End-to-End Trainable Soft Retriever for Low-resource Relation Extraction

Kohei Makino, Makoto Miwa and Yutaka Sasaki *
Toyota Technological Institute, 2-12-1 Hisakata, Tempaku-ku, Nagoya, Japan

Abstract. This study addresses a crucial challenge in instance-based relation extraction using text generation models: end-to-end training in target relation extraction task is not applicable to retrievers due to the non-differentiable nature of instance selection. We propose a novel End-to-end TRAinable Soft K-nearest neighbor retriever (ETRASK) by the neural prompting method that utilizes a soft, differentiable selection of the k nearest instances. This approach enables the end-to-end training of retrievers in target tasks. On the TACRED benchmark dataset with a low-resource setting where the training data was reduced to 10%, our method achieved a state-of-the-art F1 score of 71.5%. Moreover, ETRASK consistently improved the baseline model by adding instances for all settings. These results highlight the efficacy of our approach in enhancing relation extraction performance, especially in resource-constrained environments. Our findings offer a promising direction for future research with extraction and the broader application of text generation in natural language processing.

Keywords: Low-resource Relation extraction, Soft Retriever, Instance-based model

1. Introduction

Relation extraction is a fundamental task in Natural Language Processing (NLP). It involves identifying and classifying semantic relationships between entity mentions within text. This task plays a crucial role in understanding and interpreting the underlying meaning of sentences by analyzing how different entities, such as people, organizations, and locations, are interrelated [1–3].

The significance of relation extraction extends beyond theoretical interests; it has practical applications in several domains. For instance, in knowledge graph construction, relation extraction aids in transforming unstructured text into structured data, which can then be used to populate and enrich knowledge graphs [4]. In domain-specific scenarios, such as biomedical text mining [5], it can extract relationships between genes, diseases, and drugs, which are helpful for advanced research and discovery such as search systems [6] and prediction of novel things [7].

Developing relation extractors has been an ongoing endeavor in NLP. Relation extractors are designed to accurately identify and classify relationships from text, which requires understanding the nuanced and often complex language structures. Advances in machine learning and NLP methodologies have shaped the evolution of these extractors. Most recent extractors are based on deep learning models to obtain high-performance [8, 9], while traditional extractors are rule-based models [10] and feature-based models [11].

The Pretrained Language Models (PLMs) [12–14] have become a de-facto standard in NLP, fundamentally changing the field landscape. PLMs are models pretrained on a general task, such as masked language modeling, and are used by fine-tuning it to fit a target task. A lot of studies on relation extraction with PLM have been conducted because of the high performance. [15, 16]

*Corresponding author. E-mail: yutaka.sasaki@toyota-ti.ac.jp.

With the advent of PLM, well-trained text generation models are attracting attention [12, 17, 18], including Large Language Model (LLM) [19, 20], which are trained on a larger corpus with larger-scale parameters than PLMs. Concerning relation extraction, text generation model-based relation extractors have been used innovatively [21–23]. These models have been adapted to treat relation extraction as a question answering task [21], a text summarization task [24], or a prompt-based generation task [22], leveraging their knowledge and understanding of natural language.

Relation extraction instance utilization pushes up the performance of relation extraction with text generation. For the relation extraction with PLM encoder that embeds text into feature representations, which can adapt to the task with fine-tuning, the relation extraction with inference process of the k-nearest neighbor (kNN) algorithm [25], which is the model to find the top-k similar instances on the encoding space of PLM and predicts the class of the instance, enhances the performance for relation labels without enough training data. However, the inference with simple voting is weak in terms of the overall performance compared to other models with devised prediction methods [25].

Combining text generation models with instance utilization has been particularly effective with In-Context Learning (ICL) [19] and Retrieval-Augmented Generation (RAG) [26]. In relation extraction via ICL, instructions and instances for extracting relations are used as prompts to supervise text generation models from context and output the results by predicting the following context convertible to relation labels [22]. A typical way to create prompts in ICL is by adjusting instructions and using pre-defined instances. However, standard ICL methods cannot utilize instances adaptable to input because the prompt and the model parameters are fixed.

To solve these problems, relation extraction via RAG introduces the instances relevant to the input into the text prompt [27]. Generally, this is done by embedding the inputs and instances into dense feature space by a separate model from the base text generation model and selecting the nearest neighbors. The chosen instances are then introduced into the text prompt, allowing for prediction while adding new perspectives in ICL. It is necessary to train a separate model for retrieval to introduce desired instances. However, text prompts disconnect the differentiability essential for deep learning training, making training a retrieval model for the relation extraction task difficult. This non-differentiability poses significant limitations for end-to-end optimization, as it prevents the direct optimization of instance selection within the learning process of the target relation extraction task.

Thus, instance utilization can boost performance, where ICL and RAG use fixed and adjustably-selected instances via text generation models. kNN relation extractor enhances relation extraction performance for long-tail labels. However, the inference capability of the kNN algorithm is insufficient. In addition, when the instances are used with RAG, another model is required to use adaptive instances for the input, and its training is done separately from relation extraction. To realize a relation extractor using instances with reasoning capability, it is necessary to make two processes differentiable: the retrieval process for selecting relation extraction instances and the embedding process for prompt creation.

To overcome the non-differentiable processes, our study introduces an innovative approach that renders the selection of the k nearest instances differentiable for the retrieval process and employs soft prompt obtained from the retrieval process like neural prompting [28] for the embedding process. The differentiable k instance selection selects instances softly inspired by neural nearest neighbor networks [29]. This allows the kNN retriever to utilize k instances as differentiable soft prompts in neural prompting, thereby enabling, for the first time, the end-to-end optimization of the retriever in generation-based relation extraction, where the retriever is named ETRASK (End-to-end TRAinable Soft KNN retriever).

This study addresses these challenges by introducing the retrieval process composed of a differentiable operation and the embedding process via neural prompting. By incorporating techniques from neural nearest-neighbor networks, our method enables using selected instances as soft prompts, facilitating end-to-end optimization in generation-based relation extraction systems. Evaluation of the extraction performance confirms the improved performance in the relation extraction for the low-resource settings, with state-of-the-art performance obtained for the setting of 10% training data in the TACRED dataset. A comparison with the baseline without using instances obtained by ETRASK shows that the performance is consistently improved in all settings. Analysis of instances acquired with trained retriever confirms that instances with the same relation labels as the target of extraction and instances where the sentence contains entities are selected as relevant for relation extraction.

The remainder of this paper is organized as follows: Section 2 presents related work, further elaborating on the evolution and current state of PLM (Section 2.1), instance-based methods (Section 2.2, and relation extraction (Section 2.3). Section 3 describes our methodology and consists of a description of relation extraction (Section 3.1) and *k*NN retriever with a text generation model (Section 3.2) for preparation, and a description of the proposed method ETRASK (Section 3.3), which consists of differentiable *k*NN (Section 3.3.1), neural prompting (Section 3.3.2) and training methods (Section 3.3.3). The evaluations are performed in Section 4 for the extraction performance and Section 5 for the retriever analysis. Finally, Section 6 concludes this study and describes future directions.

