Joint Entity Linking and Relation Extraction with Neural Networks for Knowledge Base Population

Zhenyu Zhang[†‡], Xiaobo Shu[†‡], Tingwen Liu[†‡*], Zheng Fang[†‡], Quangang Li[†]
[†]*Institute of Information Engineering, Chinese Academy of Sciences.* Beijing, China
[‡]*School of Cyber Security, University of Chinese Academy of Sciences.* Beijing, China
{zhangzhenyu1996, shuxiaobo, liutingwen, fangzheng, liquangang}@iie.ac.cn

***Abstract*—Relation extraction and entity linking are two fundamental procedures to extend knowledge bases. Most existing methods typically treat them separately and ignore the semantic relevance between entities and relations. In this paper, we pioneer a general joint learning framework for relation extraction and entity linking, which allows these two tasks boost each other. Based on the framework, a demonstration model is proposed with neural networks. We conduct experiments on variants of a standard benchmark dataset (NYT-10) to verify the effectiveness of our approach. Experimental results show that our approach significantly outperforms traditional separate methods without reducing efficiency, especially on datasets with many ambiguous entity mentions. Furthermore, various mainstream methods for relation extraction and entity linking can be easily integrated into our loosely-coupled framework due to its flexible architecture.**

***Index Terms*—Joint learning, Relation extraction, Entity linking, Knowledge base population**

I. Introduction

Knowledge Bases (KBs), aiming to organize world knowledge in structural forms, are playing an increasingly important role as infrastructural facilities of natural language processing (NLP). A typical KB is usually represented as a set of triplet facts (h, r, t). Unfortunately, KBs are far from completion, knowledge base population (KBP) is the task of completing the incomplete KBs, and takes relation extraction and entity linking as two fundamental procedures [1], [2].

Relation Extraction (RE), which aims at extracting semantic triplet facts from plain texts, has drawn much attention [3]–[5]. However, entity mentions in these newly extracted facts are usually ambiguous. Our statistical results demonstrate 75% of sentences in the benchmark NYT-10 dataset [6] have ambiguous entities. Fig. 1 shows an example of this phenomenon with two facts: *(Washington, located-in, the USA)* and *(Washington, President-of, the United States)*, in which *Washington* refers to *George Washington* and *Washington(state)* respectively. To integrate these facts into existing KBs, mapping these entity mentions to their corresponding entities is an essential step, which is also named as Entity Linking (EL) [7], [8].

The KBP problem is traditionally solved as a pipeline of these two separate tasks [1], which leads to a drawback that information between entities and relations cannot be fully exploited. With this in mind, J-REED [9], leveraging probabilistic graphical models, was the first attempt to joint RE and EL. However, the relations are extracted from plain

* Corresponding author: Tingwen Liu.

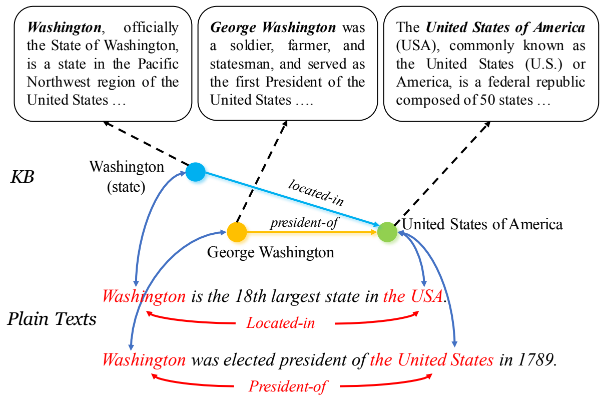
Washington, officially the State of Washington, is a state in the Pacific Northwest region of the United States …
George Washington was a soldier, farmer, and statesman, and served as the first President of the United States ….
The *United States of America* (USA), commonly known as the United States (U.S.) or America, is a federal republic composed of 50 states …
KB
Washington (state)
located-in
president-of
United States of America
George Washington
Plain Texts
Washington is the 18th largest state in the USA.
Located-in
Washington was elected president of the United States in 1789.
President-of

Fig. 1. An illustration of RE and EL. It also shows that entities are ambiguous and each entity in knowledge base corresponds to a description.

texts, making it hard to map the triplet facts to KBs directly. Nevertheless, it still shows that entity information can obviously improve RE, and relations can also be fed back to select correct entities. Another way to take advantage of such interaction is to integrate entity information into RE or vice versa. Specifically, Ji et al. [10] and Ren et al. [3] added entity descriptions to RE and made great progress, but the datasets they used have clear linking annotations, which caused their works great limitation. In reality, only after the EL can we have such anchor texts. Le et al. [11] treated relations as latent variables and showed the benefits of using relations in EL, but it suffered from low support of relation number.

To address these issues, we propose a general joint learning framework (JF-ER) to handle EL and RE in one step. Fig. 2 describes the procedure of KBP using our JF-ER framework. With the help of distant supervision, JF-ER can automatically collect training data from plain texts and extract triplet facts with linked entities and predicted relations, making it easy to be deployed for production use.

Based on the JF-ER, we introduce a demonstration model with neural networks named DM-ER, which can be divided into three modules: *LSTM Encoder*, *Entity Pair Selector* and *Relation Extractor*. Firstly, *LSTM Encoder* is used to encode the input words (sentences and entity descriptions) by taking their contexts into account. Secondly, *Entity Pair Selector* selects correct entity pair from a candidate entity pair list, in which the entity pairs are generated from the target KB for the

