Are Relational Triple Extraction Frameworks Sufficient for Hyper-relational Facts ?

Pratik Saini and **Chayan Sarkar** and **Tapas Nayak**
TCS Research, India
{pratik.saini, sarkar.chayan, nayak.tapas}@tcs.com

Abstract

Hyper-relational fact extraction involves identifying relational triples along with additional contextual information—known as qualifiers—such as time, location, or quantity. These qualifiers enable models to represent complex real-world knowledge more accurately. While numerous end-to-end models have been developed for extracting relational triples, they are not designed to handle qualifiers directly. In this work, we propose a straightforward and effective approach to extend existing end-to-end triple extraction models to also capture qualifiers. Our method reformulates qualifiers as new relations by computing the Cartesian product between qualifiers and their associated relations. This transformation allows the model to extract qualifier information as additional triples, which can later be merged to form complete hyper-relational facts. We evaluate our approach using multiple end-to-end triple extraction models on the HyperRED dataset and demonstrate its effectiveness in extracting hyper-relational facts.

1 Introduction

Relational triples extraction helps in automatically adding new knowledge in existing knowledge bases (KBs). Thus in recent times many novel deep neural frameworks are proposed such as PtrNet (Nayak and Ng, 2020), GRTE (Ren et al., 2021), TDEER (Li et al., 2021), PRGC (Zheng et al., 2021), BiRTE (Ren et al., 2022), etc to solve this task. But one drawback in this task is that they do not extract associated qualifiers with the triples such as time, quantity, location, etc. Such qualifiers along with the triples can increase the usefulness of the KBs significantly. Keeping the above drawback in relation extraction task in mind, Chia et al. (2022) proposed Hyper-Relational Facts extraction task along with a dataset. They proposed cube-filling approach that can extract the relational triples along with the qualifiers. Luo et al. (2024) proposed a N-ary relation extraction approach for this same task.

Text: Barack Obama (born August 4, 1961) is an American politician who was the 44th president of the United States from 2009 to 2017.
Gold Hyper-relation Fact: 1. <Barack Obama, position_held, President of the United States, start_time 2009, end_time 2017>
Extracted Relational Triples: 1. <Barack Obama, position_held, President of the United States> 2. <Barack Obama, position_held_start_time, 2009> 3. <Barack Obama, position_held_end_time, 2017>

Table 1: Examples of Hyper-relational facts in text.

In this work, we explore is there a need to have specialized models for hyper-relational facts extraction? Or is it possible to extend the relational triples extraction models to hyper-relational facts extraction? Relational triples extraction models such as PtrNet (Nayak and Ng, 2020), GRTE (Ren et al., 2021), TDEER (Li et al., 2021), PRGC (Zheng et al., 2021), BiRTE (Ren et al., 2022), etc only extract the triple of subject entity, object entity, and a relation. They do not extract the additional qualifiers associated with the triples.

To eliminate this drawbacks in the triples extraction models, we propose an intuitive and simple solution to extract the additional qualifiers using the triples extraction model. We convert the qualifiers into additional relations that taking Cartesian product between the original relations and qualifiers. These additional relations can encode the original relation along with the qualifier information. By doing this the Hyper-relational fact can be spread into multiple triples.

In the example show in Table 1, there is original relation 'position_held' and two qualifiers 'start_time' and 'end_time'. Taking Cartesian product between them, we obtain two additional relations 'position_held_start_time' and 'position_held_end_time'. So the original hyper-

Proceedings of the 14th International Joint Conference on Natural Language Processing and the 4th Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics, pages 112–119
December 20-24, 2025 ©2025 Association for Computational Linguistics

relational fact can be spread into three triples <Barack Hussein Obama, position_held, President of the United States>, <Barack Hussein Obama, position_held_start_time, 2009>, and <Barack Hussein Obama, position_held_end_time, 2017>. Relational triples extraction models can extract these triples without any changes in their model design and we can combine these triples to obtain the original hyper-relational fact. Following this approach, we benchmark multiple triple extraction models on hyper-relational facts extraction task to show their effectiveness. Our experimental results show that triple extraction models can achieve competitive score compared to the task specific models.

