Improving Recall of Large Language Models: A Model Collaboration Approach for Relational Triple Extraction

Zepeng Ding◇♡, **Wenhao Huang**◇♠, **Jiaqing Liang**◇♡✉,
Deqing Yang◇♡, **Yanghua Xiao**◇♠

◇Shanghai Key Laboratory of Data Science
♡School of Data Science, ♠School of Computer Science, Fudan University
{zpding22, whhuang21}@m.fudan.edu.cn
{liangjiaqing, yangdeqing, shawyh}@fudan.edu.cn

Abstract

Relation triple extraction, which outputs a set of triples from long sentences, plays a vital role in knowledge acquisition. Large language models can accurately extract triples from simple sentences through few-shot learning or fine-tuning when given appropriate instructions. However, they often miss out when extracting from complex sentences. In this paper, we design an evaluation-filtering framework that integrates large language models with small models for relational triple extraction tasks. The framework includes an evaluation model that can extract related entity pairs with high precision. We propose a simple labeling principle and a deep neural network to build the model, embedding the outputs as prompts into the extraction process of the large model. We conduct extensive experiments to demonstrate that the proposed method can assist large language models in obtaining more accurate extraction results, especially from complex sentences containing multiple relational triples. Our evaluation model can also be embedded into traditional extraction models to enhance their extraction precision from complex sentences.

1. Introduction

Relational triple extraction plays an important role in knowledge acquisition. This task aims at extracting triples *(subject, predicate, object)* (or *(s, p, o)*) from a given natural language sentence. Current large language models (LLMs) have demonstrated the capacity to effectively extract triples from simple sentences via zero-shot or few-shot learning (Wei et al., 2023; Wadhwa et al., 2023). However, it is still unsatisfactory when the sentences contain multiple relational triples or mention many entities and relations. When LLMs executing multiple triple extraction tasks, they often miss out triples, which means the **low recall** of the results (see Table 1).

In the field of relational triple extraction, numerous short sentences contain a substantial number of triples (Cheng et al., 2021). This presents a significant challenge for LLM-based extraction. Although we can meticulously design instructions or use few-shot in-context learning to improve the triple extraction capabilities of LLMs, it is still difficult to rectify the issue of incomplete extraction from complex sentences by just modifying instructions or incorporating phrases such as 'extract as many results as possible' into the prompts, as shown in Figure 1(a) and Table 1. This phenomenon could potentially be attributed to the fact that the majority of the training corpus of LLMs is composed of simple sentences, which means the distribution is significantly biased towards contain-ing few triples, leading models to overlook some correct triples when dealing with complex sentences. Previous research has demonstrated that fine-tuning a large model using task data can effectively enhance its relation extraction capabilities, yielding more accurate extraction results (Wadhwa et al., 2023). However, we observed that the fine-tuned LLMs still encounter the issue of a significantly lower recall compared to precision in the case of multiple triples, as Table 1 and Table 3 show. On one hand, the volume of fine-tuning task data is relatively small compared to the original training data of LLMs, making it insufficient to alter the bias of the triplet distribution. On the other hand, the decoding method of the generative language model is not well-suited for the extraction of multiple different relational triples. Even if we compel the large model not to generate the *<EOS>* token unless it produces enough triples, the model still lacks the capability to find more valid triples. Instead, it repeats the generated contents, as shown in Figure 1(b).

As a result, relying solely on LLMs to achieve complete extraction results in multiple triple extraction tasks is proved to be a considerable challenge. Conversely, traditional small models are prone to extract an excessive number of triples, leading to high recall but low precision results. It is because these models lack the ability to identify what triples are not mentioned (Chu et al., 2020; Jiang et al., 2020).
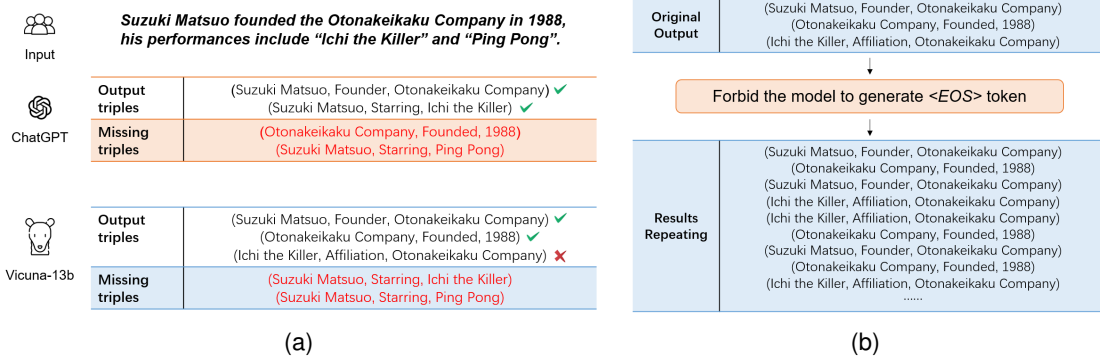
Figure 1: (a) Illustration of multiple relational triple extraction by LLMs, based on ChatGPT or Vicuna-13B. Both models are given appropriate instructions, limited predicates list and asked to extract as many as possible. (b) Compelling LLM to generate more triples results in repetitive outputs.

Dataset	Fine-tune	Precision	Recall
NYT10	w/o fine-tune	13.75	7.75
NYT10	w/ fine-tune	78.05	46.38
SKE21	w/o fine-tune	41.30	34.17
SKE21	w/ fine-tune	72.56	57.42

Table 1: Precision and recall when extracting multiple relational triples by a large language model. We only consider the complex sentences that contain more than 7 triples. The model used is Vicuna-13B for NYT10 and Qwen-7B for SKE21.

Hence, the model collaboration methods that amalgamate the strengths of small models and LLMs are a natural consideration for addressing the multiple relational triple extractions. However, as previously mentioned, traditional small models can easily generate incorrect entity pairs when dealing with complex sentences. If these results are directly incorporated into the extraction process, such as providing them to the LLMs as part of the prompt, it can easily mislead the LLMs and compromise extraction precision.

