Path-based Link Prediction on Hyper-relational Knowledge Graph

Anonymous ACL submission

Abstract

Recently, path-based Graph Neural Networks (GNNs) achieve promising performance of the link prediction task on benchmark Knowledge Graphs (KGs). However, there is no research on leveraging path-based GNNs to promote the more general case of KGs, namely hyper-relational KGs (HKGs). To bridge this research gap, we study the path-based GNNs and discover that existing path-based GNNs fail to handle HKGs because they cannot well explore the external information (i.e., qualifiers) stored in HKGs. In this paper, we propose a novel framework, **Hyper Path**-based **G**raph **N**eural **N**etwork (HyperPGNN) for HKGs. Specifically, we propose a novel Hyper2Tri conversion and hyper query learner to better enable the path-based GNNs to understand qualifiers in HKG, then capture them into the graph learning. Results show our method achieve good performance in both transductive and inductive settings. Codes are available at `https://anonymous.4open.science/r/Path_based_LP_HKG`.

1 Introduction

Hyper-relational Knowledge Graphs (HKGs) allow equipping the main triplet (h, r, t) with qualifiers $\{(qr_k : qe_k)\}$, providing contextual information for (h, r, t). As the example in Fig. 1, the fact *(Albert Einstein, employer, University of Zurich)* with qualifiers *{(start time: 1909), (end time: 1911)}* can be distinguished from the fact *(Albert Einstein, employer, Swiss Federal Institute of Intellectual Property)* with qualifiers *{(start time: 1902), (end: 1909)}*. Like KG, link prediction is a fundamental task in HKGs that predicts missing facts based on known ones. Recently, a series of methods extends the classic KG embedding models from KGs to HKGs, e.g., m-TransH (Wen et al., 2016) from TransH (Wang et al., 2014), GETD (Liu et al., 2020) from TuckER (Balazevic et al., 2019), StarE (Galkin et al., 2020) from CompGCN (Vashishth et al., 2020).

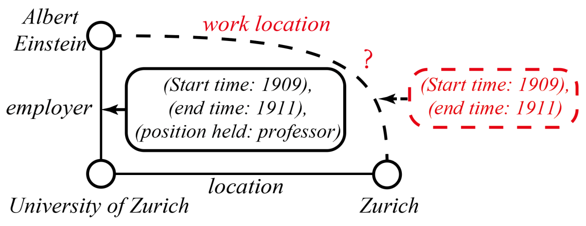
Albert Einstein
work location
?
(Start time: 1909), (end time: 1911), (position held: professor)
(Start time: 1909), (end time: 1911)
employer
location
University of Zurich
Zurich

Figure 1: Illustration of hyper-relational facts in HKGs.

Despite the success of existing methods for HKGs, those powerful path-based graph neural networks (GNNs) (Zhu et al., 2021, 2022; Zhang et al., 2023) that achieved outstanding performance on benchmark KGs have not been explored on the more general HKGs. Different from aggregating the information from neighbors (Schlichtkrull et al., 2018; Vashishth et al., 2020), path-based GNNs leverage the information from the possible paths between head and tail entities to predict the link. Inspired by their success, the path-based idea may also work on HKGs. From the main facts *(Albert Einstein, employer, University of Zurich)* with qualifiers *{(start time: 1909), (end time: 1911)}* and *(University of Zurich, location, Zurich)*, we can infer *Albert Einstein* work on *Zurich* between year *1909* to *1911* as shown in Figure 1.

However, it is a non-trivial task to extend existing path-based GNNs to HKGs. Current path-based works can only traverse and leverage the main triplets without qualifiers in HKGs. Thus, they will miss the important information stored in the qualifiers, like the working period in Figure 1. Therefore, to leverage the capability of path-based GNNs on HKGs, we propose a new model named **Hyper Path**-based **G**raph **N**eural **N**etwork (HyperPGNN) in this paper. More concretely, we propose a conversion method Hyper2Tri to covert the topology of the hyper-relational facts into the structures that can be incorporated with existing path-based

GNNs. After the topology conversion, we design a query learner to encode qualifiers to further capture the information of qualifiers. Empirical results show that HyperPGNN achieves good performance on benchmark HKG datasets under both transductive and inductive settings.