2 Proposed Approach

Joint entity and relation extraction models such as PtrNet (Nayak and Ng, 2020), GRTE (Ren et al., 2021), TDEER (Li et al., 2021), PRGC (Zheng et al., 2021), BiRTE (Ren et al., 2022) extract relational triples from text in an end-to-end fashion (Details of these models and their parameter settings are included in Appendix.). They have achieved state-of-the-art performance over the pipeline approaches. These models are not explored for extracting additional qualifiers along with the triples. In this work, we try to answer if these models can be extended to extract the qualifier information as well. We try to find out: **Is it possible to extract the qualifiers as relational triples?** The answer to that question is **YES**.

Our idea is to break a hyper-relational facts into multiple triples where the subject entity and a qualifier are clubbed together with additional relations. We create these additional relations by taking Cartesian product between the initial relation set or base relations and the qualifier labels. The rationale behind the taking the Cartesian product is to distinguish between the same qualifier of different relations. A qualifier can be associated with multiple relations. We want the models to treat the qualifiers for each relations differently from other relations. As an example a qualifier ‘Start_Time’ can be associated with relations ‘Educated_At’ and ‘President_Of’. ‘Start_Time’ refers to when a person started his/her education for the ‘Educated_At’ relation and same qualifier refers to when a person became the president of a country for the ‘President_Of’ relation. To differentiate between these two cases, we create two additional relations ‘Educated_At_Start_Time’ and ‘President_Of_Start_Time’ by taking the Cartesian product between the base relations (‘Educated_At’, and ‘President_Of’) and qualifier labels (‘Start_Time’).

Org. Rel.	Qualifier Label	Cartesian Relation
employer	point_in_time	employer-point_in_time
	location	employer-location
	position_held	employer-position_held
occupant	start_time	occupant-start_time
	point_in_time	occupant-point_in_time
	end_time	occupant-end_time

Table 3: Example of Cartesian relations formed by combining base relations and qualifier labels.

The general method of breaking the hyper-relational facts is follows. Let us consider a hyper-relational facts $F :< S, R, O, Q_1V_1, Q_2V_2 >$ where S is the subject entity, R is the relation, O is the object entity, Q_1 and Q_2 are the two qualifiers and V_1 and V_2 are the corresponding values of the Q_1 and Q_2, respectively. Now, we can break down these hyper-relational facts into three triples such as $< S, R, O >$, $< S, R\_Q_1, V_1 >$, and $< S, R\_Q_2, V_2 >$. The first triple $< S, R, O >$ is the original triple in hyper-relational fact. The second triple $< S, R\_Q_1, V_1 >$ is formed by combining the original subject entity S and the value V_1 of the qualifier Q_1 as an object entity with the new relation R_Q_1. This new relation R_Q_1 is the Cartesian product between original relation R and qualifier Q_1. Similarly, the third triple $< S, R\_Q_2, V_2 >$ constructed using the second qualifier in the fact F.

To reduce the number of Cartesian relation types, we retain only Cartesian relations observed during training. For each base relation, we identify co-occurring qualifier labels and construct relations only from these observed pairs, discarding all unobserved combinations. In this way, we can extend the relation set that includes the qualifiers as well. If original relation set has m relation and n qualifiers, new relation set may have a maximum of $m + mn$ relations. We train the triples extraction models with this new relation set and additional triples in the data and during inference, we can combine the extracted triples to re-construct the hyper-relational facts.

During inference, we can reconstruct the hyper-relational facts from the extracted triples by matching the subject entity and relation labels. Reconstruction of the hyper-relational facts becomes difficult when two or more hyper-relation facts appear in the same text with the same subject entity and

Text: Pranab Mukherjee was finance minister of India from 2009 to 2012 and President of India from 2012 to 2017.
Hyper-relation Facts: 1. <Pranab Mukherjee, position_held, finance minister of India, start_time 2009, end_time 2012> 2. <Pranab Mukherjee, position_held, President of India, start_time 2012, end_time 2017>
Relational Triples: 1. <Pranab Mukherjee, position_held, finance minister of India> 2. <Pranab Mukherjee, position_held_start_time, 2009> 3. <Pranab Mukherjee, position_held_end_time, 2012> 4. <Pranab Mukherjee, position_held, President of India> 5. <Pranab Mukherjee, position_held_start_time, 2012> 6. <Pranab Mukherjee, position_held_end_time, 2017>
Explanation: <Pranab Mukherjee, position_held_start_time> combination has two objects which are 2009 and 2012. Since there are two triples with <Pranab Mukherjee, position_held> combination, we cannot decide which triple is associated with which start_time. Similar conflict arises with the position_held_end_time as well. 3.85% samples of the HyperRED dataset (Chia et al., 2022) that we use for experiments have such conflict.