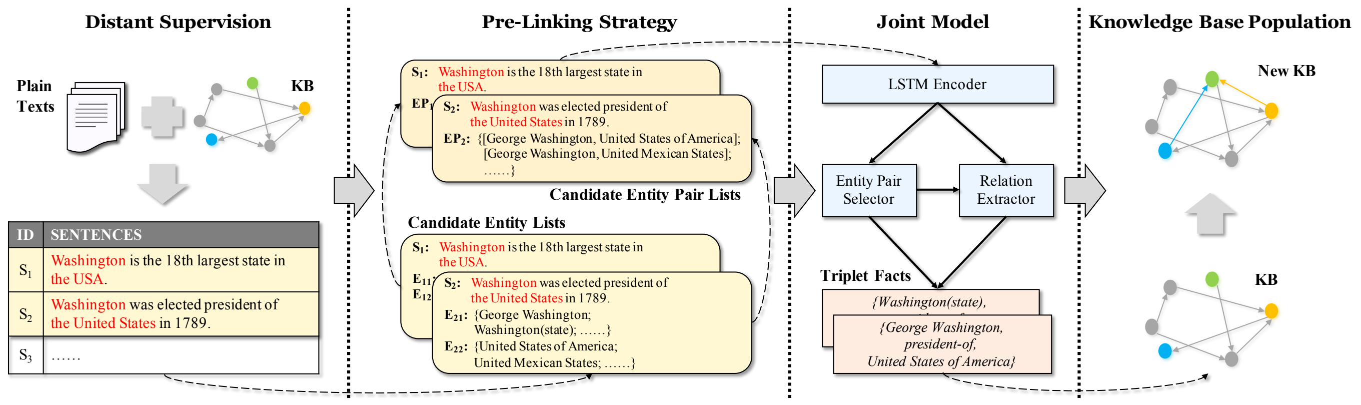
Distant Supervision
Plain Texts
KB
ID
SENTENCES
S1
Washington is the 18th largest state in the USA.
S2
Washington was elected president of the United States in 1789.
S3
……
Pre-Linking Strategy
S1: Washington is the 18th largest state in the USA.
EP1
S2: Washington was elected president of the United States in 1789.
EP2: {[George Washington, United States of America]; [George Washington, United Mexican States]; ……}
Candidate Entity Pair Lists
Candidate Entity Lists
S1: Washington is the 18th largest state in the USA.
E11
E12
S2: Washington was elected president of the United States in 1789.
E21: {George Washington; Washington(state); ……}
E22: {United States of America; United Mexican States; ……}
Joint Model
LSTM Encoder
Entity Pair Selector
Relation Extractor
Triplet Facts
{Washington(state),
{George Washington, president-of, United States of America}
Knowledge Base Population
New KB
KB

Fig. 2. An overview of the knowledge base population (KBP) procedure with our JF-ER framework.

mention pairs in sentences. The major challenge here is that the candidate list is too large for a neural EL model [8], hence we implement a simple and effective *Pre-Linking* strategy to address this problem, which is deployed before DM-ER as an indispensable component of JF-ER. Finally, *Relation Extractor* applies the linking results to enrich mention's information and utilize PCNN-based models [12] to predict relation. The most exciting part is that DM-ER has no extra model parameters, thus our joint framework has comparable efficiency compared with separate methods.

While some of these ideas seem straightforward and intuitive, as far as we know, they have not been sufficiently and systematically studied. To simulate and evaluate the KBP procedure, we construct three variant datasets based on NYT-10 [6] by attaching corresponding EL data to such RE dataset. Note that all modules interact with each other during the training process, which provides a possibility to make full use of the semantic information between entities and relations, thus our framework will work well even if there are many ambiguous mentions in the corpus. Beyond that, each module can be replaced by other advanced models, so that JF-ER can adapt to various scenes and achieve greater performances. Finally, in spite of a similar motivation shared by [9], our work is unique from the following perspectives: (1) relations in JF-ER are pre-defined based on the target knowledge base, which is more suitable for KBP [1]. (2) neural networks are applied to automatically encode sentence semantics, which is a powerful way to model large-scale noisy text.

In summary, We pioneer a joint learning framework and its demonstration model to handle EL and RE in one step. We conduct experiments on real-world dataset, and results indicate that our approaches lead to significant improvements over traditional methods and can vastly relieve the adverse effects of ambiguous mentions. Experimental results also show that our loosely-coupled framework is flexible and most existing methods for EL and RE can be easily integrated into it.

II. Task Definition

Formally, we formulate the EL task as follow: given a sentence and a list of entity mentions, for a mention m, the goal is to select the correct entity e from a candidate entity list $\{e_1, ..., e_{n_m}\}$, where n_m is the number of candidate entities of m. The RE problem is formulated as follow: given the sentence and the mentioned entity pair (m_1, m_2), it aims to predict the semantic relation r between m_1 and m_2. For convenience sake, the EL&RE is regarded as the KBP task empirically.

As to the KBP task, for one sample (see Fig. 2), the inputs of JF-ER contain a sentence s with n words, a mention pair mp and a list of candidate entity pairs $\{ep_1, ..., ep_{n_1 \times n_2}\}$, where $mp = (m_1, m_2)$ and $ep_i = (e_{i1}, e_{i2})$. The output is a relational triplet fact with linked entities and a predicted relation. Moreover, we utilize *Wikipedia* as the source KB.

III. Methodology

We propose a **J**oint Learning **F**ramework for **E**ntity Linking and **R**elation Extraction (JF-ER) to expand KBs accurately and conveniently (Figure 2 shows an overview). In this section, we first introduce our JF-ER framework (Section III-A), then describe the *Pre-Linking* strategy (Section III-B) and the **D**emonstration **M**odel for Joining **E**ntity Linking and **R**elation Extraction (DM-ER) (Section III-C) in detail.

A. The JF-ER Framework

Initially, we employ distant supervision to tag sentences from plain texts [6], and the candidate entity lists can also be obtained from the target KB in the meantime.

For each sentence, considering that entities always exist in the form of pairs after distant supervision and there may be some semantic relations between them [13], we combine two candidate entity lists freely to generate the candidate entity pair list. Unfortunately, according to the disambiguation pages of *Wikipedia*, each mention has tens of candidate entities, which means that each mention pair has hundreds of candidate entity pairs. Under such circumstances, we utilize some simple features and further propose a *Pre-Linking* strategy to reduce the size of the candidate list. Given sentences and their reduced candidate entity pair lists, *LSTM Encoder* generates distributed representations for both sentences and entity descriptions, *Entity Pair Selector* aligns mention pairs with correct candidate entity pairs, *Relation Extractor* uses entity descriptions as external knowledge to predict semantic relations. These three modules will be trained simultaneously, which allows EL and RE to boost each other. Finally, the JF-ER framework outputs relational triplet facts (*h, r, t*), in which (*h, t*) are definite KB entities and *r* is the KB relation, so that these triplet facts can be used for completing the original KB directly.

Based on the architecture, the JF-ER framework can adapt to various mainstream EL and RE approaches, including kinds of sentence encoders with or without sentence selector for RE [12], [14], [15], and most of the existing neural-based methods for EL [13], [16], [17]. That is to say, our JF-ER framework is flexible enough to focus on more complex scenarios with more advanced models, no matter the data is noisy or not.

