Multi-Aspect Explainable Inductive Relation Prediction by Sentence Transformer

Zhixiang Su[1,2,3], **Di Wang**[3,4*], **Chunyan Miao**[1,2,3,4], **Lizhen Cui**[2,5]
[1]School of Computer Science and Engineering, Nanyang Technological University (NTU), Singapore
[2]SDU-NTU Centre for Artiticial Intelligence Research (C-FAIR), Shandong University (SDU), China
[3]Joint NTU-WeBank Research Centre on Fintech, NTU, Singapore
[4]Joint NTU-UBC Research Centre of Excellence in Active Living for the Elderly (LILY), NTU, Singapore
[5]School of Software, SDU, China
{zhixiang002, wangdi, ascymiao}@ntu.edu.sg, clz@sdu.edu.cn

Abstract

Recent studies on knowledge graphs (KGs) show that path-based methods empowered by pre-trained language models perform well in the provision of inductive and explainable relation predictions. In this paper, we introduce the concepts of relation path coverage and relation path confidence to filter out unreliable paths prior to model training to elevate the model performance. Moreover, we propose Knowledge Reasoning Sentence Transformer (KRST) to predict inductive relations in KGs. KRST is designed to encode the extracted reliable paths in KGs, allowing us to properly cluster paths and provide multi-aspect explanations. We conduct extensive experiments on three real-world datasets. The experimental results show that compared to SOTA models, KRST achieves the best performance in most transductive and inductive test cases (4 of 6), and in 11 of 12 few-shot test cases.

Introduction

As an important tool for providing side information for question answering and recommendation systems (Ji et al. 2021), knowledge graph (KG) has been widely studied. A KG is typically expressed in terms of triplets $G = \{(h_i, r_i, t_i)|i = 1, 2, 3, .., m\}$, which contains entities $h_i, t_i \in E_G$ and relations $r_i \in R_G$. Because of the incompleteness of KGs in practice, knowledge graph completion (KGC) is needed to improve the quality of KGs. One of the most important KGC tasks is relation prediction. Given the target triplet (h, r, t), a relation prediction query is usually set by masking the entity h or t in the given triplet and letting the model predict the masked entity based on the other entity and the relation type.

Embedding-based methods are probably the most commonly applied SOTA models. With a fixed set of entities and relations, embedding-based methods perform fairly well in KGC tasks. However, most existing embedding-based methods are not explainable and cannot deal with inductive situations, making them not suitable for modeling real-world dynamic KGs, wherein new entities and relations may be added all the time. Inductive relation prediction requires the model to handle unseen entities unavailable in the training graph.

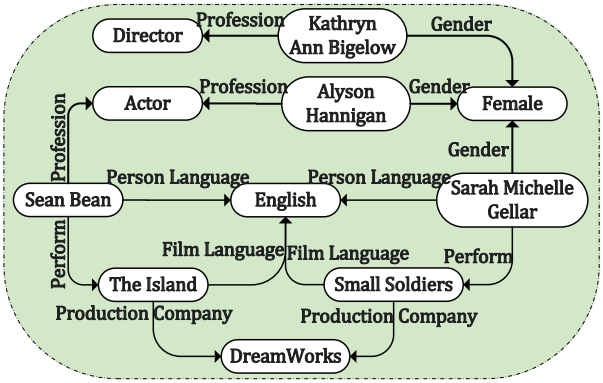


Figure 1: An example of KG.

GNN-based methods take advantage of the KG's graph connectivity, thus, are capable of predicting new entities with a sufficient number of known neighboring entities. Recent GNN-based method GRAIL (Teru, Denis, and Hamilton 2020) is shown to be capable of conducting inductive relation predictions. Nonetheless, extracting explainable rules is left for further exploration in GRAIL. So far, no evidence shows that GRAIL is explainable.

Different from GNN-based methods, path-based methods take advantage of the graph connectivity by analyzing paths between the head and tail entities. When we focus only on the relations in one path, they can be formulated as a Horn rule (Horn 1951) for knowledge reasoning. Therefore, in path-based methods, new entities can be easily modeled by applying the summarized rules. For example, w.r.t Figure 1, for triplet $(\textit{Small Soldiers}, \textit{Film Language}, \textit{English})$ and the corresponding path

$$\textit{Small Soldiers} \xrightarrow{\textit{Perform}^{-1}} \textit{Sarah} \xrightarrow{\textit{Person Language}} \textit{English}, \quad (1)$$

we can summarize that

$$(y, \textit{Perform}, x) \wedge (y, \textit{PersonLanguage}, z) \rightarrow (x, \textit{FilmLanguage}, z). \quad (2)$$

Such rule-based operations make path-based methods inductive and highly explainable. BERTRL (Zha, Chen, and Yan 2022), which employs the pre-trained language model BERT for scoring, is one typical model of such type. Using contextual descriptions of entities and relations, BERTRL

*Corresponding Author

can deal with inductive cases and provide single-path explanations, achieving the best SOTA results in the literature.

To further improve the relation prediction performance and strive for better explainability, in this paper, we propose **Knowledge Reasoning Sentence Transformer (KRST)**, which is a novel path-based model built upon the sentence transformer. The key innovations of KRST are as follows:

Path extraction. Although paths between head and tail entities can be easily extracted, most of the extracted paths may be unreliable. Moreover, unreliable short paths with words also appearing in the target triplets are usually preferred by the pre-trained language model. To assess the reliability of paths and only use reliable ones for model training, we propose the concepts of relation path coverage and relation path confidence (see Definitions 4 and 5, respectively).

