Pre-Trained Embeddings for Enhancing Multi-Hop Reasoning

Martin Drancé[1] , **Fleur Mougin**[1] , **Akka Zemmari**[2] and **Gayo Diallo**[1]
[1]Univ. Bordeaux, Inserm, BPH, U1219, Team AHeaD, F-33000 Bordeaux, France
[2]Univ. Bordeaux, CNRS, LaBRI, UMR 5800, F-33400, Talence, France
{first.last}@u-bordeaux.fr

Abstract

Knowledge graphs are an efficient way to represent heterogeneous data from multiple sources or disciplines by utilizing nodes and their relations. Nevertheless, they are frequently incomplete in terms of the subject they represent. Link prediction methods are used to discover additional links (or even to create new ones) between entities present in the Knowledge Graph (KG). In order to achieve this, multi-hop reasoning models have demonstrated good predictive performance and the ability to generate interpretable decisions, thereby enabling their application in high-stakes domains such as finance and public health. A multi-hop reasoning model usually has two tasks: 1) construct an accurate representation of the entities and relationships of the KG; 2) use these representations to explore the reasoning paths in the KG that support the newly predicted links. In this paper, we investigate how the performance of a multi-hop reasoning model changes when using pre-trained embeddings for the KG's nodes and relations. The experiments conducted on three benchmark datasets, respectively WN18RR, NELL-995 and FB15K-237, suggest that using pre-trained embeddings improves: (i) the predictive performance of multi-hop reasoning models for all three datasets, (ii) the number of newly predicted links, and (iii) the quality of paths used as explanations.

1 Introduction

A knowledge graph (KG) is a large data structure that use a graph database to describe real-world entities and their relationships. KGs represent entities and their relationships in the form of a schema, allowing entities to be linked together, and can aggregate information from various data sources and domains [Paulheim, 2017]. The facts stored in a KG are in the form of triples, *i.e.* two nodes connected by a direct and typed link, such as **(Tim Berners-Lee →wonAward →Turing Award)**. As KGs are highly incomplete [Min *et al.*, 2013], link prediction methods can be used to infer missing links between pairs of nodes, creating and updating the representation of entities and relations as embeddings.

Following the TransE [Bordes *et al.*, 2013] effort, knowledge graph embeddings (KGE) models capable of learning good representations (or embeddings) of entities and relations in a KG have been proposed. These embeddings can then be used to infer new links by evaluating the plausibility of new triples. Although they have achieved state-of-the-art results for link prediction, these methods are not transparent in nature and their use is problematic when a clear understanding of a prediction is required.

Recently, multi-hop reasoning models (referred as MHR or walk-based methods) [Das *et al.*, 2018; Liu *et al.*, 2021; Lin *et al.*, 2018; Lei *et al.*, 2020; Safavi *et al.*, 2020] have proposed to address this issue by searching for reasoning paths in KGs using reinforcement learning (RL) to learn a policy, which is then used to predict an answer to a query $(entity_1, relation_1, ?)$. As introduced in [Das *et al.*, 2018], MHR models have the strong advantage of producing directly interpretable predictions. Indeed, the paths used during inference are used to understand what are the existing links between the query and the predicted entity. Interpretability in machine learning is a growing challenge for high-stakes decisions [Rudin *et al.*, 2022] and in the context of link prediction, transparency is a desirable attribute as KGs are particularly good at storing highly symbolic concepts and relationships.

Despite the encouraging results of MHR models recently, they still struggle with the sparsity and large size of KGs. Since there is no notion of a correct path to reach a correct answer, link prediction can be performed by the agent using an incorrect path but still reach a valid entity in the graph. Also, due to the incompleteness of KGs, true predictions that are missing in the training data won't be rewarded. In both cases, these predictions will result in the alteration of entities and relations representations based on the use of false information.

In this paper, we investigate and report how using pre-trained embeddings for nodes and relations can improve the performances of a multi-hop reasoning model and discuss our findings. This improvement is analyzed in terms of link prediction scores as well as the quality of the paths generated as explanations. The use of high-quality pre-trained embeddings is intended to facilitate the training process of MHR models by bringing prior knowledge in the KG representations. In contrast to previous work and state-of-the-art models, we exclude from the MHR model the requirement

to learn entity and relationship representations from scratch. These representations are first learned by the KGE method ConvE [Dettmers *et al.*, 2018] and then used as is or fine-tuned by the MHR model. We conduct our experiments on the three benchmark datasets WN18RR [Dettmers *et al.*, 2018], NELL-995 [Das *et al.*, 2018] and FB15K-237 [Toutanova *et al.*, 2015]. The main contributions of our work are the following: (i) we show experimentally that pre-trained embeddings for entities and relations improve the performance of MHR model for three benchmark datasets WN18RR, NELL-995 and FB15K-237; (ii) we show that the use of pre-trained embeddings allows the model to find at least one reasoning path for more queries, and (iii) pre-trained embeddings increases the overall quality of the reasoning paths used as explanations.

The next section outlines related work in the area of MHR models for link prediction and KG embedding methods. Then, in section 3, we describe our proposed approach to improving multi-hop reasoning[1]. In section 4, we report its performance evaluation before discussing and interpreting it in section 5.

2 Background

In this section, we first define the properties of KGs and justify their use. Then, we outline how the main KGE models work. Finally, we outline related work in the area of multi-hop reasoning.