2. Related Work

2.1. Pretrained Language Models

PLMs are large-scale models trained on a large corpus with NLP tasks. PLMs are often used for various models for target NLP tasks with fine-tuning. Pretraining tasks are extensive such as language modeling (LM) [17], bidirectional language modeling (biLM) [12], sequence-to-sequence language modeling (Seq2Seq LM) [18], and masked language modeling (MLM) [13]. PLMs employ appropriate neural network structures such as ELMo [12] for the bidirectional language modeling task with LSTMs [30], and Transformer [31] is mainly used these days including BERT [13] for masked language modeling, T5 [18] for sequence-to-sequence language modeling, and GPT-2 [17] for language modeling. LLMs are extended PLMs to be larger scale and use a larger corpus such as Flan-T5 [32] in Seq2Seq LM and GPT-3 in LM [19].

Such PLMs are stochastic models to estimate the likelihood of text sequence based on the pretraining tasks. Since LM is a task to predict the following text from a given input, the model trained on LM can estimate the likelihood of input text sequence, formulated as $P(x)$ with input x. On the other hand, the model trained on Seq2Seq LM can assign a likelihood to input-output pairs used in pretraining, formulated as $P(y|x)$ with input x and output y. Collectively, these are text generation models.

PLMs are computationally challenging to utilize by tuning the entire parameter when adapted to the target task and are rarely used outside the text generation framework. Thus, they can be tuned by controlling the generated text with text prompts or by parameter-efficient training. Low-Rank Adaptation (LoRA) [33] inserts low-rank parameters into the base model to learn the difference between the target task. Prompt tuning [34] injects soft prompts to embedded text sequence and trains the soft prompts. The prompt tuning is extended to neural prompting [28] to soft prompts based on the instances. We utilize the LoRA in our experiment to update PLM and the neural prompting in our method to introduce instances.

2.2. Instance-based Methods

Instance-based methods, as typified by nearest neighbor method [35], have been used for various NLP tasks, such as POS tagging [36], named entity recognition [37], dependency parsing [38], and relation extraction [39]. The methods are used to mitigate a training data scarcity situation. This paper replaces the non-differentiable operations in the *k*NN algorithm with differentiable ones in order to relax to a soft operation.

Recently, PLMs are often used following the pretraining task to use instances by ICL [19], which predicts the following context from the given prompt. Since ICL is characterized by the prompt design, the instance selection performs a vital role. Since fixed prompts are usually used, the instances are also typically fixed. However, the demand to select the most appropriate instances for input has led to using RAG [26], which retrains a retriever in advance and uses the instances obtained by the retriever. Since the instance selection operation is not differentiable, the general retriever is trained separately to target the task [27]. This study tackles this separate training of the target task model and the retriever so that it can be trained end-to-end with the target task.

2.3. Relation Extraction

The fundamental relation extraction task is sentence-level relation extraction [40, 41], where only the entity pairs contained in a sentence are targets for extraction. Since sentence-level relation extraction ignores relations across sentences, it is extended to document-level relation extraction [42, 43], which also targets these relations for extraction. Since this paper focuses on retriever and a single relation, which is supposed to be a retriever query, is included in a single instance, we develop an extractor primarily on sentence-level relation extraction for simplicity.

Historically, relation extractors have evolved from traditional approaches to modern deep learning techniques. Initially, rule-based systems relied on hand-crafted rules to identify relationships [10]. Kernel-based models and feature-based approaches later emerged, offering more flexibility and better handling linguistic variations [11, 44]. Deep learning revolutionized the field, introducing models that could learn complex patterns and relationships directly from data [45].

PLMs and LLMs also become standard in the relation extraction task. PLMs are often treated as feature extractors in relation extractors and are used for direct classification based on their features [46] or connected to relation extraction-specific model [47]. On the other hand, LLMs are used as the text generation model with prompt engineering [23] or fine-tuning [22]. For example, SuRE (Summarization as Relation Extraction) [24] extracts the relation by summarization via LLM and mapping the output to relations. The SuRE measures the likelihood of a pair of text prompts of input and verbalized relations into a summary form to predict the relation with the highest likelihood.

Despite recent advancements with deep learning models, relation extraction remains challenging, primarily due to the scarcity of annotated data. Deep learning models, in particular, require large amounts of labeled data for training [48]. However, manually annotating data is expensive and time-consuming, making it a significant bottleneck in developing effective relation extraction systems [49]. This environment makes it difficult to classify with high performance for the types of relations complex to collect diverse data. In this context, the relation extractor via the nearest neighbor approach [25], where relation extractors leverage similar instances during inference, has shown promise in efficiently utilizing limited data. This study reveals that using instances, or specific instances of relations within texts, is crucial in relation extraction. By leveraging these instances, models can better generalize from limited instances and improve their ability to extract and classify relations in varied contexts accurately.

3. Methodology

We propose the end-to-end trainable soft k-nearest neighbor retriever ETRASK that retrieves virtually selected k-nearest instances composed of entirely differentiable operations, allowing end-to-end training. To make retriever selection differentiable, we use a novel algorithm for the differentiable selection of k-nearest instances to enhance neural prompting in relation extraction. The core of our method is detailed in Algorithm 1, which describes a procedure for soft selection of instances based on their embeddings.

We will explain the method in the following sections: Before describing our proposal, we generalize relation extraction models by text generation model in Section 3.1 and explain the kNN retriever in Section 3.2. Then Section 3.3 describes ETRASK consisting of differentiable k-nearest instance selection (Section 3.3.1) and integration of the instances drawn by it (Section 3.3.2). The end-to-end training and training techniques for ETRASK are shown in Section 3.3.3.

3.1. Relation Extraction with Text Generation Model

This section defines the structure and function of relation extractors using text generation models and their enhancements via a retriever. Consider an input sentence x, with h and t representing the head and tail entities of a target entity pair and $r \in R$ denoting their relationship. The standard relation extractors represented as Equation (1) formulate as a classification task using a stochastic model P.

$$\hat{r} = \operatorname*{argmax}_{r \in R} P(r|x, h, t) \tag{1}$$

Typically, for generation-based relation extraction, the input x, h, and t are reformulated into a text prompt using a pre-defined template, and the text generation model subsequently processes this prompt, as shown in Equation (2).