Pre-trained language model. Compared with the commonly applied BERT for sequence classification model, the sentence transformer adopting cosine similarity achieves a higher performance in multiple tasks (Reimers and Gurevych 2019). Therefore, we adopt sentence transformer in KRST to encode triplets and paths into embeddings, which allows us to explicitly compare between embeddings and cluster paths w.r.t various aspects.

Loss function. In KRST, we compare the similarity between paths and triplets using the cosine similarity score. In the case of negative triplets, negative scores are not necessarily close to -1. Nevertheless, commonly applied binary classification loss functions (e.g., cross-entropy) often make the model over-confident by requiring the prediction result to be close to either 1 or -1. Therefore, we use cosine embedding loss instead, aiming to better elevate the model performance.

Our key contributions in this paper are as follows:

(i) We propose two novel path extraction metrics named relation path coverage and relation path confidence, and a novel path-based model named KRST. To the best of our knowledge, KRST is the first sentence transformer model for knowledge graph path encoding.

(ii) We develop a comprehensive approach for relation prediction explanation, which enables the provision of explanations from multiple paths and multiple perspectives.

(iii) We assess the performance of KRST on three transductive and inductive datasets: WN18RR, FB15k-237, and NELL-995. KRST obtains significantly better results than SOTA models in majority cases (15 of 18).

Related Work

Embedding-based methods: Such methods (e.g., ComplEx (Trouillon et al. 2017), ConvE (Dettmers et al. 2018), and TuckER (Balaevi, Allen, and Hospedales 2019)) generate embeddings for entities and relations in the latent space. Score functions are proposed for training and evaluating within triplets. The most representative embedding-based methods are the translation methods (e.g., TransE (Bordes et al. 2013), TransH (Wang et al. 2014), TransR (Lin et al. 2015), and TransD (Ji et al. 2015)). The key idea behind the translation models is to treat the process of finding valid triplets as the translation operation of entities through relationships, define the corresponding score function, and then minimize the loss function to learn the representation of entities and relationships (Chen et al. 2020).

GNN-based methods: Such methods (e.g., CompGCN (Vashishth et al. 2019) and R-GCN (Schlichtkrull et al. 2018)) pass messages between a node and its neighbors. These approaches take advantage of the graph connectivity. They are able to deal with a particular inductive situation where the new entity is surrounded by entities already known. GRAIL (Teru, Denis, and Hamilton 2020) is proposed to handle KGs with entirely new entities. However, GRAIL is not explainable as the author stated in (Teru, Denis, and Hamilton 2020). Challenged by the number of reachable entities that grows exponentially with the search depth, when given a dense KG, GNN-based methods may not well capture the correct long path information and hence may not perform well in relation prediction tasks.

Path-based methods: Such methods aim to find one (or multiple) logical reasoning path(s) between the query head and tail entities. PRA (Lao, Mitchell, and Cohen 2011) and AMIE (Galárraga et al. 2013) generate Horn rules (Horn 1951), which have a broader definition than paths. However, due to noises in real-world KGs, their performance is limited because they are only applicable for exact matches. DeepPath (Xiong, Hoang, and Wang 2017) and MINERVA (Das et al. 2017) learn to generate paths by reinforcement learning, whereby positive rewards are given when having successful target arrivals. These approaches can be applied to new entities and are naturally explainable. Nevertheless, they are also challenged by the exponentially growing reachable entities, which leads to sparse rewards.

Methods with pre-trained language model: With the great success achieved in various NLP tasks, pre-trained language models (PLMs) (e.g., BERT (Devlin et al. 2018), GPT (Radford et al. 2018) and XLNet (Yang et al. 2019)) show great potential in dealing with contextual descriptions. KG-BERT (Yao, Mao, and Luo 2019) extends embedding-based methods by fine-tuning BERT. Using contextual descriptions for entities and relations, BERT model is able to understand a triplet and output a classification label. KG-BERT works well in inductive settings but is not explainable because of the incomprehensible embeddings. Different from KG-BERT, BERTRL (Zha, Chen, and Yan 2022) incorporates path-based methods with BERT. Specifically, BERTRL converts triplets and the corresponding paths into sentences, and fine-tunes the pre-trained BERT for sequence classification. Because BERTRL uses all paths (shorter than L) as inputs without filtering, its performance may be limited. Recently, sentence transformer models is shown to outperform BERT on common STS and transfer learning tasks (Reimers and Gurevych 2019). In addition, BERT for sequence classification requires two sentences to input together and make an implicit comparison, while sentence transformer encodes sentences separately, allowing more flexible comparisons among sentences. In this paper, we adopt sentence transformer to provide a multi-aspect explanation.

Preliminary

We start this section by introducing the commonly applied logical reasoning path. Then we introduce the definition and common settings for inductive relation prediction.

