FRUNI and FTREE synthetic knowledge graphs for evaluating explainability

Pablo Sánchez-Martín[1], **Tarek Besold**[1] **& Priyadarshini Kumari**[2]
[1]Sony AI, Barcelona, Spain
[2]Sony AI, Cupertino, USA
{pablo.sanchez2,tarek.besold,priyadarshini.kumari}@sony.com

Abstract

Research on knowledge graph completion (KGC)—i.e., link prediction within incomplete KGs—is witnessing significant growth in popularity. Recently, KGC using KG embedding (KGE) models, primarily based on complex architectures (e.g., transformers), have achieved remarkable performance. Still, extracting the *minimal and relevant* information employed by KGE models to make predictions, while constituting a major part of *explaining the predictions*, remains a challenge. While there exists a growing literature on explainers for trained KGE models, systematically exposing and quantifying their failure cases poses even greater challenges. In this work, we introduce two synthetic datasets, FRUNI and FTREE, designed to demonstrate the (in)ability of explainer methods to spot link predictions that rely on indirectly connected links. Notably, we empower practitioners to control various aspects of the datasets, such as noise levels and dataset size, enabling them to assess the performance of explainability methods across diverse scenarios. Through our experiments, we assess the performance of four recent explainers in providing accurate explanations for predictions on the proposed datasets. We believe that these datasets are valuable resources for further validating explainability methods within the knowledge graph community.

1 Introduction

Knowledge Graphs (KGs) have achieved significant attention in the research community due to their ability to facilitate efficient knowledge retrieval and reasoning [16]. A KG is typically defined as a set of triples, $\{(\texttt{head-entity}, \texttt{relation}, \texttt{tail-entity})\}$, where each triple represents a fact, for instance, $(\texttt{Insulin}, \texttt{regulates}, \texttt{blood-sugar})$. In this representation, entities correspond to nodes, and relations correspond to directed edges. KGs have been used in a wide variety of applications like recommendation systems [23, 39], question answering systems [34, 1], natural language processing tasks [18, 26, 10], and drug discovery[22, 17]. In this context, KG Completion (KGC) is a key research focus, involving the prediction and classification of missing facts in incomplete KGs. Recent advancements are linked to the emergence of KG embedding (KGE) models using powerful neural networks[5, 32], and achieving remarkable performance. However, their increasing complexity poses a challenge: the need to understand the KG information they rely on for predictions. That is, there is a growing interest to *explain* the predictions generated by these models [9]. Explainability is critical not only for understanding the model's decision-making process but also for ensuring trust, fairness, and accountability in real-world applications [9].

Existing work on explainers tailored for KGC [27] mainly focuses on extracting the most relevant facts that have the largest influence on a prediction. However, they still have several limitations, such as high computational requirements (making them impractical for large-scale KGs); con-

XAI in Action: Past, Present, and Future Applications @ NeurIPS 2023.

strained search spaces (limiting their ability to uncover complex explanations); and usage of heuristic methodologies (which may not always guarantee robust and reliable explanations).

The complexity of KGE models and KG datasets poses challenges for explainer methods and complicates the analysis of their strengths and weaknesses. Quantitatively evaluating the effectiveness of explainers and identifying their failure-cases is even more challenging; e.g., explainers may struggle with predictions reliant on indirectly connected KG links. Synthetic datasets stand out as invaluable assets in this regard, offering a controlled data generation process. This facilitates a meticulous assessment of explainers' performance across diverse scenarios, often unattainable with real-world datasets due to their inherent noise and complexity.

This paper introduces two synthetic datasets, FRUNI and FTREE, which are crafted to manifest the capabilities (or limitations) of explainer methods in scenarios where predictions depend on *indirectly connected links*. These datasets have several tweakable parameters such as noise levels and the number of entities. This flexibility allows for an evaluation of explainability methods under various conditions. Our experiments focus on assessing the performance of the latest explainers using these synthetic datasets. We aim to provide insights into the strengths and weaknesses of current explainability techniques in the context of KGC.