Motivated by the above considerations, we propose an evaluation-filtering model based on the transformer architecture to generate candidate relational entity pairs and construct an LLMs-based relational triple extraction framework in conjunction with this model. This model has the following characteristics: First, the model works at the token level, enabling the evaluation of candidate entity pairs represented with arbitrary tokens, so that it can accurately extract the positive entity pairs and tolerate noisy candidates. Second, our model can be easily integrated into the process of LLM-based relation triple extraction as a plug-in, significantly enhancing the extraction recall rate. The model can also be seamlessly combined with traditional extraction models to improve the precision. In sum-

mary, our main contributions are as follows:

- First, we construct an extraction framework that integrates both the small models and LLMs. This framework provides the filtered positive entity pairs to the LLMs as part of the prompt, thereby guiding the model to consider more entity pairs and assign proper relations to them.

- Second, we propose a fast and robust evaluation model that can be used to effectively filter wrong extracted results and generate positive entity pairs. It is loosely coupled with the extraction process and can be injected into any small model and LLM-based method to enhance the recall and F1 score of results.

- Third, we conduct extensive experiments to show that the proposed method can successfully enhance the performance of LLMs in relational triple extraction, particularly in terms of the recall rate. Additionally, supplementary experiments also indicate that the evaluation-filtering method can boost extraction precision when applied to traditional small models.

2. Related Works

2.1. Large Language Models for Relational Triple Extraction

Large Language Models (LLMs) have gained widespread attention due to their strong ability for various NLP tasks. In addition to the robust GPT series (Brown et al., 2020; OpenAI, 2023), open-source LLMs have been also widely studied and applied, including Llama series (Touvron et al., 2023a,b), Qwen (Bai et al., 2023) and Vicuna (Zheng et al., 2023). Recent studies on LLMs suggest that they perform well in a variety of downstream tasks, even when provided with only a few

examples as instructions (Agrawal et al., 2022; Je-blick et al., 2023). In extraction-related tasks, some works show that with proper prompting, ChatGPT can achieve comparable performance with the supervised methods on zero-shot or few-shot settings of extraction tasks (Wei et al., 2023; Gao et al., 2023; Tang et al., 2023). For open-source LLMs, previous work shows Flan-T5 (Chung et al., 2022) can yield outstanding performance by supervising and fine-tuning and suggests LLMs should be a standard baseline for relation extractions (Wadhwa et al., 2023). However, these studies did not specifically consider the model's extraction ability on complex sentences containing multiple relational triples. Furthermore, the manual evaluation of the results was not as rigorous as exact matching, and most of these studies focus on chatGPT and do not consider various open-source LLMs.

2.2. Model Collaboration in the Era of Large Language Models

Current methods of model collaboration involving large language models can be primarily categorized into three types. First, the output results of the small model are utilized as a component of the overall framework to assist the LLMs to perform better on downstream tasks (Xu et al., 2023; Leviathan et al., 2023). Second, large and small models are collaboratively trained based on task data to efficiently utilize the unlabeled data and minimize the bias of models (Lang et al., 2022). Third, ensembling multiple prompts or multiple LLMs to achieve more stable output results, as well as improved generalization performance (Allingham et al., 2023; Jiang et al., 2023).

In the field of relational triple extraction, research based on traditional models is relatively comprehensive, and some novel and effective multi-step or joint methods are proposed to extract multiple triples (Li et al., 2019; Wei et al., 2020; Yu et al., 2020; Xie et al., 2021). For example, RERE (Xie et al., 2021) carefully compares different types of multi-step settings and shows that the *relation-then-entity* extraction paradigm exhibits a good performance since it suffers less from the problem of data imbalance, which is often encountered in relational triple extraction tasks. However, these methods cannot fully solve complex relational triple extraction tasks. Inspired by this, we propose to design an evaluation model and integrate this small model as a plug-in within the extraction framework based on LLMs.

3. Methods

3.1. Solution Framework

Our LLM-based relational triple extraction framework comprises **two stages** (see Figure 2). In the first stage, the LLMs directly extract triples from sentences according to the provided instructions. Subsequently, in the second stage, we design an "evaluation-filtering" method, which extracts the positive entity pairs by our evaluation model and uses prompts to inform the LLMs that "these entity pairs may have certain relations in the relations list". These candidate pairs will be provided to the LLMs along with the instructions and the first-stage extraction results. LLMs will further scrutinize these candidates and assign appropriate relations based on their language comprehension capabilities, thereby achieving comprehensive and accurate extraction results. An example of the whole workflow is shown in Figure 3.

3.2. Basic Idea of Evaluation Model

The *evaluation model* (bottom-right part of Figure 2) uses a sentence (a list containing N tokens) as the input and outputs a token pair evaluation matrix ($N * N$). Each element in the matrix is an evaluation score for a token pair. The evaluation score of a token pair t_i and t_j is used to compose the evaluation score of an entity pair (s, o), where s contains t_i and o contains t_j. Obviously, for an input sentence, no matter how many candidate pairs are to be evaluated, only one inference is needed to get the evaluation matrix. Our goal is to build such a model that scores candidate entity pairs based on the sentence from which the triples are extracted, as Problem 1 shows.

Clearly, this evaluation model could be used as a filter, removing the extracted candidate entity pairs with low scores while retaining those with high scores. After this filtering process, we can obtain a set of precise and complete positive samples (i.e., truly related entity pairs), which can then be supplied to LLMs as prompts to facilitate high-precision extraction of multiple relational triples.

Problem 1 (Evaluation of entity pairs) *Given a sentence T and a candidate entity pair set C, the evaluation model outputs a score $F(s, o)$ for each pair $(s, o) \in C$.*

Moreover, in order to overcome the noisy entity problem, we use token-level representation to support any possible entities in the sentence.

