JBNU-CCLab at SemEval-2022 Task 12: Machine Reading Comprehension and Span Pair Classification for Linking Mathematical Symbols to Their Descriptions

Sung-Min Lee and **Seung-Hoon Na**
Division of Computer Science and Engineering, Jeonbuk National University, South Korea
{cap1232,nash}@jbnu.ac.kr

Abstract

This paper describes our system in the SemEval-2022 Task 12: 'linking mathematical symbols to their descriptions', achieving first on the leaderboard for all the subtasks comprising named entity extraction (NER) and relation extraction (RE). Our system is a two-stage pipeline model based on SciBERT that detects symbols, descriptions, and their relationships in scientific documents. The system consists of 1) machine reading comprehension(MRC)-based NER model, where each entity type is represented as a question and its entity mention span is extracted as an answer using an MRC model, and 2) span pair classification for RE, where two entity mentions and their type markers are encoded into span representations that are then fed to a Softmax classifier. In addition, we deploy a rule-based *symbol tokenizer* to improve the detection of the exact boundary of symbol entities. Regularization and ensemble methods are further explored to improve the RE model. [1]

1 Introduction

Mathematical symbols and descriptions appear in various forms across document section boundaries without explicit markups, and mathematical symbols appear in the form of long texts. Thus, linking mathematical symbols and their descriptions is challenging.

SemEval 2022 task 12: 'linking mathematical symbols to their descriptions (Lai et al., 2022a)', is a relation extraction task targeted at scientific documents divided into two sub-tasks: sub-task A is a **named entity recognition** (**NER**) task that aims to predict the span of symbols and descriptions, and sub-task B is a **relation extraction** (**RE**) task that aims to predict relations between symbols and descriptions.

[1]Our code is publicly available at https://github.com/ZIZUN/symlink.

Extracting these entities and relations is done to discover relational facts from unstructured texts. This problem can be decomposed into NER (Tjong Kim Sang and De Meulder, 2003; Ratinov and Roth, 2009) and RE (Zelenko et al., 2002; Bunescu and Mooney, 2005). Early works employed a two-stage relation extraction system, training one model to extract entities (Florian et al., 2004) and another model to classify relations between these entities (Zhou et al., 2005; Chan and Roth, 2011). To reduce the error propagation of NER or better capture the interactions between NER and RE, joint models have been proposed as a promising approach that are based on an end-to-end method or on the setting of multi-task learning using shared representations (Wadden et al., 2019; Lin et al., 2020; Wang and Lu, 2020).

Recently, it has been observed that RE based on the shared encoder is suboptimal, but the use of separated encoders for NER and RE has shown improved performance compared to shared encoders, reexamining the effectiveness of the simple pipelined two-stage approach (Zhong and Chen, 2021; Ye et al., 2021). From these results, we hypothesize that whereas separated encoders for NER and RE can learn customized representations useful for each task, joint models may include irrelevant information in the learned representation for NER or RE tasks, lowering the performance of the model.

These results of using distinct encoders (Zhong and Chen, 2021) encourage us to adopt the aforementioned two-stage approach for NER and RE tasks, consisting of 1) MRC-based NER and 2) span pair classification for RE, as follows:

1. **MRC-based NER using a symbol tokenizer**: Unlike the PURE system of (Zhong and Chen, 2021) that exploits the standard span-based NER of (Lee et al., 2017; Wadden et al., 2019), our NER model is based on an MRC-based model (Li et al., 2020), which treats NER as

Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022), pages 1679 - 1686

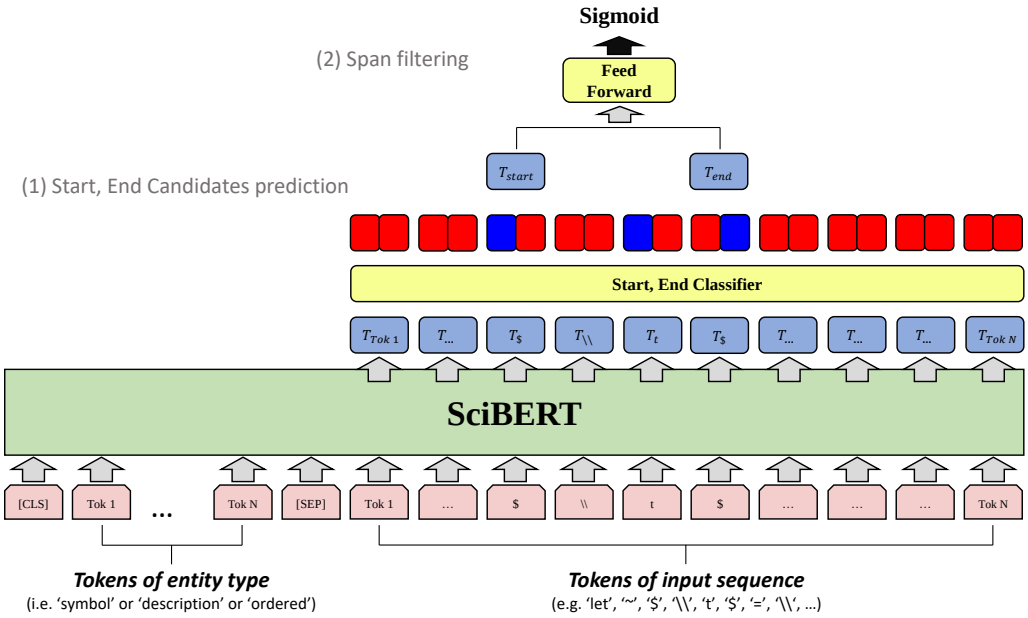

Figure 1: Our NER model architecture based on MRC

an MRC problem by providing an entity type as a question and using an MRC model to extract its entity mentions as answers. As in (Li et al., 2020), an MRC model is based on two binary classification models: first, the position classifier predicts the start and end indexes to create a set of valid answer spans, second, the span classifier determines whether each of the valid spans is an answer. As a pretrained encoder for the NER model, SciBERT of (Beltagy et al., 2019) is used. Before presenting to SciBERT's tokenizer, we apply a rule-based *symbol tokenizer* to precisely predict the span boundary of mathematical symbols that appear in scientific documents,

2. **Span pair classification for RE with solid markers**: Similar to the PURE system of (Zhong and Chen, 2021), a pair of spans resulting from the NER model is given as an input but with *solid markers*, i.e., using a typed entity marker, as in the works of (Wu and He, 2019; Zhou and Chen, 2021). The SciBERT encoder then uses this marked input to generate contextualized representations, which are then transformed to a pair of span representations and fed into a Softmax classifier. To define a set of relation types (or classes), the RE model explicitly adds a *NIL*-type class as a relation type to refer to the case in which a pair of spans has no relationship. It should be noted that the NER model's symbol tokenizer is not used in the RE model. Regularization methods such as *RDrop* (Wu et al., 2021) and *R3F* (Aghajanyan et al., 2020), as well as traditional ensemble techniques, are used to improve the performance of RE models[2].

The remainder of this paper is organized as follows: Section 2 presents our system architecture in detail, Sections 3-5 describe the experimental setting, results, and ablation studies, and Section 6 contains our concluding remarks and future works.

2 System Overview

In this section, we first describe the models of the proposed system for each sub-task.

2.1 MRC-based NER with a symbol tokenizer

Figure 1 shows our MRC-based NER model, that extracts mathematical symbols and descriptions from scientific documents.

2.1.1 Symbol tokenizer as a pre-tokenizer

We discovered in our preliminary experiment that SciBERT's tokenizer is not optimal for extracting the boundaries of mathematical symbols, because non-alphanumeric characters are important in the mentions of symbol-type entities. We perform a

[2]Regularization and ensemble methods were not adopted in the NER model.

