UniRE: A Unified Label Space for Entity Relation Extraction

Yijun Wang[*1,2], **Changzhi Sun**[*4], **Yuanbin Wu**[3], **Hao Zhou**[4], **Lei Li**[4], **and Junchi Yan**[†1,2]

[1]Department of Computer Science and Engineering, Shanghai Jiao Tong University
[2]MoE Key Lab of Artificial Intelligence, AI Institute, Shanghai Jiao Tong University
[3]School of Computer Science and Technology, East China Normal University
[4]ByteDance AI Lab
{yjwang.cs, yanjunchi}@sjtu.edu.cn ybwu@cs.ecnu.edu.cn
{sunchangzhi, zhouhao.nlp, lileilab}@bytedance.com

Abstract

Many joint entity relation extraction models setup two separated label spaces for the two sub-tasks (i.e., entity detection and relation classification). We argue that this setting may hinder the information interaction between entities and relations. In this work, we propose to eliminate the different treatment on the two sub-tasks' label spaces. The input of our model is a table containing all word pairs from a sentence. Entities and relations are represented by squares and rectangles in the table. We apply a unified classifier to predict each cell's label, which unifies the learning of two sub-tasks. For testing, an effective (yet fast) approximate decoder is proposed for finding squares and rectangles from tables. Experiments on three benchmarks (ACE04, ACE05, SciERC) show that, using only half the number of parameters, our model achieves competitive accuracy with the best extractor, and is faster.

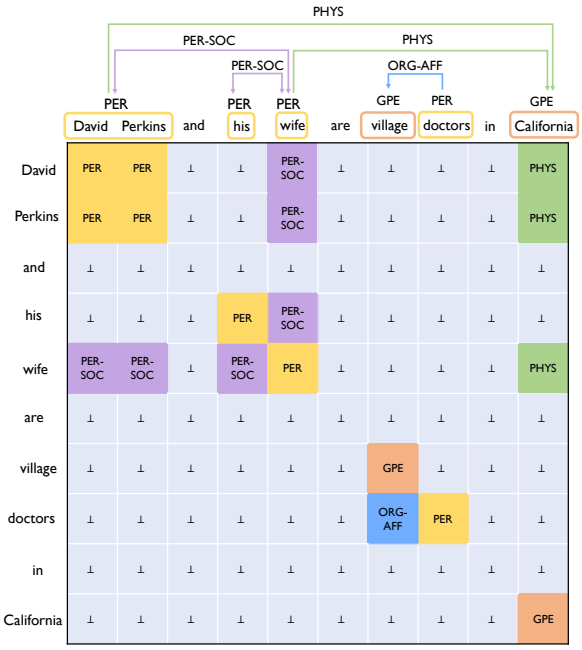

Figure 1: Example of a table for joint entity relation extraction. Each cell corresponds to a word pair. Entities are squares on diagonal, relations are rectangles off diagonal. Note that PER-SOC is a undirected (symmetrical) relation type, while PHYS and ORG-AFF are directed (asymmetrical) relation types. The table exactly expresses overlapped relations, e.g., the person entity "David Perkins" participates in two relations, ("David Perkins", "wife", PER-SOC) and ("David Perkins", "California", PHYS). For every cell, a **same** biaffine model predicts its label. The joint decoder is set to find the best squares and rectangles.

1 Introduction

Extracting structured information from plain texts is a long-lasting research topic in NLP. Typically, it aims to recognize specific entities and relations for profiling the semantic of sentences. An example is shown in Figure 1, where a person entity "David Perkins" and a geography entity "California" have a physical location relation PHYS.

Methods for detecting entities and relations can be categorized into pipeline models or joint models. In the pipeline setting, entity models and relation models are independent with disentangled feature spaces and output label spaces. In the joint setting, on the other hand, some parameter sharing of feature spaces (Miwa and Bansal, 2016; Katiyar and Cardie, 2017) or decoding interactions (Yang and Cardie, 2013; Sun et al., 2019) are imposed to explore the common structure of the two tasks. It was believed that joint models could be better since they can alleviate error propagations among sub-models, have more compact parameter sets, and uniformly encode prior knowledge (e.g., constraints) on both tasks.

However, Zhong and Chen (2020) recently show

[*]Equal contribution.
[†]Corresponding Author.

Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, pages 220–231
August 1–6, 2021.

that with the help of modern pre-training tools (e.g., BERT), separating the entity and relation model (with independent encoders and pipeline decoding) could surpass existing joint models. They argue that, since the output label spaces of entity and relation models are different, comparing with shared encoders, separate encoders could better capture distinct contextual information, avoid potential conflicts among them, and help decoders making a more accurate prediction, that is, *separate label spaces deserve separate encoders*.

In this paper, we pursue a better joint model for entity relation extraction. After revisiting existing methods, we find that though entity models and relation models share encoders, usually their label spaces are still separate (even in models with joint decoders). Therefore, parallel to (Zhong and Chen, 2020), we would ask whether *joint encoders (decoders) deserve joint label spaces*?

The challenge of developing a unified entity-relation label space is that the two sub-tasks are usually formulated into different learning problems (e.g., entity detection as sequence labeling, relation classification as multi-class classification), and their labels are placed on different things (e.g., words v.s. words pairs). One prior attempt (Zheng et al., 2017) is to handle both sub-tasks with one sequence labeling model. A compound label set was devised to encode both entities and relations. However, the model's expressiveness is sacrificed: it can detect neither overlapping relations (i.e., entities participating in multiple relation) nor isolated entities (i.e., entities not appearing in any relation).

Our key idea of defining a new unified label space is that, if we think Zheng et al. (2017)'s solution is to perform relation classification during entity labeling, we could also consider the reverse direction by seeing entity detection as a special case of relation classification. Our new input space is a two-dimensional table with each entry corresponding to a word pair in sentences (Figure 1). The joint model assign labels to each cell from *a unified label space* (union of entity type set and relation type set). Graphically, entities are squares on the diagonal, and relations are rectangles off the diagonal. This formulation retains full model expressiveness regarding existing entity-relation extraction scenarios (e.g., overlapped relations, directed relations, undirected relations). It is also different from the current table filling settings for entity relation extraction (Miwa and Sasaki, 2014; Gupta et al., 2016; Zhang et al., 2017; Wang and Lu, 2020), which still have separate label space for entities and relations, and treat on/off-diagonal entries differently.