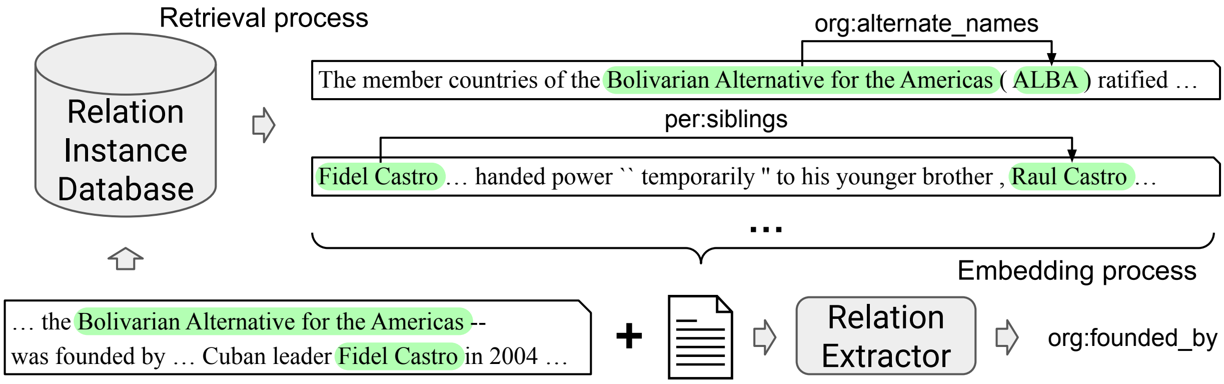

Fig. 1. Overview of relation extraction with a retriever

$$\hat{r} = \underset{r \in R}{\operatorname{argmax}} \, \text{TextGenerationModel}(r|x, h, t) \tag{2}$$

The function TextGenerationModel indicates an estimation that computes the likelihood of the verbalized input, e.g., if the PLM is pretrained on Seq2Seq LM, the PLM will compute the likelihood of the input-output pair text structured by x, h, t, and r.

We further extend this model by incorporating a retriever, as depicted in Figure 1. The retriever selects relevant instances from a database or external knowledge source D and constructs a prompt z from these instances for input into the text generation model.

$$z = \text{Retriever}(x, h, t, D), \tag{3}$$

$$\hat{r} = \underset{r \in R}{\operatorname{argmax}} \, \text{TextGenerationModel}(r|x, h, t, z) \tag{4}$$

The retriever involves two processes: the *retrieval* process, which selects pertinent instances from database D, and the *embedding* process, which transforms these instances into prompts compatible with the text generation model. In the traditional RAG framework [26], the retriever identifies neighboring instances and directly utilizes the text of these instances as prompts. Conversely, in neural prompting [28], instances are selected based on string matching and processed through a neural network to use the output as soft prompts, offering a more nuanced and adaptable approach to instance utilization.

3.2. *k-Nearest Neighbor Retriever*

Next, we define a kNN retriever, which selects k nearest instances for prompts in the embedding space. In the retriever, each instance d_i in the database D is converted into L embeddings $E_i = [e_i^1, e_i^2, \ldots, e_i^L] \in \mathbb{R}^{L \times D_{\text{emb}}}$. The retriever aims to select instances that are closely aligned with the input embedding $E_{\text{in}} \in \mathbb{R}^{L \times D_{\text{emb}}}$, which is embedded similarly to E_i using x, h, and t. The method to embed instances and the input is usually engineered to suit the purpose; for example, modern applications [50] often use PLMs such as BERT [13].Our method also uses PLM to embed instances.

To obtain k nearest instances, the distance between input and each instance s_i is calculated with distance Dist as in Equation (5).

$$s_i = \text{Dist}(E_{\text{in}}, E_i) \tag{5}$$

Algorithm 1 Differentiable Selection of k Nearest Instances

Input: k: Number of instances to select
  $E = [E_1, E_2, \ldots, E_{|D|}] \in \mathbb{R}^{|D| \times L \times D_{\text{emb}}}$: Embeddings of instances
  $E_{\text{in}} \in \mathbb{R}^{L \times D_{\text{emb}}}$: Embedding of the input
Output: $W \in \mathbb{R}^{K \times L \times D_{\text{emb}}}$: Selection weight
  for $i \in \text{range}(1, |D|)$ **do**
    $s_i \leftarrow \text{Dist}(E_{\text{in}}, E_i)$
    for $j \in [1, k]$ **do**
      $w_i \leftarrow \frac{\exp(1-s_i)}{\sum_l \exp(1-s_l)}$
      $s_i \leftarrow s_i + \log(1 - w_i)$
    end for
  end for

Since the kNN retriever objective is to emit instances within a distance of up to k-th, a set of target instances K become Equation (6), where argTopK returns a set of top-k indices:

$$K = \{d_i | i \in I, I = \underset{j}{\text{argTopK}} -s_j\} \tag{6}$$

After the retrieval, the instances are converted into a format that can be input into the model in the embedding process. The traditional retriever writes down into text prompts for PLM [26].

3.3. End-to-End Trainable Soft kNN Retriever

Traditional kNN retrievers face a crucial limitation: the retrieval process, which selects instances explicitly by the kNN algorithm, and the embedding process, which composes text-based prompts via applying templates, is non-differentiable. To overcome this, inspired by existing work on nearest neighbor networks [29] and the neural prompting [28], we introduce neural prompting which creates soft prompts using k nearest instances selected virtually by differentiable operations. Our retriever ETRASK enables differentiable instance selection by utilizing the weighted sum of embeddings derived using a softmax function over their distances. This advancement makes the retriever end-to-end trainable and allows for end-to-end training from the base model to the retriever.

3.3.1. Differentiable k-nearest Instance Selection

The differentiable k-nearest instance selection is differentiable by weighted selection over multiple instances and processing using the selection weights to construct the model, while the usual k-nearest instance selection is non-differentiable because a single discrete instance is selected as shown in Equation (6). This process ensures that the weights of the first to k-th instances are determined in order, as described in Algorithm 1. The algorithm calculates the distance s between all the target instances in advance and obtains the selection weights for virtually selecting the instances. Then, instances with heavy weights in previous steps are penalized, and the weights in subsequent steps are computed based on them so that the instances once heavily weighted process almost exclusively. We assume the distance $\text{Dist}(A, B)$ is an average of cosine distance between $A \in \mathbb{R}^{L \times D_{\text{emb}}}$ and $B \in \mathbb{R}^{L \times D_{\text{emb}}}$ as defined in Equation (7), where Dist is bounded from 0 to 1 ($0 \leqslant |\text{Dist}| \leqslant 1$).

$$\text{Dist}(A, B) = 1 - \frac{1}{L} \sum_{l=1}^{L} \frac{A_l^\top B_l}{|A_l||B_l|} \tag{7}$$

This process provides k weights for each instance, indicating the selection of the first to k-th instances. The weights W_k become a nearly one-hot vector since the softmax function computes the selection weight. When using this k-nearest instance selection, the neural model is constructed using the weights to virtually select instances with trainable settings.

3.3.2. Neural Prompting with Trainable Instance Selection

We propose a neural prompting that creates soft prompts for the text generation model in a differentiable form from the selection weights obtained via differentiable k-nearest instance selection, while the general retriever creates prompts as text. We adopt differentiable k-nearest instance selection of Section 3.3.1 as a retrieval process. As an embedding process, a soft prompt is created by weighted summing of the embeddings over the instances using the selection weights. The retriever becomes differentiable by composing with these two processes.