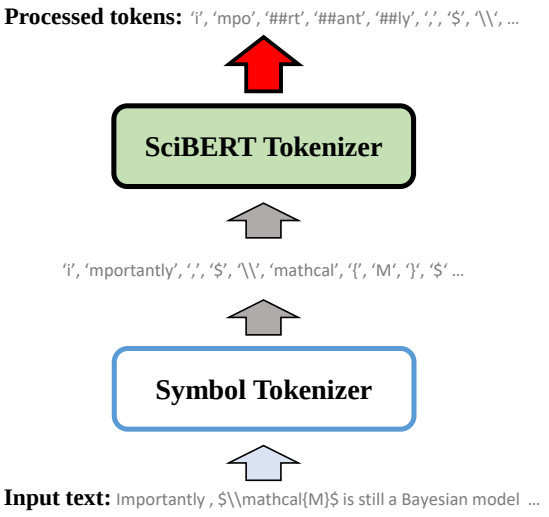

Figure 2: Two-step tokenization process for NER model with a symbol tokenizer

rule-based *symbol tokenizer* as a pre-tokenizer before applying SciBERT's tokenizer to precisely detect the boundary of symbol-type entities. Figure 2 presents this two-step tokenization adopted for the NER model.

The symbol tokenizer seperates mathematical symbols based on capital letters, numbers, and special characters (e.g., %, $, }, {). Our symbol tokenizer's rules are derived heuristically from a training dataset[3].

2.1.2 MRC models for nested NER

Given that the dataset addresses nested entities, we use the MRC model of (Li et al., 2020), that takes a *question-augmented* input. Specifically, suppose that $X = [x_1, \cdots, x_n]$ is a sequence of tokens in a scientific document, where n is the length of the sequence. Given a target entity type t, its natural language form $Q_t = [q_1, \cdots, q_m]$ is provided as a question based on Table 1, where q_i is the i-th token of Q_t and m is the length of the question. The question-augmented input X' is formulated as follows:

$X' =$[CLS], $q_1, \cdots, q_m$, [SEP], $x_1, \cdots, x_n$

Type	Text
SYMBOL	symbol
PRIMARY	description
ORDERED	ordered

Table 1: Natural language forms mapped for entity types

[3]This rule-based symbol tokenizer is also included in our codes.

Then, as a pre-trained language model, we apply SciBERT's encoder trained from scientific domain documents to obtain contextualized representations $T \in \mathbb{R}^{n \times d}$ over n tokens in a given document X, where d is the dimensionality of SciBERT's hidden representation.

The NER model predicts the probability of each token being a start or end index as follows:

$$\begin{aligned} P_{start} &= Sigmoid(FFN^{(start)}(T)) \in \mathbb{R}^n \\ P_{end} &= Sigmoid(FFN^{(end)}(T)) \in \mathbb{R}^n \end{aligned} \tag{1}$$

where $FFN^{(start)}$ ($FFN^{(end)}$) is a feed-forward neural network layer for predicting the start position and P_{start} (P_{end}) represents the probability of each index being the start (end) position of an entity, given a question entity type.

Based on Eq. (9), we obtain sets of predicted start and end indices as follows:

$$\begin{aligned} I_{start} &= \left\{ i \middle| \mathbb{1}(P^{(i)}_{start} > 0) \right\} \\ I_{end} &= \left\{ i \middle| \mathbb{1}(P^{(i)}_{end} > 0) \right\} \end{aligned} \tag{2}$$

where $P^{(i)}_{start}$ ($P^{(i)}_{end}$) is the i-th element of P^{start} (P^{end}) and $\mathbb{1}$ is an indicator function that gives 1 if an element is true, and 0 otherwise.