2 Proposed Method

A HKG can be defined as $\mathcal{G} = (\mathcal{V}, \mathcal{R}, \mathcal{D})$, where $\mathcal{V}$ is the set of entities, $\mathcal{R}$ represents the set of relations and $\mathcal{D}$ contains a set of hyper-relational facts. Each hyper-relational fact consists of a main triplet (h, r, t) and a list of qualifiers $\{(qr_k : qe_k)\}_{k=1}^{n}$, where $h, t, qe_k \in \mathcal{V}$ and $r, qr_k \in \mathcal{R}$. Link prediction on HKGs aims to predict missing head or tail entities in the query hyper-relational fact $\{(?, r, t), (qr_k : qe_k)_{k=1}^{n})\}$ or $\{(h, r, ?), (qr_k : qe_k)_{k=1}^{n})\}$. For simplicity, we use the notation of the tail prediction in the following part.

The overall framework of the model is shown in Figure 2. We propose a Hyper2Tri conversion to utilize the qualifier information in known facts (section 2.1). We design a hyper query learner to incorporate the qualifiers in the query (section 2.2). Then, we adopt an advanced path-based GNN to get an expressive representation of possible hyper-relational facts (in Appx. A).

2.1 Hyper Facts Conversion

Intuitively, we may convert the qualifiers into collections of triplets to facilitate path-based embedding with the restricting information from statement qualifiers. Existing conversion approaches, e.g., Star-to-Clique proposed by m-TransH (Wen et al., 2016), generally treat qualifier entities equally with triplet entities. They first reformulate the statement as a set of relation-entity pairs $P = (\{r_h : h\}, \{r_t : t\}, \{qr_i : qe_i\}_{i=1}^{n})$, then form a triplet $(e_1, r_1\text{-}r_2, e_2)$, for every $\{r_1 : e_1\}, \{r_2 : e_2\} \in P$. It has two major defects: 1) losing the semantic difference between triplet entities and qualifier entities, 2) introducing too much noise while expanding possible path formulation.

Thus, we propose the following Hyper2Tri (Hyper-relation to Triangle relations) conversion. Given a statement $\{(h, r, t), (qr_k : qe_k)_{k=1}^{n})\}$, every $(qr_k : qe_k)$ is converted to two labeled edges $(h, r\text{-}qr_k, qe_k)$ and $(t, r\text{-}qr_k, qe_k)$. Here, $r\text{-}qr_k$ is the new type of relation representing qualifier relation qr_k of relation r, and we denoted its type as the pair of r and qr_k. See the Hyper Facts Conversion part of Figure 2 for an illustration. In other words, a statement with n qualifiers will be decomposed into $2n+1$ triplets, including the main triplet and $2n$ triplets derived from the qualifiers. Hence, the original HKG is transformed into a knowledge graph that can be processed by previous path-based link prediction frameworks without losing information in qualifiers. Note that the newly constructed triplets will form new paths together with existing main triplets, providing more information about the qualifiers. As shown in Figure 3, the proposed conversion produce new paths related to the *Argentina national association football team*, making it possible to infer that the nationality of *Lionel Messi* is *Argentina*, and he won the *2022 FIFA World Cup*.

