *Second*, it improves the F1 score relatively by 15.9% and 2.9% over CRF on tag prediction and tuple extraction, respectively. MO outperforms the traditional statistical methods in the proposed task. *Third*, MO improves the F1 score relatively by 48.6% over the Stanford OpenIE on tuple unit prediction. It is necessary to consider conditions when structuring the statement sentences.

4.3.2 Effectiveness of Leveraging Multiple Input Sequences from Up-Stream NLP Tasks. Generally, the MIMO model that uses all the input sequences (i.e., LM, POS, and CAP) can improve the MO model (a) relatively by 9.2% on precision, 9.4% on recall, and 9.3% on F1 score for predicting the tags, and (b) relatively by 6.3% on precision, 6.3% on recall, and 6.3% on F1 score for extracting tuple units. We analyze the usefulness of each type of input sequences.

Usefulness of LM: From Table 1 we have two observations: (1) for the models that have used POS, or CAP, or both, incorporating the LM sequence will consistently improve the F1 scores on both tag prediction and tuple extraction tasks; (2) the MIMO that uses WE and LSTM-based LM [20] performs better than the one uses WE only, however, neither Bert-base nor Bert-large can improve the scores though they have been renown for the excellent performances on a wide range of NLP tasks [3]. Unfortunately, we did not have strong computational resources to support re-train Bert on the massive scientific corpus. We directly applied the Bert LMs that were trained by Google on general domains and find that the performance would not be satisfactory, which shows the significant difference between the text domains.

Usefulness of POS: Two observations: (1) for the models that have used LM, or CAP, or both, incorporating the POS tag sequence consistently improves the F1 scores; (2) the MIMO that uses WE and POS improves the F1 score relatively by 6.0% on tag prediction and by 3.9% on tuple extraction. 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.

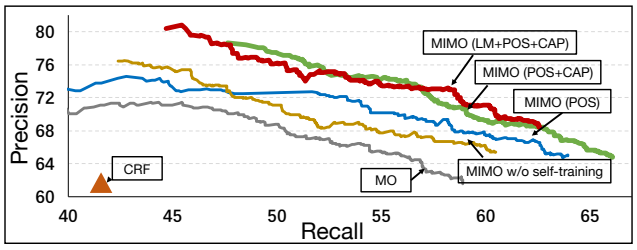
Usefulness of CAP: Table 1 shows the improvement contributed by CAP tag sequences. Given the above example, the CAP seq. is

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

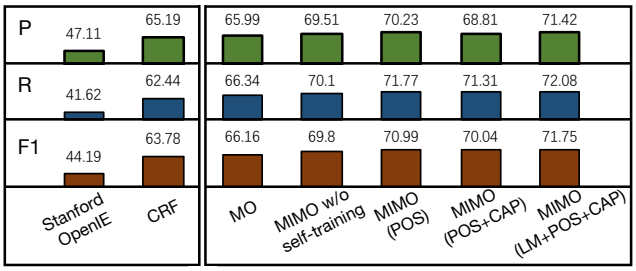
First, the concept "T cell" and attribute "activation", expected to be tagged as "[B-c3c I-c3c]" and "[B-c3a]" eventually, have been tagged as "[B-c I-c]" and "[B-a]" by concept and attribute extraction techniques. *Second*, though "oral epithelial cell-derived cytokines" was not able to be tagged as 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.

Challenges of recognizing conditions: Generally, the F1 scores on predicting condition tags are lower than those on predicting fact tags. For example, the F1 score of predicting the tag "B-c1c" is only 55–58%, while the score of predicting "B-f1c" achieves up to 75%. Recognizing the subject role in a condition needs more information than recognizing that in a fact claim. It requires modeling of global semantic dependencies in the sentence.

4.3.3 Effectiveness of Semi-Supervised Iterative Self-Training. We retrain the MIMO models through 5 iterations. As shown in Table 2, we report the best combination of the strategies. For the MIMO



(a) P-R curves on sequence tag prediction: The semi-supervised MIMOs (the red and green) perform the best. (Best viewed in color.)