In summary, these are our contributions. (i) We propose two new synthetic datasets, FRUNI and FTREE, engineered to assess the efficacy of explainers in correctly making predictions that rely on indirectly connected links. (ii) We evaluate the performance of the latest explainers in providing explanations for predictions on the introduced datasets. (iii) We open-source the datasets with the goal of fostering further research in validating explainability methods in KG tasks.

2 Background

A **knowledge graph (KG)** is a structured representation of information consisting of entities and relationships typically denoted as triples or facts (h, r, t), where h is the head entity, r is the relationship, and t is the tail entity. Mathematically, a KG is defined as a set of triples $\mathcal{T} = \{(h, r, t)\}$, where h and t belong to the entity set $\mathcal{E}$, and r belongs to the relationship set $\mathcal{R}$. **Knowledge graph embedding (KGE)** is the task of learning embeddings for entities $h \in \mathcal{E}$ and $r \in \mathcal{R}$ relationships, denoted as $\boldsymbol{e}_h$ for entity h and $\boldsymbol{r}_r$ for relationship r, where $\boldsymbol{e}_h, \boldsymbol{r}_r \in \mathbb{R}^d$, with d being the embedding dimension. **Knowledge graph completion (KGC)** is the task of predicting missing or incomplete triples in a KG. Given a KG $\mathcal{T}$ and a partial triple $(?, r, t)$, $(h, ?, t)$ or $(h, r, ?)$, the goal is to predict the missing entity/relationship. KGC aims to find:

$$h' = \arg\max_{h \in \mathcal{E}} P(h|r, t), \qquad r' = \arg\max_{r \in \mathcal{R}} P(r|h, t), \qquad t' = \arg\max_{t \in \mathcal{E}} P(t|h, r), \tag{1}$$

where $P(h|r, t)$, $P(r|h, t)$, and$P(t|h, r)$ are conditional probability distributions. In essence, KGC focuses on head, relation, or tail prediction.

Interpretability or explainability in KGC refers to providing insights into why a particular prediction was made by a KGE model. It involves identifying the minimal and relevant information from the KG that contributes to a prediction, which can be formulated as a subset of triples (h, r, t) that collectively affect the prediction. Rossi et al. [27] introduces the concepts of necessary and sufficient explanations for a prediction. A necessary explanation is a subset of triples from the KG that, when removed, causes the prediction to change; and a sufficient explanation is a subset of triples that, when added to the KG, ensures the correctness of the prediction.

3 Related work

In this section, we review the relevant literature on explainable knowledge graph completion.

Knowledge Graph Embeddings (KGE) models have been successfully used in KGC tasks. They are categorized into four groups: translation-based [3, 11]; tensor decomposition-based [36, 31]; Graph Neural Networks (GNNs) based [28, 24, 32]; and Transformers based [37, 18, 5]. Generally, more complex models achieve better performance but are less interpretable. For detailed overviews of KGE models, see [12, 33].

Knowledge Graph Reasoning has also been used to tackle KGC. These methods typically integrate logical rules or leverage path information within the KG to facilitate informed predictions. Techniques include random walks [14, 8], variational inference [4], and reinforcement-learning-based path search [35, 6, 42]. While more interpretable, these methods often suffer from higher computational complexity and reduced performance. Recently, Zhu et al. [43] focused on enhancing their scalability. For a comprehensive overview of KGR approaches, consult [16].

Post-hoc explainability methods in KGC, which aim to shed light on how black-box KGC models generate link predictions, have recently gained attention [25]. Here, we focus on local explanations, i.e., explaining the prediction of a single KG triple. Many of these approaches originate from GNN literature, including methods generating attention maps [41], highlighting the KG triples that had the most significant impact on a particular prediction; and those finding (soft) masks on the edges and nodes [38, 29, 20]. For an exhaustive survey, see Longa et al. [19] or Li et al. [15]. However, few works offer general-purpose KGE explainability methods. Some use influential functions [13, 25], data perturbation [40], or adversarial attacks [2]. Notably, Rossi et al. [27] recently proposed `Kelpie`. Although it represents the current state-of-the-art, it still faces challenges like high computational complexity and limited search space.