2.1 Knowledge Graphs

Given a set of entities $\boldsymbol{\mathcal{E}}$ and a set of binary relations $\boldsymbol{\mathcal{R}}$, a KG $\boldsymbol{\mathcal{G}} \subset \boldsymbol{\mathcal{E}} \times \boldsymbol{\mathcal{R}} \times \boldsymbol{\mathcal{E}}$ represents the collection of facts expressed as triples $\boldsymbol{(h, r, t)}$ with $\boldsymbol{h, t} \in \boldsymbol{\mathcal{E}}$ and $\boldsymbol{r} \in \boldsymbol{\mathcal{R}}$. All relations $\boldsymbol{r}$ are directed and define the role of entities in each triple, with the left node $\boldsymbol{h}$ being the head (or source) and the right node $\boldsymbol{t}$ being the tail (or target) of the triple. All entities $\boldsymbol{e}$ and relations $\boldsymbol{r}$ belong to specific categories or types, clearly defining what type of triples can be found in the KG. For example, if the KG defines that the relation $worksFor$ should link a node of type $person$ to a node of type $organization$, it can then be inferred from $(Tim, worksFor, OxfordUniversity)$ that Tim is a person and $OxfordUniversity$ is an organization.

KGs are highly flexible and can be used to represent general knowledge, such as that in Wikidata [van Veen, 2019], or more domain-specific data, such as Hetionet [Himmelstein *et al.*, 2017] which stores biomedical data or FinKG [Cheng *et al.*, 2020] which stores financial data. In 2013, Min *et al.* pointed out that KGs are mostly incomplete, with some of the entities in the graph lacking crucial information [Min *et al.*, 2013]. For example, in the Freebase KG [Bollacker *et al.*, 2008], 93.8% of the $Person$ nodes are not linked to a node of type $PlaceOfBirth$ and 78.5% of them are not linked to a node of type $Nationality$. From this observation, two main tasks emerged to discover the missing knowledge in a KG: (i) KG completion, which aims to retrieve the missing actual facts in the KG, and (ii) link prediction, which aims to discover the unknown links between the entities. These tasks have been used for specific uses cases, such as drug repurposing [Drancé *et al.*, 2021; Edwards *et al.*, 2021] or in recommendation systems for social networks [Wang *et al.*, 2015].

[1]Code to reproduce the experiments is available at https://gitub.u-bordeaux.fr/mdrance/pre-trained-embeddings-for-enhancing-multi-hop-reasoning

2.2 Knowledge Graph Embedding

Knowledge graph embedding methods aim to construct latent vector representations for each entity and relationship in the KG. Given these embeddings, each KGE model defines its own scoring function $\boldsymbol{f}$ to evaluate the correctness of a given triple $\boldsymbol{(h, r, t)}$. KGE models are trained to distinguish between true and false triples. Given that, every triple belonging to the KG is considered true and false triples are constructed by corrupting either the head or the tail of an existing triple $\boldsymbol{(h, r, t)} \rightarrow \boldsymbol{(h', r, t')}$. For each triple in the graph, the model learns to give a better score to the true triple than to its corresponding negatives. KGE models can be divided into two different categories [Ali *et al.*, 2021a], defined by how the model uses embeddings to score each triple.

Translational Distance Interaction Models. These models use the notion of distance between embeddings to compute the plausibility of a triple. For example, TransE [Bordes *et al.*, 2013] uses the relation embedding as a translation from head to tail embeddings. This translation $\boldsymbol{e_h + e_r \approx e_t}$ is defined by the scoring function as $\boldsymbol{f(h, r, t) = - \parallel e_h + e_r - e_t \parallel_p}$, with $\boldsymbol{p \in \{1, 2\}}$ the $\boldsymbol{l_p}$ norm applied to the scoring function. The simplicity of TransE makes it an efficient KGE model to work with on large KGs, but its scoring function makes it impossible to model 1:N, N:1 and N:N relations.

Following TransE, more robust models were designed, such as RotatE [Sun *et al.*, 2019], aiming at addressing the limitations of TransE. They allow in particular to model symmetry, anti-symmetry, inversion and composition using embeddings laying in complex space $\mathbb{C}$, modeling relations as rotations from the head to the tail: $\boldsymbol{e_t = e_h \odot e_r}$ with $\boldsymbol{e_h, e_r, e_t} \in \boldsymbol{\mathbb{C}^d}$ and $\odot$ being the Hadamard (or element-wise) product. Thus, the scoring function is defined as $\boldsymbol{f(h, r, t) = - \parallel e_h \odot e_r - e_t \parallel}$.

Semantic Similarity Matching Models. These models exploit the similarity of the latent features to compute the plausibility of a triple. RESCAL [Nickel *et al.*, 2011] and DistMult [Yang *et al.*, 2014] use vectors to model entities and matrices to model relations. The goal of the relation matrices $\boldsymbol{W_r} \in \mathbb{R}^{d \times d}$ is to learn the weights $\boldsymbol{w_{i,j}}$ that quantify the interaction between head entities $\boldsymbol{h} \in \mathbb{R}^d$ and tail entities $\boldsymbol{t} \in \mathbb{R}^d$. Hence, the RESCAL scoring function is given by $\boldsymbol{f(h, r, t) = h^T W_r t = \sum_{i=1}^{d} \sum_{j=1}^{d} w_{ij}^{(r)} h_i t_j}$. The sole difference between RESCAL and DistMult is that the latter reduces the scoring function by utilizing only diagonal relation matrices: $\boldsymbol{f(h, r, t) = h^T W_r t = \sum_{i=1}^{d} h_i \cdot}$