Definition 1 (Logical Reasoning Path)**.** *Given a KG* $G = \{(h_i, r_i, t_i)|i = 1, 2, 3, .., m\}$*,* $h_i, t_i \in E_G$ *and* $r_i \in R_G$*, one possible logical reasoning path* $p(h, r, t)$ *and the corresponding relation path* $R_p(h, r, t)$ *are defined as follows:*

$$p(h, r, t) = h \xrightarrow{r_1} e_1 \xrightarrow{r_2} e_2 \xrightarrow{r_3} ... \xrightarrow{r_{n-1}} e_{n-1} \xrightarrow{r_n} t, \quad (3)$$

$$R_p(h, r, t) = (r_1, r_2, ..., r_{n-1}, r_n), \quad (4)$$

where $(h, r_1, e_1), ..., (e_{n-1}, r_n, t) \in G - \{(h, r, t)\}$.

From the perspective of knowledge graph reasoning, the relation prediction task can be viewed as a logical induction problem to identify the inductive and explainable logical reasoning paths. Because logical reasoning paths are sequential in nature, we can easily convert them into sentences. However, a large proportion of paths are either illogical or entity-dependent in real-world KGs, which cannot be applied to inductive relation prediction (see Definition 2). To address this problem, we define relation path coverage and relation path confidence to filter out unreliable or meaningless paths (see Definitions 4 and 5, respectively).

Definition 2 (Inductive Relation Prediction)**.** *Given a training graph* $G_{train}(E_{G_{train}}, R_{G_{train}})$*, a testing graph* $G_{test}(E_{G_{test}}, R_{G_{test}})$ *and a query triplet* (h_q, r_q, t_q)*, a relation prediction is inductive if:*

- $E_{G_{train}} \cap E_{G_{test}} = \emptyset$,
- $R_{G_{test}} \subseteq R_{G_{train}}, r_q \in R_{G_{train}}$.

With the emergence of real-world ever-evolving KGs, dealing with new relations and entities is a necessity. We focus on the inductive setting introduced by GRAIL (Teru, Denis, and Hamilton 2020), which contains a training graph, a testing graph, and a series of query triplets. Only the training graph is visible during training. Because both GNN-based methods and path-based methods make use of the graph connectivity, the test graph is only applied to extract neighbors or paths w.r.t query triplets during testing, respectively.

Methodology

Our proposed architecture comprises three steps for relation prediction, with the intuition that a triplet and its corresponding reliable paths should have similar semantics when converted to sentences. Specifically, KRST 1) filters unreliable logical reasoning paths extracted for model training, 2) converts paths and triplets into sentences by sentence formation, and 3) measures semantic similarity scores and makes relation predictions based on them.

Path Extraction with Filtering

Following the idea of knowledge reasoning, a logical reasoning path is determined by its support evidence w.r.t the target triplet. To train the model with both positive and negative samples, we extract logical reasoning paths from KGs for both positive and negative target triplets. However, many extracted paths are unreliable. For example, for positive triplet $(Sarah, Profession, Actor)$ and negative triplet $(Sarah, Profession, Director)$, we can both extract paths with the same relation path $(Gender, Gender^{-1}, Profession)$ as follows:

$$Sarah \xrightarrow{Gender} Female \xrightarrow{Gender^{-1}} Kathryn \xrightarrow{Profession} Actor, \quad (5)$$

$$Sarah \xrightarrow{Gender} Female \xrightarrow{Gender^{-1}} Alyson \xrightarrow{Profession} Director. \quad (6)$$

Paths (5) and (6) are similar and contribute little to distinguish the profession. Therefore, we consider paths with relation path $(Gender, Gender^{-1}, Profession)$ to be unreliable. Although the unreliable paths should be excluded from model training, because they are similar to the target triplet, they may be given a high similarity score and hence mistakenly considered as reliable. For example, another path for triplet $(Sarah, Profession, Actor)$ is as follows:

$$\begin{aligned} Sarah \xrightarrow{Perform} SmallSoldiers \xrightarrow{Company^{-1}} DreamWorks \\ \xrightarrow{Company} TheIsland \xrightarrow{Perform^{-1}} Sean \xrightarrow{Profession} Actor. \end{aligned} \quad (7)$$

This path is relatively more reliable than Path (5) because it involves the person in the same company for prediction. However, after being converted to sentences, Path (5) is assumed to be more similar by PLMs (shorter and contains words in target triplet). Because $(Sarah, Profession, Actor)$ is a positive triplet, model training reinforces this bias, leading to an even higher score for Path (5) after using it for training. These unreliable paths significantly limit the model performance. To exclude unreliable extracted paths from training, we perform path filtering. How we identify unreliable logical reasoning paths are introduced as follows.

Definition 3 (Relation Path Support)**.** *Given a triplet* (h, r, t) *and a relation path* $R_{p'} = (r_1^{'}, r_2^{'}, ..., r_{m-1}^{'}, r_m^{'})$*, the support of* $R_{p'}$ *is defined as follows:*

$$supp_{R_{p'}}(h, r, t) = \#p(h, r, t), R_p(h, r, t) = R_{p'}, \quad (8)$$

where $\#p(h, r, t)$ *denotes the number of logical reasoning paths on* G *w.r.t* (h, r, t).

Relation path support $supp_{R_{p'}}(h, r, t)$ measures the number of paths containing the same relations with $R_{p'}$ between h and t. With a larger support score, $R_{p'}$ is more common among paths between h and t. However, relation path support represents an unbounded value. Entity pairs with better connectivity usually have a larger support number. To further assess the ratio of the support, we define relation path coverage and relation path confidence, respectively.