B. The Pre-Linking Strategy

Following the idea of [13], we aligned two entity mentions to the target KB in one step. However, the number of candidate entity pairs is too large for a neural EL model to handle. For this purpose, we present the *Pre-Linking* strategy to remove some unrelated candidate entity pairs. For each entity mention in sentences, we employ its corresponding vector in the word embedding table as the *mention vector*. For each candidate entity, we sort all words in the description by frequency and take the top 10 of them as entity keywords, then average all keyword vectors to obtain the *entity vector*. Inspired by [18], we employ some trivial features to rank candidate entity pairs, which can be divided into two categories:

1) Similarities: Similarity features can be divided into string similarity and semantic similarities. Firstly, the string similarity $\phi_{ss}(mp, ep_i)$ refers to *revised jaro_winkter similarity* between mention's surface forms and entity titles. To make it consistent with the following features, we change the range of similarity from [0, 1] to [-1, 1]:

$$\phi_{ss}(mp, ep_i) = sim_{jw}(\overline{m}_1, \overline{e}_{i1}) + sim_{jw}(\overline{m}_2, \overline{e}_{i2}) - 1 \quad (1)$$

where $\overline{m}$ indicates the surface form of m and $\overline{e}$ indicates the title of e. $sim_{jw}(s_1, s_2)$ is the *jaro_winkter similarity* between string s_1 and s_2, which is a classical string metric measuring distance between two sequences:

$$sim_{jw}(s_1, s_2) = sim_j(s_1, s_2) + L \times P \times (1 - sim_j(s_1, s_2)) \quad (2)$$

$$sim_j(s_1, s_2) = \begin{cases} 0 & \text{if } M = 0 \\ \frac{1}{3}\left(\frac{M}{|s_1|} + \frac{M}{|s_2|} + \frac{M-T}{M}\right) & \text{otherwise} \end{cases} \quad (3)$$

where $|s_i|$ is the length of s_i, M is the number of overlapping characters and T is the number of transpositions between s_1 and s_2. L is the length of common prefix and its maximum value is 4. P is a constant scaling factor that gives more favorable ratings to prefix and the maximum value is 0.25.

Secondly, semantic similarities include local similarity, global similarity, and maximum category similarity. Among them, the local similarity $\phi_{ls}(mp, ep_i)$ refers to the cosine similarities between mention vectors and their corresponding entity vectors; the global similarity $\phi_{ls}(mp, ep_i)$ refers to the cosine similarities between mention vectors and other entity vectors; the maximum category similarity $\phi_{cs}(mp, ep_i)$ refers to the max similarity between entity category and all words in mention's context. The calculating methods are as follows:

$$\phi_{ls}(mp, ep_i) = [cos(\boldsymbol{m}_1, \boldsymbol{e}_{i1}) + cos(\boldsymbol{m}_2, \boldsymbol{e}_{i2})]/2 \quad (4)$$

$$\phi_{gs}(mp, ep_i) = [cos(\boldsymbol{m}_1, \boldsymbol{e}_{i2}) + cos(\boldsymbol{m}_2, \boldsymbol{e}_{i1})]/2 \quad (5)$$

$$\phi_{cs}(mp, ep_i) = [\digamma(m_1, e_{i1}) + \digamma(m_2, e_{i2})]/2 \quad (6)$$

$$\digamma(m, e) = \begin{cases} 0 & \text{if } \tilde{e} = \varnothing \\ \max\{cos(\boldsymbol{c}_j, \tilde{\boldsymbol{e}})\}, \boldsymbol{c}_j \in \mathcal{C}(m) & \text{otherwise} \end{cases} \quad (7)$$

where $\boldsymbol{m} \in \mathbb{R}^{d_w}$ and $\boldsymbol{e} \in \mathbb{R}^{d_w}$ denote the mention vector and entity vector respectively, and d_w is the size of pre-trained word embeddings. $\mathcal{C}(m)$ represents the context word set of mention m. $\tilde{e}$ denotes the category of entity e, which can be obtained from the entity title (e.g., *state* for *Washington(state)*).

2) Compatibilities: We also measure the entity compatibility $\phi_{ec}(mp, ep_i)$ by computing the similarities between entities and mention's context. Analogously, we define the mention compatibility $\phi_{mc}(mp, ep_i)$ for the similarities between mentions and entity keywords:

$$\phi_{ec}(mp, ep_i) = [\frac{1}{|\mathcal{C}(m_1)|}\sum_{c_{1 \cdot j} \in \mathcal{C}(m_1)} cos(\boldsymbol{c}_{1 \cdot j}, \boldsymbol{e}_{i1}) + \frac{1}{|\mathcal{C}(m_2)|}\sum_{c_{2 \cdot j} \in \mathcal{C}(m_2)} cos(\boldsymbol{c}_{2 \cdot j}, \boldsymbol{e}_{i2})]/2 \quad (8)$$

$$\phi_{mc}(mp, ep_i) = [\frac{1}{|\mathcal{K}(e_{i1})|}\sum_{k_{i1 \cdot j} \in \mathcal{K}(e_{i1})} cos(\boldsymbol{m}_1, \boldsymbol{k}_{i1 \cdot j}) + \frac{1}{|\mathcal{K}(e_{i2})|}\sum_{k_{i2 \cdot j} \in \mathcal{K}(e_{i2})} cos(\boldsymbol{m}_2, \boldsymbol{k}_{i2 \cdot j})]/2 \quad (9)$$

where $\mathcal{K}(e)$ represents the keyword set of entity e.

3) Pre-Linking: Note that the range of values for all the features above is from -1 to 1, and the higher the feature values are, the more similar the mention pair and entity pair is. The score is normalized such that -1 equates to no similarity and 1 is an exact match. Based on the observation, we simply concatenate them as the *pre-score*, which can be denoted as $S_P(mp, ep_i)$:

$$S_P(mp, ep_i) = \sum_{x \in \mathcal{X}} \alpha_x \phi_x(mp, ep_i) \quad (10)$$

where $\mathcal{X}$=$\{ss, ls, gs, cs, ec, mc\}$ and α is the weight of each feature. These weights can be assigned empirically or by some simple machine learning methods (e.g., XGBoost [19]). Finally, only ρ entity pairs with the highest score will be retained and fed into the JF-ER framework as the ultimate candidate entity pair list.

C. The DM-ER Model

According to the JF-ER framework, we propose the DM-ER model to demonstrate how to joint EL and RE with neural networks. Fig. 2 shows that our joint model includes three components: *LSTM Encoder*, *Entity Pair Selector* and *Relation Extractor*. On this foundation, the architecture of DM-ER is further detailed in Fig. 3.