Based on the tabular formulation, our joint entity relation extractor performs two actions, *filling* and *decoding*. First, filling the table is to predict each word pair's label, which is similar to arc prediction task in dependency parsing. We adopt the biaffine attention mechanism (Dozat and Manning, 2016) to learn interactions between word pairs. We also impose two structural constraints on the table through structural regularizations. Next, given the table filling with label logits, we devise an approximate joint decoding algorithm to output the final extracted entities and relations. Basically, it efficiently finds split points in the table to identify squares and rectangles (which is also different with existing table filling models which still apply certain sequential decoding and fill tables incrementally).

Experimental results on three benchmarks (ACE04, ACE05, SciERC) show that the proposed joint method achieves competitive performances comparing with the current state-of-the-art extractors (Zhong and Chen, 2020): it is better on ACE04 and SciERC, and competitive on ACE05.[1] Meanwhile, our new joint model is fast on decoding (10x faster than the exact pipeline implementation, and comparable to an approximate pipeline, which attains lower performance). It also has a more compact parameter set: the shared encoder uses only half the number of parameters comparing with the separate encoder (Zhong and Chen, 2020).

2 Task Definition

Given an input sentence $s = x_1, x_2, \ldots, x_{|s|}$ (x_i is a word), this task is to extract a set of entities $\mathcal{E}$ and a set of relations $\mathcal{R}$. An entity e is a span ($e.\texttt{span}$) with a pre-defined type $e.\texttt{type} \in \mathcal{Y}_e$ (e.g., `PER`, `GPE`). The span is a continuous sequence of words. A relation r is a triplet (e_1, e_2, l), where e_1, e_2 are two entities and $l \in \mathcal{Y}_r$ is a pre-defined relation type describing the semantic relation among two entities (e.g., the `PHYS` relation between `PER` and `GPE` mentioned before). Here $\mathcal{Y}_e, \mathcal{Y}_r$ denote the set of possible entity types and relation types respectively.

We formulate the joint entity relation extraction

[1] Source code and models are available at https://github.com/Receiling/UniRE.

as a table filling task (multi-class classification between each word pair in sentence s), as shown in Figure 1. For the sentence s, we maintain a table $T^{|s|\times|s|}$. For each cell (i, j) in table T, we assign a label $y_{i,j} \in \mathcal{Y}$, where $\mathcal{Y} = \mathcal{Y}_e \cup \mathcal{Y}_r \cup \{\perp\}$ ($\perp$ denotes no relation). For each entity e, the label of corresponding cells $y_{i,j}(x_i \in e.\texttt{span}, x_j \in e.\texttt{span})$ should be filled in $e.\texttt{type}$. For each relation $r = (e_1, e_2, l)$, the label of corresponding cells $y_{i,j}(x_i \in e_1.\texttt{span}, x_j \in e_2.\texttt{span})$ should be filled in l.[2] While others should be filled in $\perp$. In the test phase, decoding entities and relations becomes a rectangle finding problem. Note that solving this problem is not trivial, and we propose a simple but effective joint decoding algorithm to tackle this challenge.

3 Approach

In this section, we first introduce our biaffine model for table filling task based on pre-trained language models (Section 3.1). Then we detail the main objective function of the table filling task (Section 3.2) and some constraints which are imposed on the table in training stage (Section 3.3). Finally we present the joint decoding algorithm to extract entities and relations (Section 3.4). Figure 2 shows an overview of our model architecture.[3]

3.1 Biaffine Model

Given an input sentence s, to obtain the contextual representation $\mathbf{h}_i$ for each word, we use a pre-trained language model (PLM) as our sentence encoder (e.g., BERT). The output of the encoder is

$$\{\mathbf{h}_1, \ldots, \mathbf{h}_{|s|}\} = \texttt{PLM}(\{\mathbf{x}_1, \ldots, \mathbf{x}_{|s|}\}),$$

where $\mathbf{x}_i$ is the input representation of each word x_i. Taking BERT as an example, $\mathbf{x}_i$ sums the corresponding token, segment and position embeddings. To capture long-range dependencies, we also employ cross-sentence context following (Zhong and Chen, 2020), which extends the sentence to a fixed window size W ($W = 200$ in our default settings).

To better encode direction information of words in table T, we use the deep biaffine attention mechanism (Dozat and Manning, 2016), which achieves impressive results in the dependency parsing task. Specifically, we employ two dimension-reducing MLPs (multi-layer perceptron), i.e., a head MLP and a tail MLP, on each $\mathbf{h}_i$ as

$$\mathbf{h}_i^{\text{head}} = \texttt{MLP}_{\text{head}}(\mathbf{h}_i), \quad \mathbf{h}_i^{\text{tail}} = \texttt{MLP}_{\text{tail}}(\mathbf{h}_i),$$

where $\mathbf{h}_i^{\text{head}} \in \mathbb{R}^d$ and $\mathbf{h}_i^{\text{tail}} \in \mathbb{R}^d$ are projection representations, allowing the model to identify the head or tail role of each word. Next, we calculate the scoring vector $\mathbf{g}_{i,j} \in \mathbb{R}^{|\mathcal{Y}|}$ of each word pair with biaffine model,

$$\mathbf{g}_{i,j} = \texttt{Biaff}(\mathbf{h}_i^{\text{head}}, \mathbf{h}_j^{\text{tail}}),$$
$$\texttt{Biaff}(\mathbf{h_1}, \mathbf{h_2}) = \mathbf{h}_1^T \mathbf{U}_1 \mathbf{h}_2 + \mathbf{U}_2(\mathbf{h}_1 \oplus \mathbf{h}_2) + \mathbf{b},$$

where $\mathbf{U}_1 \in \mathbb{R}^{|\mathcal{Y}|\times d\times d}$ and $\mathbf{U}_2 \in \mathbb{R}^{|\mathcal{Y}|\times 2d}$ are weight parameters, $\mathbf{b} \in \mathbb{R}^{|\mathcal{Y}|}$ is the bias, $\oplus$ denotes concatenation.

3.2 Table Filling

After obtaining the scoring vector $\mathbf{g}_{i,j}$, we feed $\mathbf{g}_{i,j}$ into the softmax function to predict corresponding label, yielding a categorical probability distribution over the label space $\mathcal{Y}$ as

$$P(\mathbf{y}_{i,j}|s) = \texttt{Softmax}(\texttt{dropout}(\mathbf{g}_{i,j})).$$

In our experiments, we observe that applying dropout in $\mathbf{g}_{i,j}$, similar to de-noising auto-encoding, can further improve the performance. [4]. We refer this trick to *logit dropout* And the training objective is to minimize

$$\mathcal{L}_{\text{entry}} = -\frac{1}{|s|^2}\sum_{i=1}^{|s|}\sum_{j=1}^{|s|}\log P(\mathbf{y}_{i,j} = y_{i,j}|s), \quad (1)$$

where the gold label $y_{i,j}$ can be read from annotations, as shown in Figure 1.