For the detailed procedure, the selection weight $W \in \mathbb{R}^{k \times L}$ is computed in the retrieval process as shown in Section 3.3.1. The source of soft prompt $P \in \mathbb{R}^{k \times L \times D_{\text{emb}}}$ for the k-th virtual instance is shown in Equation (8), where flatten P become the soft prompt $P' \in \mathbb{R}^{kL \times D_{\text{emb}}}$.

$$P_i = \sum_j W_{ij} E_j = \left[\sum_j W_{ij} e_j^1, \sum_j W_{ij} e_j^2, \dots, \sum_j W_{ij} e_i^L \right] \tag{8}$$

The prompt and a text generation model can be connected by joining the prompt to the input series embeddings. The length of the prompt kL is added to the length of the input series $|x|$, resulting in a new series with the length of $|x| + kL$.

3.3.3. Training

The training of the retriever proposed in Section 3.3.2 is simply a matter of optimizing the model created by the retriever using the soft prompt with the objective function of the target task. Since the retriever consists entirely of differentiable operations, the retriever model is end-to-end trainable. Therefore, when connected to a text generation model and trained for the target task, it can be trained end-to-end, including retriever. Our method innovatively transforms the kNN retriever into an end-to-end trainable model, enhancing its applicability and performance of relation extractors.

The primary challenge in training the retriever is its computational intensity, which requires calculating distances for all instances in D. To mitigate this during training, we employ a strategy of random sampling a subset of instances. This approach significantly reduces the computational burden while maintaining the efficacy of the retriever.

Since the base model is subsequently connected from ETRASK, a method is needed to train both the retriever and the base model stably. In particular, the performance of the retriever is easily affected by the training process of the text generation model, as it is the first stage of the text generation model process. Therefore, a warm-up step in which only the retriever is trained in advance enables stable training.

4. Evaluation of Relation Extraction Performance

This section evaluates the relation extraction performance and compares the ETRASK integrated model to existing relation extraction methods. Section 4.1 presents the settings of subsequent experiments about datasets, baseline methods, model settings, and training parameters. Based on the settings, Section 4.2 shows the performance evaluation of our proposal by comparing other methods. Section 4.3 is the ablation study to show the elements that affect performance.

4.1. Experimental Settings

To assess the effectiveness of our relation extraction method, we utilized the TACRED dataset [51], a standard benchmark in the sentence-level relation extraction, where each instance has an entity pair within a sentence and a gold relation between the pair. To understand the impact of training data size, we experimented with reduced training data scenarios for TACRED: 100%, 10%, 5%, and 1% of the entire dataset, following existing study [52]. The statistics for each scenario are shown in Table 1. Following the official evaluation settings, we evaluated the micro-averaged F1 score treating the `no_relation` class as a negative example for TACRED. The mean values are reported for three runs with different random seeds. We calculate the loss to development data for every 100

Table 1

Statistics of the TACRED Dataset with Different Proportion.

Proportion	Train	Dev.	Test
100%	68,125	22,631	15,509
10%	6,815	2,265	
5%	3,407	1,133	
1%	682	227	

The head entity is ${head entity}. The tail entity is ${tail entity}. The type of ${head entity} is ${label of head entity} . The type of ${tail entity} is ${label of tail entity} . ${input sentence} .

Fig. 2. The input template of SuRE

update steps and report the score obtained when training in early stopping with the early stopping patience of 3 for the TACRED dataset.

We compared our method against state-of-the-art models like SuRE (based on Pegasus-large) [24], DeepStruct [15], kNN-RE [25], and NLI_DeBERTa [52]. SuRE is, as described in Section 2.3, the model extracting relation with the text generation model by formulating the relation extraction task to the text summarization task using Seq2Seq LM based PLM. DeepStruct is a PLM trained for structured prediction tasks, which include relation extraction. kNN-RE performs *k*NN algorithm on the PLM embedding space of relation instances. NLI_DeBERTa extracts relations with natural language inference task, which recognizes fact inclusion in hypothesis, by identifying whether a verbalized relation is included in a target text.

We employed SuRE [24] as a generation-based relation extraction model. We introduced the ETRASK to SuRE by adding soft prompts into the input sequence between the BOS token and the following prompt. The hyperparameters of the SuRE were the same as those of the original research. The templates were the same as in the original paper: the templates for the SuRE input were in the form of Figure 2, from which the summary templates defined for each relation label are predicted. Templates for summarization were proposed in SuRE; for example, for the relation label "`org:founded_by`", "*${head entity} were founded by ${tail entity}*". The search beam width, a parameter used in SuRE classification, was set to 4, the same value as in the original paper of SuRE.

Our evaluation used the Flan-T5 large model [32] in the SuRE flame work with and without the addition of ETRASK because the tuning before experiments showed the training of Pegasus-large [53] based SuRE with ETRASK was unstable. Our method used two PLMs, the base relation extraction model and the embedding model for the retriever, so the larger models were not acceptable for our computational resources. We conducted trials introducing 10-neighbor instances as prompts, i.e., $k = 10$, pretraining the retriever for 300 steps before end-to-end training as the warm-up step. Due to computational constraints, the database D was constructed from training data relation instances, limited to randomly selected 5,000 instances. At the training time, 32 instances are randomly sampled as the subset of D. For the embeddings E, we used the entity and relation representations of PLM by averaging their spans, where the PLM input was created by concatenating the template in Figure 2 with the relation template. The entity spans were underlined parts of Figure 2. The relation span was the relation template part when the database was embedded and the EOS token when the prediction target was embedded. We applied Low-Rank Adaptation (LoRA) [33] to Flan-T5 with rank $r = 32$ and dropout rate 0.1 to all kinds of layers of Transformer. The dropout rate was set to 0.1. AdamW was used to optimize the models, with a learning rate of 5×10^{-4} for Flan-T5 and 1×10^{-3} for the other parameters, and weights of 5×10^{-6} for the bias and layer normalization parameters. The batch size was set to 64. The parameters used for evaluation were those used when early stopping completed training with patience set to 5. A single NVIDIA A100 was used for each experiment.

4.2. Extraction Performance Comparison

The results in Table 2 indicate that ETRASK consistently improved performance from the model without ETRASK for the TACRED dataset. The results confirm that ETRASK can enhance relations extraction by text

Table 2
Comparison of Relation Extraction Performance [%]

	100%	10%	5%	1%
DeepStruct [15]	76.8	–	–	–
SuRE (Pegasus) [24]	75.1	70.7	64.9	52.0
NLI_DeBERTa [52]	73.9	67.9	69.0	63.0
kNN-RE [25]	70.6	–	–	–
SuRE (Flan-T5)	71.4 ±1.6	68.5 ±1.5	65.0 ±3.1	53.5 ±1.4
+ ETRASK	73.3 ±0.5	71.5 ±0.5	68.3 ±2.0	54.6 ±1.1

Table 3
Ablation Study Results [%]

SuRE	68.5
SuRE + ETRASK	71.7
NO RETRIV. TRAINING	69.2
NO WARM-UP	70.7
RANDOM	71.0
CLS	68.8

generation. Additionally, ETRASK outperforms the existing models, SuRE and NLI_DeBERTa, in scenarios with limited training data (10%). This is the new state-of-the-art result for the setting of the TACRED dataset under the 10% setting.