Table 2: An example to explain the conflict during hyper-relational fact reconstruction from the triple representation.

the same relation such as $F_1 :< S, R, O_1, QV_1 >$ and $F_2 :< S, R, O_2, QV_2 >$. The fact F_1 is broken into $F_{11} :< S, R, O_1 >$ and $F_{12} :< S, R\_Q, V_1 >$. The fact F_2 is broken into $F_{21} :< S, R, O_2 >$ and $F_{22} :< S, R\_Q, V_2 >$. Now, if a triple extraction model extracts F_{11}, F_{12}, F_{21}, and F_{22} then we cannot determine F_{11} to be combined with F_{12} or F_{22}. Similarly, we cannot determine F_{21} to be combined with F_{12} or F_{22}. We include an example of such scenario in Table 2 for illustration purpose. We find that this conflict occurs with only 3.85% samples in the HyperRED dataset (Chia et al., 2022) that we use for our experiments. In such cases, we attach the qualifiers to the nearest object entities in terms of token-level distance between the position of the qualifier value and the object entities. We describe our hyper-relational facts reconstruction process from relational triples in Algorithm 1 that can handle such ambiguous cases. We include few diverse examples in Table 7 in Appendix to show how the algorithm works.

2.1 Hyper-relational Facts Reconstruction Algorithm

We present our decoding procedure in Algorithm 1, which reconstructs hyper-relational facts from extracted relational triples. These relational triples are categorized into two distinct sets: (i) Base relational triples, comprising the initial set of relations (base relations), and (ii) Cartesian relational triples, which contain composite relation labels formed by taking the Cartesian product of base relations and qualifier labels. The algorithm begins by processing each Cartesian triple, decomposing its composite relation label into a base relation and a qualifier label. This step yields a set of qualifier candidates. Both base triples and qualifier candidates are then grouped according to their shared subject and base relation. Within each group, the algorithm aligns qualifiers to their corresponding base triples. If a

Algorithm 1 Reconstruction of hyper-relational facts from extracted relational triples.

Inputs:
- Base triples: $\mathcal{T}_{\text{base}} = \{(s, r, o)\}$
- Cartesian triples: $\mathcal{T}_{\text{cart}} = \{(s, r\#q_l, o)\}$
- Tokenized sentence: $\mathcal{X}$

Output: Decoded quintuples: $\mathcal{Q} = \{(s, r, o, q_l, q_v)\}$
Initialize empty list: candidate qualifiers $\mathcal{Q}_{\text{cand}}$
for each $(s, r\#q_l, q_v)$ in $\mathcal{T}_{\text{cart}}$ **do**
  Add (s, r, q_l, q_v) to $\mathcal{Q}_{\text{cand}}$
end for
Group $\mathcal{T}_{\text{base}}$ and $\mathcal{Q}_{\text{cand}}$ by (s, r)
for each group (s, r) **do**
  Let $\mathcal{T}_{(s,r)}$ be triples in $\mathcal{T}_{\text{base}}$ with subject s and relation r
  Let $\mathcal{Q}_{(s,r)}$ be qualifiers in $\mathcal{Q}_{\text{cand}}$ with subject s and base relation r
  if $|\mathcal{T}_{(s,r)}| = 1$ **then**
    Assign all qualifiers in $\mathcal{Q}_{(s,r)}$ to the single triple in $\mathcal{T}_{(s,r)}$
  else
    for each qualifier (s, r, q_l, q_v) in $\mathcal{Q}_{(s,r)}$ **do**
      For each triple (s, r, o) in $\mathcal{T}_{(s,r)}$, compute token distance between o and q_v in $\mathcal{X}$
      Assign qualifier to triple with smallest distance
    end for
  end if
end for
For each matched triple and qualifier, form quintuple (s, r, o, q_l, q_v)
return Set of decoded quintuples

#Ent	#R	#Q	$\#R \times Q$	All		Train		Dev		Test	
				#Sent	#Fact	#Sent	#Fact	#Sent	#Fact	#Sent	#Fact
40,293	62	44	422	44,840	45,994	39,840	39978	1,000	1,220	4,000	4,796

Table 4: HyperRED dataset statistics, where the columns represent the number of entities, relations, Cartesian relations, and hyper-relational facts across the entire dataset, training set, validation set and test set respectively.