(b) The proposed models outperform OpenIE and CRF methods on tuple extraction.

Figure 5: Visualizing performance comparison of competitive methods on (a) tag prediction and (b) tuple extraction.

that uses all the four types of input sequences, this semi-supervised design improves Precision from 65.09 to 67.37, Recall from 60.24 to 64.36, and F1 score from 62.57 to 65.82, on tag prediction; and it improves Precision from 69.51 to 71.42, Recall from 70.10 to 72.08, and F1 score from 69.80 to 71.75, on tuple unit extraction.

We have two observations: *First*, with iterative self-training, the F1 scores of MIMOs were improved relatively by 3.2–6.5% on tag prediction but only 0.8–4.1% on tuple unit extraction. The tuple's unit would be evaluated as correct only when every token's tag was correctly recognized. So, tuple unit extraction is difficult to improve by self-training, without a large volume of labelled data.

Second, we find that *association rule-based correction* (AR) and *short sentence only selection* (SH) are the most effective strategies: All the MIMO models choose to apply these two strategies.

The association rules have two categories: The left-hand-side are sequential patterns of either POS or CAP tag sequence; the right-hand-side are sequential patterns of final tags. We find from the predictions that the CAP-related patterns are often well learned, while the POS tags may still be useful for correcting the newly expanded training set.

SH discarded the sentences whose number of tokens tagged as "O" was above a pre-defined parameter. Only partial information might be recognized by the sequence labeling model. So if put into the training set, they would reduce the model's recall.

We observe from Table 2 that both correction and selection strategies are meaningful. The correction strategies are able to generate correctly labeled, unseen examples. The selection strategies are useful for reducing the amount of noise in the newly labeled data.

Summary of performance comparisons: Figure 5(a) and (b) agree at the point that (1) MO performs better than CRF/OpenIE, (2) Multi-input MO performs better than MO, and (3) semi-supervised MIMO performs better than MIMO w/o self-training.

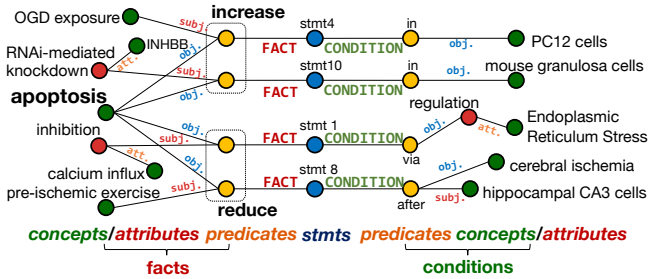


Figure 6: Our knowledge graph provides a visualized, comprehensive understanding of what increased/reduced "apoptosis" under what kind of conditions.

4.4 Efficiency and Case Study on SciKG

Efficiency: Our experiments were conducted with 28 GPU cards (GeForce GTX 1080 Ti) parallel computing. We ran 1,000 epochs for training and 5 iterations of self-training, which took 20.8 hours. Given the 20.4GB corpus data, it took 5.7 hours to extract all the fact/condition tuples for SciKG construction.

Case study: Suppose we are interested in what *increased*/*decreased* "apoptosis". The SciKG provides us a snapshot in Figure 6. We conclude that (1) "OGD exposure" and the "RNAi-mediated knockdown" of "INHBB" 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 SciKG will enable effective scientific knowledge inference and reasoning.

5 RELATED WORK

In this section, we review the literature in three related topics: SciKG, OpenIE, and Sequence Labeling. We point out the uniqueness of our proposed approach compared to each.

Scientific Knowledge Graphs: SciKGs have been constructed broadly in sciences such as biomedicine [16] and computer science [12, 28]. Life-iNet [16] 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.* [12] 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 SciKG that contains scientific facts as well as conditions of the facts being observed and valid.