Rationality for providing entity pairs Note that we only evaluate entity pairs (s, o) and provide them to LLMs, ignoring the predicate p. The rationality of ignoring the predicate is as follows.
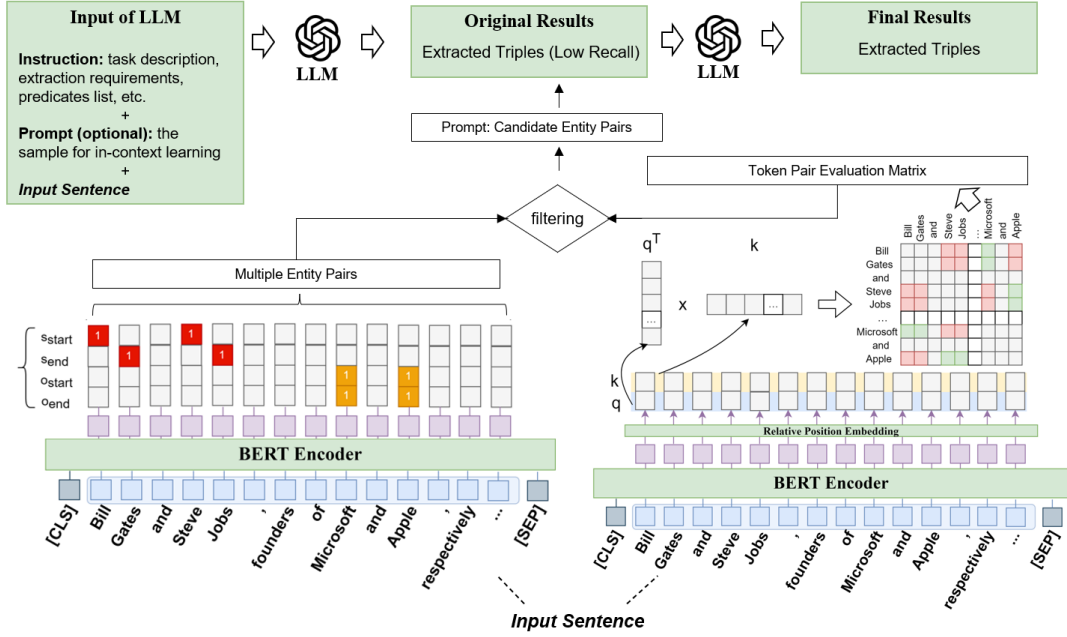
Figure 2: Model framework. On the bottom left is an arbitrary entity-extraction model. On the bottom right is our evaluation model, which outputs a token pair scoring matrix.
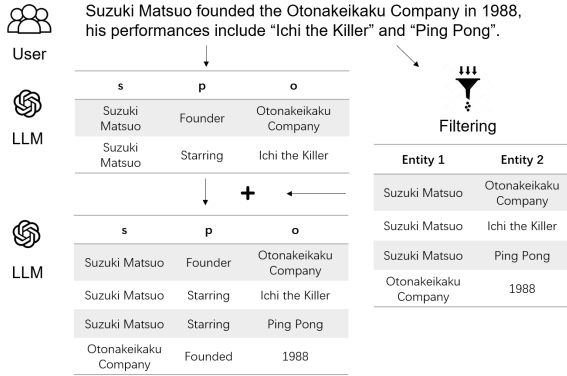

Figure 3: An example of the workflow of our Evaluation-Filtering method.

First, we find that in most real datasets, the entity pairs are more accurate than predicates in a labeled sample[1]. Second, the model structure based on entity pairs evaluation is more straightforward. It only needs to generate one evaluation matrix for a sentence, while the evaluation model for entire (s, p, o) triples requires to generate k matrices, where k represents the number of relations contained within the sentence.

Rationality of token based representation Our model aims at evaluating any candidate entity pairs

---

[1]E.g., in the NYT11 training set, there are 20% wrong triples, but only 6% wrong entity pairs.

in the sentences. However, extracting the entity span accurately is still a problem (Dixit and Al-Onaizan, 2019; Ji et al., 2020). For example, "*Gates and Steve*" might be wrongly identified as an entity (in Figure 4). Thus, it is necessary to evaluate the candidate "entities" represented by arbitrary tokens.

3.3. Self Labeling

The evaluation model has to distinguish between correct entity pairs and wrong entity pairs, which is a binary classification task, thus we need positive and negative training samples. In original extraction datasets, a sentence is labeled with some triples, which correspond to some entity pairs with one of the target relations. Obviously, these entity pairs are positive samples ($y = 1$). Then, it is important to obtain negative samples. We generate negative samples with the following assumption:

Assumption 1 *If a labeled sentence contains multiple triples, which involve multiple entities, then the unlabeled entity pairs are negative samples.*

Rationality of Assumption 1 Assumption 1 will generate a false negative entity pair (e_1, e_2) *only* when all the four conditions are satisfied ("*" means any relations or entities):

- $(e_1, *, e_2)$ is mentioned in the sentence.

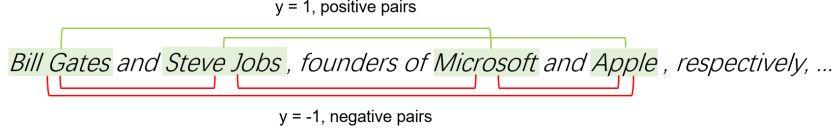- Any triples $(e_1, *, e_2)$ are not labeled in the sentence.

Figure 4: This sentence contains 6 entity pairs, but only 2 pairs are positive.

- Triple $(e_1, *, *)$ or $(*, *, e_1)$ is labeled in the sentence.

- Triple $(e_2, *, *)$ or $(*, *, e_2)$ is labeled in the sentence.

However, it is seldom the case that the four conditions are simultaneously met. The false negative case means that an annotator (no matter whether it is distant supervision via knowledge base or hand annotation via human) labels other triples in a sentence for both e_1 and e_2, but only misses the relation between them.

Token-level labeling The above process generates entity pair samples, and the labeled token pairs can be simply and effectively obtained. For example, in Figure 4, we simply split the negative entity pair *(Microsoft, Steve Jobs)* into token pairs *(Microsoft, Steve)* and *(Microsoft, Jobs)*, which are labeled as negative token pairs. This process not only increases the number of training pairs, but also enables the model to evaluate unseen or wrong entities, or even any token sequence.

Note that, for any other token pairs in the sentence (e.g. *(founders, Microsoft)*), they are not labeled as negative or positive ($y = 0$). They will be masked in the training process of our evaluation model since we have no information about whether they are positive or negative.

3.4. Evaluation Model Structure

Following the Transformer architecture, our evaluation model adopts a BERT-based encoder and an attention-like 2-dim decoder (as shown in the right part of Figure 2).

3.4.1. Encoder

We use a regular Transformer model as our encoder. Specifically, we use BERT (Devlin et al., 2019) for English and RoBERTa (Liu et al., 2019) for Chinese. They have the same network structure.

More formally, for an input sentence with N tokens $[t_1, t_2, ..., t_N]$, where $t_1 = [CLS]$ and $t_N = [SEP]$ are fixed special tokens, the BERT encoder converts these tokens into hidden vectors $[\mathbf{h}_1, \mathbf{h}_2, ..., \mathbf{h}_N]$, where each $\mathbf{h}_i$ is a d_1-dimension vector. In the BERT-base structure, $d_1 = 768$.