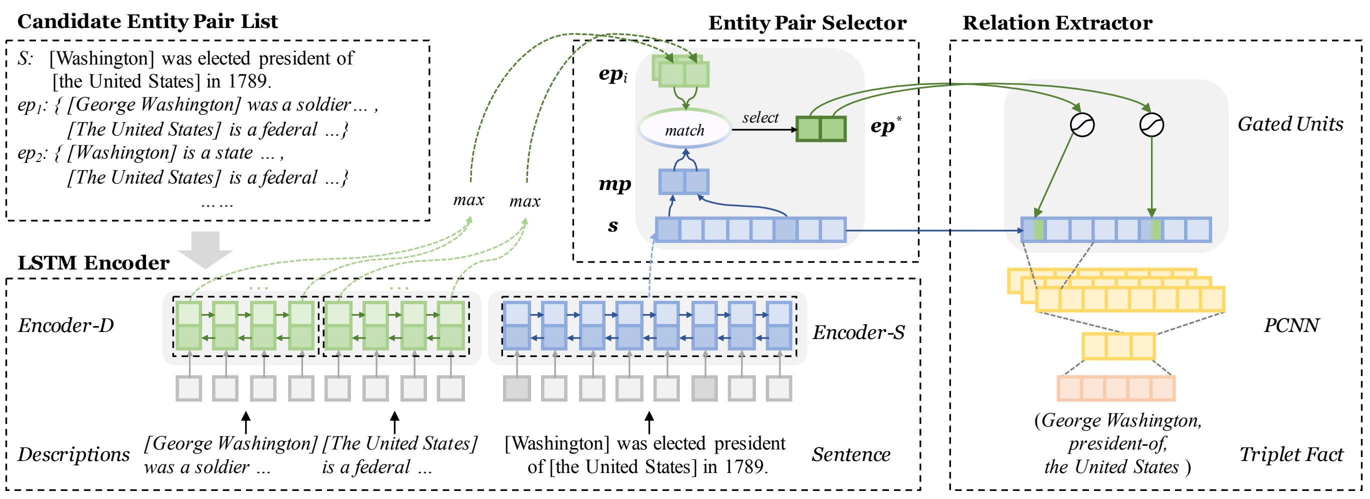
Candidate Entity Pair List
S: [Washington] was elected president of [the United States] in 1789.
ep1: { [George Washington] was a soldier… , [The United States] is a federal …}
ep2: { [Washington] is a state … , [The United States] is a federal …}
LSTM Encoder
Encoder-D
Encoder-S
Descriptions
[George Washington] was a soldier …
[The United States] is a federal …
[Washington] was elected president of [the United States] in 1789.
Sentence
max
max
Entity Pair Selector
epi
match
select
ep*
mp
s
Relation Extractor
Gated Units
PCNN
(George Washington, president-of, the United States)
Triplet Fact

Fig. 3. The architecture of DM-ER model. It has three parts: *LSTM Encoder* encodes sentences and descriptions. *Entity Pair Selector* sorts all candidate entity pairs and selects the pair with maximum score. *Relation Extractor* mixes the chosen entity descriptions into original mentions, predicts the semantic relations and generates triplet facts.

1) LSTM Encoder: Bidirectional LSTM utilizes both previous and future context by processing the sequence in two directions, and generate two independent sequences of LSTM output vectors. One processes the input sequence in the forward direction, while the other processes the input in the reverse direction. The output at each time step is the concatenation of the two output vectors from both directions. For simplicity, we denote the operation of an LSTM unit on $\boldsymbol{x}_i$ as LSTM($\boldsymbol{x}_i$). Thus, the contextualized word representation is obtained as $\boldsymbol{h}_i = [\overrightarrow{\text{LSTM}}(\boldsymbol{x}_i); \overleftarrow{\text{LSTM}}(\boldsymbol{x}_i)], i \in [1, n_s]$ where n_s is the length of sequences (i.e., sentences or entity descriptions), $\boldsymbol{h}_i \in \mathbb{R}^{2\times d_h}$ and d_h indicates the dimension of hidden state for LSTM. In doing so, we can efficiently make use of past features (via forward states) and future features (via backward states) for a specific time.

Finally, *LSTM Encoder* generates distributed representations for both sentences and entity descriptions independently, which are denoted as Encoder-S (Encoder for Sentences) and Encoder-D (Encoder for Descriptions). What we have to mention is that we use the same parameters to encode descriptions for all candidate entities.

2) Entity Pair Selector: The input of this module includes sentences, mention pairs, and candidate entity pair lists, the goal is to select an entity pair ep^* which corresponding to the mention pair from the candidate list. For the output of Encoder-S, vectors which have the same indexes with mentions (m_1, m_2) are considered as mention vectors $(\boldsymbol{m}_1, \boldsymbol{m}_2)$. In addition, we employ a max pooling layer behind Encoder-D to capture all features and semantics in the descriptions, so that we can obtain the entity vectors $(\boldsymbol{e}_{i1}, \boldsymbol{e}_{i2})$. EL is usually treated as a ranking problem and utilizes cosine similarity to measure the distance between mentions and entities [2], [7]. Therefore, we calculate the *joint-score* of each entity pair, then rank all candidate entity pairs by $S_J(mp, ep_i)$, select the i-th pair with the maximum score as ep^* and apply them in the next module, in which:

$$S_J(mp, ep_i) = cos(\boldsymbol{m}_1, \boldsymbol{e}_{i1}) + cos(\boldsymbol{m}_2, \boldsymbol{e}_{i2}) \tag{11}$$

$$i = \arg\max_i S_J(mp, ep_i) \tag{12}$$

Let Θ_{EL} represent all learnable parameters in EL module, the conditional probability of selected entity pair ep^* is:

$$p(ep^*|s, \Theta_{EL}) = \frac{\exp(S_J(mp, ep^*))}{\sum_{i=1}^{\rho} \exp(S_J(mp, ep_i))} \tag{13}$$

3) Relation Extractor: Entity descriptions can provide rich background knowledge for entities and eliminate the ambiguous mentions. However, *Entity Pair Selector* inevitably has some wrong choices. Here, we leverage the gated mechanism to suppress the transmission of error information and update the representation of mention pair (m_1, m_2):

$$\hat{\boldsymbol{m}}_i = (1-\sigma)\boldsymbol{m}_i + \sigma \boldsymbol{e}_i^* \quad i \in \{1, 2\} \tag{14}$$

where σ refers to the weight of entity description, which can be calculated by Equation 13. It is intuitive that, if *Entity Pair Selector* has high confidence in selecting e_i^*, the updated mention vectors will be blended with more information from the selected entities and vice versa.