Comparing Pegasus-based and T5-based SuREs, the Pegasus-based SuREs performed better. This is because Pegasus is a model created specifically for the summarization task, which matches SuRE's objective. However, when ETRASK was introduced, the training was not stable for the Pegasus-based model, the learning collapsed, and the relation was not predicted well. Even in this situation, the addition of ETRASK boosted the performance of the Flan-T5-based model and achieved the best performance on the 10% setting.

Compared to another instance-based method, kNN-RE, adding generation-based prediction of relations to neighborhood method-based inference confirms the improved extraction performance. This proves that simple instance utilization is insufficient and that inference capability is essential.

4.3. Ablation Study

This section delves into the factors influencing the extraction performance observed in Section 4.2 and investigates the model's behavior. We conducted an ablation study in the 10% training instance setting for the TACRED dataset, where our method showed notable improvements.

Our ablation study aimed to pinpoint the elements critical to our method's enhanced performance. We examined various scenarios: employing k-nearest neighbor instances without retriever training (NO RETRIV. TRAINING), omitting the warm-up phase in Retriever training (NO WARM-UP), using randomly chosen instances (RANDOM), and utilizing CLS token representations (CLS).NO RETRIV. TRAINING aims to investigate the usefulness of the end-to-end trainable retriever by using the initial parameter for the retriever and operations of ETRASK. NO WARM-UP omits the warm-up step but trains the retriever, which checks the stability effect of the warm-up. RANDOM picks up instances randomly and uses soft prompts in the same embedding process as ETRASK, where the experiment checks retrieval process training effectiveness. CLS changes the embedding process not to use relation extraction-specific representation by using the CLS token representations. The other settings of experiments are the same as settings in Section 4.1 except for the number of runs for evaluation that changed from 3 runs to 1 run.

The results in Table 3 reveal that omitting any of these components results in lower F1 scores, underscoring their collective importance. The NO RETRIV. TRAINING caused a performance loss of 2.2 percentage points, which was not much different from the performance of SuRE without any retrievers. This indicates that in the relation extraction with text generation model, the retriever needs to be trained for the relation extraction objective to improve performance when relation instances are introduced with the retriever.

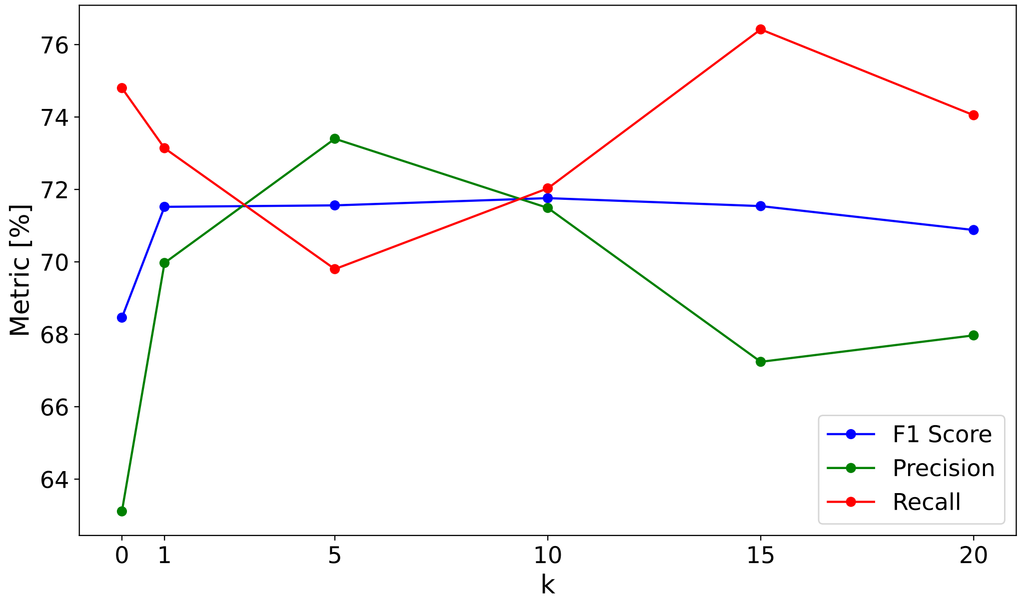

Fig. 3. Change in Extraction Performance with the Number of Instances k Used in Prompts

In the No Warm-up case, performance was decreased by 1.0 percentage points; the decrease was relatively smaller than those in the other cases, No Retriv. Training and CLS. This may be due to the lack of treatment for convergence stability, although similar processing and training of the model is carried out.

In the case of Random, where random instances are used without training the retrieval process, the performance drop is relatively small, around 0.7 percentage points. Compared to the case where no retrieval process is used (i.e., SuRE), performance improvement is observed by introducing randomly selected instances. This suggests that the embedding process and the text generation model can pick and discard instances and obtain helpful information from them.

For comparison of representations used in ETRASK, the performance with the representation of CLS degraded by 2.9 percentage points from the representation engineered for relation extraction. Additionally, the performance was almost the same as one of SuRE without a retriever. This confirms that engineering a representation specializing in relation extraction is essential to using ETRASK.

Overall, this analysis highlights the factors that contribute to the performance of our proposed method, emphasizing the importance of instance selection and training in achieving optimal relation extraction outcomes.

5. Retriever Analysis

Section 4 showed the performance improvement by our proposed ETRASK that was integrated into the text generation model (Section 4.2). The ablation studies in Section 4.3 showed the factors that contributed to performance. However, it is not yet clear what happened inside ETRASK that led to the improvements. Therefore, we also analyzed the retriever from the perspective of instances. Section 5.1 confirms the impact of the instances by changing the number of virtual instances created by the retriever and checking their behavior at that time; Section 5.2 shows what instances were retrieved and used after training by statistically analyzing the instances to analyze the actual retrieved instances directly.

5.1. Performance Variation with Number of Instances

We investigated the sensitivity of the number of instances k by varying k from 0 to 20, where 0 instance means ETRASK is not used. The results of k versus F1 score, Precision, and Recall are shown in Figure 3. Although the f1

Table 4
What retrieved instances contain. Label means the relation label is the same. Entity means either head entity or tail entity is contained. Related means either head entity, tail entity, or relation label is contained.

	Label	Head entity	Tail entity	Entity	Related
top-1	72.2	7.1	3.4	10.0	73.6
top-3	67.9	8.3	4.4	12.0	77.4
top-5	61.8	10.9	6.3	15.7	78.9
top-10	60.4	18.6	10.9	25.8	82.1

score reached its maximum at $k = 10$, there was no significant change in the $1 \leqslant k \leqslant 15$ interval. On the other hand, the balance between precision and recall changed significantly. In the $0 \leqslant k \leqslant 5$ interval, precision increased, and recall decreased as overall performance improved. Conversely, recall gradually increased, and precision decreased in the interval $k > 5$, except for $k = 20$. Since precision and recall were balanced at $k = 10$, the F1 score was considered the largest, defined as the harmonic mean of the precision and recall.