3.4.2. 2-dim Decoder

For an input sentence with N tokens, the BERT-based encoder encodes the tokens into N vectors $[\mathbf{h}_1, \mathbf{h}_2, ..., \mathbf{h}_N]$. Then, following the computation of the attention matrix in Transformers, we use a one-head self-attention to compute the 2-dim attention matrix as the output of the decoder. In detail, we first use two linear layers to convert the vectors $\mathbf{h}_i$ to d_2-dimension vectors $\mathbf{q}_i$ and $\mathbf{k}_i$:

$$\mathbf{q}_i = \mathbf{W}^{(q)} \mathbf{h}_i + \mathbf{b}^q,$$
$$\mathbf{k}_i = \mathbf{W}^{(k)} \mathbf{h}_i + \mathbf{b}^k, \qquad (1)$$

where $\mathbf{W}$ and $\mathbf{b}$ are trainable parameters of the linear layers, and $d_2 = 64$. Then, we compute their scaled dot-product attention as the output:

$$A_{ij} = \mathbf{q}_i^T \mathbf{k}_j / \sqrt{d_2}. \qquad (2)$$

As proposed by Roformer (Su et al., 2021), it is advantageous to add relative position embeddings (RoPE) before computing the attention output. The relative position embeddings $\mathbf{R}_i$ are realized by constructing sine and cosine functions that satisfy $\mathbf{R}_i^T \mathbf{R}_j = \mathbf{R}_{j-i}$, we refer the readers to (Su et al., 2021) for technical details. The intuition is that when encoding positional information R_i and R_j at position i and j, the output attention will naturally contain the relative positional information. The final form of the attention output is:

$$A'_{ij} = (\mathbf{R}_i \mathbf{q}_i)^T (\mathbf{R}_j \mathbf{k}_j) / \sqrt{d_2}$$
$$= \mathbf{q}_i^T \mathbf{R}_{j-i} \mathbf{k}_j / \sqrt{d_2}. \qquad (3)$$

Recall that this decoder (without the relative position embeddings) is only a part of regular one-head self-attention, although it has $O(N^2)$ outputs, its computational cost is smaller than the Transformer-based encoder. Hence, the cost of training such an evaluation model is lower than a Transformer-based extraction model.

3.5. Loss Function

Since our task is a classification task with positive and negative labels, we use the binary cross-entropy loss function to train our evaluation model:

$$L = - \sum_{i,j: y_{ij}=1} \log(\sigma(A'_{ij})) - \sum_{i,j: y_{ij}=-1} \log(1 - \sigma(A'_{ij})), \qquad (4)$$

Dataset	$\|T\|>=0$			$\|T\|>=30$			$\|T\|>=50$			$\|T\|>=70$			$\|T\|>=100$		
	avgE	avgR	#sen	avgE	avgR	#sen	avgE	avgR	#sen	avgE	avgR	#sen	avgE	avgR	#sen
NYT10	2.2	1.7	5000	2.2	1.8	4091	2.2	1.8	1798	2.3	1.9	441	2.3	2.1	51
NYT11-HRL	2.0	1.0	369	2.0	1.0	283	2.0	1.0	120	2.0	1.0	28	2.0	1.0	3
SKE21	3.3	2.4	1150	3.5	2.6	901	3.9	3.0	423	3.9	3.0	202	4.0	3.2	80
WikiKBP	2.1	1.1	182	2.2	1.2	98	2.1	1.0	25	2.3	1.2	6	-	-	-

Dataset	$\|T\|>=50$			$\|T\|>=100$			$\|T\|>=150$			$\|T\|>=200$			$\|T\|>=250$		
	avgE	avgR	#sen	avgE	avgR	#sen	avgE	avgR	#sen	avgE	avgR	#sen	avgE	avgR	#sen
HacRED	7.1	7.4	1500	7.4	7.7	1372	8.2	8.8	1012	9.1	10.0	693	10.2	11.4	410

Table 2: The statistics of complex sentences of testing datasets. $\|T\|$ means the number of tokens in the sentences. $\|T\| >= x$ only reports results for sentences with at least x tokens. $avgE$, $avgR$ denote the average numbers of labeled entities, labeled triples in the sentence, respectively. $\#sen$ denotes the number of sentences.

where y_{ij} is the label of the token pair (t_i, t_j), and σ is the sigmoid function. Note that, our task is not a pure binary classification task, since there are many unlabeled token pairs in our task. Thus, the positive and negative examples are not complementary. In the implementation of the loss, we ignore the part of unlabeled pairs (i.e. $y = 0$).

3.6. Candidate Pairs Evaluation

After training such an evaluation model, we adapt the model to an existing extraction method to obtain better extraction results. We score each candidate extracted pairs (s, o), where $s = [t_{s_{st}}, t_{s_{st}+1}, ..., t_{s_{ed}}]$ and $o = [t_{o_{st}}, t_{o_{st}+1}, ..., t_{o_{ed}}]$ are sub-token-sequences in the given sentence. Recall that, our evaluation model outputs a token pair evaluation matrix A'. Based on this matrix, we compute the score between s and o by the mean of the matching scores of their tokens:

$$F(s,o) = \frac{\sum_{k=s_{st}}^{s_{ed}} \sum_{l=o_{st}}^{o_{ed}} A'_{kl}}{(s_{ed} - s_{st} + 1)(o_{ed} - o_{st} + 1)} \quad (5)$$

where s_{st} and o_{st} are the indexes of the first element of token lists s and o, s_{ed} and o_{ed} are the indexes of the last element of s and o.

Finally, only (s, o) satisfying $F(s,o) > 0$ will be added to the result. Note that, we only need to predict the matrix A' once for each sentence, no matter how many triples of this sentence should be evaluated, thus the evaluation process is efficient.

3.7. Parameter-Efficient Fine-Tuning for LLMs

For multiple relational triple extractions, employing instruction-tuning or in-context learning (ICL) to guide LLMs, as is done for general tasks, often yields unsatisfactory results. This is because LLMs possess strong generalization capabilities and language comprehension, leading them to inexactly recognize the span of entities or relations. For instance, they may extract predicates not presented in the predicate list, or consider book titles as part of an entity, even when their extraction range is explicitly limited through prompts. Consequently, parameter fine-tuning is necessary to adapt the model to the corresponding datasets and to potentially non-natural language representations of predicates (e.g. NYT10 dataset).