Position embedding [14] is used to embed the relative distances of each word to the two entity mentions into two d_p-dimensional vectors. For each word in s, we concatenate its original representation and position embedding $\boldsymbol{p}_i \in \mathbb{R}^{2\times d_p}$ to get new embedding $\boldsymbol{h}_i \in \mathbb{R}^{2\times d_p + 2\times d_h}$, where $2 \times d_h$ is the size of original vectors after *LSTM Encoder*.

In DM-ER, we choose PCNN [12] to extract the semantic relations between the two entity mentions as an example. As a widely used baseline, PCNN is an extension of CNN, which slides a convolution kernel with the window size w over the input sequence $\{\boldsymbol{h}_1, ..., \boldsymbol{h}_{n_s}\}$ to update the hidden embeddings:

$$\hat{\boldsymbol{h}}_i = \text{CNN}(\boldsymbol{h}_{i-\frac{w-1}{2}}, ..., \boldsymbol{h}_{i+\frac{w+1}{2}}) \tag{15}$$

A piecewise max-pooling mechanism is applied over the hidden embeddings:

$$\begin{aligned} [p^{(1)}]_j &= \max_{1 \le i \le i_1} \{[\hat{\boldsymbol{h}}_i]_j\} \\ [p^{(2)}]_j &= \max_{i_1 < i \le i_2} \{[\hat{\boldsymbol{h}}_i]_j\} \\ [p^{(3)}]_j &= \max_{i_2 < i \le n} \{[\hat{\boldsymbol{h}}_i]_j\} \end{aligned} \tag{16}$$

where $[\cdot]_j$ is the j-th value of a vector, i_1 and i_2 are entity positions. The sentence embedding is achieved by concatenating these three pooling results as $\boldsymbol{g} = [p^{(1)}; p^{(2)}; p^{(3)}]$. Finally, we feed the sentence vector into a fully connected layer:

$$\boldsymbol{o} = \boldsymbol{W} \cdot \boldsymbol{g} + b \tag{17}$$

where $\boldsymbol{o} \in \mathbb{R}^{n_r}$ is the final output, n_r is the number of relation labels, $\boldsymbol{W}$ is the weight matrix and b is the bias. Let Θ_{RE} represent all parameters in RE, the conditional probability of i-th relation is:

$$p(r_i|s, \Theta_{RE}) = \frac{\exp(o_i)}{\sum_{j=1}^{n_r} \exp(o_j)} \tag{18}$$

4) Training Objective: For a sample, in other words, a sentence s, we denote its relation label as r^* and correct entity pair as $ep^* = (e_1^*, e_2^*)$. We introduce our training objective in three folds as follows.

Firstly, we define the objective function by using cross-entropy for EL and RE respectively:

$$\min \mathcal{L}_{EL} = -\log p(ep^*|s, \Theta_{EL}) \tag{19}$$

$$\min \mathcal{L}_{RE} = -\log p(r^*|s, \Theta_{RE}) \tag{20}$$

Secondly, we hope the selected entities (e_1^*, e_2^*) and relation r^* conform to the translation hypothesis [20], except the case where these two entities have no expected relation (i.e., r^* = NA). Hence, we define the following energy function:

$$\min \mathcal{L}_{TS} = ||\mathbf{e}_1^* + \mathbf{r}^* - \mathbf{e}_2^*||_{L_2} \tag{21}$$

Following that, we present the final objective function of the joint model by summing up these three parts:

$$\min \mathcal{L} = \lambda_1 \mathcal{L}_{EL} + \lambda_2 \mathcal{L}_{RE} + \delta \mathcal{L}_{TS} \tag{22}$$

where $\lambda_1, \lambda_2, \delta \ge 0$ are weights of the three parts respectively.

By applying the linking results (Equation 14) and the energy function (Equation 21), our model establishes the interplay between EL and RE, calculates the losses and updates the parameters together. Beyond that, the *Dropout* strategy and *ADAM* optimizer are also adopted to train our model.

IV. Experiments

A. Datasets

Considering that there is no open-source dataset that can evaluate the joint task of EL and RE (KBP), we evaluate our approach on variants of the NYT-10 dataset [6], which is developed by aligning *Freebase* entities and relations with the *New York Times* corpus. The NYT-10 dataset has been widely used in RE as a standard benchmark dataset, and it can be modified for EL by just adding some negative examples since it has linking annotation (*Freebase ID* for each entity), which can be regarded as positive samples for EL.

TABLE I
Statistics of NYT-A/AL and NYT-U.

	NYT-A/AL		NYT-U	
	Train	Test	Train	Test
Sentences	106,978	31,000	249,681	78,409
Total Entity Pairs	30,011	10,679	159,009	54,461
Positive Entity Pairs	2,818	521	15,589	1,193
Relations	26	26	26	26

To make the dataset can imitate the KBP procedure veritably, two steps are needed. First, *Freebase IDs* in NYT-10 are mapped to *Wikidata* through an open source mapping file [1]. Based on the hyper links in the *Wikidata*, we extract entity descriptions from their corresponding *Wikipedia* pages. Second, we get candidate entities from disambiguation pages of *Wikipedia*, all entities in which pages except the ground-truth can be negative samples. We follow [12] to filter the NYT dataset. Besides, if mentions cannot be mapped to *Wikipedia*, the corresponding sentences are also omitted. The dataset obtained here is denoted as NYT-Total (NYT-T), it can be divided into NYT-Ambiguous (NYT-A) and NYT-Unambiguous (NYT-U) by judging whether entity mentions are ambiguous. Statistics of them are shown in Table I. On this basis, we attach the candidate entity pair list to NYT-A and name the final dataset as NYT-Ambiguous with Linking annotation (NYT-AL), which can be used in KBP directly.

B. Experiment Setup

1) The Pre-Linking Strategy: In order to remove the irrelevant candidate entity pairs, top 10 words with highest frequency are chosen as entity keywords and the size of mention's context window is set to 5. We employ the XGBoost model [19] to calculate feature weights and heuristically assign α_{ss} = α_{cs} = 3, α_{ls} = α_{mc} = 2, α_{ec} = α_{gs} = 1. By defining R_{pl} to represent the recall of EL in test dataset, the results go as follows: when k = 1, R_{pl} = 0.866, when k = 5, $R_{pl} = 0.971$, and when k = 10, R_{pl} = 0.988. We empirically retain the top 5 entity pairs as the input of JF-ER. Word embeddings released by [15] are used as the pre-trained word embeddings in the *Pre-Linking* stage.