3.3 Constraints

In fact, Equation 1 is based on the assumption that each label is independent. This assumption simplifies the training procedure, but ignores some structural constraints. For example, entities and relations correspond to squares and rectangles in the table. Equation 1 does not encode this constraint explicitly. To enhance our model, we propose two intuitive constraints, *symmetry* and *implication*, which are detailed in this section. Here we introduce a new notation $\mathcal{P} \in \mathbb{R}^{|s|\times|s|\times|\mathcal{Y}|}$, denoting the stack of $P(\mathbf{y}_{i,j}|s)$ for all word pairs in sentence s.[5]

[2] Assuming no overlapping entities in one sentence.

[3] We only show three labels of $\mathcal{Y}$ in Figure 2 for simplicity and clarity.

[4] We set dropout rate $p = 0.2$ by default.

[5] $\mathcal{P}$ without *logit dropout* mentioned in Section 3.2 to preserve learned structure.

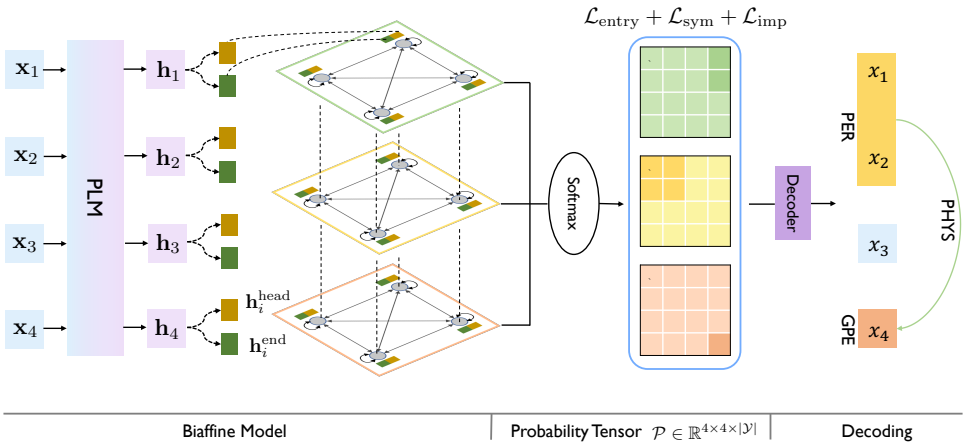

Figure 2: Overview of our model architecture. One main objective ($\mathcal{L}_{\text{entry}}$) and two additional objectives ($\mathcal{L}_{\text{sym}}, \mathcal{L}_{\text{imp}}$) are imposed on probability tensor $\mathcal{P}$ and optimized jointly.

Dataset	$\mathcal{Y}_{\text{sym}}$ Ent	Rel
ACE04/ ACE05	PER,ORG,LOC, FAC,WEA,VEH,GPE	PER-SOC
SciERC	Task,Method,Metric, Material,Generic, OtherScientificTerm	COMPAREP, CONJUNCTION

Table 1: Symmetrical label set $\mathcal{Y}_{\text{sym}}$ for used datasets.

Symmetry We have several observations from the table in the tag level. Firstly, the squares corresponding to entities must be symmetrical about the diagonal. Secondly, for symmetrical relations, the relation triples (e_1, e_2, l) and (e_2, e_1, l) are equivalent, thus the rectangles corresponding to two counterpart relation triples are also symmetrical about the diagonal. As shown in Figure 1, the rectangles corresponding to ("his", "wife", PER-SOC) and ("wife", "his", PER-SOC) are symmetrical about the diagonal. We divide the set of labels $\mathcal{Y}$ into a symmetrical label set $\mathcal{Y}_{\text{sym}}$ and an asymmetrical label set $\mathcal{Y}_{\text{asym}}$. The matrix $\mathcal{P}_{:,:,t}$ should be symmetrical about the diagonal for each label $t \in \mathcal{Y}_{\text{sym}}$. We formulate this tag-level constraint as symmetrical loss,

$$\mathcal{L}_{\text{sym}} = \frac{1}{|s|^2} \sum_{i=1}^{|s|} \sum_{j=1}^{|s|} \sum_{t \in \mathcal{Y}_{\text{sym}}} |\mathcal{P}_{i,j,t} - \mathcal{P}_{j,i,t}|.$$

We list all $\mathcal{Y}_{\text{sym}}$ in Table 1 for our adopted datasets.

Implication A key intuition is that if a relation exists, then its two argument entities must also exist. In other words, it is impossible for a relation to exist without two corresponding entities. From the perspective of probability, it implies that the probability of relation is not greater than the probability of each argument entity. Since we model entity and relation labels in a unified probability space, this idea can be easily used in our model as the implication constraint. We impose this constraint on $\mathcal{P}$: for each word in the diagonal, its maximum possibility over the entity type space $\mathcal{Y}_e$ must not be lower than the maximum possibility for other words in the same row or column over the relation type space $\mathcal{Y}_r$. We formulate this table-level constraint as implication loss,

$$\mathcal{L}_{\text{imp}} = \frac{1}{|s|} \sum_{i=1}^{|s|} \left[\max_{l \in \mathcal{Y}_r} \{\mathcal{P}_{i,:,l}, \mathcal{P}_{:,i,l}\} - \max_{t \in \mathcal{Y}_e} \{\mathcal{P}_{i,i,t}\} \right]_*$$

where $[u]_* = \max(u, 0)$ is the hinge loss. It is worth noting that we do not add margin in this loss function. Since the value of each item is a probability and might be relatively small, it is meaningless to set a large margin.

Finally, we jointly optimize the three objectives in the training stage as $\mathcal{L}_{\text{entry}} + \mathcal{L}_{\text{sym}} + \mathcal{L}_{\text{imp}}$.[6]

3.4 Decoding

In the testing stage, given the probability tensor $\mathcal{P} \in \mathbb{R}^{|s|\times|s|\times|\mathcal{Y}|}$ of the sentence s,[7] how to decode all rectangles (including squares) corresponding to entities or relations remains a non-trivial problem. Since brute force enumeration of all rectangles is intractable, a new joint decoding algorithm is needed. We expect our decoder to have,

[6] We directly sum the three losses to avoid introducing more hyper-parameters.

[7] For the symmetrical label $t \in \mathcal{Y}_{\text{sym}}$, we set $\mathcal{P}_{i,j,t} = \mathcal{P}_{j,i,t} = (\mathcal{P}_{i,j,t} + \mathcal{P}_{j,i,t})/2$.