In this paper, we mainly adopt the LoRA technology (Hu et al., 2021). LoRA is a parameter-efficient fine-tuning (PEFT) method. It freezes the large-scale parameters of a pre-trained model and simulates parameter changes through low-rank decomposition of the matrix, thereby adapting the model to downstream tasks with small-scale parameter adjustments. Compared to full-parameter fine-tuning, this method is more time-efficient and requires less computing resources and storage.

3.8. Instruction Template

To better guide LLMs in performing multiple relational triple extraction tasks, we design an instruction template that explicitly includes the task description, the restricted range of extracted predicates, the output format, and other requirements. We also explicitly require the model to extract as many relation triples as possible. Additionally, for the original large model without PEFT, a complete input-output example can also be placed after the instruction template for in-context learning.

After the evaluation model extracts candidate entity pairs, these candidates will be provided to the LLMs as part of the prompt, along with the aforementioned instructions and the first-stage extraction results, to guide the model in completing the extraction. See Appendix C for detailed prompts.

4. Experiments

4.1. Datasets

We evaluate our methods on several public and downloadable complex extraction datasets, including NYT series (Riedel et al., 2010; Takanobu et al., 2019), Wiki-KBP (Ling and Weld, 2012), SKE21 (Xie et al., 2021) and HacRED (Cheng et al., 2021), which are challenging for many ex-

	SKE21			HacRED			NYT10			NYT11-HRL			WikiKBP		
	Prec.	Reca.	F1	Prec.	Reca.	F1	Prec.	Reca.	F1	Prec.	Reca.	F1	Prec.	Reca.	F1
Qwen-7b (w/o peft)	17.59	26.18	21.04	5.72	6.10	5.90	8.18	6.12	7.00	8.08	16.19	10.78	6.64	19.50	9.91
+Ours	42.21	40.14	**41.15**	19.56	15.06	**17.02**	10.07	6.75	**8.08**	15.87	24.13	**19.14**	9.02	18.00	**12.02**
#Triples>=t	31.53	29.77	30.63	6.04	4.37	5.07	12.89	2.96	4.81	5.63	10.67	7.37	7.46	12.82	9.43
+Ours, #Triples>=t	54.62	39.10	**45.57**	21.43	13.98	**16.93**	13.13	5.20	**7.45**	12.50	16.67	**14.29**	8.47	12.82	**10.20**
Qwen-7b (w/ peft)	50.34	70.16	58.62	44.51	41.47	42.94	66.30	56.27	60.88	56.48	58.01	57.24	42.54	48.50	45.33
+Ours	48.17	77.90	**59.53**	42.89	49.95	**46.15**	65.54	75.37	**70.11**	56.95	67.24	**61.67**	39.58	56.00	**46.38**
#Triples>=t	63.27	65.40	64.31	51.54	39.77	44.89	81.49	43.30	56.55	76.74	44.00	55.93	34.78	20.51	25.81
+Ours, #Triples>=t	61.36	73.84	**67.02**	50.84	47.37	**49.04**	78.26	66.53	**71.92**	60.94	52.00	**56.12**	35.00	35.90	**35.44**
Llama-13b (w/ peft)	33.65	24.05	28.05	12.89	9.28	10.79	18.05	54.90	27.17	13.85	52.93	21.95	18.84	76.00	30.19
+Ours	30.16	36.06	**32.84**	27.98	15.52	**19.96**	19.04	49.78	**27.54**	24.13	65.92	**35.32**	20.96	76.00	**32.86**
#Triples>=t	37.08	21.35	27.10	17.03	8.33	11.19	36.68	47.60	41.43	38.83	53.33	44.94	20.22	46.16	28.12
+Ours, #Triples>=t	44.50	29.17	**35.24**	31.35	13.67	**19.04**	51.38	56.35	**53.75**	59.09	52.00	**55.32**	32.00	61.54	**42.10**
Vicuna-13b (w/ peft)	69.30	51.75	59.25	34.36	33.35	33.85	71.02	62.89	**66.71**	32.92	56.36	41.57	17.06	18.00	17.52
+Ours	68.20	67.96	**68.08**	53.49	40.88	**46.34**	60.98	65.86	63.33	34.20	61.37	**43.93**	37.45	47.00	**41.69**
#Triples>=t	84.55	29.62	43.88	45.86	30.97	36.97	76.29	43.53	55.43	55.41	54.67	55.03	30.00	23.08	26.09
+Ours, #Triples>=t	74.59	60.90	**67.05**	61.98	38.68	**47.63**	70.40	51.76	**59.66**	59.49	62.67	**61.04**	41.38	30.77	**35.29**

Table 3: The main extraction results. The "w/ peft" means that parameter-efficient fine-tuning (LoRA) of base LLMs is done before triples extracting, based on a part of the train set (about 800 sentences). Better exact match F1 scores are marked **bold**. The threshold t is 2 for WikiKBP and NYT11-HRL since the most complex sentence only contains 4 triples in these datasets, and is 5 for other datasets.

traction methods. Table 2 shows their statistics and shows that sentences in these datasets often have more than one triple. A brief introduction to these datasets is provided in Appendix A.

For the NYT series, the relations in the original data are labeled in structured formats (e.g. */location/location/contains*), which is not easily comprehensible to large models. Therefore, before conducting experiments, we converted these relations into natural language with similar meanings(e.g. *location contains*) to enhance the model's understanding.

4.2. Comparing Methods and Metrics

We apply our method to some recently popular LLMs such as Qwen-7B, Vicuna-13B, and Llama-13B, and compare the triple extraction performance of our framework including evaluation-filtering with the base large models. To better assess the effectiveness of our method in complex scenarios, we have also specifically calculated the metrics in cases involving multiple triples.

Additionally, we also apply our methods to pretrained-language-model-based approaches including CasRel (Wei et al., 2020), TPLinker (Wang et al., 2020), and RERE (Xie et al., 2021), to examine the effectiveness of the evaluation model and evaluate its improvement over small models.

We report standard precision (Prec.), recall (Reca.), and F1 scores for all the experiments. We mainly focus on the **exact match result**, which is also the main consideration of current extraction methods. Note that, some triples extracted by the model may be deemed errors when calculating metrics due to the synonyms or the addition of certain symbols. These could also be considered as correct results by manual evaluation. However, employing exact matching makes the evaluation and comparison of results more rigorous and credible.