This characteristic is useful in real-world applications and can be used to make performance trade-offs to suit the situation. For example, an application that requires users to retrieve necessary relational information could use a larger k for coverage.

5.2. Retrieved Instances

Since the characteristics of retrievers are most evident in retrieved instances, we investigate the retrieved instances in ETRASK. However, because instances are virtually selected instead of selected in ETRASK, it is impossible to indicate which instances were chosen explicitly. Therefore, we analyze k nearest instances specified by the actual kNN algorithm on the feature space after training of ETRASK, which is similar to instances used in ETRASK.

Statistics are taken in association with the extraction target, the retrieval query, to analyze retrieved instances. The targets for statistics are the relation labels, head entities, and tail entities, which are closely related to the relation extraction. The relation labels are judged to determine whether the retrieved instances' relation labels match the extraction target's correct relation label. For the entities, check whether the entity is included in the instance as a string. The statistic is the proportion of the top-k instances containing the object.

The statistics in Table 4 show the ratio of the retrieved instances that contain objects related to relation extraction. First, for the Label column, the percentage gradually decreases from top-1 to top-10. This indicates that the feature space of the retrieval is structured based on the type of relation labels and that instances with the same relation labels are placed close to each other. For the Entity column, the percentage gradually increases from top-1 to top-10, indicating that the inclusion of entities is a second retrieval perspective, although the percentages are less than those in the Label column. The results for the Head and Tail entity columns show that instances containing the head entity are more intensive. From the results of the top-10 row in the Related column, more than 80% have been selected that contain labels or entities relevant for relation extraction. This may be due to the use of relation and entity features as the features used for retrieval. As a result of end-to-end training from these features, the distance becomes smaller when the relation labels match or entities are included.

6. Conclusions

This study introduced a novel approach to relation extraction using the text generation model by implementing an end-to-end trainable retriever through differentiable k-nearest neighbor selection. Existing retrievers cannot train end-to-end due to the non-differentiable environments of the instance selection part and the integration part of instances. Therefore, we propose a fully differentiable end-to-end trainable soft kNN retriever ETRASK by differentiable k-nearest neighbor selection and integration as a soft prompt. Our method, centered around neural prompting, significantly enhances the retriever's capability to select instances for use in prompts.

Our experimental findings underscore this approach's effectiveness, particularly in scenarios with limited training data. We evaluated the model with ETRASK and compared the model without ETRASK and existing methods with the TACRED dataset. Our experiment showed our proposal ETRASK consistently improved from the baseline model without ETRASK. Moreover, the model reported outstanding performance in low-resource settings, especially the new state-of-the-art for the TACRED dataset in the 10% training data setting.

Our analysis confirms that the number of virtually composed instances introduced by ETRASK can balance the precision-recall trade-off. We also confirmed that the end-to-end trained retriever referred to the instances involved in relation extraction. However, our study also identified limitations in our method's performance when dealing with sufficient training instances.

Future work could focus on refining the retriever's training process to adapt more effectively to varying sizes of training datasets and exploring ways to optimize instance selection for a broader range of data scenarios.

Acknowledgements

This work was supported by Grant-in-Aid for JSPS Fellows Grant Number JP22J14025. Computational resource of AI Bridging Cloud Infrastructure (ABCI) provided by National Institute of Advanced Industrial Science and Technology (AIST) was used.

References

[1] N.A. Chinchor, Overview of MUC-7, in: *Seventh Message Understanding Conference (MUC-7): Proceedings of a Conference Held in Fairfax, Virginia, April 29 - May 1, 1998*, 1998. https://aclanthology.org/M98-1001.

[2] E.F. Tjong Kim Sang and F. De Meulder, 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*, 2003, pp. 142–147. https://www.aclweb.org/anthology/W03-0419.

[3] L. Wang, Z. Cao, G. de Melo and Z. Liu, Relation Classification via Multi-Level Attention CNNs, in: *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Association for Computational Linguistics, Berlin, Germany, 2016, pp. 1298–1307. doi:{10.18653/v1/P16-1123}. https://aclanthology.org/P16-1123.

[4] H. Paulheim, Knowledge graph refinement: A survey of approaches and evaluation methods, *Semantic Web* **8** (2016), 489–508. doi:{10.3233/SW-160218}.

[5] A.P. Davis, C.G. Murphy, C.A. Saraceni-Richards, M.C. Rosenstein, T.C. Wiegers and C.J. Mattingly, Comparative Toxicogenomics Database: a knowledgebase and discovery tool for chemical–gene–disease networks, *Nucleic Acids Research* **37**(suppl_1) (2008), D786–D792. doi:{10.1093/nar/gkn580}.

[6] K. Zhu, J. Huang and K.C.-C. Chang, Descriptive Knowledge Graph in Biomedical Domain, in: *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, Y. Feng and E. Lefever, eds, Association for Computational Linguistics, Singapore, 2023, pp. 462–470. doi:{10.18653/v1/2023.emnlp-demo.42}. https://aclanthology.org/2023.emnlp-demo.42.

[7] Y. Long, M. Wu, Y. Liu, Y. Fang, C.K. Kwoh, J. Chen, J. Luo and X. Li, Pre-training graph neural networks for link prediction in biomedical networks, *Bioinformatics* **38**(8) (2022), 2254–2262. doi:{10.1093/bioinformatics/btac100}.

[8] G. Nan, Z. Guo, I. Sekulic and W. Lu, Reasoning with Latent Structure Refinement for Document-Level Relation Extraction, in: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Association for Computational Linguistics, Online, 2020, pp. 1546–1557. doi:{10.18653/v1/2020.acl-main.141}. https://www.aclweb.org/anthology/2020.acl-main.141.

[9] Y. Ma, A. Wang and N. Okazaki, DREEAM: Guiding Attention with Evidence for Improving Document-Level Relation Extraction, in: *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, A. Vlachos and I. Augenstein, eds, Association for Computational Linguistics, Dubrovnik, Croatia, 2023, pp. 1971–1983. doi:{10.18653/v1/2023.eacl-main.145}. https://aclanthology.org/2023.eacl-main.145.

[10] E. Riloff, Automatically constructing a dictionary for information extraction tasks, in: *AAAI*, Vol. 1, Citeseer, 1993, pp. 2–1.

[11] D. Zelenko, C. Aone and A. Richardella, Kernel methods for relation extraction, *Journal of machine learning research* **3**(Feb) (2003), 1083–1106.

[12] M. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee and L. Zettlemoyer, Deep Contextualized Word Representations, in: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, Association for Computational Linguistics, New Orleans, Louisiana, 2018, pp. 2227–2237. doi:{10.18653/v1/N18-1202}. https://www.aclweb.org/anthology/N18-1202.