OpenIE Systems: OpenIE systems are proposed to extend information extraction to open domains without requiring any relation-specific schema in advance [1, 11, 19, 23]. 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 [1]. Stanovsky *et al.* [19] obtained

labeled OpenIE data from semantic role labeling, making supervised neural sequence labeling possible for OpenIE. Our representation of scientific 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 [13, 27, 30]. Ma *et al.* [13] 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.* [27] leveraged continuous representations of KBs to enhance LSTM for sequence labeling. Zheng *et al.* [30] 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 [30]. We use BiLSTM-LSTMd for the sequence labeling module in our approach.

6 CONCLUSIONS

In this work, we proposed a novel representation of SciKG with the role of condition. The SciKG had three layers: concept/attribute nodes, fact/condition tuples and statement nodes. Inspired by a recent work that considers open information extraction as a sequence labeling task, we proposed a semi-supervised Multi-Input Multi-Output (MIMO) sequence labeling model that learned complex dependencies between the sequence tags from multiple signals to generate output sequences for fact/condition tuples. Experiments showed that our model outperformed existing methods.

ACKNOWLEDGMENTS

The authors would like to thank the anonymous reviewers for their valuable comments and insightful suggestions. This work was supported in part by NSF Grants IIS-1447795 and CNS-1622914, and National Natural Science Foundation of China (NSFC) No. 61632011 and No. 61772156.

REFERENCES

[1] Gabor Angeli, Melvin Jose Johnson Premkumar, and Christopher D. Manning. 2015. Leveraging Linguistic Structure For Open Domain Information Extraction. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics (ACL)*. 344–354.

[2] Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. 2018. Convolutional 2D knowledge graph embeddings. In *32nd AAAI Conference on Artificial Intelligence (AAAI)*. 1811–1818.

[3] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*.

[4] Kata Gábor, Davide Buscaldi, Anne-Kathrin Schumann, Behrang QasemiZadeh, Haifa Zargayouna, and Thierry Charnois. 2018. Semeval-2018 Task 7: Semantic relation extraction and classification in scientific papers. In *Proceedings of The 12th International Workshop on Semantic Evaluation (SemEval)*. 679–688.

[5] Sonal Gupta and Christopher Manning. 2011. Analyzing the Dynamics of Research by Extracting Key Aspects of Scientific Papers. In *Proceedings of 5th International Joint Conference on Natural Language Processing (IJCNLP)*. 1–9.

[6] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*. 770–778.

[7] Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. In *Proceedings of the 56rd Annual Meeting of the Association for Computational Linguistics (ACL)*. 328–339.

[8] Meng Jiang, Jingbo Shang, Taylor Cassidy, Xiang Ren, Lance M Kaplan, Timothy P Hanratty, and Jiawei Han. 2017. Metapad: Meta pattern discovery from massive text corpora. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*. ACM, 877–886.

[9] Matthieu Labeau, Kevin Löser, and Alexandre Allauzen. 2015. Non-lexical neural architecture for fine-grained POS tagging. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 232–237.

[10] John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In *Proceedings of the Eighteenth International Conference on Machine Learning (ICML)*. 282–289.

[11] Qi Li, Meng Jiang, Xikun Zhang, Meng Qu, Timothy P Hanratty, Jing Gao, and Jiawei Han. 2018. Truepie: Discovering reliable patterns in pattern-based information extraction. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 1675–1684.

[12] Yi Luan, Luheng He, Mari Ostendorf, and Hannaneh Hajishirzi. 2018. Multi-task identification of entities, relations, and coreference for scientific knowledge graph construction. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 3219–3232.

[13] Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional lstm-cnns-crf. In *Proceedings of the 54rd Annual Meeting of the Association for Computational Linguistics (ACL)*. 1064–1074.

[14] David L Miller. 1947. The Nature of Scientific Statements. *Philosophy of Science* 14, 3 (1947), 219–223.

[15] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 1532–1543.