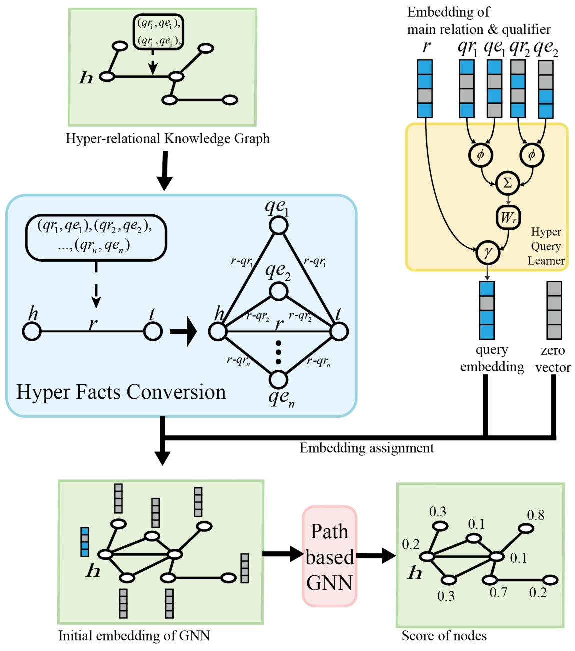

Figure 2: An overview of our pipeline

Compared with Star-to-Clique (Wen et al., 2016) Hyper2Tri strengthens the connection between the primal entity and qualifier entities while weakening the bonds among qualifier entities. And Hyper2Tri creates no new bonds between qualifier entities, thus reducing information noise. Besides, another possible design is to directly encode the hyper-relations in the paths and using these representations to conduct predictions. However, the performance will be limited without collecting path-based information about qualifiers.

2.2 Hyper Query Learner

Existing path-based GNNs on KGs initialize the head node with the embedding of query relation, and use the embedding of candidate nodes after message passing to compute the scores of candidates. Zhang et al. (2021) shows that this initialization can be considered as a labeling trick, making the network highly expressive. Likewise, we design a hyper query learner to achieve the embedding $\mathbf{q}$ of a given query $\{(h, r, ?), (qr_k : qe_k)_{k=1}^{n})\}$ [1]:

$$\mathbf{q} \leftarrow \gamma(\mathbf{r}, \mathbf{W}_r \cdot \Sigma_k \phi(\mathbf{qr}_k, \mathbf{qe}_k)), \tag{1}$$

where $\mathbf{qr}_k, \mathbf{qe}_k$ and $\mathbf{r}$ are the learned embedding of qr_k, qe_k and r respectively/ $\phi(\cdot)$ is the function to composite the embedding of qualifier keys and qualifier values, and can be any function of the form $\mathbb{R}^n \times \mathbb{R}^n \rightarrow \mathbb{R}^n$. We sum the composition results of all qualifiers to get the embedding of qualifiers, and then transform with a relation-specific matrix W_r. $\gamma(\cdot)$ is another composition function.

In this way, we integrate the information from the qualifiers included in the query into the embeddings, enabling the final prediction results to better satisfy the constraints imposed by the qualifiers. Note that the above process does not need the embedding of the head entity h , which is crucial for applying to the inductive setting.

2.3 Path-based GNN

Given a HKG $\mathcal{G}$, we convert it to a knowledge graph $\tilde{\mathcal{G}}$ as described in Section 2.1. For a query $\{(h, r, ?), (qr_k : qe_k)_{k=1}^{n})\}$, we initialize the representation on node h with the hyper query embedding $\mathbf{q}$ obtained in Equation 1. The representations of other nodes are initialized with zero vectors. Then the initial representations are fed into a Path-based GNN. The two modules proposed above can be integrated with any path-based methods designed for KGs. We adopt the NBFNet (Zhu et al., 2021) as the backbone path-based GNN.

3 Experiment

We use a 24GB NVIDIA GTX 3090 GPU to conduct the experiments. The reported results are averaged over five runs.

3.1 Dataset

Experiments are conducted under both transductive and inductive settings. For the transductive setting, we use Wikipeople (Guan et al., 2019a), JF17K (Rosso et al., 2020), and WD50K (Galkin et al., 2020). For the inductive setting, we use WP-IND, JF-IND, and MFB-IND proposed by Yadati (2020). In a transductive setting during testing, all the entities and relations in the main triplet and qualifiers are limited to those that have occurred in the training set. However, in the inductive setting, the model is required to predict query facts that may involve new head entities and new tail entities as potential candidates.

[1]We use the same notation for the query facts here to maintain symbol simplicity.