[13] J. Devlin, M.-W. Chang, K. Lee and K. Toutanova, 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)*, Association for Computational Linguistics, Minneapolis, Minnesota, 2019, pp. 4171–4186. doi:{10.18653/v1/N19-1423}. https://www.aclweb.org/anthology/N19-1423.

[14] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer and V. Stoyanov, RoBERTa: A Robustly Optimized BERT Pretraining Approach, 2019.

[15] C. Wang, X. Liu, Z. Chen, H. Hong, J. Tang and D. Song, DeepStruct: Pretraining of Language Models for Structure Prediction, in: *Findings of the Association for Computational Linguistics: ACL 2022*, S. Muresan, P. Nakov and A. Villavicencio, eds, Association for Computational Linguistics, Dublin, Ireland, 2022, pp. 803–823. doi:{10.18653/v1/2022.findings-acl.67}. https://aclanthology.org/2022.findings-acl.67.

[16] J.Y. Huang, B. Li, J. Xu and M. Chen, Unified Semantic Typing with Meaningful Label Inference, in: *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, M. Carpuat, M.-C. de Marneffe and I.V. Meza Ruiz, eds, Association for Computational Linguistics, Seattle, United States, 2022, pp. 2642–2654. doi:{10.18653/v1/2022.naacl-main.190}. https://aclanthology.org/2022.naacl-main.190.

[17] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei and I. Sutskever, Language Models are Unsupervised Multitask Learners (2019).

[18] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li and P.J. Liu, Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer, *J. Mach. Learn. Res.* **21**(1) (2020).

[19] T. Brown, B. Mann, N. Ryder, M. Subbiah, J.D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever and D. Amodei, Language Models are Few-Shot Learners, in: *Advances in Neural Information Processing Systems*, Vol. 33, H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan and H. Lin, eds, Curran Associates, Inc., 2020, pp. 1877–1901.

[20] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, A. Rodriguez, A. Joulin, E. Grave and G. Lample, LLaMA: Open and Efficient Foundation Language Models, 2023.

[21] A.D. Cohen, S. Rosenman and Y. Goldberg, Supervised relation classification as twoway span-prediction, in: *4th Conference on Automated Knowledge Base Construction*, 2022.

[22] S. Wadhwa, S. Amir and B. Wallace, Revisiting Relation Extraction in the era of Large Language Models, in: *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, A. Rogers, J. Boyd-Graber and N. Okazaki, eds, Association for Computational Linguistics, Toronto, Canada, 2023, pp. 15566–15589. doi:{10.18653/v1/2023.acl-long.868}. https://aclanthology.org/2023.acl-long.868.

[23] Z. Wan, F. Cheng, Z. Mao, Q. Liu, H. Song, J. Li and S. Kurohashi, GPT-RE: In-context Learning for Relation Extraction using Large Language Models, in: *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, H. Bouamor, J. Pino and K. Bali, eds, Association for Computational Linguistics, Singapore, 2023, pp. 3534–3547. doi:{10.18653/v1/2023.emnlp-main.214}. https://aclanthology.org/2023.emnlp-main.214.

[24] K. Lu, I.-H. Hsu, W. Zhou, M.D. Ma and M. Chen, Summarization as Indirect Supervision for Relation Extraction, in: *Findings of the Association for Computational Linguistics: EMNLP 2022*, Y. Goldberg, Z. Kozareva and Y. Zhang, eds, Association for Computational Linguistics, Abu Dhabi, United Arab Emirates, 2022, pp. 6575–6594. doi:{10.18653/v1/2022.findings-emnlp.490}. https://aclanthology.org/2022.findings-emnlp.490.

[25] Z. Wan, Q. Liu, Z. Mao, F. Cheng, S. Kurohashi and J. Li, Rescue Implicit and Long-tail Cases: Nearest Neighbor Relation Extraction, in: *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, Y. Goldberg, Z. Kozareva and Y. Zhang, eds, Association for Computational Linguistics, Abu Dhabi, United Arab Emirates, 2022, pp. 1731–1738. doi:{10.18653/v1/2022.emnlp-main.113}. https://aclanthology.org/2022.emnlp-main.113.

[26] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-t. Yih, T. Rocktäschel, S. Riedel and D. Kiela, Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks, in: *Advances in Neural Information Processing Systems*, Vol. 33, H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan and H. Lin, eds, Curran Associates, Inc., 2020, pp. 9459–9474.

[27] X. Li, Z. Liu, C. Xiong, S. Yu, Y. Gu, Z. Liu and G. Yu, Structure-Aware Language Model Pretraining Improves Dense Retrieval on Structured Data, in: *Findings of the Association for Computational Linguistics: ACL 2023*, A. Rogers, J. Boyd-Graber and N. Okazaki, eds, Association for Computational Linguistics, Toronto, Canada, 2023, pp. 11560–11574. doi:{10.18653/v1/2023.findings-acl.734}. https://aclanthology.org/2023.findings-acl.734.

[28] Y. Tian, H. Song, Z. Wang, H. Wang, Z. Hu, F. Wang, N.V. Chawla and P. Xu, Graph Neural Prompting with Large Language Models, 2023.

[29] T. Plötz and S. Roth, Neural Nearest Neighbors Networks, in: *Advances in Neural Information Processing Systems*, Vol. 31, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi and R. Garnett, eds, Curran Associates, Inc., 2018.

[30] S. Hochreiter and J. Schmidhuber, Long Short-Term Memory, *Neural Comput.* **9**(8) (1997), 1735–1780–. doi:{10.1162/neco.1997.9.8.1735}.

[31] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, L.u. Kaiser and I. Polosukhin, Attention is All you Need, in: *Advances in Neural Information Processing Systems 30*, I. Guyon, U.V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan and R. Garnett, eds, Curran Associates, Inc., 2017, pp. 5998–6008. http://papers.nips.cc/paper/7181-attention-is-all-you-need.pdf.

[32] H.W. Chung, L. Hou, S. Longpre, B. Zoph, Y. Tay, W. Fedus, Y. Li, X. Wang, M. Dehghani, S. Brahma, A. Webson, S.S. Gu, Z. Dai, M. Suzgun, X. Chen, A. Chowdhery, A. Castro-Ros, M. Pellat, K. Robinson, D. Valter, S. Narang, G. Mishra, A. Yu, V. Zhao, Y. Huang, A. Dai, H. Yu, S. Petrov, E.H. Chi, J. Dean, J. Devlin, A. Roberts, D. Zhou, Q.V. Le and J. Wei, Scaling Instruction-Finetuned Language Models, 2022.

[33] E.J. Hu, yelong shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang and W. Chen, LoRA: Low-Rank Adaptation of Large Language Models, in: *International Conference on Learning Representations*, 2022. https://openreview.net/forum?id=nZeVKeeFYf9.

[34] B. Lester, R. Al-Rfou and N. Constant, The Power of Scale for Parameter-Efficient Prompt Tuning, in: *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, M.-F. Moens, X. Huang, L. Specia and S.W.-t. Yih, eds, Association for Computational Linguistics, Online and Punta Cana, Dominican Republic, 2021, pp. 3045–3059. doi:{10.18653/v1/2021.emnlp-main.243}. https://aclanthology.org/2021.emnlp-main.243.