4.3. Effectiveness of Our Method

The results in Table 3 demonstrate that with the assistance of our evaluation-filtering model, the triple extraction results of various LLMs on different Chinese and English datasets have been significantly enhanced. By using the filtered candidate pairs as prompts, compared to the basic LLMs, the recall rate in the multiple relational triple extractions task can be stably and significantly improved (more than 10%) in most cases, with only a risk of slightly reduced precision. In fact, the precision of models that include evaluation-filtering will also be improved in many cases.

We specifically focused on relational triple extraction in more complex scenarios, setting a minimum number of triples for each dataset (2 for WikiKBP and NYT11-HRL, 5 for others), and only considering sentences containing more than this number of triples to assess the model's extraction effect on them. The results indicate that when a sentence contains a substantial number of triples, the direct application of LLMs to extract relational triples based on instructions often yields poor results, irrespective of whether fine-tuning on labeled data. Notably, the recall in most cases is significantly lower than the precision. In contrast, on complex sentences (number of triples>=t) in various datasets, and the most complex dataset HacRED, our evaluation-filtering method can significantly enhance the recall of the extraction results, while also improving precision in most cases.

In Figure 5, we use the NYT10 dataset, categorizing the test sentences based on the number of triples they contain, and utilize the Qwen-7B

	NYT10			NYT10-HRL			SKE21			HacRED		
	Prec.	Reca.	F1	Prec.	Reca.	F1	Prec.	Reca.	F1	Prec.	Reca.	F1
TPLinker	84.96	89.66	87.25	74.31	61.06	67.04	72.73	77.94	75.24	54.64	61.21	57.74
+ Ours	86.87	89.36	**88.10**	74.79	60.86	**67.11**	81.31	77.63	**79.43**	61.78	59.04	**60.38**
CasRel	83.82	87.63	85.69	70.25	65.11	67.58	84.24	67.50	74.95	62.62	34.62	44.59
+ Ours	88.23	87.40	**87.81**	72.35	64.88	**68.41**	84.89	67.42	**75.15**	69.48	34.18	**45.82**
RERE	81.28	89.16	85.04	68.66	63.77	66.12	81.01	82.15	81.58	46.42	61.37	52.86
+ Ours	87.03	88.80	**87.90**	71.32	63.61	**67.42**	83.44	81.68	**82.55**	69.92	59.37	**64.21**

Table 4: The main evaluation results of different small models. We only report the results for sentences with at least 50 tokens. Best exact match F1 scores are marked **bold**.

model to extract triples from sentences of different complexity levels. The results show that as the number of triples within a sentence increases, our model demonstrates a progressively noticeable improvement in the recall of relational triple extraction results, compared to the base model. Moreover, it can maintain the F1 score of the results at a relatively high level. This suggests that our method is particularly effective for extracting multiple relational triples from complex sentences, and it can sustain a high level of precision of results.

In addition, the results of the small extraction model in Table 4 show that our method achieves a large precision improvement with a small recall decline, which leads to a better F1 score. This indicates that our evaluation model can accurately and reliably obtain candidate pairs, which can be applied to the traditional small extraction model to improve the performance of multiple relational triple extraction.
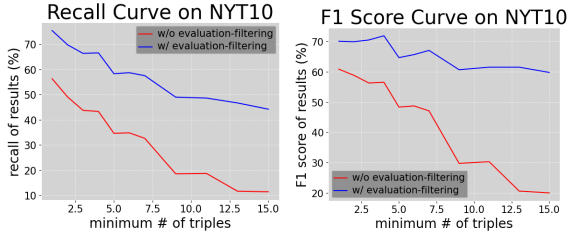

Figure 5: Recall and F1-score curve of Qwen-7B (w/ peft) on NYT10, with and without our evaluation-filtering method. Minimum # of triples means we only consider sentences that contain a number of triples greater than this value. Note that the coordinates do not start from 0.

4.4. Further Analysis

In Table 5, we try three ablation settings. First, we remove the second stage of the framework, that is, the LLMs extraction part after receiving the candidate pairs prompt. Instead, we incorporated an additional relation classification model prior to the evaluation model. In other words, we use a small extraction model (RERE (Xie et al., 2021)) to extract triples, which are then filtered according to

the evaluation model. The filtered triples are combined with the first-stage LLMs' extraction results as the final results. The results indicate that the omission of the LLMs in the second stage leads to a decrease in the precision and F1 score of triple extraction results. Therefore, a large model in the second stage is still necessary for judgment and relation identification.

Second, we remove the first stage of the framework, that is, when inputting instructions and original sentences, we also provide the LLMs evaluation-filtering prompt, which will strictly limit the scope of triple extraction to the candidates provided by the evaluation model. The results show that our model can still enhance the recall rate of multiple triple extraction, but less effectively compared to the complete framework. This could be attributed to the presence of positive entity pairs that the evaluation model fails to recognize. However, without stringent restrictions, LLMs are capable of identifying and retaining these results.

Third, we remove the filtering step in the framework, that is, directly provide all entity pairs recognized by the entity-extraction model as prompts to the LLMs. The results show that the precision and F1 score of extraction results significantly decrease. This suggests that our evaluation-filtering method is indispensable.

Models (w/ peft)	SKE21 (t=7)			NYT11-HRL (t=3)		
	Prec.	Reca.	F1	Prec.	Reca.	F1
Qwen-7b + Ours	68.32	74.46	**71.26**	60.76	60.95	**60.86**
w/o stage 2	64.69	75.01	69.47	53.85	62.23	57.73
w/o stage 1	72.77	68.86	70.76	63.33	50.67	56.30
w/o pairs filtering	65.12	66.04	65.57	58.86	59.05	58.95
Llama-13b + Ours	53.79	28.06	36.88	52.37	60.00	55.92
w/o stage 2	40.23	41.19	**40.70**	36.38	62.20	45.91
w/o stage 1	63.11	25.06	35.87	68.11	52.00	**58.97**
w/o pairs filtering	44.75	27.25	33.87	37.76	49.33	42.77
Vicuna-13b + Ours	71.49	56.83	**63.33**	58.82	57.14	**57.97**
w/o stage 2	65.26	59.90	62.92	44.00	61.33	51.24
w/o stage 1	92.76	36.06	51.93	70.91	41.67	52.49
w/o pairs filtering	63.97	57.67	60.66	32.38	55.18	40.82

Table 5: The ablation experiments results of different LLMs. We only report the results for sentences containing at least t triples. Best exact match F1 scores are marked **bold**.