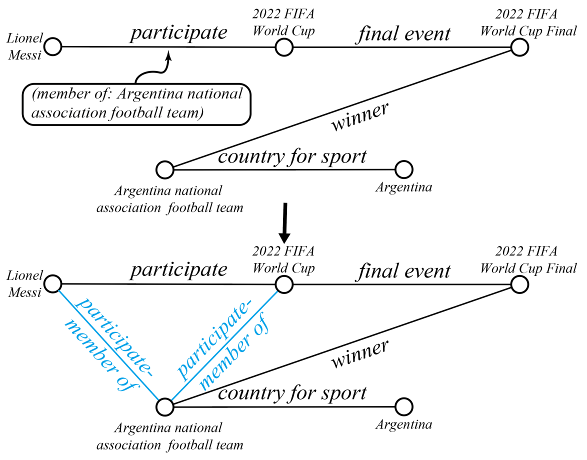

Figure 3: Hyper2Tri will form new paths for qualifiers

3.2 Experimental Setup

Base models. For the transductive setting, We compare the proposed method with StarE (Galkin et al., 2020), Hy-Transformer (Yu and Yang, 2021), GRAN (Wang et al., 2021) and QUAD (Shomer et al., 2022). For the inductive setting, we compare our results with GMPNN (Yadati, 2020).

Metric. We report the mean reciprocal ranking(MRR) and Hit@1,10 for the transductive setting for comparison with previous work. For inductive setting, we report MRR and Hit@1,3 to keep in line with existing work.

Hyperparameter Setting. We employ Adam (Kingma and Ba, 2014) to train our model for a maximum number of 1000 epochs, the learning rate is set to 0.001. We set the embedding size to be 32. For the hyper query learner, we use the rotate function in RotateE (Sun et al., 2019) as the $\phi(\cdot)$ in Equation 1 and $\gamma(\cdot)$ is a weighted sum function with the weight of qualifiers is set to 0.4. In the path-based GNN part, we use a GNN with 6 layers to compute the embedding. We use the mult function proposed by DisMult (Yang et al., 2015) as the message function and the pna (Corso et al., 2020) as the aggregation function. A two-layer MLP with a hidden layer of size 32 is used to compute the score of each candidate.

Table 1: Comparison on transductive datasets. The best results are in bold. the second-best results are underlined

Method	WD50K			WikiPeople			JF17K		
	MRR	Hit@1	Hit@10	MRR	Hit@1	Hit@10	MRR	Hit@1	Hit@10
m-TransH	-	-	-	0.063	0.063	0.300	0.206	0.206	0.463
StarE	0.349	0.271	0.496	0.491	0.398	**0.648**	0.574	0.496	0.725
Hy-Transformer	0.356	0.281	0.498	0.501	0.426	0.634	0.582	0.501	0.742
GRAN-hete	-	-	-	**0.503**	0.438	0.620	0.617	0.539	0.770
QUAD	0.348	0.270	0.497	0.466	0.365	0.624	0.582	0.502	0.740
QUAD(Parallel)	0.349	0.275	0.489	0.497	**0.431**	0.617	0.596	0.519	0.751
Ours	**0.362**	**0.283**	**0.505**	0.501	0.430	**0.648**	**0.657**	**0.585**	**0.771**

Table 2: Comparison on inductive datasets, The best results are in bold. the second-best results are underlined

Method	WP-IND			JF-IND			MFB-IND		
	MRR	Hit@1	Hit@10	MRR	Hit@1	Hit@10	MRR	Hit@1	Hit@10
G-MPNN-sum	0.010	0.051	0.185	0.240	0.188	0.331	0.292	0.203	0.479
G-MPNN-mean	0.093	0.029	0.189	0.146	0.084	0.221	0.242	0.151	0.416
G-MPNN-max	0.160	0.093	0.293	0.224	0.159	0.315	0.268	0.191	0.283
Ours	**0.312**	**0.262**	**0.419**	**0.463**	**0.412**	**0.487**	**0.507**	**0.432**	**0.531**

3.3 Performance Comparison

Table 1 and Table 2 show the performance comparison in the transductive setting and inductive setting respectively. The results of the baselines were gathered from the original literature, and "-" indicates the results were not reported. We omit the comparison with RAE (Zhang et al., 2018), NaLP-Fix (Guan et al., 2019b), HINGE (Rosso et al., 2020) in Table 1 since StarE consistently obtains better results than them. We select the GRAN-hete For GRAN (Wang et al., 2021) since it consistently outperforms other variants, namely GRAN-homo and GRAN-complete. For the comparison with G-MPNN, the results in the original literature take the prediction of values in qualifiers into consideration. we retrain the model to predict head and tail entities in main triplets with the reported hyperparameters and get slightly better results. It can be seen that our model consistently outperforms the state-of-the-art.