$diag(W_r)_i \cdot t_i$.
As with RotatE, ComplEx [Trouillon *et al.*, 2016] improves DistMult by learning representations for entities and relations in $\mathbb{C}$. The scoring function is defined as $f(h, r, t) = Re(e_h \odot e_r \odot e_t)$ where $Re(e_t)$ is the real component of the complex valued vectors representing t.
Finally, ConvE [Dettmers *et al.*, 2018] makes use of a convolutional neural network layer to learn interactions between h and r. ConvE first takes the head and relation of each triple in the batch and concatenates them to create a matrix $B \in \mathbb{R}^{m \times n}$ where the first half $m/2$ rows represent all the h and the second half $m/2$ rows represent their corresponding r. Convolutional filters are then applied on B to capture interactions between h and r. These interactions are reshaped in order to obtain a feature vector v, that is then mapped to the entity space using a linear transformation W, finally creating conjoint representations of each pair of (h, r): $e_{h,r} = v^T W$. This representation of the first half of the triple is then scored against all the potential tails using the scoring function $f(h, r, t) = e_{h,r} \cdot e_t$.

To the best of our knowledge, KGE models are the most efficient methods for the link prediction task. State-of-the-art results are obtained by using these models or improving existing KGE models [Lu *et al.*, 2022; Zhang *et al.*, 2019; Chen *et al.*, 2021; Pan and Wang, 2021]. The main issue is that embedding-based methods are not interpretable, although they can handle uncertainty and noise in the data. This is because each embedding represents a combination of latent factors, encoding the meaning of the entity/relation [Bianchi *et al.*, 2020].

2.3 Related Work

MHR models are neuro-symbolic approaches in that they combine learning the vector representation of entities and relations with reasoning properties and interpretabilty provided to the user via the reasoning paths exploited for each prediction. MHR aims to create a new direct link between two entities using reasoning paths supporting the new direct link. To do so, a policy network (typically a fully-connected neural network) is trained to find the next best action for the agent given its current state. The sequence of successive actions then forms the reasoning path explaining the prediction. DeepPath [Xiong *et al.*, 2017] is the first attempt to model the path finding problem in a KG using the Monte-Carlo Policy Gradient algorithm, REINFORCE [Williams, 1992]. The particularity of DeepPath is that the source and target entities must be known, *i.e.* DeepPath learns to find correct reasoning paths between the two entities (e_{source}, e_{target}), but not to predict a new link between unconnected nodes in the graph. MINERVA [Das *et al.*, 2018] is the first MHR model to actually address the link prediction problem using reinforcement learning. Indeed, instead of focusing on finding true paths between two entities, MINERVA trains its policy to reach a correct answer node given a query $(e_{source}, r_{query}, ?)$, by traversing the best sequence of relations and entities supporting the choice of the predicted node. PoLo [Liu *et al.*, 2021] is the first approach that attempts to decrease the effect of a noisy reward signal received by the agent when spurious paths are used but lead to a correct prediction. Since there exist many paths to connect two nodes, some of them may not be valid to support the final prediction. However, in MINERVA the reward is only binary $\{0, 1\}$, which allows the agent to use false paths for valid predictions, *i.e.* paths that are meaningless regarding the new predicted link. Based on the implementation of MINERVA and to help guide the agent and remove noise from the reward signal, PoLo uses a set of known and efficient logical rules as a reward shaping mechanism: if a prediction uses one of these correct rules, the reward is increased. MultiHopKG [Lin *et al.*, 2018] takes it further and proposes two modeling advances for MHR methods. First, the authors tackle the problem of missing true facts in the KG by adopting a new reward shaping mechanism. Because KGs are incomplete [Min *et al.*, 2013], the agent may arrive at a correct answer that is not present in the training data, and thus not receive a reward for that prediction. Instead of using a binary $\{0, 1\}$ reward, a trained KGE model is used to estimate a soft reward for predictions not existing in the training KG. For each new triple (h, r, t) predicted by the MHR model, if the triple is not present in the training KG, the KGE model scoring function $f(h, r, t)$ is used to evaluate the plausibility of the prediction and shape the reward accordingly. Second, they enhance the agent's exploration ability by adding an action dropout for each training step. As REINFORCE is a policy-based learning algorithm, the agent may be inclined to use spurious but rewarding paths encountered earlier in the training phase. Action dropout randomly disconnects some outgoing edges at each step, forcing the exploration of more diverse paths. RuleGuider [Lei *et al.*, 2020] proposes another strategy to improve MultiHopKG's reward shaping mechanism using high-quality logical rules, also dissociating the walk-based agent between a relation agent and an entity agent. As with PoLo, a symbolic method first mines the logical rules, which are then used to modify the agent's reward based on the confidence of the type of logical rule used to make the prediction. In addition, the relation agent will now select the type of outgoing relation to choose for the next step, then the entity agent will select the best node to move to, based on the starting node and the relation selected by the relation agent. This method significantly prunes the search space when selecting the next action.

Dataset	#Nodes	#Relations	#Facts	Degree
WN18RR	40,945	11	86,835	2
NELL-995	75,492	200	154,213	1
FB15K-237	14,505	237	272,115	14

Table 1: Characteristics of the KGs used in our experiments. Degree corresponds to the median degree of nodes in the KG.

3 Approach of Exploiting Pre-trained Embeddings

The current work proposes to analyze how pre-trained embeddings for entities and relations can be useful to enhance the performance of the link prediction task with MHR models. We tested this new approach on the state of the art model MultiHopKG, introduced in subsection 2.3, which we

call hereafter PT-MultiHopKG. We describe here how MHR methods, and in particular MultiHopKG, model the problem of link prediction on a KG, followed by the details of our contribution based on the use of pre-trained embeddings.