Definition 4 (Relation Path Coverage)**.** *Given a triplet* (h, r, t) *and a relation path* $R_{p'}$*, the head and tail relation path coverage of* $R_{p'}$ *is defined as follows:*

$$cover^{h}_{R_{p'}}(h, r, t) = \frac{supp_{R_{p'}}(h, r, t)}{\#p(h, r, t'), |p(h, r, t')| = |R_{p'}|}, \quad (9)$$

$$cover^{t}_{R_{p'}}(h, r, t) = \frac{supp_{R_{p'}}(h, r, t)}{\#p(h', r, t), |p(h', r, t)| = |R_{p'}|}, \quad (10)$$

where $|\cdot|$ *denotes the length of the path.*

In (9) and (10), the number of paths starting from h or ending at t with length $|R_{p'}|$ is adopted as the denominator, respectively.

Definition 5 (Relation Path Confidence)**.** *Given a triplet (h, r, t), the head and tail relation path confidence of a relation path $R_{p'}$ is defined as follows:*

$$conf^{h}_{R_{p'}}(h, r, t) = \frac{supp_{R_{p'}}(h, r, t)}{\sum_{t' \in E_G - \{h\}} supp_{R_{p'}}(h, r, t')}, \quad (11)$$

$$conf^{t}_{R_{p'}}(h, r, t) = \frac{supp_{R_{p'}}(h, r, t)}{\sum_{h' \in E_G - \{t\}} supp_{R_{p'}}(h', r, t)}. \quad (12)$$

In (11) and (12), the total number of all reachable entities starting from h or ending at t w.r.t relation path $R_{p'}$ is adopted as the denominator, respectively.

Relation path coverage measures the ratio over all paths with the same source (or destination) and length, while relation path confidence measures the ratio over all entities that satisfy the target relation paths. To show the effectiveness of relation path confidence, we refer back to Paths (5) and (7). The relation path confidence score for Path (7) is much larger than Path (5). Because in the relation path of $(Gender, Gender^{-1}, Profession)$, $Gender^{-1}$ can lead to multiple people with various professions. Therefore, the number of paths satisfying the relation path of Path (5) is significantly more than that of Path (7), leading to a relatively smaller relation path confidence score for Path (5).

For path extraction, we adopt the breadth-first search, with the maximum search depth L and the maximum number of paths per triplet M. These parameters are set to avoid the generation of an unnecessarily large number of paths. Also, paths with excessive length are highly likely illogical. In addition, to ensure a sufficient number of paths per triplet are generated for effective model training, we synchronously perform path filtering and path extraction (see Algorithm 1).

Algorithm 1: Path Extraction

Input: KG G, query triplet (h, r, t), filter threshold α, filter function $f()$, max search depth L, max number of paths M
Output: List of extracted paths P

// Initialize search queue and state list
q = *Queue*(); *visited* = *List*(); *prev* = *List*()
q.push($(h, 0)$)
visited$[h]$ = *True*
// Breadth-first search
while q is not empty **do**
  $u, l = q$.pop()
  // Check whether search depth exceeds L
  if $l >= L$ **then**
    continue
  // Give priority to less frequent relations
  for v in $G[u]$ sorted by frequency $G[u][v]$['*relation*'] **do**
    if $v == t$ **then**
      if $u == h \&\& G[u][v]$['*relation*'] $== r$ **then**
        continue
      p =generatePath(*prev*, h, t)
      if $f(p) \geq \alpha$ **then**
        P.add(p)
    else if *visited*$[v] == 0$ **then**
      q.push($(v, l + 1)$); *visited*$(v) = 1$; *prev*$(v) = u$
    // Early break when generated path is enough
    if $|P| > M$ **then**
      break
  if $|P| > M$ **then**
    break
return P

Sentence Formation

To leverage the pre-trained parameters of the sentence transformer, KRST converts paths and triplets into sentences. Our key considerations are as follows:

Entity description selection. Both long (more than 20 words on average) and short entity descriptions in text are available in various datasets. Long descriptions usually make the sentence description imbalance between entities and relations when being modeled by PLMs, hence, they are less effective under inductive situations. In addition, sentences converted using long descriptions usually exceed the maximum sequence length of the PLM, which need to be truncated as incomplete. Therefore, only short descriptions are applied to KRST.

Inverse relation. To provide KRST with sequential order for positional encoding, entities' and relations' order in the paths should be preserved after being converted into sentences. A straightforward way is to place the descriptions following the entities' and relations' order in paths. However, inverse relations (e.g., *Gender*$^{-1}$ in Path (5)) occur in the paths and their descriptions are not available. Empirically, a systematic description in KG usually starts with descriptions related to the head entity and ends with descriptions related to the tail entity. Thus, we choose to use the inverse order of words in the sequential relation for inverse relations. Moreover, by doing so, we preserve the similarity between relations and inverse ones, making it easier for the model to understand symmetric relations (e.g., *friend*, *spouse*, and *teammate*).

Description concatenation. To obtain a complete sentence, we need to concatenate descriptions of entities and relations in the path. A natural language pattern makes the sentence closer to human expressions. An example w.r.t triplet $(Sarah, Profession, Actor)$ is shown as follows:

> *Question:*
>
> ***Sarah Michelle Gellar** is the **person profession** of what?*
>
> *Is the correct answer **Actor**?*

However, our preliminary results show that formulating complete sentences does not yield better performance than simply combining entities and relations together using semicolons as follows:

> *Sarah Michelle Gellar; person profession; Actor*

This is because the latter approach better preserves the sequential order and relative positions in the paths in PLMs.