2) Implementation Details: Following popular choices and previous studies, we tune the model with three-fold cross-validation. For the vector representation, we set the dimension of word embedding to 300 and dimension of position feature to 50. For parameters of the DM-ER, the batch size is 256, the number of LSTM units is 128, the learning rate is $5e^{-4}$ with gradient clipping 10. At the convolutional layers of PCNN, the window size w is 5, and the number of feature maps is 200. We set λ_1 = λ_2 = 1 and let δ equal 10^{-4}, the probability of dropout is set to 0.5. Besides, it should be mentioned that, if

[1]https://developers.google.com/freebase/

TABLE II
HELD-OUT EVALUATION OF DIFFERENT APPROACHES ON EL, RE AND KBP TASKS. WE USE ("+") TO DENOTE THE SEPARATE METHODS AND ("×") TO OUR JF-ER FRAMEWORK. *Normalized* SCORES CONSIDER ONLY MENTIONS WHICH CONTAIN THE GOLD ENTITIES IN THE CANDIDATE SET.

Methods	Entity Linking		Relation Extraction			Knowledge Base Population		
	Accuracy(*standard*)	Accuracy(*normalized*)	Precision	Recall	F1	Precision	Recall	F1
PCNN + XEL	0.866	0.866	0.259	0.245	0.252	0.224	0.213	0.218
PCNN + NEL	0.840	0.865	0.259	0.245	0.252	0.219	0.207	0.213
PCNN × NEL (*DM-ER*)	0.888	**0.914**	0.320	0.275	**0.296**	0.298	0.253	**0.274**
PCNN + HEL	0.902	0.929	0.259	0.245	0.252	0.232	0.221	0.226
PCNN × HEL	0.926	**0.954**	0.338	0.295	**0.315**	0.310	0.276	**0.291**
PCNN(ATT) + XEL	0.866	0.866	0.346	0.238	0.282	0.307	0.209	0.248
PCNN(ATT) + NEL	0.840	0.865	0.346	0.238	0.282	0.302	0.204	0.243
PCNN(ATT) × NEL	0.894	**0.921**	0.395	0.273	**0.322**	0.370	0.247	**0.296**
PCNN(ATT) + HEL	0.902	0.929	0.346	0.238	0.282	0.324	0.220	0.262
PCNN(ATT) × HEL	0.935	**0.963**	0.404	0.296	**0.342**	0.383	0.278	**0.322**

there are many sentences in a distantly supervised bag, we add the *pre-score* / *joint-score* of each sentence to obtain the final score for a candidate entity pair in the *Pre-Linking* strategy and *Entity Pair Selector* module.

C. Comparing with Previous Work

To illustrate the effectiveness of JF-ER and DM-ER, various mainstream EL and RE models are freely combined through traditional separate methods and our joint framework in turn.

1) Baseline Methods: To show the performance and flexibility of our joint framework, we integrate several traditional and representative approaches toward EL and RE as follows:

- **XEL** is a XGBoost model for EL, which adopt six representative features and is detailed in the *Pre-Linking* strategy (see also section III-B).
- **NEL** [21] is a general neural network model for the answer selection task, which can be adopted to EL and DM-ER without manual features.
- **HEL** [17] is one of the latest neural models for EL, which is combined with hierarchical losses to use the rich hierarchies over types.
- **PCNN** [12] is the original distant supervision model with neural networks for RE, which proposes piece-wise CNN to obtain sentence embeddings.
- **PCNN(ATT)** [15] is an extension of PCNN, which leverages PCNN to encode and classify each sentence, and then consolidates the results of different sentences using an attention mechanism for distantly supervised RE.

Previous EL models usually depend on lots of manual features and external information, which makes it difficult to migrate them to our datasets. Thus we re-implement XEL and NEL following the main ideas of previous work. Based on these 5 above models, we implement 6 separate methods as baselines and 4 joint methods as the examples of JF-ER, then compare them on the NYT-AL dataset. For KBP task, only when the labels of relations and entities are predicted correctly at the same time, can we take them as a positive result.

2) Results: Comparing the performance of different methods in Table II, we observe that JF-ER significantly outperforms all the corresponding baselines, not only on the EL and RE tasks individually but also on the KBP task. Especially, by using the JF-ER framework, the RE task achieves a remarkable performance in F_1 score, which has 20% to 30% relative improvement than the separate baseline methods. This observation demonstrates that the background knowledge of entities can indeed help improve the performance of RE. Furthermore, NEL with our JF-ER framework obtains superior performance than XEL while the performance of vanilla NEL is quite limited. It is noteworthy that there are neither hand-designed features nor extra information in the NEL model, which proves that relations extracted by *Relation Extractor* provide lots of valuable information for EL in turn. Therefore, we could draw the conclusion that our JF-ER framework can utilize semantic information between relations and entities effectively. Meanwhile, we also notice that the performance of our PCNN×NEL (i.e., DM-ER) is quite higher than PCNN+NEL (+6.1% in terms of F_1 score) on the KBP task. Similarly, the F_1 score of PCNN(ATT)×NEL is also 6.5% higher than PCNN(ATT)+NEL and there are some similar phenomena when the JF-ER framework employs HEL as *Entity Pair Selector*. This is strong evidence that existing methods for EL and RE can be easily integrated into our framework, and existing advanced EL or RE models can actually achieve greater performances.

TABLE III
MANUAL EVALUATION OF RELATION EXTRACTION TASK WITH PCNN AND DM-ER MODELS ON DIFFERENT DATASETS.

	P@50	P@100	P@200	Mean
PCNN on NYT-10	0.87	0.84	0.78	0.830
PCNN on NYT-T	0.88	0.83	0.76	0.823
PCNN on NYT-U	0.91	0.86	0.80	0.857
PCNN on NYT-A	0.75	0.71	0.65	0.703
DM-ER on NYT-AL	0.84	0.78	0.71	0.776

D. Influences of Ambiguous Mentions

In Table II, PCNN and PCNN(ATT) show poor performances than original papers. The most notable here is the proportion of ambiguous mentions, so we adopt PCNN on different versions of the NYT-10 dataset to explore the influences

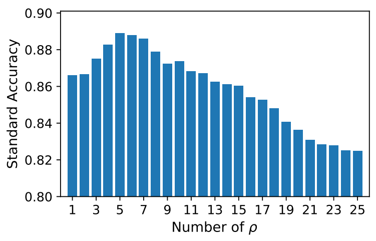

Fig. 4. Standard accuracy of NEL when taking different values of ρ.

of ambiguous mentions. Moreover, we also adapt our DM-ER model, which employs PCNN as *Relation Extractor*, on the NYT-AL dataset. To escape the influence of false negative, we manually evaluate the performance with P@N metric to show the performance of our approach on such different datasets.