3.4 Ablation Study

We compare with multiple variants of our method to demonstrate the effectiveness of our design. (i) **BaseModel** eliminates the Hyper2Tri conversion, dropping all qualifiers in the known facts, and removes the Hyper Query Learner, relying solely on the main relation r for the query embedding. (ii) **NoConversion** utilize the hyper query learner but removes the Hyper2Tri conversion. (iii) **Star2Clique** substitute the Hyper2Tri conversion with Star2Clique conversion while keeping the Hyper Query Learner. (iv) **NoQueryLeaner** keeps the Hyper2Tri conversion and elimates Hyper Query Learner. (v) **FullModel** represent exactly the proposed method.

Table 3: Abation study in transductive setting, here we use H@10 as abbreviation for Hit@10 to save space

Method	WD50K		WP		JF17K	
	MRR	H@10	MRR	H@10	MRR	H@10
BaseModel	0.153	0.295	0.251	0.468	0.341	0.570
NoConversion	0.203	0.345	0.322	0.477	0.419	0.619
Star2Clique	0.297	0.425	0.416	0.508	0.543	0.748
NoQueryLeaner	0.276	0.462	0.383	0.471	0.597	0.763
FullModel	**0.362**	**0.505**	**0.499**	**0.648**	**0.657**	**0.771**

Table 3 presents the results of the variants in transductive setting. By comparing the FullModel, the NoConversion and the Star2Clique, we show the efficiency of the proposed Hyper2Tri conversion. By Comparing the FullModel and the No queryLearner, we show the importance of the hyper query learner. By comparing the Basemodel with Noconversion and NoQueryLearner, we show the two module contribute to performance independently.

4 Conclusion

We introduce a path-based link prediction model for HKGs. Our model uses a Hyper2Tri conversion to incorporate the information of the qualifiers in the known facts and utilizes a hyper query learner to embed the qualifiers in the query fact. Experiments conducted on multiple datasets demonstrate that the proposed model achieves state-of-the-art performance in both the inductive and transductive settings.

References

Ivana Balazevic, Carl Allen, and Timothy Hospedales. 2019. TuckER: Tensor factorization for knowledge graph completion. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5185–5194, Hong Kong, China. Association for Computational Linguistics.

Gabriele Corso, Luca Cavalleri, Dominique Beaini, Pietro Liò, and Petar Veličković. 2020. Principal neighbourhood aggregation for graph nets. *Advances in Neural Information Processing Systems*, 33:13260–13271.

Mikhail Galkin, Priyansh Trivedi, Gaurav Maheshwari, Ricardo Usbeck, and Jens Lehmann. 2020. Message passing for hyper-relational knowledge graphs. *arXiv preprint arXiv:2009.10847*.

Saiping Guan, Xiaolong Jin, Yuanzhuo Wang, and Xueqi Cheng. 2019a. Link prediction on n-ary relational data. In *Proceedings of the 28th International Conference on World Wide Web (WWW'19)*, pages 583–593.

Saiping Guan, Xiaolong Jin, Yuanzhuo Wang, and Xueqi Cheng. 2019b. Link prediction on n-ary relational data. In *The world wide web conference*, pages 583–593.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Yu Liu, Quanming Yao, and Yong Li. 2020. Generalizing tensor decomposition for n-ary relational knowledge bases. In *Proceedings of the web conference 2020*, pages 1104–1114.

Paolo Rosso, Dingqi Yang, and Philippe Cudré-Mauroux. 2020. Beyond triplets: hyper-relational knowledge graph embedding for link prediction. In *Proceedings of the web conference 2020*, pages 1885–1896.

Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *The Semantic Web: 15th International Conference, ESWC 2018, Heraklion, Crete, Greece, June 3–7, 2018, Proceedings 15*, pages 593–607. Springer.