3.1 Problem Definition

Environment and States. The KG $\mathcal{G}$ represents the environment, such that $\mathcal{G} \subset \mathcal{E} \times \mathcal{R} \times \mathcal{E}$ where $\mathcal{E}$ represents the set of all entities in $\mathcal{G}$ and $\mathcal{R}$ represents the set of all relations in $\mathcal{G}$. The link prediction task consists in finding the set of all possible answers $e_o \in E_o$ given a query $(e_s, r_q, ?)$, where e_s is the source node and r_q is the query relation, such that each (e_s, r_q, e_o) is a missing triple in $\mathcal{G}$. These answer nodes are selected after the agent moves successively from one node to another until it reaches a plausible target. The state s_t of the agent at step t should encode the source entity, the query relation and the location of the agent e_t at step t. Thus, the current state s_t is defined by $s_t = (e_s, r_q, e_t) \in S$.

Action Space. The action space at step t comprises all outgoing edges from the current location e_t: $A_t = \{(r_{t'}, e_{t'}) | e_s, r_q, e_t\}$ with $r_{t'} \in \mathcal{R}$ and $e_{t'} \in \mathcal{E}$. At each step, the action selection is made by selecting an outgoing edge, knowing the type of $r_{t'}$ and the next node $e_{t'}$. For each starting node e_s, the search is limited to a certain number of steps T. To allow the agent to stay at its current position if it reaches a plausible answer at step $t < T$, an additional action "NO_OP" is added, which corresponds to a self-loop on the current node e_t.

Rewards. The base reward function defines a reward of 1 if the agent reaches a correct target entity and 0 otherwise: $R = \mathbb{1}\{(e_s, r_q, e_o) \in \mathcal{G}\}$. For MultiHopKG, the reward is shaped using the scoring function $f(e_s, r_q, e_o)$ of a KGE model. The idea is to give a reward of 1 if the triple $(e_s, r_q, e_o) \in \mathcal{G}$, otherwise the reward is only defined by the KGE scoring function: $R' = R + (1 - R) f(e_s, r_q, e_o)$.

Policy Network. As defined in [Das *et al.*, 2018], the policy network makes use of three pieces of information to choose the appropriate action: the agent's current position e_t, the query relation r_q and the history of all previous actions of the agent. Each entity and relation in $\mathcal{G}$ is respectively assigned an embedding $e \in \mathbb{R}^d$ and $r \in \mathbb{R}^d$, all actions A_t at step t are represented as $a_t = [r_{t'}; e_{t'}]$ where $[;]$ is the vector concatenation of the chosen relation and its corresponding destination node. The history $h_t \in \mathbb{R}^{2d}$, encoded using an LSTM [Hochreiter and Schmidhuber, 1997], represents the sequence of past observations and actions performed up to step t and is defined as follows:

$$h_0 = LSTM(0, [r_0; e_s]) \quad (1)$$

$$h_t = LSTM(h_{t-1}, a_{t-1}) \quad (2)$$

where equation (1) is used at step t_0 to encode the "starting action" in the history with r_0 a special "starting relation", and equation (2) is used otherwise. Based on this history h_t, the current state s_t and the query relation r_q, the policy network π is defined as a two-layers feed-forward network that outputs the probability distributions of all possible actions A_t at step t:

$$\pi_\theta(a_t | s_t) = \sigma(A_t \times W_2 ReLU(W_1 [h_t; e_t; r_q])) \quad (3)$$

with σ being the softmax function.
Equations (2) and (3) are repeated for each transition step, until the maximum number of steps T is reached. The learnable parameters are the LSTM parameters, the feed-forward network parameters W_1, W_2 and the entity and relation embeddings e, r.

3.2 Training

To train the policy network and find the best parameters θ, the REINFORCE algorithm is used to maximize the expected reward $\mathbb{E}$ as follows:

$$J(\theta) = \mathbb{E}_{(e_s, r_q, e_o) \in \mathcal{G}} [\mathbb{E}_{a_1, a_2, \ldots, a_T \sim \pi_\theta} [R(S_T | e_s, r_q)]] \quad (4)$$

using the following stochastic gradient:

$$\nabla_\theta J(\theta) \approx \nabla_\theta \sum_t R(s_T | e_s, r_q) log \pi_\theta(a_t | s_t) \quad (5)$$

During training, MultiHopKG proposes to add an action dropout mechanism to randomly hide some outgoing edges in the sampling step of REINFORCE. The goal is to encourage the agent to search for more diverse paths, as exploration in equation (5) can be biased towards spurious paths leading to correct answers.

3.3 Pre-trained Embeddings

The purpose of MHR models can be divided into two distinct parts:

- The learning of the entity and relation embeddings e and r as latent vector representations, describing the information carried by each node and each relation in the KG. This task is also performed in the KGE models.
- The learning of the LSTM parameters in equation (2) and the feed-forward parameters W_1 and W_2 in equation (3), given the information of the agent history h_t, the current node e_t and the query relation r_q. This task corresponds specifically to the walk-based method.

During the training process, node and relation representations e and r are updated using equations (4) and (5), which are dependent on the rewards obtained at steps T. This training procedure implies that, in some cases, the representations will be updated using false information. First, when a false path is used to reach a good target node, the nodes and relations embeddings are updated using a positive reward based on a path that is completely illogical. Secondly, for correct predictions not present in the training data due to the incompleteness of the KGs, the representations will be updated using a wrong reward signal caused by a false negative.