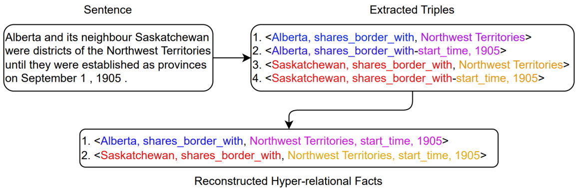

Figure 1: Visual illustration of our proposed approach for hyper-relational facts extraction.

group contains only one base triple, all associated qualifiers are directly assigned to it. In cases with multiple base triples, qualifiers are matched using a token-level proximity heuristic—comparing the object of each triple with the qualifier value in the original text. This heuristic helps disambiguate the correct pairing between base triples and qualifiers. The final output is a set of hyper-relational quintuples of the form (S, R, O, Q_l, Q_v), effectively enriching relational triples with additional qualifier information. Illustrative examples are provided in Table 7 in the Appendix to demonstrate the algorithm's functionality.

3 Experiments

3.1 Datasets & Evaluation

We use the HyperRED (Chia et al., 2022) dataset for our experiments. We experiment in two settings: (i) HyperRED_ORG: In this setting entities are nor marked. Models need to identify the entities along with relations and qualifiers. (ii) HyperRED_EM: Luo et al. (2024) used the Hyper_RED dataset is a different way where they marked the entities in the input text. Models do no need to identify the entities here. They only need to pair up the entities and qualifiers. We include the dataset statistics in Table 4. There are 62 relations ($\#R$) and 44 types of qualifiers ($\#Q$) and out of all the possible Cartesian product between relations and qualifier types ($\#R \times Q$), 422 has some valid mentions in the training data.

We use facts level precision, recall, and F1 score for evaluation. We consider a quintuple of subject entity, object entity, a relation, a qualifier type, a qualifier value as a fact. We use a strict evaluation where a predicted quintuple is considered correct if its elements matches (exact match only, no partial matching) with a ground truth quintuple.

3.2 Baseline Models

We compare the performance of our proposed approach against the following baseline models.

Pipeline Approach: This baseline performance is taken from Chia et al. (2022). First, UniRE model (Wang et al., 2021) is trained to extract relational triples from each input sentence. Separately, a span extraction model based on BERT-Tagger (Devlin et al., 2019) that is conditioned on the input sentence and a relational triples to extract the value entities and corresponding qualifier label.

Generative Approach: This baseline performance is also taken from Chia et al. (2022). Cabot and Navigli (2021) proposed a generative approach for relation extraction task. They represent the relational triples in a pre-defined text sequence and use a generative models such as BART (Lewis et al., 2019) to output the desired text sequence. Same approach can be used for hyper-relational facts as well.

CubeRE (Chia et al., 2022): Chia et al. (2022) proposed a cube-filling approach for hyper-relational facts extraction. Similar to the table-filling approach of GRTE, they added another dimension to represent the qualifier information. The cube contains multiple planes of tables where front most table table stores the entity and relation information and following tables store the qualifier information.

Text2NKG (Luo et al., 2024): The input to Text2NKG consists of natural language tokens annotated with entity spans at the sentence level. Each ordered span-tuple of three entities (subject, object, and qualifier) is classified using a multi-label classification approach. It employs a hetero-ordered merging strategy which involves aggregating the label probabilities across all $3! = 6$ permutations of the same entity set to filter out inconsistent facts. The valid 3-ary facts are merged into hyper-relational facts.