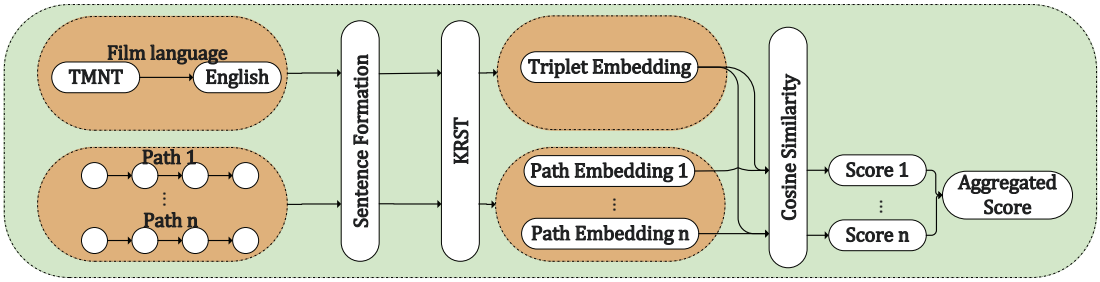


Figure 2: Overall architecture of relation prediction in KRST.

Therefore, we choose to use semicolons for description concatenation. For example, the corresponding sentence of Path (5) is formulated as follows:

Sarah Michelle Gellar; person gender; Female; gender person; Alyson Hannigan; person profession; Actor

KRST Prediction

After sentence formation, KRST is able to generate embeddings and make relation predictions. Figure 2 shows the overall model architecture.

For each positive query triplet, multiple reliable logical reasoning paths are extracted correspondingly. After being formatted into sentences, triplets and the corresponding paths have similar semantics, which are measured by the cosine similarity:

$$cos(s_1, s_2) = \frac{s_1 \cdot s_2}{||s_1||_2 \cdot ||s_2||_2}, \tag{13}$$

where s_1 and s_2 are the corresponding sentence embeddings.

KRST extracts at most $|M|$ paths per triplet and converts paths and triplets into embeddings. The similarity score between each path and the corresponding triplet is computed and the path with the highest score is deemed as most reasonable for relation prediction. Therefore, we use the highest score among all paths as the score for each triplet:

$$score(h, r, t) = \max_{p \in P} \{cos(s((h, r, t)), s(p))\}, \tag{14}$$

where P denotes the corresponding path set w.r.t triplet (h, r, t), and $s(\cdot)$ denotes the embedding function (i.e., KRST) for triplets and paths.

During training, negative triplets are processed with the label of -1. As aforementioned, commonly applied binary loss functions (e.g., cross-entropy and hinge) require the negative scores to be close to -1 (or 0). This is not appropriate in our scenario because the unmatched pair of triplet and path are not necessarily perpendicular to each other. To relax the penalty for loss, we use the cosine embedding loss function:

$$\mathscr{L}(s_1, s_2, y) = \begin{cases} 1 - cos(s_1, s_2), & y = 1, \\ \max(0, cos(s_1, s_2) - margin), & y = -1, \end{cases} \tag{15}$$

where $margin \in (-1, 1)$ and $y \in \{1, -1\}$ denotes the label. Equation 15 makes the positive score to be close to 1, maximizing similar semantics. On the other hand, the negative score is only constrained to be smaller than *margin*.

Experiments

We evaluate the performance of KRST in three different settings: transductive, inductive, and few-shot. Then, we demonstrate multi-aspect comprehensive explanations by clustering the embeddings of paths generated by KRST. We implement KRST[1] with a SOTA sentence transformer (all-mpnet-base-v2[2]) on a Tesla V100 GPU with 16GB RAM.

Following the evaluation tasks conducted in (Teru, Denis, and Hamilton 2020) and (Zha, Chen, and Yan 2022), in this paper, we measure the rank and hit rate of one positive triplet among 49 negative triplets. We only randomly generate negative triplets and use them for training and validation. For a fair comparison, we use the negative triplets provided by (Zha, Chen, and Yan 2022) for testing.

Datasets

To evaluate the transductive and inductive performance of KRST, we use all three datasets adopted in (Zha, Chen, and Yan 2022), which were introduced by (Teru, Denis, and Hamilton 2020)[3]. These datasets are commonly adopted by various inductive approaches and they are the respective subsets of WN18RR, FB15k-237, and NELL-995. In the inductive setting, training entities have no overlap with testing entities.

For few-shot evaluation, we use the corresponding few-shot transductive and few-shot inductive datasets given in (Zha, Chen, and Yan 2022).

Transductive and Inductive Relation Prediction

In transductive cases, we extract paths in the training graph for all training, validation, and testing triplets. However, in inductive cases, paths for testing triplets are not available from the training graph, because entities used for testing do not appear in the training graph (see Definition 2). Instead, we use the inductive graphs given in (Teru, Denis, and Hamilton 2020) for path extraction.