In this work, we propose to use pre-trained embeddings for

Method / Dataset	WN18RR			NELL-995			FB15K-237		
	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10	MRR
MultiHopKG (ConvE)	41.4	51.7	44.8	65.6	84.4	72.7	32.7	56.4	40.7
MultiHopKG (ComplEx)	42.5	52.6	46.1	64.4	81.6	71.2	**32.9**	54.4	39.3
PT-MultiHopKG$^+$	43.0	52.4	46.0	67.3	84.6	74.0	32.7	**58.1**	**41.2**
PT-MultiHopKG$^-$	**44.1**	**52.8**	**46.8**	**68.1**	**85.6**	**74.9**	31.1	57.2	39.9

Table 2: Performance comparison on MultiHopKG with and without pre-trained embeddings. PT-MultiHopKG$^+$ corresponds to trainable pre-trained embeddings, PT-MultiHopKG$^-$ corresponds to frozen pre-trained embeddings. Best results in bold.

	Unique Paths		Diversity		Path Recall	
Datasets	MultiHopKG	PT-MultiHopKG	MultiHopKG	PT-MultiHopKG	MultiHopKG	PT-MultiHopKG
WN18RR	**1918**	1790	2.76	**2.78**	0.62	**0.63**
NELL-995	29,091	**29,396**	2.94	**2.96**	0.66	**0.67**
FB15K-237	82,426	**82,992**	2.94	**2.95**	0.74	**0.76**

Table 3: Path analysis. The number of unique paths used as explanations, the diversity of these paths and the path recall on the test set are reported.

entities and relations in the KG, generated by the KGE model ConvE. Consequently, the e, r embeddings in equations (1), (3), (4) and (5) are substituted with the corresponding pre-trained representations e^+, r^+ corresponding to the pre-trained learnable parameters, or e^-, r^- corresponding to the pre-trained non-learnable parameters. The goal of this approach is to mitigate the impact of false information on entities and relations embeddings during the training process.

4 Experiments

4.1 Experimental Setup

Datasets. To compare our approach to other MHR methods, we evaluated its performance on the following three benchmark datasets: WN18RR [Dettmers *et al.*, 2018], Nell-995 [Das *et al.*, 2018] and FB15K-237 [Toutanova *et al.*, 2015]. Their characteristics are given in Table 1. For each existing triple (h, r, t) in $\mathcal{G}$, the inverse triple (t, r^{-1}, h) is added to allow bidirectional movement of the agent. At each step, the maximum number of outgoing edges is limited to a number n, to avoid GPU memory overflow. The top-n neighbors are selected using their PageRank scores [Page *et al.*, 1999].

Hyperparameters. As previously indicated, we used the pre-trained embeddings generated by ConvE as defined in [Lin *et al.*, 2018]. We keep the size of all embeddings at 200. We performed a grid search on embedding dropout rate [0, 0.5], feed-forward layers [0, 0.5], action dropout rate [0.1, 0.9] and learning rate [0.001, 0.003], respectively. We conducted the experiment using pre-trained embeddings as learnable parameters e^+ or having them frozen e^-.

KGE model. Experiments were conducted using the embeddings generated by the KGE model used for reward shaping, resulting in no computational overhead. Tests carried out with the KGE model implemented using the Pykeen librairy [Ali *et al.*, 2021b] showed exactly the same results as with the KGE model used for reward shaping. The process of training and testing the KGE model is made on exactly the same data splits used for training and testing the MHR model.

4.2 Path Analysis

A key feature of MHR models is their ability to provide reasoning paths to explain each prediction. MHR methods are expected to be more reliable than KGE methods, as the reason for each triple prediction can be easily understood. However, it has been shown that these reasoning paths are often unreasonable, *i.e.* paths that do not make sense but lead to a correct answer, or incomplete [Lv *et al.*, 2021]. We chose to compare how the addition of pre-trained embeddings on MultiHopKG changes the reasoning paths supporting each prediction. We measured: 1) the number of unique paths used as explanations, 2) the diversity of these paths, and 3) the number of test triples for which the model found an answer.

Unique Paths. MHR models often give more than one reasoning path to support each prediction, which leads to many explanations that decrease the interpretability of the results. Given two different queries, the same reasoning paths are often found for both predictions because, even though the entities present in the paths are different, the relations used are the same. The goal here is to abstract the reasoning paths into logical rules using only the relations, and measure how many unique logical rules are used for the explanations. For each path $p = (h, r, t) \leftarrow (h, r_1, e_1) \wedge (e_1, r_2, e_2) \wedge (e_2, r_3, t)$, the corresponding logical rule is $l = (r_1 \wedge r_2 \wedge r_3)$. We then calculate the number of unique logical rules used on the set of predictions, which represents the model's ability to generalize the paths encountered during training onto the entities in the test set.