To evaluate Explainable AI approaches for KGC, the selection of an appropriate **dataset** is of great importance. Previous research mostly used established KG datasets like FB15k [3], WN18 [3], FB15k-237 [30], WN18RR [7], and YAGO3 [21]. However, the inherent complexity of real-world KGs complicates the identification of failure cases in the explanations. Synthetic datasets can play a critical role in this regard, as they can be carefully designed to control various facets of the KG. Unfortunately, they are scarce. To the best of our knowledge, the `UserItemAttr` dataset, introduced by Zhang et al. [41], is the sole synthetic KG available. It emulates user purchase decisions based on item attributes. Nonetheless, we need more synthetic datasets that can simulate different scenarios and foster robust research within the KGC community.

4 Proposed synthetic datasets

In this section, we describe the proposed synthetic datasets designed for the evaluation of explainability methods in the context of KGC. We would like to highlight that while our focus is on KGC, these datasets can find utility in other tasks such as entity prediction. Each of the datasets can be represented as a set of entities $\mathcal{E}$, relations $\mathcal{R}$, and triples $\mathcal{T}$. Alternative, they can be described as a $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, i.e., as a set of nodes $\mathcal{V}$ an edges $\mathcal{E}$.

For every dataset proposed, we offer a semantic example to illustrate the meaning of its entities and relations. Furthermore, we provide a mathematical formulation of the generation process in the form of an algorithm in App. A. Additionally, we describe the hyperparameters that practitioners have the flexibility to manipulate. The code for both generating and visualizing these datasets can be accessed at `https://github.com/SonyResearch/synthetic_knowledge_graphs`.

4.1 FRUNI dataset

The "Friends and Universities" (FRUNI) dataset comprises N_u university entities, categorized into two types: `uniF` and `uniI`. Two illustrative knowledge graph examples, each centered around one of these university types, are depicted in Fig. 1. In this dataset, every university is associated with exactly two `student`, also represented as entities. Consequently, each university is linked to two triples of the form $(\texttt{uni}, \texttt{enrolls}, \texttt{student})$. Furthermore, each student has $K \sim \text{Poisson}(\lambda_\text{f}) + 1$ `friend` entities, leading to triples of the form $(\texttt{student}, \texttt{friendOf}, \texttt{friend})$.

Importantly, we know that N_f universities are of the type `uniF`, which actively foster friendships among students' friends. In other words, every friend of a student becomes friends with all the friends of the other student from the same university. This arrangement generates triples of the form $(\texttt{friendA}, \texttt{friendOf}, \texttt{friendB})$. It is important to clarify that `friendA` and `friendB` cannot both be friends of the same student. The remaining $N_u - N_f$ university entities, denoted as `uniI`, impose isolation among the friends of the students, thus no facts exist among them.

The rationale behind this design choice is to require a two-hop reasoning process to determine whether two friends are connected. First, we need to confirm that they are friends of different students within the same university. Second, we must determine whether the

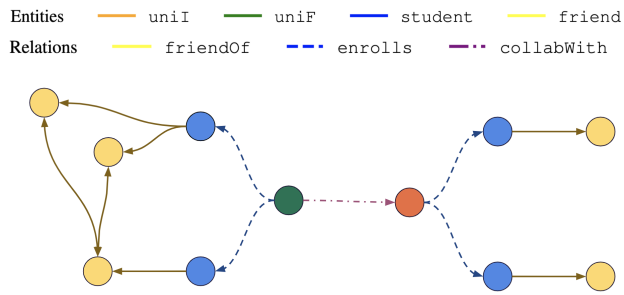

Figure 1: Par of the FRUNI KG containing the facts from a uniF and a uniI university entities.