For any start index $i_{start} \in I_{start}$ and $i_{end} \in I_{end}$, a binary classifier is applied to predict whether the span of (i_{start}, i_{end}) becomes an answer, as follows:

$$\begin{aligned} &P_{i_{start}, i_{end}} = \\ &Sigmoid\left(FFN^{(span)}(T_{i_{start}}; T_{i_{end}})\right) \end{aligned} \tag{3}$$

where ; is the concatenation operator and $FFN^{(span)}$ is an additional feed-forward neural network layer for the span prediction.

Training As in (Li et al., 2020), the loss function for predicting the start and end positions is based on the cross-entropy term, which is formulated with probabilities of indexes being the start and end positions, as follows:

$$\begin{aligned} \mathcal{L}_{start} &= CE(P_{start}, Y_{start}) \\ \mathcal{L}_{end} &= CE(P_{end}, Y_{end}) \end{aligned} \tag{4}$$

where $Y_{start} \in \{0, 1\}^n$ and $Y_{end} \in \{0, 1\}^n$ represent the gold start and end positions, respectively of input tokens. The loss function for span probability is formulated as follows:

$$\mathcal{L}_{span} = CE(P_{start,end}, Y_{start,end}) \tag{5}$$

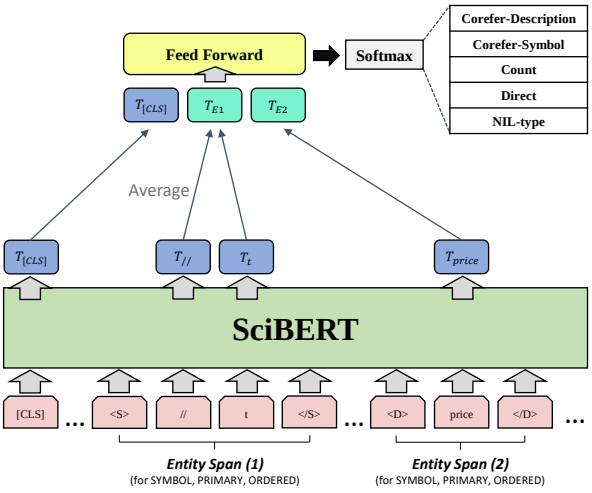

Figure 3: Span pair classification models for RE

where $Y_{start,end}$ represents the gold span of the input tokens. The overall loss is formulated as follows:

$$\mathcal{L} = \lambda_1 \mathcal{L}_{start} + \lambda_2 \mathcal{L}_{end} + \lambda_3 \mathcal{L}_{span} \quad (6)$$

where $\mathcal{L}_{start}$, $\mathcal{L}_{end}$, and $\mathcal{L}_{span}$ are the loss functions for predicting the start, end positions, and for the span prediction task, respectively, and λ_i is the weight for each loss function.

2.2 Span pair classification for RE with solid markers

Figure 3 represents the RE model based on the span pair classification of (Wu and He, 2019; Zhou and Chen, 2021), that classifies a pair of entity spans extracted from the MRC-based NER model in Section 2.1.2.

Like the NER model, we use SciBERT as a pre-trained language model for the RE model, but keep separate parameters that are not shared with the NER's encoder, following the work of (Zhong and Chen, 2021).

In the RE model, a *type-marked* document is provided as input. Specifically, suppose that e_1 and e_2 are a pair of entity spans (i.e., sequences of tokens), and their types are t_1 and t_2, respectively. Then, a type-marked document $\hat{X}$ is defined by prepending and appending *type markers* before and after each entity span, as follows:

$\hat{X}$ =[CLS] $\cdots$ <t_1> e_1 </t_1> $\cdots$ <t_2> e_2 </t_2> $\cdots$ where <t_i> and </t_1> are type markers.