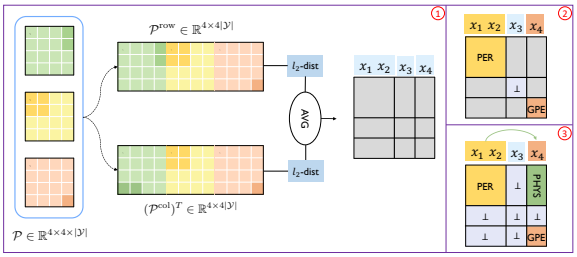

Figure 3: Overview of our joint decoding algorithm. It consists of three steps: span decoding, entity type decoding, and relation type decoding.

- Simple implementation and fast decoding. We permit slight decoding accuracy drops for scalability.

- Strong interactions between entities and relations. When decoding entities, it should take the relation information into account, and vice versa.

Inspired by the procedures of (Sun et al., 2019), We propose a three-steps decoding algorithm: decode *span* first (entity spans or spans between entities), and then decode *entity type* of each span, and at last decode *relation type* of each entity pair (Figure 3). We consider each cell's probability scores on all labels (including entity labels and relation labels) and predict spans according to a threshold. Then, we predict entities and relations with the highest score. Our heuristic decoding algorithm could be very efficient. Next we will detail the entire decoding process, and give a formal description in the Appendix A.

Span Decoding One crucial observation of a ground-truth table is that, for an arbitrary entity, its corresponding rows (or columns) are exactly the same in the table (e.g., row 1 and row 2 of Figure 1 are identical), not only for the diagonal entries (entities are squares), but also for the off-diagonal entries (if it participates in a relation with another entity, all its rows (columns) will spot that relation label in the same way). In other words, if the adjacent rows/columns are different, there must be an entity boundary (i.e., one belonging to the entity and the other not belonging to the entity). Therefore, if our biaffine model is reasonably trained, given a model predicted table, we could use this property to find split positions of entity boundary. As expected, experiments (Figure 4) verify our assumption. We adapt this idea to the 3-dimensional probability tensor $\mathcal{P}$.

Dataset	#sents	#ents(#types)	#rels(#types)
ACE04	8,683	22,519(7)	4,417(6)
ACE05	14,525	38,287(7)	7,691(6)
SciERC	2,687	8,094(6)	5,463(7)

Table 2: The statistics of the adopted datasets.

Specifically, we flatten $\mathcal{P} \in \mathbb{R}^{|s|\times|s|\times|\mathcal{Y}|}$ as a matrix $\mathcal{P}^{\text{row}} \in \mathbb{R}^{|s|\times(|s|\cdot|\mathcal{Y}|)}$ from row perspective, and then calculate the Euclidean distances (l_2 distances) of adjacent rows. Similarly, we calculate the other Euclidean distances of adjacent columns according to a matrix $\mathcal{P}^{\text{col}} \in \mathbb{R}^{(|s|\cdot|\mathcal{Y}|)\times|s|}$ from column perspective, and then average the two distances as the final distance. If the distance is larger than the threshold α ($\alpha = 1.4$ in our default settings), this position is a split position. In this way, we can decode all the spans in $\mathcal{O}(|s|)$ time complexity.

Entity Type Decoding Given a span (i, j) by span decoding,[8] we decode the entity type $\hat{t}$ according to the corresponding square symmetric about the diagonal: $\hat{t} = \arg\max_{t\in\mathcal{Y}_e\cup\{\perp\}} \texttt{Avg}(\mathcal{P}_{i:j,i:j,t})$. If $\hat{t} \in \mathcal{Y}_e$, we decode an entity. If $\hat{t} = \perp$, the span (i, j) is not an entity.

Relation Type Decoding After entity type decoding, given an entity e_1 with the span (i, j) and another entity e_2 with the span (m, n), we decode the relation type $\hat{l}$ between e_1 and e_2 according to the corresponding rectangle. Formally, $\hat{l} = \arg\max_{l\in\mathcal{Y}_r\cup\{\perp\}} \texttt{Avg}(\mathcal{P}_{i:j,m:n,l})$. If $\hat{l} \in \mathcal{Y}_r$, we decode a relation $(e_1, e_2, \hat{l})$. If $\hat{l} = \perp$, e_1 and e_2 have no relation.

4 Experiments

Datasets We conduct experiments on three entity relation extraction benchmarks: ACE04 (Doddington et al., 2004),[9] ACE05 (Walker et al., 2006),[10] and SciERC (Luan et al., 2018).[11] Table 2 shows the dataset statistics. Besides, we provide detailed dataset specifications in the Appendix B.

Evaluation Following suggestions in (Taillé et al., 2020), we evaluate Precision (P), Recall (R), and F1 scores with micro-averaging and adopt the **Strict Evaluation** criterion. Specifically, a predicted entity is correct if its type and boundaries are correct, and a predicted relation is correct if its

[8] i and j denote start and end indices of the span.
[9] https://catalog.ldc.upenn.edu/LDC2005T09
[10] https://catalog.ldc.upenn.edu/LDC2006T06
[11] http://nlp.cs.washington.edu/sciIE/