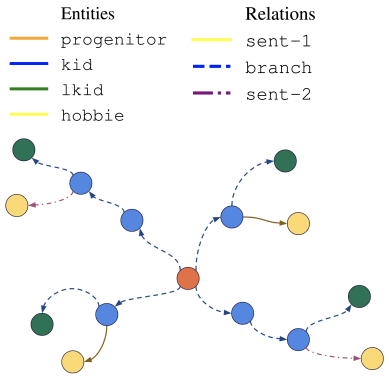

Figure 2: Example of a family with the root entity progenitor.

university promotes friendship or isolation. Consequently, an explanation for the triples (friendA, friendOf, friendB) must include information about the university to which the friends are connected through their respective students.

Furthermore, universities in this dataset have the capability to collaborate, with this collaboration occurring randomly with probability α_u. It is important to note that such collaborations result in triples like (uni1, collabWith, uni2) but do not impact the remainder of the knowledge graph, they just introduce noise. For a formal description of the generative process, please refer to Alg. 1 in App. A. In the following, we will elaborate on the elements of this synthetic knowledge graph.

Entities. There are four distinct types of entities: universities fostering friendship (uniF) or imposing isolation (uniI), students (student), and friends (friend).

Relations. We define three types of relations: 1) friendOf: represents the friendship relationship between students and friends. This relation can only appear in facts with head is student or friend and tail entity is friend; 2) collabWith: denotes collaborations between universities. This relation exclusively appears in facts where the head/tail entities are of type uni; 3) enrolls: signifies the association between students and universities. This relation is found in facts with the head entity of type uniF or uniI and the tail entity of type student.

Hyperparameters. The dataset incorporates four essential hyperparameters, each governing distinct aspects of the dataset: N_u is the total number of university entities; λ_f is the mean of the Poisson distribution that models the number of friends of each student; α_u is the probability of collaboration between universities, which determines the expected number of triples containing the collabWith relation, expressed as $\alpha_u \times N_u^2$; and, N_f is the number of universities that foster friendship, resulting in triples between friend entities of students.

It is worth noting that certain facts depend solely on the entities, such as (uni, collabWith, uni), while the existence of triples like (friend, friendOf, friend) is dependant upon the type of university to which the friends are affiliated. A robust explainer should capture these distinctions.

4.2 FTREE dataset

The family tree (FTREE) dataset comprises N_t isjoint family trees. An illustrative example of a family tree is depicted in Fig. 2, where progenitor represents the root entity of the tree structure. The progenitor entity has $B \sim \text{Poisson}(\lambda_b) + 2$ offspring, each of which corresponds to an entity kid. These entities form triples of the form (progenitor, ancestorOf, kid).

In this dataset, every lineage, defined as a distinct branch within a family tree, spans a depth of $L \sim \text{Uni}(\{1, \ldots, N_d\})$ generations, with each generation yielding a single offspring. For instance, if $N_d = 2$, some lineages originating from progenitor contain information up to the level of children, while others extend to the grandchildren, as exemplified in Fig. 2. Regardless of lineage depth, all triples arising from these lineages consistently exhibit the ancestorOf relation.

Table 1: Statistics of the datasets.

	Entities	Relations	Facts
FRUNI	5000	3	4974
FTREE	524	3	519

Table 2: Performance in terms of MRR of the trained KGE models.

	ConvE	ComplEx	TransE
FRUNI	1.0	1.0	1.0
FTREE	1.0	1.0	1.0

For the last generation (the final node in the branch), we are aware of two crucial facts: they possess a last-kid and they have a hobbie, e.g., tennis. The nature of their relationship with hobbie varies according to $L \in \{1, \ldots, N_d\}$ and can encompass sentiments such as liking, disliking, and so on. We shall denote these relationships as $\{\texttt{sent-1}, \ldots, \texttt{sent-N}_d\}$. For example, Fig. 2 shows the case for $N_d = 2$ leading to sent-1 and sent-2 relations. It is important to note that this relationship depends on the length of the path traced back to the root entity. Therefore, knowledge of the path length is indispensable for accurately predicting the relationship for the triple containing the hobbie entity (which is a leaf node in the tree). In App. A, we provide a formal description of the generative process in Alg. 2. In the following, we describe the key components of this synthetic KG.