Rule Diversity. The number of unique logical rules provides a measure of the variety of explanations, but does not take into account the variety of each reasoning path. Some of

Method / Dataset	WN18RR	NELL-995	FB15K-237
MultiHopKG -RS	46.2	72.2	32.4
MultiHopKG +RS	44.8 (-3%)	72.7 (+0.5%)	40.7 (+25%)
PT-MultiHopKG$^+$ -RS	48.1 (+4%)	73.3 (+1.5%)	36.2 (+12%)
PT-MultiHopKG$^-$ -RS	49.0 (+6%)	71.4(-1%)	35.2 (+9%)

Table 4: Performance comparison (MRR) of the reward shaping and the impact of pre-trained embeddings on the model. MultiHopKG -RS corresponds to the model without reward shaping, MultiHopKG +RS corresponds to the model with ConvE reward shaping, Ours PT-MultiHopKG$^+$ -RS and Ours PT-MultiHopKG$^-$ -RS correspond respectively to trainable pre-trained embeddings and frozen pre-trained embeddings without reward shaping. The percentage of improvement for each method is provided in parentheses.

them are highly redundant because they use the same type of relation for more than one step. Diversity in the number of relation types employed by each logical rule results in more informative explanations, as it corresponds to rules with a greater variety of node types and semantically distinct relations. We quantify diversity as follows:

$$\boldsymbol{d} = \frac{\sum_l \boldsymbol{Unique(l)}}{|\boldsymbol{l}|} \tag{6}$$

where $\boldsymbol{Unique}$ represents the number of unique relations $\boldsymbol{r} \in \boldsymbol{\mathcal{R}}$ found in the logical rules $\boldsymbol{l}$.

Path Recall. We use the path recall score as defined in [Lv *et al.*, 2021] to quantify the number of triples in the test set that can be retrieved by the model. A significant path recall indicates that the model accurately predict and explain a greater proportion of the test triples. The path recall is defined as follows:

$$\boldsymbol{PR} = \frac{\sum_{(h,r,t)\in T^{test}} \boldsymbol{Cnt(h,r,t)}}{|\boldsymbol{T^{test}}|} \tag{7}$$

where $\boldsymbol{Cnt(h,r,t)} = \mathbf{1}$ if the model finds at least one path from $\boldsymbol{h}$ to $\boldsymbol{t}$, and $\mathbf{0}$ otherwise.

5 Results and Discussion

5.1 Model Comparison

Table 2 reports the performance comparison between MultiHopKG and PT-MultiHopKG. For all three datasets, the addition of pre-trained embeddings increases the predictive performance of the model. For NELL-995 and WN18RR, the use of frozen embeddings provides the best results, indicating that the ConvE-constructed representations are most efficient than using embeddings as model parameters. For NELL-995, using pre-trained embeddings increases the performances on both setup. For FB15K-237, the best performance is obtained by fine-tuning the embeddings. FB15K-237 is a more complex KG than both WN18RR and NELL-995 due to the number of triples it contains. Overall, although the two other datasets have more unique nodes, FB15K-237 contains more facts and each entity is much more connected to its neighbors (median degree of 14), making it a more complex KG to work with.

Table 3 shows the results of path analysis produced for MultiHopKG with and without pre-trained embeddings. Except for WN18RR, the addition of pre-trained embeddings increases the number of unique paths that serve as explanations for the model. This suggests that the improvement in performance is not solely attributable to the model's ability to predict more facts using the same rules, but rather to the model's use of different rules. The pre-trained embeddings improve path diversity and recall across all datasets. First, this indicates that the relationships between the logical rules employed are generally more diverse, *i.e.* the model is able to reuse a larger number of different logical paths seen during training when predicting new links. Second, the path recall values indicate that the model is able to identify at least one path for a greater number of query triples in the test dataset. In the case of FB15K-237, for instance, our model was able to provide a prediction and a reasoning path for 409 additional test queries.

5.2 Ablation Study and Model Variations

Using ComplEx. We conducted the same tests than previously, with the exception that we replaced ConvE with ComplEx to generate the pre-trained embeddings. We used the best configuration for each dataset: frozen pre-trained embeddings for WN18RR and NELL-995 and learnable pre-trained embeddings for FB15K-237. Our results indicate that using ConvE is the best option and yields the best results.

Reward Shaping. The purpose of using pre-trained embeddings is to have prior information about each node and relation in the form of its embedding. Reward shaping mechanisms are also a way to use prior or external knowledge to assist the MHR model during training. As pointed out in [Lin *et al.*, 2018], reward shaping improves performance on FB15K-237 and NELL-995, but decreases it for WN18RR. We compared the effectiveness of the two approaches by excluding the reward-shaping mechanism, as their goals are identical. Table 4 shows the results obtained for MultiHopKG with and without reward shaping mechanism using or not using pre-trained embeddings. First, for all three datasets, the use of pre-trained embeddings increases the performance of the base model, highlighting that these embeddings fulfil their role in providing prior knowledge during the MHR training process. Secondly, reward shaping has a stronger impact on FB15K-237 but a negative impact on WN18RR, where the use of trainable pre-trained embeddings still has a positive impact on the MRR.

A summary of all results and a comparison with other symbolic and MHR methods can be found in Table 5.