We benchmark the performance of KRST against SOTA inductive methods, SOTA embedding-based methods, and SOTA reinforcement learning methods. Table 1 shows the results of transductive and inductive relation prediction. Compared with SOTA methods, KRST methods achieve the

[1] github.com/ZhixiangSu/KRST

[2] huggingface.co/sentence-transformers/all-mpnet-base-v2

[3] github.com/kkteru/grail

		Transductive			Inductive		
		WN18RR	FB15k-237	NELL-995	WN18RR	FB15k-237	NELL-995
MRR	RuleN	0.669	0.674	0.736	0.780	0.462	0.710
	GRAIL	0.676	0.597	0.727	0.799	0.469	0.675
	MINERVA	0.656	0.572	0.592	-	-	-
	TuckER	0.646	0.682	0.800	-	-	-
	KG-BERT	-	-	-	0.547	0.500	0.419
	BERTRL	0.683	0.695	0.781	0.792	0.605	**0.808**
	KRST (No filter)	0.881	0.671	0.730	0.883	0.713	0.753
	KRST (Coverage)	0.897	0.709	**0.803**	**0.902**	0.704	0.696
	KRST (Confidence)	**0.899**	**0.720**	0.800	0.890	**0.716**	0.769
Hit@1	RuleN	0.646	0.603	0.636	0.745	0.415	0.638
	GRAIL	0.644	0.494	0.615	0.769	0.390	0.554
	MINERVA	0.632	0.534	0.553	-	-	-
	TuckER	0.600	0.615	**0.729**	-	-	-
	KG-BERT	-	-	-	0.436	0.341	0.244
	BERTRL	0.655	0.620	0.686	0.755	0.541	**0.715**
	KRST (No filter)	0.807	0.576	0.618	0.803	**0.602**	0.633
	KRST (Coverage)	0.831	0.624	0.692	**0.835**	0.573	0.554
	KRST (Confidence)	**0.835**	**0.639**	0.694	0.809	0.600	0.649

Table 1: Transductive and inductive relation prediction results

		Transductive						Inductive					
		WN18RR		FB15k-237		NELL-995		WN18RR		FB15k-237		NELL-995	
		1000	2000	1000	2000	1000	2000	1000	2000	1000	2000	1000	2000
MRR	RuleN	0.567	0.625	0.434	0.577	0.453	0.609	0.681	0.773	0.236	0.383	0.334	0.495
	GRAIL	0.588	0.673	0.375	0.453	0.292	0.436	0.652	0.799	0.380	0.432	0.458	0.462
	MINERVA	0.125	0.268	0.198	0.364	0.182	0.322	-	-	-	-	-	-
	TuckER	0.258	0.448	0.457	0.601	0.436	0.577	-	-	-	-	-	-
	KG-BERT	-	-	-	-	-	-	0.471	0.525	0.431	0.460	0.406	0.406
	BERTRL	0.662	0.673	0.618	0.667	0.648	0.693	0.765	0.777	0.526	0.565	0.736	**0.744**
	KRST (Confidence)	**0.871**	**0.882**	**0.696**	**0.701**	**0.743**	**0.781**	**0.886**	**0.878**	**0.679**	**0.680**	**0.745**	0.738
Hit@1	RuleN	0.548	0.605	0.374	0.508	0.365	0.501	0.649	0.737	0.207	0.344	0.282	0.418
	GRAIL	0.489	0.633	0.267	0.352	0.198	0.342	0.516	0.769	0.273	0.351	0.295	0.298
	MINERVA	0.106	0.248	0.170	0.324	0.152	0.284	-	-	-	-	-	-
	TuckER	0.230	0.415	0.407	0.529	0.392	0.520	-	-	-	-	-	-
	KG-BERT	-	-	-	-	-	-	0.364	0.404	0.288	0.317	0.236	0.236
	BERTRL	0.621	0.637	0.517	0.583	0.526	0.582	0.713	0.731	0.441	0.493	0.622	0.628
	KRST (Confidence)	**0.790**	**0.810**	**0.611**	**0.602**	**0.628**	**0.678**	**0.811**	**0.793**	**0.537**	**0.524**	**0.637**	**0.629**

Table 2: Few-shot transductive and inductive relation prediction results

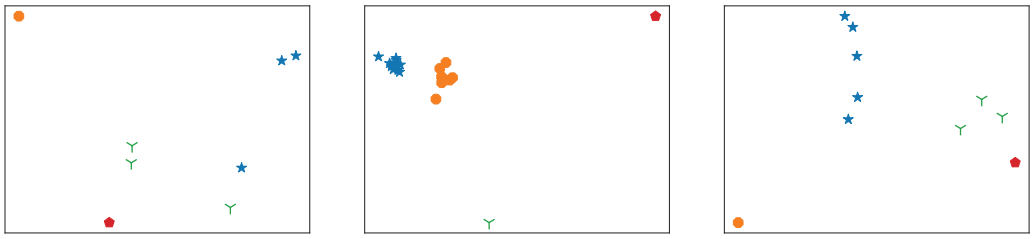
best performance under *MRR* (5 of 6) and *Hit@1* (4 of 6) metrics. Specifically, KRST methods achieve significant improvement in the transductive case of WN18RR (+0.216 for *MRR* and +18.0% for *Hit@1*), inductive case of WN18RR (+0.103 for *MRR* and +6.6% for *Hit@1*) and inductive case of FB15k-237 (+0.111 for *MRR* and +5.9% for *Hit@1*).

Among KRST methods, the majority of the best results are achieved by KRST with relation path confidence (8 of 12). Only in 1 of 12 cases, KRST with no filter performs the best. So we empirically show that filtering (especially relation path confidence) elevates model performance. As for the relatively inferior performance of relation path coverage, this is because long paths usually lead to exponentially increasing numbers of reachable entities, making the relation path coverage prefer shorter paths. However, short paths are not necessarily reliable. Therefore, the performance is limited by unreliable paths unfiltered by relation path coverage.