Based on Table III, we can draw the following conclusions: (1) PCNN model achieves comparable performances on NYT-10 and NYT-T, indicating that removing data in the same proportion does not affect the data distribution and results evidently. (2) PCNN model on NYT-U outperforms other datasets, especially on NYT-A, which suggests that ambiguous mentions will have negative effects on RE. That is the more ambiguous words, the poorer performance for RE. (3) DM-ER model on NYT-AL achieves better results than PCNN on NYT-A, proving that the combination of RE and EL can effectively alleviate the impact of ambiguous words. As a result, we argue that it is necessary to focus on not only models but also the semantic information and the distribution of datasets.

E. Balancing Each Module

In this section, DM-ER is employed as the evaluation model. We adjust the candidate number ρ and loss weight λ_1, λ_2 to explore the influences of these hyper-parameters.

1) Pre-Linking and DM-ER: We employ *Pre-Linking* strategy to obtain better performances of EL. There is no doubt that how to choose the number of retained candidate entity pairs is essential. For this consideration, we vary ρ from 1 to 25, increased by 1. Results with different numbers of retained pairs are given in Fig. 4. In the procedure of transferring ρ, the standard accuracy curve displays a trend of rising first and then decreasing, when ρ equals 5, it reaches the peak. It can be explained intuitively: when ρ is small, the recall of candidate entity pair list is very low, which limits the upper bound of accuracy; when ρ becomes larger, NEL cannot work very well. Hence, a moderate ρ is crucial to the follow-up processes.

2) Entity Linking and Relation Extraction: We investigate how the loss weights λ_1 and λ_2 affect the performance of *Entity Pair Selector* and *Relation Extractor*. Obviously, EL and RE tasks are treated equally when $\lambda_1 = \lambda_2 = 1$. In the process of λ_1 becoming larger than λ_2, our model gradually pays more attention to EL. On the contrary, RE makes more effect to the joint model when λ_1/λ_2 becomes smaller. Experimental results show that when λ_1 and λ_2 change, the F_1 score of KBP only undulates slightly, thus we ignore the variance and deem that our DM-ER model is relatively stable overall.

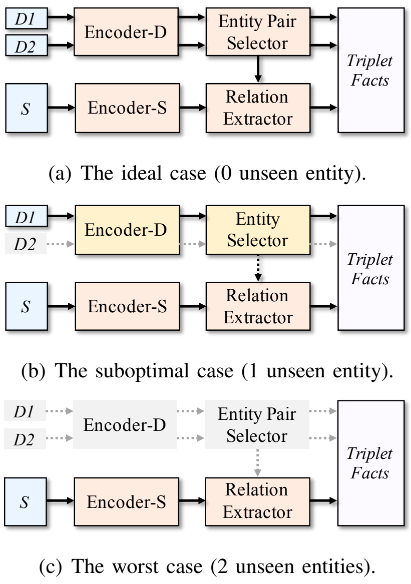

Fig. 5. The illustration of the usage of each module in JF-ER when facing different unseen entity problem. The gray boxes indicate that these modules are not used and yellow indicate simplified.

F. Universality Analysis

In the above statement, we assume that each entity mention could obtain a list of candidate entities from the source KB. However, in the production environment, we are bound to face the problem of ***unseen entity***, which means that an entity mention cannot retrieve a candidate entity (list) from the KB. Fortunately, our JF-ER can automatically adapt to such situations without any model-level modification. Based on the number of unseen entities, we consider three situations when deploying the system as follows:

In the ideal case, there is no unseen entity in the sentence, the original JF-ER does not need any modification (see Fig. 5(a)). There is no doubt that it will consistently outperform over strong baselines. In the suboptimal case, there is one unseen entity in the sentence, and the other entity mention with the candidate entity list is recorded as m^*. All averaging and summing operations between two entities in *Pre-Linking* and *Entity Pair Selector* are simplified, and only the information from m^* is fed into *Relation Extractor* (see Fig. 5(b)), which will partially enhance the performances. In the worst case, there are two unseen entities in the sentence, the JF-ER will degenerate to a vanilla RE model (e.g. PCNN) with an LSTM Encoder (Encoder-S) (see Fig. 5(c)). It will not lead to any notable improvement compared to baselines.

Overall, the JF-ER framework can lead to dramatic improvements in most cases, except for the worst case. Thus it is very clear that this technique is useful in general.

V. Related Work

Relation Extraction: RE is a historic task in NLP community. Recent studies usually uses a pre-defined relation set to complete KBs [14], especially under the distant supervision scenarios [12], [15]. The problem is that they considered RE as an independent task and lost a lot of effective external information. Liu et al. [22] and Vashishth et al. [23] involved entity type into RE, while Ji et al. [10] and Ren et al. [3] utilized entity description to enrich entity representation. However, the datasets they used have clear labels. In fact, there are no such data with type annotation or linking information before aligning the dataset with KBs.

Entity Linking: EL is a vital stage in KBP process. Traditional approaches rely on superior feature selections [24], [25]. Global features and collective linking [7], [26] have achieved outstanding performances. Phan et al. [13] hold the view that each linking only relevant to another linking. However, EL is considered as a single task and missed much effective information. Recently, researchers tend to employ all kinds of information, such as entity types [5], [17] and relations [11]. For instance, Le et al. [11] proved that relations are effective information in EL. But it only supports a few relations, which makes this approach suffer from a great limitation.

Joint Learning for Information Extraction: Joint learning has been widely used in many NLP tasks, especially on entity and relation extraction [27]–[29] or entity extraction and linking [30], [31]. There are also some methods to link entities and relations at once in question answering over knowledge base (KBQA) problem [32], but usually, there are only two elements of the (*h, r, t*) triplet in query sentences and we need to retrieval the remaining one from KB, hence those methods are not suitable for KBP. When it comes to joint RE and EL, J-REED [9] is the first attempt using probabilistic graphical models. However, it needs to extract relation words from free texts, which cannot be used in KBP directly.

VI. Conclusion

In this paper, we propose a joint learning framework and its demonstration model to extract relations and link entities in one step. Our approach can easily capture the semantic relevance between entities and relations so that entity linking can provide extra entity information to predict relations, and the relations extracted by relation extraction can also improve entity linking in turn. We evaluate our model on variants of the NYT-10 dataset and originally discuss the influences of ambiguous mentions for relation extraction. Experiment results suggest that our approach offers significant improvements over traditional separate methods and most of the existing methods can be integrated into our framework.