5. Conclusion

In this paper, we propose an evaluation model that can act as a filter to assess and identify entity pairs that have relations, thereby providing high-precision candidates for the subsequent extraction process.

We incorporate this evaluation model into our proposed evaluation-filtering framework for LLMs multiple relation triple extraction. The candidates filtered by the evaluation model are integrated into the extraction process of LLMs in the form of prompts. This effectively addresses the issue of low recall rate in triple extraction tasks performed by LLMs, without diminishing precision.

The experimental results that derived from multiple LLMs and datasets validate the effectiveness and completeness of our framework. Additionally, we confirm that our evaluation model can also be implemented in traditional small extraction models to enhance their precision and F1 score.

Acknowledgements

Limitations

Extraction Performance   Despite the effectiveness of our model, the overall extraction results may still miss some correct triples and contain errors. On the one hand, a small amount of related entity pairs are not correctly evaluated by the evaluation model. On the other hand, it is difficult for LLMs to completely avoid errors or omissions in the second stage, although we prompt them to pick the correct candidate pairs and recheck the original results. Subsequent research could explore the optimization of the evaluation model, as well as further improvements in the extraction precision and recall of the model collaboration approach.

Complexity of Our Method   Our framework involves multiple components and requires the LLMs to perform extraction twice. Our method is more complicated and more time-consuming with the direct application of LLMs, and its inference time roughly doubled. To obtain stable effect improvement when dealing with complex sentences, our method is more suitable, while for simple extraction tasks, we suggest single-stage direct extraction.

Bibliographical References

Monica Agrawal, Stefan Hegselmann, Hunter Lang, Yoon Kim, and David Sontag. 2022. Large language models are zero-shot clinical information extractors. *arXiv preprint arXiv:2205.12689*.

James Urquhart Allingham, Jie Ren, Michael W Dusenberry, Xiuye Gu, Yin Cui, Dustin Tran, Jeremiah Zhe Liu, and Balaji Lakshminarayanan. 2023. A simple zero-shot prompt weighting technique to improve prompt ensembling in text-image models. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 547–568. PMLR.

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, et al. 2023. Qwen technical report. *arXiv preprint arXiv:2309.16609*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Qiao Cheng, Juntao Liu, Xiaoye Qu, Jin Zhao, Jiaqing Liang, Zhefeng Wang, Baoxing Huai, Nicholas Jing Yuan, and Yanghua Xiao. 2021. HacRED: A large-scale relation extraction dataset toward hard cases in practical applications. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 2819–2831, Online. Association for Computational Linguistics.

Zhendong Chu, Haiyun Jiang, Yanghua Xiao, and Wei Wang. 2020. Insrl: A multi-view learning framework fusing multiple information sources for distantly-supervised relation extraction. *arXiv preprint arXiv:2012.09370*.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Eric Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. 2022. Scaling instruction-finetuned language models. *arXiv preprint arXiv:2210.11416*.

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.

Kalpit Dixit and Yaser Al-Onaizan. 2019. Span-level model for relation extraction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5308–5314, Florence, Italy. Association for Computational Linguistics.

Joe Ellis, Xuansong Li, Kira Griffitt, Stephanie M Strassel, and Jonathan Wright. 2012. Linguistic resources for 2013 knowledge base population evaluations. In *TAC*.

Jun Gao, Huan Zhao, Changlong Yu, and Ruifeng Xu. 2023. Exploring the feasibility of chatgpt for event extraction. *arXiv preprint arXiv:2303.03836*.

R. Hoffmann, Congle Zhang, Xiao Ling, Luke Zettlemoyer, and Daniel S. Weld. 2011. Knowledge-based weak supervision for information extraction of overlapping relations. In *Proceedings of ACL*.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

Katharina Jeblick, Balthasar Schachtner, Jakob Dexl, Andreas Mittermeier, Anna Theresa Stüber, Johanna Topalis, Tobias Weber, Philipp Wesp, Bastian Oliver Sabel, Jens Ricke, et al. 2023. Chatgpt makes medicine easy to swallow: an exploratory case study on simplified radiology reports. *European Radiology*, pages 1–9.

Bin Ji, Jie Yu, Shasha Li, Jun Ma, Qingbo Wu, Yusong Tan, and Huijun Liu. 2020. Span-based joint entity and relation extraction with attention-based span-specific and contextual semantic representations. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 88–99, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Dongfu Jiang, Xiang Ren, and Bill Yuchen Lin. 2023. LLM-blender: Ensembling large language models with pairwise ranking and generative fusion. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14165–14178, Toronto, Canada. Association for Computational Linguistics.

Haiyun Jiang, JunTao Liu, Sheng Zhang, Deqing Yang, Yanghua Xiao, and Wei Wang. 2020. Surface pattern-enhanced relation extraction with

global constraints. *Knowledge and Information Systems*, 62(12):4509–4540.

Hunter Lang, Monica N Agrawal, Yoon Kim, and David Sontag. 2022. Co-training improves prompt-based learning for large language models. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 11985–12003. PMLR.

Yaniv Leviathan, Matan Kalman, and Yossi Matias. 2023. Fast inference from transformers via speculative decoding. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 19274–19286. PMLR.

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.

Xiao Ling and Daniel S Weld. 2012. Fine-grained entity recognition. In *Twenty-Sixth AAAI Conference on Artificial Intelligence*.

Liyuan Liu, Xiang Ren, Qi Zhu, Shi Zhi, Huan Gui, Heng Ji, and Jiawei Han. 2017. Heterogeneous supervision for relation extraction: A representation learning approach. *arXiv preprint arXiv:1707.00166*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

OpenAI. 2023. Gpt-4 technical report.

Xiang Ren, Zeqiu Wu, Wenqi He, Meng Qu, Clare R Voss, Heng Ji, Tarek F Abdelzaher, and Jiawei Han. 2017. Cotype: Joint extraction of typed entities and relations with knowledge bases. In *Proceedings of the 26th International Conference on World Wide Web*, pages 1015–1024.

Sebastian Riedel, Limin Yao, and Andrew McCallum. 2010. Modeling relations and their mentions without labeled text. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 148–163. Springer.