Entities. There exist four distinct types of entities. 1) The original couple, denoted as progenitor, always serves as the root entity within each family tree. 2) Descendants, represented as kid, function as internal nodes in the tree structure. 3) The last descendant, designated as last-kid, 4) and hobbies, indicated as hobbie, both act as leaf nodes within the trees.

Relations. We establish $1 + N_d$ distinct types of relations within the dataset. ancestorOf signifies a parent-to-child relationship. This relation manifests from progenitor to kid, from kid to another kid, or from kid to last-kid. On the other hand, sent-i, for $i \in \{1, \ldots, N_d\}$ indicates that a kid exhibits a particular sentiment (dependant on the depth) towards a hobbie.

Hyperparameters. The FTREE dataset incorporates three hyperparameters. N_t represents the total count of family trees. λ_b represents the mean parameter of the Poisson distribution used to model the number of children for each progenitor entity in the dataset. N_d denotes the number of distinct relations of type sent. It also determines the length of the longest path from the progenitor node to the leaves (last-kid or hobbie), which is set to $N_d + 1$.

4.3 Discussion

While it is true that these are two made-up examples and may not be very common in practice, it is also very naive to assume that they will not happen. Thus, we should aim to build KGC models and explanation approaches that are able to handle such types of KGs.

5 Experiments

In this section, we utilize the FRUNI and FTREE datasets to empirically assess the performance of four recent explainers.

Experimental setup. We utilized the open-source code provided by Rossi et al. [27] [1] for running the experiments. We integrated the proposed datasets and trained three different KGE models: ComplEx [31], ConvE [7], and TransE [3]. Notably, we used the same triples for both training and testing phases, as our primary objective is to assess the performance of the explainer, not the performance capabilities of the KGE models.

Dataset. We conducted our experiments on two distinct instantiations of the proposed datasets. Specifically, for the FRUNI dataset, we configured the following parameters: $N_u = 1000$, $\lambda_f = 1$, $\alpha_u = 0.0$, and $N_f = 500$. On the other hand, for the FTREE dataset, we set the following parameters: $N_t = 5$, $\lambda_b = 30$, and $N_d = 2$. A summary of dataset statistics is presented in Tab. 1. The datasets created for these experiments are publicly available in our GitHub repository.

Explainers. Following Rossi et al. [27], we compare the following explainers: Kelpie [27], Kelpie limited to single-fact explanations (K1), Data Poisoning [40], and Criage [25].

[1] https://github.com/AndRossi/Kelpie

Table 3: MMR effectiveness of necessary (the lower the better) and sufficient (the higher the better) explanations. Results for `Criage` and `TransE` are missing due to their incompatibility.

		FRUNI				FTREE			
		K1	DP	Criage	Kelpie	K1	DP	Criage	Kelpie
Necess.	TransE	-0.77	-0.78	-	-0.89	-0.66	-0.88	-	-0.86
	ComplEx	-0.65	-0.69	-0.46	-1.00	-0.91	-0.95	-0.87	-0.99
	ConvE	-0.78	-0.74	-0.62	-1.00	-0.89	-0.93	-0.67	-0.99
Suffic.	TransE	0.83	0.83	-	0.83	0.61	0.61	-	0.61
	ComplEx	0.82	1.00	0.70	0.82	0.87	0.95	0.74	0.87
	ConvE	1.00	0.93	0.55	1.00	0.97	0.78	0.71	0.97

Table 4: Example of an incorrect (top) and correct (bottom) necessary explanation generated by `Kelpie`.

Pred. to explain	Necessary explanation
(kid-2-16-1, sent-2, hobbie-2-16)	(kid-2-16-0, sent-2, hobbie-2-16) (kid-2-16-0, ancestorOf, kid-2-16-1) (kid-2-16-1, ancestorOf, lkid-2-16)
(kid-2-24-0, ancestorOf, kid-2-24-1)	(kid-2-24-0, ancestorOf, kid-2-24-1)

Metrics. We evaluate the performance of the explainers based on the MRR *effectiveness*[27] defined as $\Delta\mathrm{MRR}$, meaning the variation in Mean Reciprocal Rank (MRR). For *necessary explanations*, smaller values of $\Delta\mathrm{MRR}$ are indicative of better performance. This is because a smaller $\Delta\mathrm{MRR}$ signifies that the removal of the explanation triples from the KG $\mathcal{G}$, followed by retraining, results in a deterioration of performance. In contrast, for *sufficient explanations*, larger values of $\Delta\mathrm{MRR}$ are desirable. A larger $\Delta\mathrm{MRR}$ indicates that adding the triples of the explanations to a random head or tail entity effectively converts the prediction to the correct one.