Model	Precision	Recall	F1
Generative$_{bart_base}$	0.646 ± 0.005	0.597 ± 0.004	0.620 ± 0.002
Pipeline$_{bert_base}$	0.690 ± 0.005	0.576 ± 0.002	0.628 ± 0.003
CubeRE$_{bert_base}$	0.658 ± 0.008	0.643 ± 0.003	0.650 ± 0.003
TE_PtrNet$_{bert_base}$	0.650 ± 0.003	0.586 ± 0.001	0.616 ± 0.002
TE_GRTE$_{bert_base}$	0.658 ± 0.027	0.624 ± 0.011	0.640 ± 0.014
TE_TDEER$_{bert_base}$	0.680 ± 0.011	0.608 ± 0.013	0.642 ± 0.012
TE_PRGC$_{bert_base}$	0.727 ± 0.012	0.598 ± 0.013	0.656 ± 0.006
TE_BiRTE$_{bert_base}$	0.725 ± 0.013	0.594 ± 0.025	0.653 ± 0.011
Generative$_{bart_large}$	0.672 ± 0.004	0.646 ± 0.006	0.658 ± 0.003
Pipeline$_{bert_large}$	0.692 ± 0.006	0.643 ± 0.002	0.667 ± 0.003
CubeRE$_{bert_large}$	0.664 ± 0.010	0.671 ± 0.007	0.668 ± 0.003
TE_PtrNet$_{bert_large}$	0.679 ± 0.012	0.604 ± 0.006	0.640 ± 0.005
TE_GRTE$_{bert_large}$	0.713 ± 0.008	0.690 ± 0.007	0.701 ± 0.005
TE_TDEER$_{bert_large}$	0.698 ± 0.011	0.665 ± 0.005	0.681 ± 0.005
TE_PRGC$_{bert_large}$	0.750 ± 0.002	0.628 ± 0.006	0.684 ± 0.004
TE_BiRTE$_{bert_large}$	0.721 ± 0.007	0.622 ± 0.059	0.667 ± 0.038

Table 5: Performance comparison of the models on HyperRED_ORG dataset. TE=Triple Extraction.

Model	Precision	Recall	F1
Text2NKG$_{bert_base}$	0.908 ± 0.006	0.775 ± 0.003	0.836 ± 0.006
TE_PtrNet$_{bert_base}$	0.839 ± 0.004	0.763 ± 0.008	0.799 ± 0.003
TE_GRTE$_{bert_base}$	0.837 ± 0.018	0.822 ± 0.004	0.829 ± 0.011
TE_TDEER$_{bert_base}$	0.830 ± 0.009	0.799 ± 0.005	0.814 ± 0.006
TE_PRGC$_{bert_base}$	0.868 ± 0.005	0.811 ± 0.005	0.839 ± 0.004
TE_BiRTE$_{bert_base}$	0.865 ± 0.011	0.793 ± 0.014	0.827 ± 0.011
Text2NKG$_{bert_large}$	0.911 ± 0.008	0.776 ± 0.005	0.838 ± 0.005
TE_PtrNet$_{bert_large}$	0.847 ± 0.002	0.775 ± 0.005	0.810 ± 0.004
TE_GRTE$_{bert_large}$	0.866 ± 0.004	0.842 ± 0.005	0.854 ± 0.004
TE_TDEER$_{bert_large}$	0.857 ± 0.007	0.828 ± 0.008	0.842 ± 0.007
TE_PRGC$_{bert_large}$	0.879 ± 0.005	0.825 ± 0.010	0.851 ± 0.003
TE_BiRTE$_{bert_large}$	0.876 ± 0.006	0.826 ± 0.001	0.850 ± 0.002

Table 6: Performance comparison of the models on HyperRED_EM dataset. TE=Triple Extraction.

4 Results & Discussion

We include our experimental results with the original HyperRED dataset in Table 5. We see that with the BERT-base encoder, CubeRE achieved F1 score of 0.650. PRGC model with triple extraction approach achieves 0.656 F1 score which is marginally better than CubeRE. With the BERT-large encoder, CubeRE achieves F1 score 0.668. Three out of the five triple extraction models, GRTE, TDEER, and PRGC outperform CubeRE by 1-3%. GRTE model outperforms CubeRE by 3.3% F1 score in this setting of the dataset.

Text2NKG model used the HyperRED dataset in their experiments in a different way. They marked all the entity mentions in the input text, as we call it HyperRED_EM. We experiment with this dataset as well and include the results in Table 6. With the BERT-base encoder, PRGC model with triples extraction approach marginally outperforms the Text2NKG model. With the BERT-large encoder, four out of the five triple extraction models, GRTE, TDEER, PRGC, BiRTE outperform Text2NKG by more than 1%. GRTE model achieves the highest F1 score of 0.854 in this settings of the HyperRED dataset. Our experimental results establish that relational triples extraction frameworks with our proposed Cartesian product based relation extension approach are sufficient for the hyper-relational facts.