Method / Dataset	WN18RR			NELL-995			FB15K-237		
	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10	MRR
AnyBURL†	42.9	53.7	-	44.0	57.0	-	26.9	52.0	-
MINERVA†	41.3	51.3	44.8	66.3	83.1	72.5	21.7	45.6	29.3
MultiHopKG (ConvE)	41.4	51.7	44.8	65.6	84.4	72.7	32.7	56.4	40.7
MultiHopKG (ComplEx)	42.5	52.6	46.1	64.4	81.6	71.2	**32.9**	54.4	39.3
RuleGuider (ConvE)†	42.2	53.6	46.0	66.0	85.1	73.1	31.6	57.4	40.8
RuleGuider (ComplEx)†	44.3	55.5	48.0	66.4	**85.9**	73.6	31.3	56.4	39.5
PT-MultiHopKG$^+$	43.0	52.4	46.0	67.3	84.6	74.0	32.7	**58.1**	**41.2**
PT-MultiHopKG$^-$	44.1	52.8	46.8	**68.1**	85.6	**74.9**	31.1	57.2	39.9
PT-MultiHopKG$^+$ -RS	44.8	54.4	48.1	66.8	83.5	73.3	28.5	51.6	36.2
PT-MultiHopKG$^-$ -RS	**45.5**	**55.8**	**49.0**	63.9	84.6	71.4	27.2	51.2	35.2

Table 5: Summary of our results and comparison with other symbolic and MHR models. Best scores are in bold. † Results taken from [Lei *et al.*, 2020]

5.3 Discussion

Pre-training has been widely studied in the past few years as a way to improve the predictive performance of a model. From image classification, where model pre-training is used to enhance feature detection [Russakovsky *et al.*, 2015], to natural language processing, where pre-trained word embeddings are fine-tuned to match a specific domain [Lee *et al.*, 2019], pre-training allows for prior knowledge to be used or for the model to be tailored to a specific task. We believe that the experimental evaluation results presented in subsection 5.1 suggest that pre-trained embeddings have a positive impact on MultiHopKG. As in other areas of machine learning, model parameters need to be fine-tuned when applied to more complex tasks. Consequently, while embeddings can be used as is for the WN18RR and NELL-995 datasets, they must be included in the model parameters for the FB15k-237 dataset, which is significantly more complex.

Since the value of MHR models lies in their interpretability, the analysis of reasoning paths has shown the interest of using pre-trained embeddings regarding explanations. Possessing more query triples for which a prediction can be made is a significant characteristic, but having more interpretable reasoning routes is even more crucial. Each prediction can be supported by hundreds of alternative reasoning paths, however these paths are frequently redundant in the type of relation they employ, traversing via various nodes while employing the exact same type of relation. Diverse reasoning paths enable more sophisticated and exhaustive explanations. Explanations can be more complex and exhaustive when there are a variety of reasoning paths.

Compared to reward shaping, subsection 5.2 showed that pre-trained embeddings consistently improve the model performance, while reward shaping hurt performance for the WN18RR dataset. For FB15K-237, reward shaping has a greater impact than pre-trained embeddings, confirming the complexity of the link prediction task on this dataset, as it benefits the most from the addition of prior knowledge, but the best results were obtained using both methods simultaneously. Finally, when testing with the KGE model ComplEx to build the embeddings, we observed inferior results to those obtained with ConvE. It is important to note that, when it comes to link prediction, ConvE always produces better results than ComplEx, as shown in [Ali *et al.*, 2021a].

6 Conclusion

The current study investigates the impact of pre-trained embeddings on the state-of-the-art model MultiHopKG for the link prediction task. We demonstrated that adding pre-trained embeddings improves the performance of such model on the three benchmark datasets WN18RR, NELL-995 and FB15K-237. In addition to improving the results of the link prediction task, the results demonstrate that the pre-trained embeddings enable the model to predict new links for a larger number of queries using a larger number of unique and more diverse paths. This suggests that the method is ideally suited for use cases in which explanations are required, such as biomedical KGs. An ablation study shows that pre-trained embeddings are a valid method for providing prior knowledge during the MHR model training process. Furthermore, these results suggest that KGE methods are appropriate for building these pre-trained embeddings. In future work, we plan to investigate the use of other methods to construct these pre-trained embeddings and validate the lessons learned from this study.

Ethical Statement

This work does not rely on sensitive personal data that could raise any ethical concern. However, the authors are fully aware that a great attention should be paid to ensure positive ethical and societal consequences of machine learning-based technologies.

References

Mehdi Ali, Max Berrendorf, Charles Hoyt, Laurent Vermue, Mikhail Galkin, Sahand Sharifzadeh, Asja Fischer, Volker Tresp, and Jens Lehmann. Bringing light into the dark: A large-scale evaluation of knowledge graph embedding models under a unified framework. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PP:1–1, 11 2021.

Mehdi Ali, Max Berrendorf, Charles Tapley Hoyt, Laurent Vermue, Sahand Sharifzadeh, Volker Tresp, and Jens

Lehmann. PyKEEN 1.0: A Python Library for Training and Evaluating Knowledge Graph Embeddings. *Journal of Machine Learning Research*, 22(82):1–6, 2021.

Federico Bianchi, Gaetano Rossiello, Luca Costabello, Matteo Palmonari, and Pasquale Minervini. Knowledge graph embeddings and explainable AI. *ArXiv*, abs/2004.14843, 2020.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. Freebase: A collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, SIGMOD '08, page 1247–1250, New York, NY, USA, 2008. Association for Computing Machinery.

Antoine Bordes, Nicolas Usunier, Alberto García-Durán, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. In *Advances in Neural Information Processing Systems 26*, 2013.

Yihong Chen, Pasquale Minervini, Sebastian Riedel, and Pontus Stenetorp. Relation prediction as an auxiliary training objective for improving multi-relational graph representations. In *Proceedings of the International Conference on Automated Knowledge Base Construction*, 10 2021.

Dawei Cheng, Fangzhou Yang, Xiaoyang Wang, Ying Zhang, and Liqing Zhang 0001. Knowledge graph-based event embedding framework for financial quantitative investments. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*, pages 2221–2230. ACM, 2020.