[16] Xiang Ren, Jiaming Shen, Meng Qu, Xuan Wang, Zeqiu Wu, Qi Zhu, Meng Jiang, Fangbo Tao, Saurabh Sinha, David Liem, et al. 2017. Life-iNet: A Structured Network-Based Knowledge Exploration and Analytics System for Life Sciences. *Proceedings of the 55rd Annual Meeting of the Association for Computational Linguistics (ACL)* (2017), 55–60.

[17] Michael Schmitz, Robert Bart, Stephen Soderland, Oren Etzioni, et al. 2012. Open language learning for information extraction. In *Proceedings of the 2012 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 523–534.

[18] Jingbo Shang, Jialu Liu, Meng Jiang, Xiang Ren, Clare R Voss, and Jiawei Han. 2018. Automated phrase mining from massive text corpora. *IEEE Transactions on Knowledge and Data Engineering (TKDE)* 30, 10 (2018), 1825–1837.

[19] Gabriel Stanovsky, Julian Michael, Luke Zettlemoyer, and Ido Dagan. 2018. Supervised open information extraction. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL)*. 885–895.

[20] Martin Sundermeyer, Ralf Schlüter, and Hermann Ney. 2012. LSTM neural networks for language modeling. In *Thirteenth annual conference of the international speech communication association (ISCA)*.

[21] Victor N Tomilin, Alena L Cherezova, Yuri A Negulyaev, and Svetlana B Semenova. 2016. TRPV5/V6 channels mediate Ca2+ influx in jurkat T cells under the control of extracellular pH. *Journal of cellular biochemistry* 117, 1 (2016), 197–206.

[22] Ioannis Tsochantaridis, Thorsten Joachims, Thomas Hofmann, and Yasemin Altun. 2005. Large margin methods for structured and interdependent output variables. *Journal of machine learning research* 6, Sep (2005), 1453–1484.

[23] Xueying Wang, Haiqiao Zhang, Qi Li, Yiyu Shi, and Meng Jiang. 2019. A Novel Unsupervised Approach for Precise Temporal Slot Filling from Incomplete and Noisy Temporal Contexts. In *The World Wide Web Conference*.

[24] Xuan Wang, Yu Zhang, Qi Li, Yinyin Chen, and Jiawei Han. 2018. Open Information Extraction with Meta-pattern Discovery in Biomedical Literature. In *Proceedings of the 2018 ACM International Conference on Bioinformatics, Computational Biology, and Health Informatics (ACM BCB)*. 291–300.

[25] Wenhan Xiong, Thien Hoang, and William Yang Wang. 2017. Deeppath: A Reinforcement Learning Method for Knowledge Graph Reasoning. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 564–573.

[26] Mohamed Yahya, Steven Whang, Rahul Gupta, and Alon Halevy. 2014. Renoun: Fact extraction for nominal attributes. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 325–335.

[27] Bishan Yang and Tom Mitchell. 2017. Leveraging knowledge bases in lstms for improving machine reading. In *Proceedings of the 55rd Annual Meeting of the Association for Computational Linguistics (ACL)*. 1436–1446.

[28] Wenhao Yu, Zongze Li, Qingkai Zeng, and Meng Jiang. 2019. Tablepedia: Automating PDF Table Reading in an Experimental Evidence Exploration and Analytic System. In *The World Wide Web Conference*.

[29] Yuyu Zhang, Hanjun Dai, Zornitsa Kozareva, Alexander J Smola, and Le Song. 2018. Variational reasoning for question answering with knowledge graph. In *32nd AAAI Conference on Artificial Intelligence (AAAI)*. 6069–6076.

[30] Suncong Zheng, Feng Wang, Hongyun Bao, Yuexing Hao, Peng Zhou, and Bo Xu. 2017. Joint Extraction of Entities and Relations Based on a Novel Tagging Scheme. In *Proceedings of the 55rd Annual Meeting of the Association for Computational Linguistics (ACL)*. 1227–1236.