Given $\hat{X}$, we apply SciBERT's encoder to obtain contextualized representations $T^{(rel)}$. We then obtain span representations for e_1 and e_2 by mean-pooling over their contextual representations, as follows:

$$H_{e_i} = \frac{1}{(end_{e_i} - start_{e_i} + 1)} \sum_{j=start_{e_i}}^{end_{e_i}} T_j^{(rel)} \quad (7)$$

where $start_{e_i}$ and end_{e_i} represent the start and end positions of e_i in the type-marked document $\hat{X}$, respectively. Finally, the model predicts the $P_{relation} \in \mathbb{R}^{l+1}$ probabilities over the relation types of e_1 and e_2 as follows:

$$P_{relation} = softmax(FFN^{(rel)}(T^{(rel)}_{[CLS]}; H_{e_1}; H_{e_2})) \quad (8)$$

where l is the number of relation types, NIL-type of relation is presented as $l+1$-th type, $FFN^{(rel)}$ is an additional feed-forward neural network for relation classification, and $T^{(rel)}_{[CLS]}$ (i.e., the contextual representation of the [CLS] token of $\hat{X}$) is concatenated to provide a global context over the entire document.

During training, the loss function for relation classification uses a cross-entropy function, which is formulated as follows:

$$\mathcal{L} = CE(P_{relation}, Y_{relation}) \quad (9)$$

where $Y_{relation} \in \{0,1\}^{l+1}$ represents a one-hot vector for the gold-relation label of a given pair of entities.

Tokenization Unlike the NER model in Section 2.1.2, only SciBERT's WordPiece tokenizer is exploited.

2.2.1 Automatic creation of examples for NIL-type class

Because we do not have explicit training examples for the NIL-type class, we use a simple negative sampling method to train the RE model. When a pair of entities appear in the context within the maximum length of tokens, they are considered *negative* samples (i.e., examples for the NIL-type class) when they do not have any relationship. The number of NIL-type samples collected in this manner, however, was more than 10 times that of normal samples. To correct the data imbalance, we use *oversampling* of (Chawla et al., 2002) on normal positive samples. Oversampling is also used to balance the positive, negative examples in the development set.

Model	NER				RE		
	Strict	**Exact**	**Partial**	**Type**	**Precision**	**Recall**	**F1 score**
Our System	-	-	47.61	47.70	32.09	38.56	35.03
+ *RDrop*	-	-	-	-	33.40	38.66	35.84
+ *R3F*	-	-	-	-	33.77	38.56	36.00
+ *R3F*, Ensemble	-	-	-	-	38.20	36.23	37.19

Table 2: Performances of final submission runs of our NER and RE models on the test dataset

3 Experimental setup

3.1 Dataset

We use the SemEval-2022 Task 12 dataset(Lai et al., 2022b) in our experiments.

The NER dataset contains three entity types: SYMBOL, PRIMARY, ORDERED. SYMBOL is a mathematical symbol, PRIMARY is a primary description, and ORDERED is a description of multiple terms.

The RE dataset contains four relation types: DIRECT, COUNT, COREFER-DESCRIPTION, and COREFER-SYMBOL. The dataset annotation guidelines[4] state that the relations should be directional; however, some of the relations, such as COREFER-SYMBOL, are *unidirectional*. COREFER-SYMBOL($\mathbf{E_1}$, $\mathbf{E_2}$) is the same as COREFER-SYMBOL($\mathbf{E_2}$, $\mathbf{E_1}$),

The directions of the other relations are defined based on the entity types. Such examples include COUNT($\mathbf{E_1}$, $\mathbf{E_2}$) and DIRECT($\mathbf{E_1}$, $\mathbf{E_2}$) where $\mathbf{E_1}$ is a symbol-type entity and $\mathbf{E_2}$ is a description-type entity. In postprocessing, the directions of these relations are automatically determined based on entity types in a post-processing manner. In other words, for the RE model, the order of the two entity spans e_1 and e_2 is determined based on their corresponding entity types.