Dataset	Model	Encoder	Entity			Relation		
			P	R	F1	P	R	F1
ACE04	Li and Ji (2014)	-	83.5	76.2	79.7	60.8	36.1	45.3
	Miwa and Bansal (2016)	LSTM	80.8	82.9	81.8	48.7	48.1	48.4
	Katiyar and Cardie (2017)	LSTM	81.2	78.1	79.6	46.4	45.3	45.7
	Li et al. (2019)	$\text{BERT}_{\text{LARGE}}$	84.4	82.9	83.6	50.1	48.7	49.4
	Wang and Lu (2020)	$\text{ALBERT}_{\text{XXLARGE}}$	-	-	88.6	-	-	59.6
	Zhong and Chen (2020)$^\diamond$	$\text{BERT}_{\text{BASE}}$	-	-	89.2	-	-	60.1
	Zhong and Chen (2020)$^\diamond$	$\text{ALBERT}_{\text{XXLARGE}}$	-	-	**90.3**	-	-	62.2
	UNIRE$^\diamond$	$\text{BERT}_{\text{BASE}}$	87.4	88.0	87.7	62.1	58.0	60.0
	UNIRE$^\diamond$	$\text{ALBERT}_{\text{XXLARGE}}$	**88.9**	**90.0**	89.5	**67.3**	**59.3**	**63.0**
ACE05	Li and Ji (2014)	-	85.2	76.9	80.8	65.4	39.8	49.5
	Miwa and Bansal (2016)	LSTM	82.9	83.9	83.4	57.2	54.0	55.6
	Katiyar and Cardie (2017)	LSTM	84.0	81.3	82.6	55.5	51.8	53.6
	Sun et al. (2019)	LSTM	86.1	82.4	84.2	68.1	52.3	59.1
	Li et al. (2019)	$\text{BERT}_{\text{LARGE}}$	84.7	84.9	84.8	64.8	56.2	60.2
	Wang et al. (2020)	$\text{BERT}_{\text{BASE}}$	-	-	87.2	-	-	63.2
	Wang and Lu (2020)	$\text{ALBERT}_{\text{XXLARGE}}$	-	-	89.5	-	-	64.3
	Zhong and Chen (2020)$^\diamond$	$\text{BERT}_{\text{BASE}}$	-	-	90.2	-	-	64.6
	Zhong and Chen (2020)$^\diamond$	$\text{ALBERT}_{\text{XXLARGE}}$	-	-	**90.9**	-	-	**67.8**
	UNIRE$^\diamond$	$\text{BERT}_{\text{BASE}}$	88.8	88.9	88.8	67.1	**61.8**	64.3
	UNIRE$^\diamond$	$\text{ALBERT}_{\text{XXLARGE}}$	**89.9**	**90.5**	90.2	**72.3**	60.7	66.0
SciERC	Wang et al. (2020)	SciBERT	-	-	68.0	-	-	34.6
	Zhong and Chen (2020)$^\diamond$	SciBERT	-	-	68.2	-	-	36.7
	UNIRE$^\diamond$	SciBERT	**65.8**	**71.1**	**68.4**	**37.3**	**36.6**	**36.9**

Table 3: Overall evaluation. $\diamond$ means that the model leverages cross-sentence context information.

relation type is correct, as well as the boundaries and types of two argument entities are correct.

Implementation Details We tune all hyper-parameters based on the averaged entity F1 and relation F1 on ACE05 development set, then keep the same settings on ACE04 and SciERC. For fair comparison with previous works, we use three pre-trained language models: `bert-base-uncased` (Devlin et al., 2019), `albert-xxlarge-v1` (Lan et al., 2019) and `scibert-scivocab-uncased` (Beltagy et al., 2019) as the sentence encoder and fine-tune them in training stage.[12]

For the MLP layer, we set the hidden size as $d = 150$ and use GELU as the activation function. We use AdamW optimizer (Loshchilov and Hutter, 2017) with $\beta_1 = 0.9$ and $\beta_2 = 0.9$, and observe a phenomenon similar to (Dozat and Manning, 2016) in that setting β_2 from 0.9 to 0.999 causes a significant drop on final performance. The batch size is 32, and the learning rate is 5e-5 with weight decay 1e-5. We apply a linear warm-up learning rate scheduler with a warm-up ratio of 0.2. We train our model with a maximum of 200 epochs (300 epochs for SciERC) and employ an early stop strategy. We perform all experiments on an Intel(R) Xeon(R) W-3175X CPU and a NVIDIA Quadro RTX 8000 GPU.

[12]The first two are for ACE04 and ACE05, and the last one is for SciERC.

4.1 Performance Comparison

Table 3 summarizes previous works and our UNIRE on three datasets.[13] In general, UNIRE achieves the best performance on ACE04 and SciERC and a comparable result on ACE05. Comparing with the previous best joint model (Wang and Lu, 2020), our model significantly advances both entity and relation performances, i.e., an absolute F1 of +0.9 and +0.7 for entity as well as +3.4 and +1.7 for relation, on ACE04 and ACE05 respectively. For the best pipeline model (Zhong and Chen, 2020) (current SOTA), our model achieves superior performance on ACE04 and SciERC and comparable performance on ACE05. Comparing with ACE04/ACE05, SciERC is much smaller, so entity performance on SciERC drops sharply. Since (Zhong and Chen, 2020) is a pipeline method, its relation performance is severely influenced by the poor entity performance. Nevertheless, our model is less influenced in this case and

[13]Since (Luan et al., 2019a; Wadden et al., 2019) neglect the argument entity type in relation evaluation and underperform our baseline (Zhang et al., 2020), we do not compare their results here.

Settings	ACE05		SciERC	
	Ent	Rel	Ent	Rel
Default	88.8	**64.3**	**68.4**	36.9
w/o symmetry loss	88.9	64.0	67.3	35.5
w/o implication loss	**89.0**	63.3	68.0	**37.1**
w/o logit dropout	88.8	61.8	66.9	34.7
w/o cross-sentence context	87.9	62.7	65.3	32.1
hard decoding	74.0	34.6	46.1	17.8

Table 4: Results (F1 score) with different settings on ACE05 and SciERC test sets. Note that we use $\text{BERT}_{\text{BASE}}$ on ACE05.

Model	Parameters	W	ACE05		SciERC	
			Rel (F1)	Speed (sent/s)	Rel (F1)	Speed (sent/s)
Z&C(2020)	219M	100	**64.6**	14.7	36.7	19.9
Z&C(2020)†	219M	100	-	237.6	-	194.7
UniRE	110M	100	63.6	**340.6**	34.0	**314.8**
UniRE	110M	200	64.3	194.2	**36.9**	200.1
hard decoding	110M	200	34.6	139.1	17.8	113.0

Table 5: Comparison of accuracy and efficiency on ACE05 and SciERC test sets with different context window sizes. † denotes the approximation version with a faster speed and a worse performance.

achieves better performance. Besides, our model can achieve better relation performance even with worse entity results on ACE04. Actually, our base model ($\text{BERT}_{\text{BASE}}$) has achieved competitive relation performance, which even exceeds prior models based on $\text{BERT}_{\text{LARGE}}$ (Li et al., 2019) and $\text{ALBERT}_{\text{XXLARGE}}$ (Wang and Lu, 2020). These results confirm the proposed unified label space is effective for exploring the interaction between entities and relations. Note that all subsequent experiment results on ACE04 and ACE05 are based on $\text{BERT}_{\text{BASE}}$ for efficiency.

4.2 Ablation Study

In this section, we analyze the effects of components in UniRE with different settings (Table 4). Particularly, we implement a naive decoding algorithm for comparison, namely "hard decoding", which takes the "intermediate table" as input. The "intermediate table" is the hard form of probability tensor $\mathcal{P}$ output by the biaffine model, i.e., choosing the class with the highest probability as the label of each cell. To find entity squares on the diagonal, it first tries to judge whether the largest square ($|s| \times |s|$) is an entity. The criterion is simply counting the number of different entity labels appearing in the square and choosing the most frequent one. If the most frequent label is $\perp$, we shrink the size of square by 1 and do the same work on two $(|s|-1) \times (|s|-1)$ squares and so on. To avoid entity overlapping, an entity will be discarded if it overlaps with identified entities. To find relations, each entity pair is labeled by the most frequent relation label in the corresponding rectangle.