5 Conclusion

In this work, we investigate if the relational triples extraction frameworks can be utilized for hyper-relational facts or not. We propose to extract the qualifier information in hyper-relational facts as relational triples only. We break the hyper-relational facts into multiple triples by creating new relations by doing Cartesian product between the original relations and qualifier labels. We experiment with five state-of-the-art such frameworks with the HyperRED dataset. Our experiments show that many of these triples extraction frameworks achieve SOTA F1 score on the HyperRED outperforming task specific models such as CubeRE or Text2NKG.

Limitations

There are so many relational triples extraction models that are proposed in recent times. But due to limited computing capabilities, we could only experiment with five of such models.

Ethics Statement

There are no ethical concerns regarding this work.

References

Pere-Lluís Huguet Cabot and Roberto Navigli. 2021. Rebel: Relation extraction by end-to-end language generation. In *EMNLP*.

Yew Ken Chia, Lidong Bing, Sharifah Mahani Aljunied, Luo Si, and Soujanya Poria. 2022. A dataset for hyper-relational extraction and a cube-filling approach. In *EMNLP*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*.

Raphael 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 the 49th Annual Meeting of the Association for Computational Linguistics*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdel rahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *ACL*.

Xianming Li, Xiaotian Luo, Cheng Jie Dong, Daichuan Yang, Beidi Luan, and Zhen He. 2021. TDEER: An efficient translating decoding schema for joint extraction of entities and relations. In *EMNLP*.

Haoran Luo, E. Haihong, Yuhao Yang, Tianyu Yao, Yikai Guo, Zichen Tang, Wentai Zhang, Kaiyang Wan, Shiyao Peng, Meina Song, and Wei Lin. 2024. Text2nkg: Fine-grained n-ary relation extraction for n-ary relational knowledge graph construction. In *NeurIPS*.

Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *ACL*.

Makoto Miwa and Mohit Bansal. 2016. End-to-end relation extraction using LSTMs on sequences and tree structures. In *ACL*.

Tapas Nayak and Hwee Tou Ng. 2019. Effective attention modeling for neural relation extraction. In *Proceedings of the Conference on Computational Natural Language Learning*.

Tapas Nayak and Hwee Tou Ng. 2020. Effective modeling of encoder-decoder architecture for joint entity and relation extraction. In *Proceedings of The Thirty-Fourth AAAI Conference on Artificial Intelligence*.

Feiliang Ren, Longhui Zhang, Shujuan Yin, Xiaofeng Zhao, Shilei Liu, Bochao Li, and Yaduo Liu. 2021. A novel global feature-oriented relational triple extraction model based on table filling. In *EMNLP*.

Feiliang Ren, Longhui Zhang, Xiaofeng Zhao, Shujuan Yin, Shilei Liu, and Bochao Li. 2022. A simple but effective bidirectional framework for relational triple extraction. *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*.

Yijun Wang, Changzhi Sun, Yuanbin Wu, Hao Zhou, Lei Li, and Junchi Yan. 2021. UniRE: A unified label space for entity relation extraction. In *ACL*.

Daojian Zeng, Kang Liu, Yubo Chen, and Jun Zhao. 2015. Distant supervision for relation extraction via piecewise convolutional neural networks. In *EMNLP*.

Daojian Zeng, Kang Liu, Siwei Lai, Guangyou Zhou, and Jun Zhao. 2014. Relation classification via convolutional deep neural network. In *Proceedings of the 25th International Conference on Computational Linguistics*.

Xiangrong Zeng, Daojian Zeng, Shizhu He, Kang Liu, and Jun Zhao. 2018. Extracting relational facts by an end-to-end neural model with copy mechanism. In *ACL*.

Heng Zheng, Rui Wen, Xi Chen, Yifan Yang, Yunyan Zhang, Ziheng Zhang, Ningyu Zhang, Bin Qin, Ming Xu, and Yefeng Zheng. 2021. PRGC: Potential relation and global correspondence based joint relational triple extraction. In *ACL*.

Suncong Zheng, Feng Wang, Hongyun Bao, Yuexing Hao, Peng Zhou, and Bo Xu. 2017. Joint extraction of entities and relations based on a novel tagging scheme. In *ACL*.

A Appendix

A.1 Details of Triple Extraction Models

PtrNet (Nayak and Ng, 2020): PtrNet employs a sequence-to-sequence framework combined with pointer network-based decoding to jointly extract entities and their relations. The triples are represented using the start and end indices of both the subject and object entities, along with the corresponding relation label. Pointer networks are used to find such indices in the input text.