Our system is evaluated separately for the NER and RE tasks. For NER, we use the entity-based strict/exact/partial/type from SemEval 2013 Task 9.1 (Segura-Bedmar et al., 2013). We use the standard precision, recall, f1-score metrics for RE.

3.2 Regularization and ensemble for RE model[5]

We use two regularization methods to improve the performance of the RE model: *RDrop* (Wu et al., 2021) and *R3F* (Aghajanyan et al., 2020). *Rdrop* is a regularization method that reduces the difference between representations at inference and training time caused by dropout, and *R3F* is a regularization method that maintains more generalizable representations of the pretrained language model during fine-tuning.

We train 10 RE models using different random seeds for the ensemble inference and and then perform maximum voting for each entity pair.

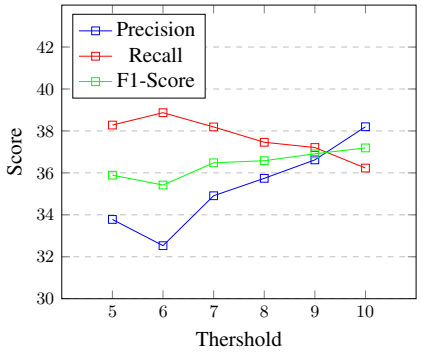

Figure 4: Performance comparison of ensembles of 10 RE models varying thresholds

4 Experimental results

Table 2 presents the final results on the blind test dataset[6].

As shown in Table 2, using the regularization method improves performance over the baseline model. Among the two methods, *R3F* is better than *RDrop*; thus, we use *R3F* for the submission of the RE model.

Overall, the recall is relatively higher than the precision for the RE model. In our preliminary experiments, we observed a similar tendency with high recall for the ensemble method, despite the fact that the ensemble method was shown to be effective in terms of F1 score.

We use a *voting threshold* to increase the pre-

[4]Official annotation guidelines are available at http://nlp.uoregon.edu/download/symlink/guideline.pdf

[5]These regularization and ensemble methods were not applied to NER model.

[6]Due to a submission error, we do not report strict/exact scores at the NER task.

cision of the ensemble RE model and adjust the ensemble inference so that a NIL-type class is assigned either when the size of the majority votes from the models does not exceed the voting threshold or when the class from the majority votes is NIL-type.

Figure 4 shows the performance of the ensemble method across different voting threshold values. In this case, we observe that as the voting threshold is raised, the F1 score gradually increases, while precision increases and recall decreases. As a result, when voting threshold is 10, the best performance of the F1 score is obtained. This run was finally submitted.

5 Analysis

In this section, we examine the effects of some of the components of our system as well as additional trials.

5.1 Effect of symbol tokenizer on NER task

Symbol Tokenizer	**Exact**	**Not Exact**	**Recall**
Used	18668	85	99.54
Unused	18412	341	98.18

Table 3: Frequencies and recalls of SYMBOL-type entities whose sequences of tokens are exact gold spans, with and without *symbol tokenizer*.

To examine the effect of the symbol tokenizer, Table 3 compares the frequencies and recalls of SYMBOL-type entities whose exact gold spans are correctly obtained when and without the symbol tokenizer. In this case, recall is defined as the ratio of the number of symbol-type entities whose exact span boundaries are *extractable* using a given tokenizer to the total number of symbol-type entities.

5.2 Effect of removing non-relational entities

Method	NER			
	Strict	**Exact**	**Partial**	**Type**
Not Excluded	-	-	47.61	47.70
Excluded	-	-	47.18	47.31

Table 4: Performances of NER models when including or excluding non-relational entities that have no relationship with other entities.

Assuming that mathematical symbols and descriptions must have one or more relations according to the annotation guideline, our additional trial is to exclude *non-relational* entities that have no relationship with other entities. Table 4 shows the performance of our NER models when those non-relational entities are included or excluded. However, it is observed that removing non-relational entities reduces the performance of the NER model.