From the ablation study, we get the following observations.

- When one of the additional losses is removed, the performance will decline with varying degrees (line 2-3). Specifically, the symmetrical loss has a significant impact on SciERC (decrease 1.1 points and 1.4 points for entity and relation performance). While removing the implication loss will obviously harm the relation performance on ACE05 (1.0 point). It demonstrates that the structural information incorporated by both losses is useful for this task.

- Comparing with the "Default", the performance of "w/o logit dropout" and "w/o cross-sentence context" drop more sharply (line 4-5). Logit dropout prevents the model from overfitting, and cross-sentence context provides more contextual information for this task, especially for small datasets like SciERC.

- The "hard decoding" has the worst performance (its relation performance is almost half of the "Default") (line 6). The major reason is that "hard decoding" separately decodes entities and relations. It shows the proposed decoding algorithm jointly considers entities and relations, which is important for decoding.

4.3 Inference Speed

Following (Zhong and Chen, 2020), we evaluate the inference speed of our model (Table 5) on ACE05 and SciERC with the same batch size and pre-trained encoders ($\text{BERT}_{\text{BASE}}$ for ACE05 and SciBERT for SciERC). Comparing with the pipeline method (Zhong and Chen, 2020), we obtain a more than $10\times$ speedup and achieve a comparable or even better relation performance with $W = 200$. As for their approximate version, our inference speed is still competitive but with better performance. If the context window size is set the same as (Zhong and Chen, 2020) ($W = 100$), we can further accelerate model inference with slight performance drops. Besides, "hard decoding" is

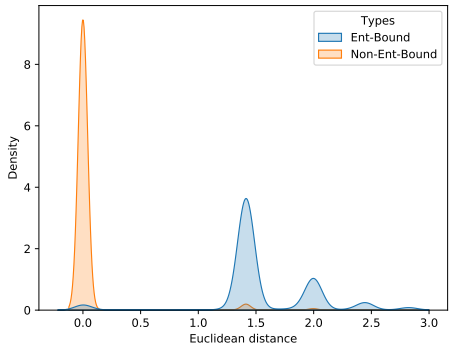

Figure 4: Distributions of adjacent rows' distances for two categories with respect to the threshold α on ACE05 dev set.

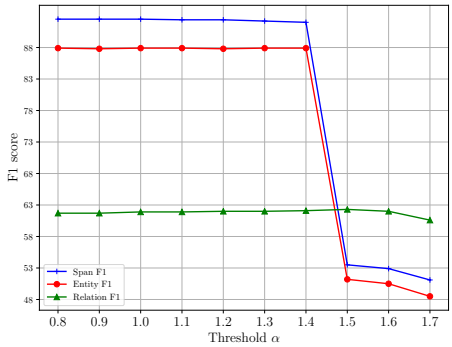

Figure 5: Performances with respect to the threshold α on ACE05 dev set.

much slower than UNIRE, which demonstrates the efficiency of the proposed decoding algorithm.

4.4 Impact of Different Threshold α

In Figure 4, the distance between adjacent rows not at entity boundary ("Non-Ent-Bound") mainly concentrates at 0, while that at entity boundary ("Ent-Bound") is usually greater than 1. This phenomenon verifies the correctness of our span decoding method. Then we evaluate the performances, with regard to the threshold α in Figure 5.[14] Both span and entity performances sharply decrease when α increases from 1.4 to 1.5, while the relation performance starts to decline slowly from $\alpha = 1.5$. The major reason is that relations are so sparse that many entities do not participate in any relation, so the threshold of relation is much higher than that of entity. Moreover, we observe a similar phenomenon on ACE04 and SciERC, and $\alpha = 1.4$ is a general best setting on three datasets. It shows the stability and generalization of our model.

[14]We use an additional metric to evaluate span performance, "Span F1", is Micro-F1 of predicted split positions.

	Value	ACE05 Ent	ACE05 Rel	SciERC Ent	SciERC Rel
W	100	87.4	**62.4**	69.0	36.7
	200	**87.9**	62.1	**70.6**	**38.3**
	300	87.2	60.8	69.4	35.4
p	0.1	87.4	61.8	**71.1**	37.8
	0.2	**87.9**	**62.1**	70.6	**38.3**
	0.3	87.2	**62.1**	67.8	33.5
	0.4	87.4	62.0	70.6	35.8

Table 6: Results (F1 scores) with respect to the context window size and the logit dropout rate on ACE05 and SciERC dev sets.

4.5 Context Window and Logit Dropout Rate

In Table 4, both cross-sentence context and logit dropout can improve the entity and relation performance. Table 6 shows the effect of different context window size W and logit dropout rate p. The entity and relation performances are significantly improved from $W = 100$ to $W = 200$, and drop sharply from $W = 200$ to $W = 300$. Similarly, we achieve the best entity and relation performances when $p = 0.2$. So we use $W = 200$ and $p = 0.2$ in our final model.

4.6 Error Analysis

We further analyze the remaining errors for relation extraction and present the distribution of five errors: span splitting error (SSE), entity not found (ENF), entity type error (ETE), relation not found (RNF), and relation type error (RTE) in Figure 6. The proportion of "SSE" is relatively small, which proves the effectiveness of our span decoding method. Moreover, the proportion of "not found error" is significantly larger than that of "type error" for both entity and relation. The primary reason is that the table filling suffers from the class imbalance issue, i.e., the number of $\perp$ is much larger than that of other classes. We reserve this imbalanced classification problem in the future.

Finally, we give some concrete examples in Figure 7 to verify the robustness of our decoding algorithm. There are some errors in the biaffine model's prediction, such as cells in the upper left corner (first example) and upper right corner (second example) in the intermediate table. However, these errors are corrected after decoding, which demonstrates that our decoding algorithm not only recover all entities and relations but also corrects errors leveraging table structure and neighbor cells' information.

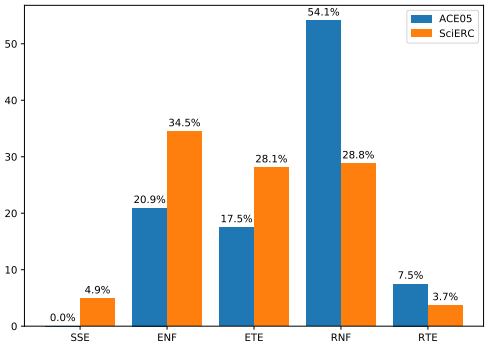

Figure 6: Distribution of five relation extraction errors on ACE05 and SciERC test data.