Few-shot Relation Prediction

In the few-shot settings, wherein only subsets of the entire datasets are given for training, we conduct similar path extractions as done for the entire datasets. Specifically, for transductive cases, training, validation, and testing paths are all extracted from the entire training graph. For inductive cases, paths for inductive training are extracted from the entire training graph, while paths for inductive validation and inductive testing are both extracted from the inductive graph.

Because KRST with relation path confidence achieves the best performance on the entire datasets, we apply it for all few-shot settings. As shown in Table 2, KRST with relation path confidence outperforms SOTA methods in 11 of 12 cases. The average transductive and inductive improvement for *MRR* and *Hit@1* is 0.119 and 0.082 (+3.1% and +5.0%), respectively. In addition, the performance gap between few-



(a) Clusters for triplet (Atlantic City, Film Genre, Drama).

(b) Clusters for triplet (California, Location Contain, Carlsbad).

(c) Clusters for triplet (Kelsey, Person Language, English).

Figure 3: Clustering result for a multi-aspect explanation.

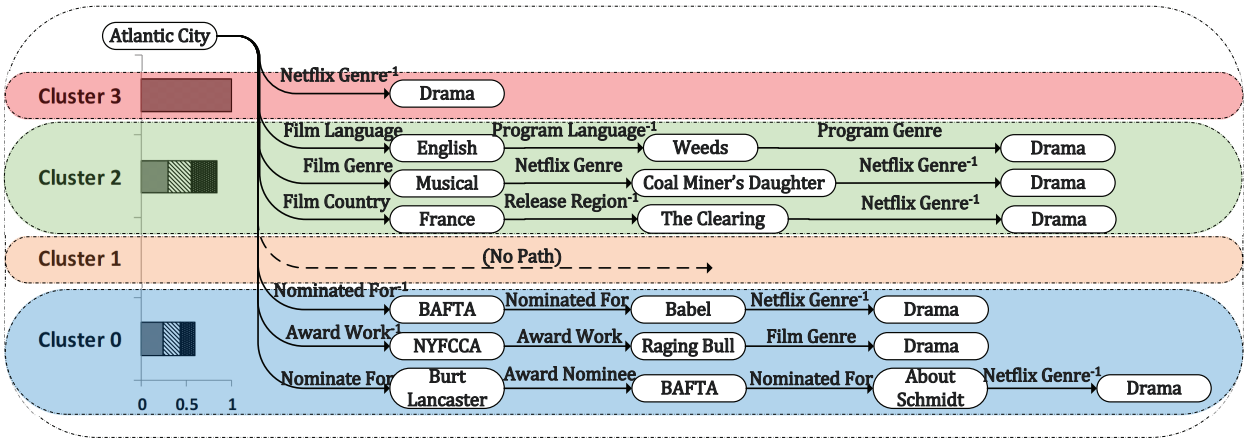


Figure 4: Paths clustered by similarity scores for a multi-aspect explanation of triplet (Atlantic City, Film Genre, Drama).

shot-1000 and few-shot-2000 samples is small (smaller than 0.015 for *MRR* in 5 of 6 cases). These results illustrate the strong generalization capability of KRST, which requires a lesser amount of samples to achieve on-par performance.

Multi-Aspect Explanation

As afore-introduced, KRST is able to provide a multi-aspect explanation of the relation prediction results. This is because KRST generates an embedding for each path, allowing us to quantifiably analyze the relations among paths. Although paths are supposed to be similar to the given query triplet, they are not necessarily similar to each other. By grouping them into different clusters, we could provide a multi-aspect comprehensive explanation.

Figure 3 visualizes the clustering results on the reduced dimensions after applying Linear Discriminant Analysis (LDA) for three different triplets. Clusters are generated using the K-Means algorithm. As shown in Figures 3(a) and 4, KRST successfully provides a multi-aspect explanation for triplet $(AtlanticCity, FilmGenre, Drama)$ based on the similarity score (pre-processed after min-max scaling). The paths grouped into each cluster are shown in Figure 4 and presented using the same color in Figure 3(a). As shown, the only path in Cluster 3 illustrates the explanation provided by the external knowledge from Netflix that *AtlanticCity* is a *Drama* available on Netflix. This explanation is straightforward and convincing, and is considered to be the most reliable (score of 1.0). For paths in Clusters 2, explanations are provided using similar films with the same attributes (e.g., language and country), which is similar to the human analogical reasoning. Cluster 0 explains the target triplet using the knowledge of awards won or nominated regarding *AtlanticCity* in three paths, which is a distinctive piece of side information. KRST considers Clusters 2 and 0 as relatively less convincing and assigns relatively lower scores (0.837 and 0.593 on average, respectively). We also input an empty path for comparison, which individually constitutes Cluster 1. KRST correctly gives it the lowest similarity score. In summary, based on Figure 4, we can easily perceive different explanations in the corresponding aspects of Netflix platform, similar films, and awards.

Conclusion

In this paper, we propose a novel architecture named KRST which outperforms SOTA models in most transductive and inductive relation prediction tasks (15 of 18). In addition, we perform clustering on KRST generated embeddings and provide a comprehensive multi-aspect explanation. Nonetheless, KRST is a model based on BERT, which requires relatively large memory usage and computational resources. Going forward, we plan to solve this computational intensive issue by proposing a more parameter-efficient model.