Rajarshi Das, Shehzaad Dhuliawala, Manzil Zaheer, Luke Vilnis, Ishan Durugkar, Akshay Krishnamurthy, Alex Smola, and Andrew McCallum. Go for a walk and arrive at the answer: Reasoning over paths in knowledge bases using reinforcement learning. In *Proceedings of the International Conference on Learning Representations*, 2018.

Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. Convolutional 2d knowledge graph embeddings. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence*. AAAI Press, 2018.

Martin Drancé, Marina Boudin, Fleur Mougin, and Gayo Diallo. Neuro-symbolic XAI for computational drug repurposing. In *Proceedings of the 13th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management*, pages 220–225. INSTICC, SciTePress, 2021.

Gavin Edwards, Sebastian Nilsson, Benedek Rozemberczki, and Eliseo Papa. Explainable biomedical recommendations via reinforcement learning reasoning on knowledge graphs. *arXiv e-prints*, pages arXiv–2111, 2021.

Daniel Scott Himmelstein, Antoine Lizee, Christine Hessler, Leo Brueggeman, Sabrina L. Chen, Dexter Hadley, Ari Green, Pouya Khankhanian, and Sergio E. Baranzini. Systematic integration of biomedical knowledge prioritizes drugs for repurposing. *eLife*, 6:e26726, Sep 2017.

Sepp Hochreiter and Jürgen Schmidhuber. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780, 11 1997.

Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. BioBERT: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36(4):1234–1240, 09 2019.

Deren Lei, Gangrong Jiang, Xiaotao Gu, Yuning Mao, and Xiang Ren. Learning collaborative agents with rule guidance for knowledge graph reasoning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, pages 8541–8547. Association for Computational Linguistics, 01 2020.

Xi Victoria Lin, Richard Socher, and Caiming Xiong. Multi-hop knowledge graph reasoning with reward shaping. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3243–3253. Association for Computational Linguistics, 10-11 2018.

Yushan Liu, Marcel Hildebrandt, Mitchell Joblin, Martin Ringsquandl, Rime Raissouni, and Volker Tresp. Neural multi-hop reasoning with logical rules on biomedical knowledge graphs. In *Eighteenth Extended Semantic Web Conference - Research Track*, 2021.

Haonan Lu, Hailin Hu, and Xiaodong Lin. DensE: An enhanced non-commutative representation for knowledge graph embedding with adaptive semantic hierarchy. *Neurocomputing*, 476:115–125, 2022.

Xin Lv, Yixin Cao, Lei Hou, Juanzi Li, Zhiyuan Liu, Yichi Zhang, and Zelin Dai. Is multi-hop reasoning really explainable? Towards benchmarking reasoning interpretability. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8899–8911. Association for Computational Linguistics, 11 2021.

Bonan Min, Ralph Grishman, Li Wan, Chang Wang, and David Gondek. Distant supervision for relation extraction with an incomplete knowledge base. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 777–782. Association for Computational Linguistics, 6 2013.

Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. A three-way model for collective learning on multi-relational data. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, page 809–816. Omnipress, 2011.

Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The PageRank Citation Ranking: Bringing order to the web. Technical Report 1999-66, Stanford InfoLab, 11 1999.

Zhe Pan and Peng Wang. Hyperbolic hierarchy-aware knowledge graph embedding for link prediction. In *Findings of the Association for Computational Linguistics: EMNLP*

2021, pages 2941–2948. Association for Computational Linguistics, 11 2021.

Heiko Paulheim. Knowledge graph refinement: A survey of approaches and evaluation methods. *Semantic Web*, 8:489–508, 2017.

Cynthia Rudin, Chaofan Chen, Zhi Chen, Haiyang Huang, Lesia Semenova, and Chudi Zhong. Interpretable machine learning: Fundamental principles and 10 grand challenges. *Statistics Surveys*, 16(none):1 – 85, 2022.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015.

Tara Safavi, Danai Koutra, and Edgar Meij. Evaluating the Calibration of Knowledge Graph Embeddings for Trustworthy Link Prediction. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, pages 8308–8321. Association for Computational Linguistics, 11 2020.

Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. RotatE: Knowledge graph embedding by relational rotation in complex space. In *Proceedings of the International Conference on Learning Representations*, 2019.

Kristina Toutanova, Danqi Chen, Patrick Pantel, Hoifung Poon, Pallavi Choudhury, and Michael Gamon. Representing text for joint embedding of text and knowledge bases. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1499–1509. Association for Computational Linguistics, 9 2015.

Théo Trouillon, Johannes Welbl, Sebastian Riedel, Eric Gaussier, and Guillaume Bouchard. Complex embeddings for simple link prediction. In *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 2071–2080. PMLR, 6 2016.

Theo van Veen. Wikidata. *Information Technology and Libraries*, 38(2):72–81, Jun. 2019.

Peng Wang, Baowen Xu, Yurong Wu, and Xiaoyu Zhou. Link prediction in social networks: the state-of-the-art. *Science China Information Sciences*, 58(1):1–38, 2015.

Ronald J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8(3–4):229–256, 5 1992.

Wenhan Xiong, Thien Hoang, and William Yang Wang. DeepPath: A reinforcement learning method for knowledge graph reasoning. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 564–573. Association for Computational Linguistics, 9 2017.

Bishan Yang, Wen tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. Embedding entities and relations for learning and inference in knowledge bases. *CoRR*, abs/1412.6575, 2014.

Shuai Zhang, Yi Tay, Lina Yao, and Qi Liu. Quaternion knowledge graph embeddings. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*. Curran Associates Inc., 12 2019.