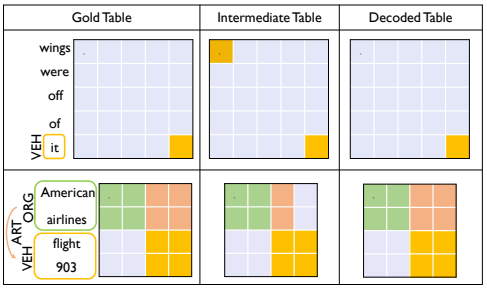

Figure 7: Examples showing the robustness of our decoding algorithm. "Gold Table" presents the gold label. "Intermediate Table" presents the biaffine model's prediction (choosing the label with the highest probability for each cell). "Decoded Table" presents the final results after decoding.

5 Related Work

Entity relation extraction has been extensively studied over the decades. Existing methods can be roughly divided into two categories according to the adopted label space.

Separate Label Spaces This category study this task as two separate sub-tasks: entity recognition and relation classification, which are defined in two separate label spaces. One early paradigm is the pipeline method (Zelenko et al., 2003; Miwa et al., 2009) that uses two independent models for two sub-tasks respectively. Then joint method handles this task with an end-to-end model to explore more interaction between entities and relations. The most basic joint paradigm, parameter sharing (Miwa and Bansal, 2016; Katiyar and Cardie, 2017), adopts two independent decoders based on a shared encoder. Recent span-based models (Luan et al., 2019b; Wadden et al., 2019) also use this paradigm. To enhance the connection of two decoders, many joint decoding algorithms are proposed, such as ILP-based joint decoder (Yang and Cardie, 2013), joint MRT (Sun et al., 2018), GCN-based joint inference (Sun et al., 2019). Actually, table filling method (Miwa and Sasaki, 2014; Gupta et al., 2016; Zhang et al., 2017; Wang et al., 2020) is a special case of parameter sharing in table structure. These joint models all focus on various joint algorithms but ignore the fact that they are essentially based on separate label spaces.

Unified Label Space This family of methods aims to unify two sub-tasks and tackle this task in a unified label space. Entity relation extraction has been converted into a tagging problem (Zheng et al., 2017), a transition-based parsing problem (Wang et al., 2018), and a generation problem with Seq2Seq framework (Zeng et al., 2018; Nayak and Ng, 2020). We follow this trend and propose a new unified label space. We introduce a 2D table to tackle the overlapping relation problem in (Zheng et al., 2017). Also, our model is more versatile as not relying on complex expertise like (Wang et al., 2018), which requires external expert knowledge to design a complex transition system.

6 Conclusion

In this work, we extract entities and relations in a unified label space to better mine the interaction between both sub-tasks. We propose a novel table that presents entities and relations as squares and rectangles. Then this task can be performed in two simple steps: filling the table with our biaffine model and decoding entities and relations with our joint decoding algorithm. Experiments on three benchmarks show the proposed method achieves not only state-of-the-art performance but also promising efficiency.

Acknowledgement

The authors wish to thank the reviewers for their helpful comments and suggestions. This work was (partially) supported by National Key Research and Development Program of China (2018AAA0100704), NSFC (61972250, 62076097), STCSM (18ZR1411500), Shanghai Municipal Science and Technology Major Project (2021SHZDZX0102), and the Fundamental Research Funds for the Central Universities.

References

Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. Scibert: A pretrained language model for scientific text. *arXiv preprint arXiv:1903.10676*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

George Doddington, Alexis Mitchell, Mark Przybocki, Lance Ramshaw, Stephanie Strassel, and Ralph Weischedel. 2004. The automatic content extraction (ACE) program – tasks, data, and evaluation. In *Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC'04)*, Lisbon, Portugal. European Language Resources Association (ELRA).

Timothy Dozat and Christopher D Manning. 2016. Deep biaffine attention for neural dependency parsing. *arXiv preprint arXiv:1611.01734*.

Pankaj Gupta, Hinrich Schütze, and Bernt Andrassy. 2016. Table filling multi-task recurrent neural network for joint entity and relation extraction. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 2537–2547, Osaka, Japan. The COLING 2016 Organizing Committee.

Arzoo Katiyar and Claire Cardie. 2017. Going out on a limb: Joint extraction of entity mentions and relations without dependency trees. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 917–928, Vancouver, Canada. Association for Computational Linguistics.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*.

Qi Li and Heng Ji. 2014. Incremental joint extraction of entity mentions and relations. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 402–412, Baltimore, Maryland. Association for Computational Linguistics.

Xiaoya Li, Fan Yin, Zijun Sun, Xiayu Li, Arianna Yuan, Duo Chai, Mingxin Zhou, and Jiwei Li. 2019. Entity-relation extraction as multi-turn question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1340–1350, Florence, Italy. Association for Computational Linguistics.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

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*, pages 3219–3232, Brussels, Belgium. Association for Computational Linguistics.

Yi Luan, Dave Wadden, Luheng He, Amy Shah, Mari Ostendorf, and Hannaneh Hajishirzi. 2019a. A general framework for information extraction using dynamic span graphs. *arXiv preprint arXiv:1904.03296*.

Yi Luan, Dave Wadden, Luheng He, Amy Shah, Mari Ostendorf, and Hannaneh Hajishirzi. 2019b. A general framework for information extraction using dynamic span graphs. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3036–3046, Minneapolis, Minnesota. Association for Computational Linguistics.

Makoto Miwa and Mohit Bansal. 2016. End-to-end relation extraction using LSTMs on sequences and tree structures. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1105–1116, Berlin, Germany. Association for Computational Linguistics.

Makoto Miwa, Rune Sætre, Yusuke Miyao, and Jun'ichi Tsujii. 2009. A rich feature vector for protein-protein interaction extraction from multiple corpora. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 121–130, Singapore. Association for Computational Linguistics.

Makoto Miwa and Yutaka Sasaki. 2014. Modeling joint entity and relation extraction with table representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1858–1869.

Tapas Nayak and Hwee Tou Ng. 2020. Effective modeling of encoder-decoder architecture for joint entity and relation extraction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8528–8535.

Changzhi Sun, Yeyun Gong, Yuanbin Wu, Ming Gong, Daxin Jiang, Man Lan, Shiliang Sun, and Nan Duan. 2019. Joint type inference on entities and relations via graph convolutional networks. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1361–1370, Florence, Italy. Association for Computational Linguistics.