5.3 Analysis of RE model

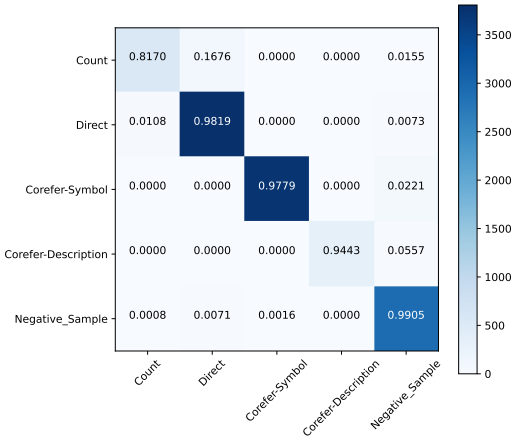

Figure 5: Confusion matrix of RE model on development dataset

Figure 5 shows the confusion matrix of the RE model. It is observed that the discrimination between COUNT and DIRECT is particularly challenging, and the effectiveness of COREFER-DESCRIPTION is relatively low. For this reason, there may be a low number of examples for COUNT and COREFER-DESCRIPTION labels. Given our assumption that these weak performances come from a lack of sufficient number of examples, data augmentation may need to be necessary to improve the performances of these relation labels.

6 Conclusion

Our system shows first for all subtasks of SemEval-2022 Task 12: 'linking mathematical symbols to their descriptions'. MRC-based NER and span pair classification for NER are part of our system that uses SciBERT as a backbone encoder. To improve the performance, the symbol tokenizer for NER model, regularization, and ensemble methods, for RE model are used.

To improve the performance further, future work should look into data augmentation and mathematical symbol and description-aware pretraining.

Acknowledgement

This work was supported by Institute of Information communications Technology Planning Evaluation (IITP) grant funded by the Korea government(MSIT) (No.2021-0-02068, Artificial Intelligence Innovation Hub)

References

Armen Aghajanyan, Akshat Shrivastava, Anchit Gupta, Naman Goyal, Luke Zettlemoyer, and Sonal Gupta. 2020. Better fine-tuning by reducing representational collapse. *arXiv preprint arXiv:2008.03156*.

Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. SciBERT: A pretrained language model for scientific text. 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 3615–3620, Hong Kong, China. Association for Computational Linguistics.

Razvan Bunescu and Raymond Mooney. 2005. A shortest path dependency kernel for relation extraction. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 724–731, Vancouver, British Columbia, Canada. Association for Computational Linguistics.

Yee Seng Chan and Dan Roth. 2011. Exploiting syntactico-semantic structures for relation extraction. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 551–560, Portland, Oregon, USA. Association for Computational Linguistics.

Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, and W. Philip Kegelmeyer. 2002. Smote: Synthetic minority over-sampling technique. 16(1):321–357.

R. Florian, H. Hassan, A. Ittycheriah, H. Jing, N. Kambhatla, X. Luo, N. Nicolov, and S. Roukos. 2004. A statistical model for multilingual entity detection and tracking. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics: HLT-NAACL 2004*, pages 1–8, Boston, Massachusetts, USA. Association for Computational Linguistics.

Viet Lai, Amir Pouran Ben Veyseh, Franck Dernoncourt, and Thien Huu Nguyen. 2022a. Semeval 2022 task 12: Symlink: Linking mathematical symbols to their descriptions. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Viet Dac Lai, Amir Pouran Ben Veyseh, Franck Dernoncourt, and Thien Huu Nguyen. 2022b. Symlink: A new dataset for scientific symbol-description linking.

Kenton Lee, Luheng He, Mike Lewis, and Luke Zettlemoyer. 2017. End-to-end neural coreference resolution. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 188–197, Copenhagen, Denmark. Association for Computational Linguistics.

Xiaoya Li, Jingrong Feng, Yuxian Meng, Qinghong Han, Fei Wu, and Jiwei Li. 2020. A unified MRC framework for named entity recognition. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5849–5859, Online. Association for Computational Linguistics.