[35] E.L. Kaplan and P. Meier, Nonparametric Estimation from Incomplete Observations, *Journal of the American Statistical Association* **53**(282) (1958), 457–481. doi:{10.1080/01621459.1958.10501452}.

[36] A. Søgaard, Semi-supervised condensed nearest neighbor for part-of-speech tagging, in: *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, D. Lin, Y. Matsumoto and R. Mihalcea, eds, Association for Computational Linguistics, Portland, Oregon, USA, 2011, pp. 48–52. https://aclanthology.org/P11-2009.

[37] Y. Yang and A. Katiyar, Simple and Effective Few-Shot Named Entity Recognition with Structured Nearest Neighbor Learning, in: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, B. Webber, T. Cohn, Y. He and Y. Liu, eds, Association for Computational Linguistics, Online, 2020, pp. 6365–6375. doi:{10.18653/v1/2020.emnlp-main.516}. https://aclanthology.org/2020.emnlp-main.516.

[38] H. Ouchi, J. Suzuki, S. Kobayashi, S. Yokoi, T. Kuribayashi, M. Yoshikawa and K. Inui, Instance-Based Neural Dependency Parsing, *Transactions of the Association for Computational Linguistics* **9** (2021), 1493–1507. doi:{10.1162/tacl_a_00439}. https://aclanthology.org/2021.tacl-1.89.

[39] I. Mani and I. Zhang, kNN approach to unbalanced data distributions: a case study involving information extraction, in: *Proceedings of workshop on learning from imbalanced datasets*, Vol. 126, ICML, 2003, pp. 1–7.

[40] C. Walker and L.D. Consortium, *ACE 2005 Multilingual Training Corpus*, LDC corpora, Linguistic Data Consortium, 2005. ISBN 9781585633760. https://books.google.co.jp/books?id=SbjjuQEACAAJ.

[41] I. Hendrickx, S.N. Kim, Z. Kozareva, P. Nakov, D. Ó Séaghdha, S. Padó, M. Pennacchiotti, L. Romano and S. Szpakowicz, SemEval-2010 Task 8: Multi-Way Classification of Semantic Relations between Pairs of Nominals, in: *Proceedings of the 5th International Workshop on Semantic Evaluation*, Association for Computational Linguistics, Uppsala, Sweden, 2010. https://www.aclweb.org/anthology/S10-1006.

[42] S.K. Sahu, F. Christopoulou, M. Miwa and S. Ananiadou, Inter-sentence Relation Extraction with Document-level Graph Convolutional Neural Network, in: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, Association for Computational Linguistics, Florence, Italy, 2019, pp. 4309–4316. doi:{10.18653/v1/P19-1423}. https://www.aclweb.org/anthology/P19-1423.

[43] F. Christopoulou, M. Miwa and S. Ananiadou, Connecting the Dots: Document-level Neural Relation Extraction with Edge-oriented Graphs, 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)*, Association for Computational Linguistics, Hong Kong, China, 2019, pp. 4925–4936. doi:{10.18653/v1/D19-1498}. https://www.aclweb.org/anthology/D19-1498.

[44] M. Miwa and Y. Sasaki, Modeling Joint Entity and Relation Extraction with Table Representation, in: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Association for Computational Linguistics, Doha, Qatar, 2014, pp. 1858–1869. doi:{10.3115/v1/D14-1200}. https://www.aclweb.org/anthology/D14-1200.

[45] D. Zeng, K. Liu, S. Lai, G. Zhou and J. Zhao, Relation Classification via Convolutional Deep Neural Network, in: *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, J. Tsujii and J. Hajic, eds, Dublin City University and Association for Computational Linguistics, Dublin, Ireland, 2014, pp. 2335–2344. https://aclanthology.org/C14-1220.

[46] W. Zhou, K. Huang, T. Ma and J. Huang, Document-Level Relation Extraction with Adaptive Thresholding and Localized Context Pooling, in: *Proceedings of the AAAI Conference on Artificial Intelligence*, 2021.

[47] T.-J. Fu, P.-H. Li and W.-Y. Ma, GraphRel: Modeling Text as Relational Graphs for Joint Entity and Relation Extraction, in: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, Association for Computational Linguistics, Florence, Italy, 2019, pp. 1409–1418. doi:{10.18653/v1/P19-1136}. https://www.aclweb.org/anthology/P19-1136.

[48] M. Miwa and M. Bansal, 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)*, Association for Computational Linguistics, Berlin, Germany, 2016, pp. 1105–1116. doi:{10.18653/v1/P16-1105}. https://www.aclweb.org/anthology/P16-1105.

[49] X. Han, H. Zhu, P. Yu, Z. Wang, Y. Yao, Z. Liu and M. Sun, FewRel: A Large-Scale Supervised Few-Shot Relation Classification Dataset with State-of-the-Art Evaluation, in: *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, E. Riloff, D. Chiang, J. Hockenmaier and J. Tsujii, eds, Association for Computational Linguistics, Brussels, Belgium, 2018, pp. 4803–4809. doi:{10.18653/v1/D18-1514}. https://aclanthology.org/D18-1514.

[50] Z. Wu, Y. Wang, J. Ye, Z. Wu, J. Feng, J. Xu and Y. Qiao, OpenICL: An Open-Source Framework for In-context Learning, in: *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, D. Bollegala, R. Huang and A. Ritter, eds, Association for Computational Linguistics, Toronto, Canada, 2023, pp. 489–498. doi:{10.18653/v1/2023.acl-demo.47}. https://aclanthology.org/2023.acl-demo.47.

[51] Y. Zhang, V. Zhong, D. Chen, G. Angeli and C.D. Manning, Position-aware Attention and Supervised Data Improve Slot Filling, in: *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics, Copenhagen, Denmark, 2017, pp. 35–45. doi:{10.18653/v1/D17-1004}. https://aclanthology.org/D17-1004.

[52] O. Sainz, O. Lopez de Lacalle, G. Labaka, A. Barrena and E. Agirre, Label Verbalization and Entailment for Effective Zero and Few-Shot Relation Extraction, in: *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, M.-F. Moens, X. Huang, L. Specia and S.W.-t. Yih, eds, Association for Computational Linguistics, Online and Punta Cana, Dominican Republic, 2021, pp. 1199–1212. doi:{10.18653/v1/2021.emnlp-main.92}. https://aclanthology.org/2021.emnlp-main.92.

[53] J. Zhang, Y. Zhao, M. Saleh and P. Liu, PEGASUS: Pre-training with Extracted Gap-sentences for Abstractive Summarization, in: *Proceedings of the 37th International Conference on Machine Learning*, H.D. III and A. Singh, eds, Proceedings of Machine Learning Research, Vol. 119, PMLR, 2020, pp. 11328–11339. https://proceedings.mlr.press/v119/zhang20ae.html.