Acknowledgment

This work is supported by the National Key Research and Development Program of China (No.2016YFB0801003) and the Strategic Priority Research Program of Chinese Academy of Sciences (No.XDC02040400).

References

[1] M. Surdeanu, D. McClosky, J. Tibshirani, J. Bauer, A. X. Chang, V. I. Spitkovsky, and C. D. Manning, “A simple distant supervision approach for the tac-kbp slot filling task.” in *TAC Workshop*, 2010.

[2] W. Shen, J. Wang, and J. Han, “Entity linking with a knowledge base: Issues, techniques, and solutions,” *TKDE*, 2015.

[3] F. Ren, D. Zhou, Z. Liu, Y. Li, R. Zhao, Y. Liu, and X. Liang, “Neural relation classification with text descriptions,” in *Proc. of COLING*, 2018.

[4] B. Yu, Z. Zhang, T. Liu, B. Wang, S. Li, and Q. Li, “Beyond word attention: Using segment attention in neural relation extraction,” in *Proc. of IJCAI*, 2019.

[5] Z. Zhang, X. Shu, B. Yu, T. Liu, J. Zhao, Q. Li, and L. Guo, “Distilling knowledge from well-informed soft labels for neural relation extraction,” in *Proc. of AAAI*, 2020.

[6] S. Riedel, L. Yao, and A. McCallum, “Modeling relations and their mentions without labeled text,” in *Proc. of ECML*, 2010.

[7] Z. Fang, Y. Cao, D. Zhang, Q. Li, Z. Zhang, and Y. Liu, “Joint entity linking with deep reinforcement learning,” in *Proc. of WWW*, 2019.

[8] Z. Fang, Y. Cao, R. Li, Z. Zhang, Y. Liu, and S. Wang, “High quality candidate generation and sequential graph attention network for entity linking,” in *Proc. of WWW*, 2020.

[9] D. B. Nguyen, M. Theobald, and G. Weikum, “J-reed: Joint relation extraction and entity disambiguation,” in *Proc. of CIKM*, 2017.

[10] G. Ji, K. Liu, S. He, J. Zhao *et al.*, “Distant supervision for relation extraction with sentence-level attention and entity descriptions.” in *Proc. of AAAI*, 2017.

[11] P. Le and I. Titov, “Improving entity linking by modeling latent relations between mentions,” in *Proc. of ACL*, 2018.

[12] D. Zeng, K. Liu, Y. Chen, and J. Zhao, “Distant supervision for relation extraction via piecewise convolutional neural networks,” in *Proc. of EMNLP*, 2015.

[13] M. C. Phan, A. Sun, Y. Tay, J. Han, and C. Li, “Neupl: Attention-based semantic matching and pair-linking for entity disambiguation,” in *Proc. of CIKM*, 2017.

[14] D. Zeng, K. Liu, S. Lai, G. Zhou, and J. Zhao, “Relation classification via convolutional deep neural network,” in *Proc. of COLING*, 2014.

[15] Y. Lin, S. Shen, Z. Liu, H. Luan, and M. Sun, “Neural relation extraction with selective attention over instances,” in *Proc. of ACL*, 2016.

[16] N. Gupta, S. Singh, and D. Roth, “Entity linking via joint encoding of types, descriptions, and context,” in *Proc. of EMNLP*, 2017.

[17] S. Murty, P. Verga, L. Vilnis, I. Radovanovic, and A. McCallum, “Hierarchical losses and new resources for fine-grained entity typing and linking,” in *Proc. of ACL*, 2018.

[18] Y. Cao, L. Hou, J. Li, and Z. Liu, “Neural collective entity linking,” in *Proc. of COLING*, 2018.

[19] T. Chen and C. Guestrin, “Xgboost: A scalable tree boosting system,” in *Proc. of SIGKDD*, 2016.

[20] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, and O. Yakhnenko, “Translating embeddings for modeling multi-relational data,” in *Proc. of NIPS*, 2013.

[21] M. Tan, C. d. Santos, B. Xiang, and B. Zhou, “Lstm-based deep learning models for non-factoid answer selection,” in *ICLR Workshop*, 2016.

[22] Y. Liu, K. Liu, L. Xu, and J. Zhao, “Exploring fine-grained entity type constraints for distantly supervised relation extraction,” in *Proc. of COLING*, 2014.

[23] S. Vashishth, R. Joshi, S. S. Prayaga, C. Bhattacharyya, and P. Talukdar, “Reside: Improving distantly-supervised neural relation extraction using side information,” in *Proc. of EMNLP*, 2018.

[24] L. Ratinov, D. Roth, D. Downey, and M. Anderson, “Local and global algorithms for disambiguation to wikipedia,” in *Proc. of ACL*, 2011.

[25] L. Chen, J. Liang, C. Xie, and Y. Xiao, “Short text entity linking with fine-grained topics,” in *Proc. of CIKM*, 2018.

[26] X. Han, L. Sun, and J. Zhao, “Collective entity linking in web text: a graph-based method,” in *Proc. of SIGIR*, 2011.

[27] X. Ren, Z. Wu, W. He, M. Qu, C. R. Voss, H. Ji, T. F. Abdelzaher, and J. Han, “Cotype: Joint extraction of typed entities and relations with knowledge bases,” in *Proc. of WWW*, 2017.

[28] K. Dixit and Y. Al-Onaizan, “Span-level model for relation extraction,” in *Proc. of ACL*, 2019.

[29] B. Yu, Z. Zhang, X. Shu, Y. Wang, T. Liu, B. Wang, and S. Li, “Joint extraction of entities and relations based on a novel decomposition strategy,” in *Proc. of ECAI*, 2020.

[30] Y. Luan, L. He, M. Ostendorf, and H. Hajishirzi, “Multi-task identification of entities, relations, and coreference for scientific knowledge graph construction,” in *Proc. of EMNLP*, 2018.

[31] A. Pappu, R. Blanco, Y. Mehdad, A. Stent, and K. Thadani, “Lightweight multilingual entity extraction and linking,” in *Proc. of WSDM*, 2017.

[32] M. Dubey, D. Banerjee, D. Chaudhuri, and J. Lehmann, “Earl: Joint entity and relation linking for question answering over knowledge graphs,” in *Proc. of ISWC*, 2018.