Results. In Tab. 2, we present the performance of the KGE models in terms of MRR. Notably, all models achieve a perfect score, indicating their capability to overfit the data. However, it remains uncertain whether these models can accurately predict facts through proper reasoning. In Tab. 3, we show the performance of the explainers in terms of *effective* MRR for both necessary and sufficient explanations. We can observe variability in the scores among the explainers. `Criage` consistently underperforms compared to others, while `Kelpie`, the state-of-the-art method, outperforms the rest. In terms of KGE models, `ConvE` consistently outperforms others. Nevertheless, it's crucial to emphasize that achieving a high $\Delta\mathrm{MRR}$ score does not guarantee the model's correctness in predicting ground-truth explanations. To delve deeper into this matter, we examine in Tab. 4 specific instance of explanations for FTREE generated by the best-performing combination of `Kelpie` with `ConvE`. Note that the necessary explanation for (`kid-2-16-1`, `sent-2`, `hobbie-2-16`) fails to include all relevant facts up to the corresponding `progenitor` entity. This limitation is expected as explanations in `Kelpie` are restricted to triples containing the head or tail entity of the prediction. Yet, this observation underscores the need for synthetic datasets that facilitate the assessment of failure cases.

6 Conclusion

We introduced the FRUNI and FTREE synthetic knowledge graphs to evaluate the performance of explainers in scenarios where fact predictions rely on indirectly connected links. Our experiments reveal some limitations of current explainability methods. We leave for feature work the design and analysis of a KGE model that can perfectly solve the KGC task on the proposed datasets. A model of this nature would help to dismiss the failure cases on the KGE model, and better understand the effectiveness of explainability methods. We expect FRUNI and FTREE to benefit the knowledge graph community, enabling researchers and practitioners to assess and improve the explainability of KGE models, ultimately advancing the field of knowledge graph completion.

References

[1] Lisa Bauer, Yicheng Wang, and Mohit Bansal. Commonsense for generative multi-hop question answering tasks. In *Conference on Empirical Methods in Natural Language Processing*, 2018.

[2] Patrick Betz, Christian Meilicke, and Heiner Stuckenschmidt. Adversarial explanations for knowledge graph embeddings. In *International Joint Conference on Artificial Intelligence*, 2022.

[3] Antoine Bordes, Nicolas Usunier, Alberto García-Durán, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. In *NIPS*, 2013.

[4] Wenhu Chen, Wenhan Xiong, Xifeng Yan, and William Yang Wang. Variational knowledge graph reasoning. *ArXiv*, abs/1803.06581, 2018.

[5] Xiang Chen, Ningyu Zhang, Lei Li, Shumin Deng, Chuanqi Tan, Changliang Xu, Fei Huang, Luo Si, and Huajun Chen. Hybrid transformer with multi-level fusion for multimodal knowledge graph completion. *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2022.

[6] 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. *ArXiv*, abs/1711.05851, 2017.

[7] Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. Convolutional 2d knowledge graph embeddings. *ArXiv*, abs/1707.01476, 2017.

[8] Matt Gardner, Partha Pratim Talukdar, Jayant Krishnamurthy, and Tom Michael Mitchell. Incorporating vector space similarity in random walk inference over knowledge bases. In *Conference on Empirical Methods in Natural Language Processing*, 2014.

[9] Prashant Gohel, Priyanka Singh, and Manoranjan Mohanty. Explainable ai: current status and future directions. *ArXiv*, abs/2107.07045, 2021.