Acknowledgements

This research is supported, in part, by the Joint SDU-NTU Centre for Artificial Intelligence Research (C-FAIR), Shandong University, China and by the Joint NTU-WeBank Research Centre on Fintech (Award No: NWJ-2020-010), Nanyang Technological University, Singapore. This research is also supported, in part, by the National Research Foundation, Prime Minister's Office, Singapore under its NRF Investigatorship Programme (NRFI Award No. NRF-NRFI05-2019-0002). Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not reflect the views of National Research Foundation, Singapore. This work is also supported, in part, by the National Key R&D Program of China No.2021YFF0900800; NSFC No.91846205; Shandong Provincial Key Research and Development Program (Major Scientific and Technological Innovation Project) (NO.2021CXGC010108).

References

Balaevi, I.; Allen, C.; and Hospedales, T. M. 2019. Tucker: Tensor factorization for knowledge graph completion. *arXiv preprint arXiv:1901.09590*.

Bordes, A.; Usunier, N.; Garcia-Duran, A.; Weston, J.; and Yakhnenko, O. 2013. Translating embeddings for modeling multi-relational data. In *Proceedings of Advances in Neural Information Processing Systems*, 27872795.

Chen, Z.; Wang, Y.; Zhao, B.; Cheng, J.; Zhao, X.; and Duan, Z. 2020. Knowledge graph completion: A review. *IEEE Access*, 8: 192435–192456.

Das, R.; Dhuliawala, S.; Zaheer, M.; Vilnis, L.; Durugkar, I.; Krishnamurthy, A.; Smola, A.; and McCallum, A. 2017. Go for a walk and arrive at the answer: Reasoning over paths in knowledge bases using reinforcement learning. *arXiv preprint arXiv:1711.05851*.

Dettmers, T.; Minervini, P.; Stenetorp, P.; and Riedel, S. 2018. Convolutional 2d knowledge graph embeddings. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 1811–1819.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Galárraga, L. A.; Teflioudi, C.; Hose, K.; and Suchanek, F. 2013. AMIE: association rule mining under incomplete evidence in ontological knowledge bases. In *Proceedings of the International Conference on World Wide Web*, 413–422.

Horn, A. 1951. On sentences which are true of direct unions of algebras1. *The Journal of Symbolic Logic*, 14–21.

Ji, G.; He, S.; Xu, L.; Liu, K.; and Zhao, J. 2015. Knowledge graph embedding via dynamic mapping matrix. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics and the International Joint Conference on Natural Language Processing*, 687–696.

Ji, S.; Pan, S.; Cambria, E.; Marttinen, P.; and Philip, S. Y. 2021. A survey on knowledge graphs: Representation, acquisition, and applications. *IEEE Transactions on Neural Networks and Learning Systems*, 494–514.

Lao, N.; Mitchell, T.; and Cohen, W. 2011. Random walk inference and learning in a large scale knowledge base. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 529–539.

Lin, Y.; Liu, Z.; Sun, M.; Liu, Y.; and Zhu, X. 2015. Learning entity and relation embeddings for knowledge graph completion. In *Proceedings of AAAI Conference on Artificial Intelligence*, 2181–2187.

Radford, A.; Narasimhan, K.; Salimans, T.; and Sutskever, I. 2018. Improving language understanding by generative pre-training. https://openai.com/blog/language-unsupervised/. Accessed 2023-03-17.

Reimers, N.; and Gurevych, I. 2019. Sentence-BERT: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

Schlichtkrull, M.; Kipf, T. N.; Bloem, P.; Berg, R. v. d.; Titov, I.; and Welling, M. 2018. Modeling relational data with graph convolutional networks. In *Proceedings of European Semantic Web Conference*, 593–607.

Teru, K.; Denis, E.; and Hamilton, W. 2020. Inductive relation prediction by subgraph reasoning. In *Proceedings of International Conference on Machine Learning*, 9448–9457.

Trouillon, T.; Dance, C. R.; Welbl, J.; Riedel, S.; Gaussier, .; and Bouchard, G. 2017. Knowledge graph completion via complex tensor factorization. *arXiv preprint arXiv:1702.06879*.

Vashishth, S.; Sanyal, S.; Nitin, V.; and Talukdar, P. 2019. Composition-based multi-relational graph convolutional networks. *arXiv preprint arXiv:1911.03082*.

Wang, Z.; Zhang, J.; Feng, J.; and Chen, Z. 2014. Knowledge graph embedding by translating on hyperplanes. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 1112–1119.

Xiong, W.; Hoang, T.; and Wang, W. Y. 2017. DeepPath: A reinforcement learning method for knowledge graph reasoning. *arXiv preprint arXiv:1707.06690*.

Yang, Z.; Dai, Z.; Yang, Y.; Carbonell, J.; Salakhutdinov, R. R.; and Le, Q. V. 2019. XLNet: Generalized autoregressive pretraining for language understanding. In *Proceedings of Advances in Neural Information Processing Systems*, 57535763.

Yao, L.; Mao, C.; and Luo, Y. 2019. KG-BERT: BERT for knowledge graph completion. *arXiv preprint arXiv:1909.03193*.

Zha, H.; Chen, Z.; and Yan, X. 2022. Inductive relation prediction by BERT. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 5923–5931.