Ying Lin, Heng Ji, Fei Huang, and Lingfei Wu. 2020. A joint neural model for information extraction with global features. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7999–8009, Online. Association for Computational Linguistics.

Lev Ratinov and Dan Roth. 2009. Design challenges and misconceptions in named entity recognition. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL-2009)*, pages 147–155, Boulder, Colorado. Association for Computational Linguistics.

Isabel Segura-Bedmar, Paloma Martínez, and María Herrero-Zazo. 2013. SemEval-2013 task 9 : Extraction of drug-drug interactions from biomedical texts (DDIExtraction 2013). In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 341–350, Atlanta, Georgia, USA. Association for Computational Linguistics.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, pages 142–147.

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.

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.

Lijun Wu, Juntao Li, Yue Wang, Qi Meng, Tao Qin, Wei Chen, Min Zhang, Tie-Yan Liu, et al. 2021. R-drop: regularized dropout for neural networks. *Advances in Neural Information Processing Systems*, 34.

Shanchan Wu and Yifan He. 2019. Enriching pre-trained language model with entity information for relation classification. In *Proceedings of the 28th ACM international conference on information and knowledge management*, pages 2361–2364.

Deming Ye, Yankai Lin, and Maosong Sun. 2021. Pack together: Entity and relation extraction with levitated marker. *arXiv preprint arXiv:2109.06067*.

Dmitry Zelenko, Chinatsu Aone, and Anthony Richardella. 2002. Kernel methods for relation extraction. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP 2002)*, pages 71–78. Association for Computational Linguistics.

Zexuan Zhong and Danqi Chen. 2021. A frustratingly easy approach for entity and relation extraction. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 50–61, Online. Association for Computational Linguistics.

GuoDong Zhou, Jian Su, Jie Zhang, and Min Zhang. 2005. Exploring various knowledge in relation extraction. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 427–434, Ann Arbor, Michigan. Association for Computational Linguistics.

Wenxuan Zhou and Muhao Chen. 2021. An improved baseline for sentence-level relation extraction. *arXiv preprint arXiv:2102.01373*.

A Comparison of regularization methods

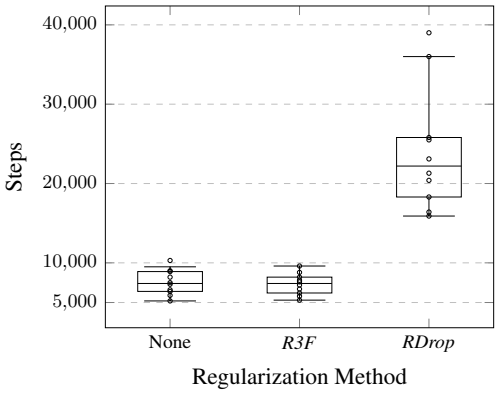

Figure 6: Comparison of the number of steps required for regularization methods in RE models.

We tried *RDrop* and *R3F* as regularization methods, and there were differences in terms of not only performance but also the *training time*. To compare training time, we measured the number of steps required for training our RE model. The results are shown in Figure 6.

B Hyper-parameters

NER model	
Sliding window	100
Dropout rate	0.1
Learning rate	3e-5
λ_1, λ_2, λ_3	1, 1, 0.1
Warmup steps	1000
Scheduler	OneCycle
Optimizer	AdamW
Max length	512
Batch size	2
Accumulation steps	5
Span classifier Inter hidden	2048
RE model	
Dropout rate	0.1
Learning rate	4e-5
Warmup steps	1000
Scheduler	Cosine
Optimizer	AdamW
Max length	512
Batch size	32
Accumulation steps	2

Table 5: Hyper-parameter settings

Table 5 shows the setup of hyper-parameters of our NER and RE models. We ran the experiments using 4 TITAN RTX(24GB) GPUs.