[10] Bin He, Di Zhou, Jinghui Xiao, Xin Jiang, Qun Liu, Nicholas Jing Yuan, and Tong Xu. Integrating graph contextualized knowledge into pre-trained language models. *ArXiv*, abs/1912.00147, 2019.

[11] Shizhu He, Kang Liu, Guoliang Ji, and Jun Zhao. Learning to represent knowledge graphs with gaussian embedding. *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, 2015.

[12] Shaoxiong Ji, Shirui Pan, E. Cambria, Pekka Marttinen, and Philip S. Yu. A survey on knowledge graphs: Representation, acquisition, and applications. *IEEE Transactions on Neural Networks and Learning Systems*, 33:494–514, 2020.

[13] Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *International Conference on Machine Learning*, 2017.

[14] N. Lao and William W. Cohen. Relational retrieval using a combination of path-constrained random walks. *Machine Learning*, 81:53–67, 2010.

[15] Yiqiao Li, Jianlong Zhou, Sunny Verma, and Fang Chen. A survey of explainable graph neural networks: Taxonomy and evaluation metrics. *ArXiv*, abs/2207.12599, 2022.

[16] Kenny Ye Liang, Lingyuan Meng, Meng Liu, Yue Liu, Wenxuan Tu, Siwei Wang, Sihang Zhou, Xinwang Liu, and Fu Sun. A survey of knowledge graph reasoning on graph types: Static, dynamic, and multimodal. 2022.

[17] Xuan Lin, Zhe Quan, Zhi-Jie Wang, Tengfei Ma, and Xiangxiang Zeng. Kgnn: Knowledge graph neural network for drug-drug interaction prediction. In *International Joint Conference on Artificial Intelligence*, 2020.

[18] Weijie Liu, Peng Zhou, Zhe Zhao, Zhiruo Wang, Qi Ju, Haotang Deng, and Ping Wang. K-bert: Enabling language representation with knowledge graph. In *AAAI Conference on Artificial Intelligence*, 2019.

[19] Antonio Longa, Steve Azzolin, Gabriele Santin, Giulia Cencetti, Pietro Lio', Bruno Lepri, and Andrea Passerini. Explaining the explainers in graph neural networks: a comparative study. *ArXiv*, abs/2210.15304, 2022.

[20] Dongsheng Luo, Wei Cheng, Dongkuan Xu, Wenchao Yu, Bo Zong, Haifeng Chen, and Xiang Zhang. Parameterized explainer for graph neural network. *ArXiv*, abs/2011.04573, 2020.

[21] Farzaneh Mahdisoltani, Joanna Asia Biega, and Fabian M. Suchanek. Yago3: A knowledge base from multilingual wikipedias. In *Conference on Innovative Data Systems Research*, 2015.

[22] Sameh K. Mohamed, Vit Novavcek, and Aayah Nounu. Discovering protein drug targets using knowledge graph embeddings. *Bioinformatics*, 2019.

[23] Cataldo Musto, Pierpaolo Basile, and Giovanni Semeraro. Embedding knowledge graphs for semantics-aware recommendations based on dbpedia. *Adjunct Publication of the 27th Conference on User Modeling, Adaptation and Personalization*, 2019.

[24] Deepak Nathani, Jatin Chauhan, Charu Sharma, and Manohar Kaul. Learning attention-based embeddings for relation prediction in knowledge graphs. In *Annual Meeting of the Association for Computational Linguistics*, 2019.

[25] Pouya Pezeshkpour, Yifan Tian, and Sameer Singh. Investigating robustness and interpretability of link prediction via adversarial modifications. In *North American Chapter of the Association for Computational Linguistics*, 2018.

[26] IV RobertL.Logan, Nelson F. Liu, Matthew E. Peters, Matt Gardner, and Sameer Singh. Barack's wife hillary: Using knowledge graphs for fact-aware language modeling. *ArXiv*, abs/1906.07241, 2019.

[27] Andrea Rossi, Donatella Firmani, Paolo Merialdo, and Tommaso Teofili. Kelpie: an explainability framework for embedding-based link prediction models. *Proc. VLDB Endow.*, 15:3566–3569, 2022.