Oscar Sainz, Haoling Qiu, Oier Lopez de Lacalle, Eneko Agirre, and Bonan Min. 2022. ZS4IE: A

toolkit for zero-shot information extraction with simple verbalizations. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: System Demonstrations*, pages 27–38, Hybrid: Seattle, Washington + Online. Association for Computational Linguistics.

Jianlin Su, Yu Lu, Shengfeng Pan, Bo Wen, and Yunfeng Liu. 2021. Roformer: Enhanced transformer with rotary position embedding. *arXiv preprint arXiv:2104.09864*.

Ryuichi Takanobu, Tianyang Zhang, Jiexi Liu, and Minlie Huang. 2019. A hierarchical framework for relation extraction with reinforcement learning. In *Proceedings of AAAI*, volume 33, pages 7072–7079.

Ruixiang Tang, Xiaotian Han, Xiaoqian Jiang, and Xia Hu. 2023. Does synthetic data generation of llms help clinical text mining? *arXiv preprint arXiv:2303.04360*.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023a. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023b. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Somin Wadhwa, Silvio Amir, and Byron Wallace. 2023. 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)*, pages 15566–15589, Toronto, Canada. Association for Computational Linguistics.

Yucheng Wang, Bowen Yu, Yueyang Zhang, Tingwen Liu, Hongsong Zhu, and Limin Sun. 2020. TPLinker: Single-stage joint extraction of entities and relations through token pair linking. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1572–1582, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. 2021. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*.

Xiang Wei, Xingyu Cui, Ning Cheng, Xiaobin Wang, Xin Zhang, Shen Huang, Pengjun Xie, Jinan Xu, Yufeng Chen, Meishan Zhang, et al. 2023. Zero-shot information extraction via chatting with chatgpt. *arXiv preprint arXiv:2302.10205*.

Zhepei Wei, Jianlin Su, Yue Wang, Yuan Tian, and Yi Chang. 2020. A novel cascade binary tagging framework for relational triple extraction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1476–1488, Online. Association for Computational Linguistics.

Chenhao Xie, Jiaqing Liang, Jingping Liu, Chengsong Huang, Wenhao Huang, and Yanghua Xiao. 2021. Revisiting the negative data of distantly supervised relation extraction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3572–3581, Online. Association for Computational Linguistics.

Canwen Xu, Yichong Xu, Shuohang Wang, Yang Liu, Chenguang Zhu, and Julian McAuley. 2023. Small models are valuable plug-ins for large language models. *arXiv preprint arXiv:2305.08848*.

Bowen Yu, Zhenyu Zhang, Xiaobo Shu, Yubin Wang, Tingwen Liu, Bin Wang, and Sujian Li. 2020. Joint extraction of entities and relations based on a novel decomposition strategy. In *Proceedings of ECAI*.

Chenhan Yuan, Qianqian Xie, and Sophia Ananiadou. 2023. Zero-shot temporal relation extraction with ChatGPT. In *The 22nd Workshop on Biomedical Natural Language Processing and BioNLP Shared Tasks*, pages 92–102, Toronto, Canada. Association for Computational Linguistics.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. *arXiv preprint arXiv:2306.05685*.

A. Dataset Introduction

NYT series   NYT is based on the articles in New York Times. There are many derived datasets with better labeling. NYT10 (Riedel et al., 2010) and NYT11 (Hoffmann et al., 2011) label the complete entities. Moreover, NYT10-HRL and NYT11-HRL (Takanobu et al., 2019) are better versions that are processed by optimizing the relation labels.

HacRED   HacRED (Cheng et al., 2021)[2] is a novel challenging extraction dataset. It analyzes the performance gap between popular datasets and practical applications, and carefully selects and designs more hard cases. HacRED consists of 65,225 relational facts annotated from 9,231 wiki documents with sufficient and diverse hard cases, which poses a very high challenge to many current complex extraction methods.

SKE21   SKE19[3] is published by Baidu, and is currently the largest dataset available for complex relational triple extraction. Since its testing set is unpublished, and there are some errors in the validation set, a version named SKE21 is published by Xie et al. (2021). The testing set of SKE21 is carefully manually relabeled and contains 1,150 sentences and 2,765 annotated triples.

Wiki-KBP   Wiki-KBP (Ling and Weld, 2012) is based on the articles in Wikipedia. There're 1.5M sentences in training set which are automatically labeled using distant supervision and handcrafted patterns by (Liu et al., 2017), and the test set contains 289 sentences selected by the author of (Ren et al., 2017) from the manual annotations in 2013 KBP slot filling results (Ellis et al., 2012).

B. Experiment Details

Our experiments are conducted on two A800 GPUs. All deep models, including the LLMs and the evaluation model, are fine-tuned or implemented using the PyTorch framework. We employed AdamW optimizer as the optimizer. For the evaluation model, we first initialize the model with bert-base-cased and chinese-roberta-wwm-ext respectively, then train 20 epochs in English corpus task, and 40 epochs in Chinese. For the fine-tuning of LLMs, we randomly select 1500 items from the train set for each dataset and train 30 epochs. Our codes and hyper-parameters can be found at https://github.com/Ding-Papa/Evaluating-filtering-coling24.

C. Instruction Template

Here we provide the instruction templates that guide the LLMs for relational triple extraction. First is the template for directly using the LLMs to perform extraction, i.e., the first stage of our method.
Template for the first stage:

```
Pre-define the following relation
list r, please extract all triples
containing the above relations from
the given sentence S.
Note that the relation name of the
triple must be selected from the
above list, and other relations
not listed are not considered.
Please output according to the
specified format: [{"s": subject1,
"o": object1, "p": relation1},
{"s": subject2, "o": object2, "p":
relation2},...]
(Optional) Here are some examples:
...
Now given the following input,
please complete the extracting task.
Please output as many triples as
possible that meet the requirements.
Input:   S_i, r_i
```

In the second stage, the input of the LLMs consists of the first-stage extraction results and candidate pairs extracted by the evaluation model. The LLMs are prompted to recheck the original results, assign relations to the appropriate candidate entity pairs, and output the final extracted triples to complete the extraction.
Template for the second stage:

```
Pre-define the following relation
list r. We want to extract all
triples containing the above
relations from the given sentence
S. Here are the original extraction
results A.
Now we claim that the entity pairs
that may be related in the above
sentence are (s_1,o_1),(s_2,o_2),...
Please check the original results
and fill in the missing triples,
remove the wrong triples and output
the final results.
Constraints and output format are
the same as stage 1.
Please output according to the
specified format.
```