GRTE (Ren et al., 2021): GRTE uses a table-filling approach for triples extraction. It maintains separate tables for each relation, where each cell in the table indicates whether a relation exists between a given pair of tokens. To improve the table-filling process, GRTE introduces two types of global features: one capturing the association between entity pairs and the other focusing on relations. Initially, a table feature is generated for each relation, which is then integrated across all relations to produce two global features—one for the subject and one for the object. These global features are refined in an iterative way. In the final step, the completed tables are used to extract all relevant relational triples.

TDEER (Li et al., 2021): Li et al. (2021) proposed a multi-stage sequence labeling framework for the triple extraction task. In the first stage, a binary (0/1) tagging scheme is applied to extract subject and object entities, while a multi-label classification method is used to identify all potential relations. Next, for each subject–relation pair, the start position of the corresponding object entity is predicted. If this predicted position matches one of the object entities extracted in the initial stage, the resulting triple is considered valid and retained.

PRGC (Zheng et al., 2021): This model begins by identifying a set of candidate relations within the sentence and constructing a global correspondence matrix between subject and object entities. In the next stage, relation-specific sequence taggers are used to label subject and object entities with fine-grained annotations, allowing for precise entity recognition. Finally, the global correspondence matrix guides the selection process, helping to determine which triples are valid.

BiRTE (Ren et al., 2022): This paper proposes a multi-stage bidirectional tagging approach. BiRTE first detects subject entities and then identifies the corresponding object entities based on them. It then reverses the process—starting with object entities and locating the related subject entities. In the final stage, the model classifies the relation between each subject-object pair.

A.2 Parameter Settings

We use the code released by authors of PtrNet (Nayak and Ng, 2020), GRTE (Ren et al., 2021), TDEER (Li et al., 2021), PRGC (Zheng et al., 2021), and BiRTE (Ren et al., 2022) to run our experiments. We use batch size of 8 for BERT-base experiments and batch size of 6 for BERT-large experiments. We train the models with Adam optimizer and learning rate of $2e - 05$. We report average precision, recall, F1 score and their standard deviation of three runs of these models in Table 5 and 6.

A.3 Related Work

Extracting relational facts from text is a key task in natural language processing, aimed at building or enriching knowledge bases (KBs) with new information. Most research in this area focuses on extracting relational triples—comprising a subject entity, a relation, and an object entity—from textual data. Mintz et al. (2009); Hoffmann et al. (2011); Zeng et al. (2014, 2015); Nayak and Ng (2019) proposed two-step pipeline approach to solve this task.

Miwa and Bansal (2016) introduced an end-to-end neural network that jointly performs entity extraction and relation classification within a unified framework. Building on this idea, Zheng et al. (2017) proposed a sequence tagging approach for extracting relational triples, while Zeng et al. (2018) and Nayak and Ng (2020) employed sequence-to-sequence models for the same task. Further advancements came from Zheng et al. (2021); Ren et al. (2021, 2022); Li et al. (2021), who developed various BERT-based neural architectures to address relational triple extraction.

Generative models like BART (Lewis et al., 2019) were leveraged by Cabot and Navigli (2021), who used different linearization strategies to extract relational triples at both the sentence and document levels. Extending beyond standard triples, Chia et al. (2022) tackled the extraction of hyper-relational facts by incorporating additional qualifiers such as time, location, and quantity. To further support hyper-relational fact extraction, Luo et al. (2024) proposed an N-ary relation extraction framework capable of capturing complex relationships involving multiple entities and qualifiers.