Changzhi Sun, Yuanbin Wu, Man Lan, Shiliang Sun, Wenting Wang, Kuang-Chih Lee, and Kewen Wu.

2018. Extracting entities and relations with joint minimum risk training. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2256–2265, Brussels, Belgium. Association for Computational Linguistics.

Bruno Taillé, Vincent Guigue, Geoffrey Scoutheeten, and Patrick Gallinari. 2020. Let's stop incorrect comparisons in end-to-end relation extraction! In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3689–3701, Online. Association for Computational Linguistics.

David Wadden, Ulme Wennberg, Yi Luan, and Hannaneh Hajishirzi. 2019. Entity, relation, and event extraction with contextualized span representations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5784–5789, Hong Kong, China. Association for Computational Linguistics.

Christopher Walker, Stephanie Strassel, Julie Medero, and Kazuaki Maeda. 2006. Ace 2005 multilingual training corpus. *Linguistic Data Consortium, Philadelphia*, 57:45.

Jue Wang and Wei Lu. 2020. Two are better than one: Joint entity and relation extraction with table-sequence encoders. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1706–1721, Online. Association for Computational Linguistics.

Shaolei Wang, Yue Zhang, Wanxiang Che, and Ting Liu. 2018. Joint extraction of entities and relations based on a novel graph scheme. In *IJCAI*, pages 4461–4467.

Yijun Wang, Changzhi Sun, Yuanbin Wu, Junchi Yan, Peng Gao, and Guotong Xie. 2020. Pre-training entity relation encoder with intra-span and inter-span information. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1692–1705, Online. Association for Computational Linguistics.

Bishan Yang and Claire Cardie. 2013. Joint inference for fine-grained opinion extraction. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1640–1649.

Dmitry Zelenko, Chinatsu Aone, and Anthony Richardella. 2003. Kernel methods for relation extraction. *Journal of machine learning research*, 3(Feb):1083–1106.

Xiangrong Zeng, Daojian Zeng, Shizhu He, Kang Liu, and Jun Zhao. 2018. Extracting relational facts by an end-to-end neural model with copy mechanism. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 506–514.

Haoran Zhang, Qianying Liu, Aysa Xuemo Fan, Heng Ji, Daojian Zeng, Fei Cheng, Daisuke Kawahara, and Sadao Kurohashi. 2020. Minimize exposure bias of seq2seq models in joint entity and relation extraction. *arXiv preprint arXiv:2009.07503*.

Meishan Zhang, Yue Zhang, and Guohong Fu. 2017. End-to-end neural relation extraction with global optimization. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1730–1740, Copenhagen, Denmark. Association for Computational Linguistics.

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. *arXiv preprint arXiv:1706.05075*.

Zexuan Zhong and Danqi Chen. 2020. A frustratingly easy approach for joint entity and relation extraction. *arXiv preprint arXiv:2010.12812*.

Algorithm 1 Decoding Algorithm

Input: Probability tensor $\mathcal{P} \in \mathbb{R}^{|s|\times|s|\times|\mathcal{Y}|}$ of sentence s
Output: A set of entities $\mathcal{E}$ and a set of relations $\mathcal{R}$

$\mathcal{E}_{\text{split}} = [], \mathcal{E} = \texttt{set}(), \mathcal{R} = \texttt{set}()$
$\mathcal{P}^{\text{row}} \leftarrow \mathcal{P}.\texttt{view}(n, n * |\mathcal{Y}|)$
$\mathcal{P}^{\text{col}} \leftarrow \mathcal{P}.\texttt{transpose}(0, 1).\texttt{view}(n, n * |\mathcal{Y}|)$
$i \leftarrow 1$
while $i<|s|$ **do**
    $d \leftarrow \frac{||\mathcal{P}_i^{\text{row}} - \mathcal{P}_{i+1}^{\text{row}}||_2^2 + ||\mathcal{P}_i^{\text{col}} - \mathcal{P}_{i+1}^{\text{col}}||_2^2}{2}$
    if $d>\alpha$ **then**
        $\mathcal{E}_{\text{split}}.\texttt{append}(i)$
    end if
    $i \leftarrow i + 1$
end while
$\mathcal{E}_{\text{split}}.\texttt{append}(|s|)$
$i \leftarrow 1$
for $j \in \mathcal{E}_{\text{split}}$ **do**
    $\hat{t} = \arg\max_{t\in\mathcal{Y}_e\cup\{\perp\}} \texttt{Avg}(\mathcal{P}_{i:j,i:j,t})$
    if $\hat{t} \neq \perp$ **then**
        new e: $e.\texttt{span} = (i, j)$ and $e.\texttt{type} = \hat{t}$
        $\mathcal{E}.\texttt{add}(e)$
    end if
    $i \leftarrow j + 1$
end for
for $e_1, e_2 \in \mathcal{E}, e_1 \neq e_2$ **do**
    $(i, j) = e_1.\texttt{span}$
    $(m, n) = e_2.\texttt{span}$
    $\hat{l} = \arg\max_{l\in\mathcal{Y}_r\cup\{\perp\}} \texttt{Avg}(\mathcal{P}_{i:j,m:n,l})$
    if $\hat{l} \neq \perp$ **then**
        $\mathcal{R}.\texttt{add}((e_1, e_2, \hat{l}))$
    end if
end for

A Decoding Algorithm

A formal description are shown in Algorithm 1.

B Datasets

The ACE04 and ACE05 corpora are collected from various domains, such as newswire and online forums. Both corpora annotate 7 entity types and 6 relation types. we use the same data splits and preprocessing as (Li and Ji, 2014; Miwa and Bansal, 2016), i.e., 5-fold cross-validation for ACE04, and 351 training, 80 validating, and 80 testing for ACE05.[15] Besides, we randomly sample 10% of training set as the development set for ACE04.

The SciERC corpus collects 500 scientific abstracts taken from AI conference/workshop proceedings. This dataset annotates 6 entity types and 7 relation types. We adopt the same data split protocol as in (Luan et al., 2019b) (350 training, 50 validating, and 100 testing). Detailed dataset specifications are shown in Table 2.

[15]We use the pre-processing scripts provided by (Wang and Lu, 2020) at https://github.com/LorrinWWW/two-are-better-than-one/tree/master/datasets.

Moreover, we correct the annotations of undirected relations for three datasets, regarding each undirected relation as two directed relation instances, e.g., for the undirected relation PER-SOC, only one relation triplet ("his", wife", PER-SOC) is annotated in the original dataset, we will add another relation triplet ("wife", "his", PER-SOC) in our corrected datasets for symmetry. In this case, each undirected relation corresponds to two rectangles, which are symmetrical about the diagonal.