Harry Shomer, Wei Jin, Juanhui Li, Yao Ma, and Jiliang Tang. 2022. Learning representations for hyper-relational knowledge graphs. *arXiv preprint arXiv:2208.14322*.

Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. 2019. Rotate: Knowledge graph embedding by relational rotation in complex space. *arXiv preprint arXiv:1902.10197*.

Shikhar Vashishth, Soumya Sanyal, Vikram Nitin, and Partha Talukdar. 2020. Composition-based multi-relational graph convolutional networks. In *International Conference on Learning Representations*.

Quan Wang, Haifeng Wang, Yajuan Lyu, and Yong Zhu. 2021. Link prediction on n-ary relational facts: A graph-based approach. *arXiv preprint arXiv:2105.08476*.

Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. Knowledge graph embedding by translating on hyperplanes. In *Proceedings of the AAAI conference on artificial intelligence*, volume 28.

Jianfeng Wen, Jianxin Li, Yongyi Mao, Shini Chen, and Richong Zhang. 2016. On the representation and embedding of knowledge bases beyond binary relations. *arXiv preprint arXiv:1604.08642*.

Naganand Yadati. 2020. Neural message passing for multi-relational ordered and recursive hypergraphs. In *Advances in Neural Information Processing Systems (NeurIPS) 33*. Curran Associates, Inc.

Bishan Yang, Scott Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2015. Embedding entities and relations for learning and inference in knowledge bases. In *Proceedings of the International Conference on Learning Representations (ICLR) 2015*.

Donghan Yu and Yiming Yang. 2021. Improving hyper-relational knowledge graph completion. *arXiv preprint arXiv:2104.08167*.

Muhan Zhang, Pan Li, Yinglong Xia, Kai Wang, and Long Jin. 2021. Labeling trick: A theory of using graph neural networks for multi-node representation learning. *Advances in Neural Information Processing Systems*, 34:9061–9073.

Richong Zhang, Junpeng Li, Jiajie Mei, and Yongyi Mao. 2018. Scalable instance reconstruction in knowledge bases via relatedness affiliated embedding. In *Proceedings of the 2018 world wide web conference*, pages 1185–1194.

Yongqi Zhang, Zhanke Zhou, Quanming Yao, Xiaowen Chu, and Bo Han. 2023. Adaprop: Learning adaptive propagation for graph neural network based knowledge graph reasoning. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 3446–3457.

Zhaocheng Zhu, Xinyu Yuan, Louis-Pascal Xhonneux, Ming Zhang, Maxime Gazeau, and Jian Tang. 2022. Learning to efficiently propagate for reasoning on knowledge graphs. *arXiv preprint arXiv:2206.04798*.

Zhaocheng Zhu, Zuobai Zhang, Louis-Pascal Xhonneux, and Jian Tang. 2021. Neural bellman-ford networks: A general graph neural network framework for link prediction. *Advances in Neural Information Processing Systems*, 34:29476–29490.

A Supplementary Materials for the Method

In this section, we briefly describe the NBFNet (Zhu et al., 2021) to make the paper self-contained. The message passing process of layer l is defined as:

$$\mathbf{e}_i^{(l)} \leftarrow agg(\{msg(\mathbf{e}_j^{(l-1)}, w_{r,\mathbf{q}})\} \cup \mathbf{e}_i^{(0)}). \quad (2)$$

The agg represents the aggregation function and the msg is the massage function. e_j is the node in the neighborhood of node e_i. The edge embedding $w_{r,\mathbf{q}}$ is related to the type of edge between them(denoted by r) and the embedding of query $\mathbf{q}$. By utilizing this message passing scheme, the final embedding of the node e_i only depends on the query embedding and the embedding of edges in the paths between h and e_i. Note the final embedding is also independent on the embedding of node h and e_i, making it possible to apply our model to inductive setting.

After message passing, the representation of the pair (h, e_i) for the query is the embedding of e_i in the last layer, i.e. $\mathbf{e}_i^L$. Then the score of $\{(h, r, e_i), (qr_k : qe_k)_{k=1}^n)\}$ is computed using an MLP:

$$score \leftarrow MLP(\mathbf{e}_i^L). \quad (3)$$