Text: Alberta and its neighbour Saskatchewan were districts of the Northwest Territories until they were established as provinces on September 1 , 1905 .
Ground-truth Hyper-relation Facts: 1. <Alberta, shares_border_with, Northwest Territories, start_time, 1905> 2. <Saskatchewan, shares_border_with, Northwest Territories, start_time, 1905>
Extracted Relational Triples: 1. <Alberta, shares_border_with, Northwest Territories> 2. <Alberta, shares_border_with-start_time, 1905> 3. <Saskatchewan, shares_border_with, Northwest Territories> 4. <Saskatchewan, shares_border_with-start_time, 1905>
Reconstructed Hyper-relation Facts: 1. <Alberta, shares_border_with, Northwest Territories, start_time, 1905> 2. <Saskatchewan, shares_border_with, Northwest Territories, start_time, 1905>
Explanation: There is no ambiguity present in this example. The combination of <Alberta, shares_border_with> includes only one triple with object value <Northwest Territories>. The corresponding qualifier value <start_time, 1905> gets attached to this triple. Same goes for the combination of <Saskatchewan, shares_border_with> with object value <Northwest Territories>.
Text: The idea for the Woody Allen story came from David ' s experiences working with Allen ; he briefly appeared in Radio Days (1987) and New York Stories (1989) .
Ground-truth Hyper-relation Facts: 1. <Woody Allen, notable_work, New York Stories, point_in_time, 1989> 2. <Woody Allen, notable_work, Radio Days, point_in_time, 1987>
Extracted Relational Triples: 1. <Woody Allen, notable_work, New York Stories> 2. <Woody Allen, notable_work-point_in_time, 1989> 3. <Woody Allen, notable_work, Radio Days> 4. <Woody Allen, notable_work-point_in_time, 1987>
Reconstructed Hyper-relation Facts: 1. <Woody Allen, notable_work, New York Stories, point_in_time, 1989> 2. <Woody Allen, notable_work, Radio Days, point_in_time, 1987>
Explanation: The combination of <Woody Allen, notable_work> features in two triples with object values <New York Stories> and <Radio Days>. The corresponding qualifier values <point_in_time, 1989> and <point_in_time, 1987> could be associated with either of these two triples. In this case, our distance-based heuristics algorithm correctly maps the <point_in_time, 1989> to <Woody Allen, notable_work, New York Stories> and <point_in_time, 1987> to <Woody Allen, notable_work, Radio Days>.
Text: Today , the record is held by Gareth Bale , who , in 2013 became the first player to cost € 100m when he transferred from Tottenham Hotspur to Real Madrid .
Ground-truth Hyper-relation Facts: 1. <Gareth Bale, member_of_sports_team, Real Madrid, start_time, 2013> 2. <Gareth Bale, member_of_sports_team, Tottenham Hotspur, end_time, 2013>
Extracted Relational Triples: 1. <Gareth Bale, member_of_sports_team, Real Madrid> 2. <Gareth Bale, member_of_sports_team-start_time, 2013> 3. <Gareth Bale, member_of_sports_team, Tottenham Hotspur> 4. <Gareth Bale, member_of_sports_team-end_time, 2013>
Reconstructed Hyper-relation Facts: 1. <Gareth Bale, member_of_sports_team, Tottenham Hotspur, start_time, 2013> 2. <Gareth Bale, member_of_sports_team, Tottenham Hotspur, end_time, 2013>
Explanation: The combination of <Gareth Bale, member_of_sports_team> appears in two triples with object values <Real Madrid> and <Tottenham Hotspur>. The corresponding qualifier values <start_time, 2013> and <end_time, 2013> could apply to either of these triples. However, our distance-based heuristics algorithm erroneously maps both qualifiers to <Gareth Bale, member_of_sports_team, Tottenham Hotspur> because it calculates the token distances, finding that both qualifiers were closest to this object.
Text: Gurung Hill is a mountain near the Line of Actual Control in the region of Aksai Chin that is controlled by China but claimed by India .
Ground-truth Hyper-relation Facts: 1. <Aksai Chin, country, India, statement_disputed_by, China> 2. <Aksai Chin, country, China, statement_disputed_by, India>
Extracted Relational Triples: 1. <Aksai Chin, country, India> 2. <Aksai Chin, country-statement_disputed_by, China> 3. <Aksai Chin, country, China> 4. <Aksai Chin, country-statement_disputed_by, India>
Reconstructed Hyper-relation Facts: 1. <Aksai Chin, country, China, statement_disputed_by, China> 2. <Aksai Chin, country, India, statement_disputed_by, India>
Explanation: The combination of <Aksai Chin, country> appears in two triples with object values <India> and <China>. The associated qualifier values <statement_disputed_by, China> and <statement_disputed_by, India> could potentially apply to either of these triples. However, our distance-based heuristics algorithm makes an error in mapping these qualifiers. It assigns <statement_disputed_by, China> to <Aksai Chin, country, China> and <statement_disputed_by, India> to <Aksai Chin, country, India>.

Table 7: Few examples to explain the working of our distance-based heuristics algorithm in hyper-relational fact reconstruction process from the extracted triples.