[28] M. Schlichtkrull, Thomas Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. Modeling relational data with graph convolutional networks. In *Extended Semantic Web Conference*, 2017.

[29] M. Schlichtkrull, Nicola De Cao, and Ivan Titov. Interpreting graph neural networks for nlp with differentiable edge masking. *ArXiv*, abs/2010.00577, 2020.

[30] Kristina Toutanova and Danqi Chen. Observed versus latent features for knowledge base and text inference. In *Workshop on Continuous Vector Space Models and their Compositionality*, 2015.

[31] Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. Complex embeddings for simple link prediction. *ArXiv*, abs/1606.06357, 2016.

[32] Shikhar Vashishth, Soumya Sanyal, Vikram Nitin, and Partha Pratim Talukdar. Composition-based multi-relational graph convolutional networks. *ArXiv*, abs/1911.03082, 2019.

[33] Meihong Wang, Linling Qiu, and Xiaoli Wang. A survey on knowledge graph embeddings for link prediction. *Symmetry*, 13:485, 2021.

[34] Ruijie Wang, M. Wang, Jun Liu, Weitong Chen, Michael Cochez, and S. Decker. Leveraging knowledge graph embeddings for natural language question answering. In *International Conference on Database Systems for Advanced Applications*, 2019.

[35] Wenhan Xiong, Thi-Lan-Giao Hoang, and William Yang Wang. Deeppath: A reinforcement learning method for knowledge graph reasoning. In *Conference on Empirical Methods in Natural Language Processing*, 2017.

[36] 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.

[37] Liang Yao, Chengsheng Mao, and Yuan Luo. Kg-bert: Bert for knowledge graph completion. *ArXiv*, abs/1909.03193, 2019.

[38] Rex Ying, Dylan Bourgeois, Jiaxuan You, Marinka Zitnik, and Jure Leskovec. Gnnexplainer: Generating explanations for graph neural networks. *Advances in neural information processing systems*, 32:9240–9251, 2019.

[39] Fuzheng Zhang, Nicholas Jing Yuan, Defu Lian, Xing Xie, and Wei-Ying Ma. Collaborative knowledge base embedding for recommender systems. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016.

[40] Hengtong Zhang, T. Zheng, Jing Gao, Chenglin Miao, Lu Su, Yaliang Li, and Kui Ren. Data poisoning attack against knowledge graph embedding. In *International Joint Conference on Artificial Intelligence*, 2019.

[41] Shichang Zhang, Jiani Zhang, Xiang Song, Soji Adeshina, Da Zheng, Christos Faloutsos, and Yizhou Sun. Page-link: Path-based graph neural network explanation for heterogeneous link prediction. *Proceedings of the ACM Web Conference 2023*, 2023.

[42] Anjie Zhu, Deqiang Ouyang, Shuang Liang, and Jie Shao. Step by step: A hierarchical framework for multi-hop knowledge graph reasoning with reinforcement learning. *Knowl. Based Syst.*, 248:108843, 2022.

[43] Zhaocheng Zhu, Xinyu Yuan, Mikhail Galkin, Sophie Xhonneux, Ming Zhang, Maxime Gazeau, and Jian Tang. A*net: A scalable path-based reasoning approach for knowledge graphs. 2022.

A Data generation process

In this section, we provide a detailed description of the data generation process for the two synthetic datasets introduced in this work, namely FRUNI and FTREE.

A.1 Data Generation for FRUNI

The data generation process for FRUNI is outlined in Alg. 1. In this context:

- We represent a `uni` entity as a single integer, denoted as `i`.
- For `student` entities, we use two integers, `i-j`, where the first integer signifies the university `i`, and the second integer represents the student's identifier within that university.
- We denote `friend` entities with three integers, `i-j-k`, with the first two integers referring to the university and student entities that the friend is connected to, and the last integer (`k`) indicating the identifier of the friend within friends of student `i-j`.

Alg. 2 contains the data generation process for FTREE. Note that we identify a `progenitor` entity with a single integer `i`. We identify the descendants with the integers `i-j-k`, the first one corresponds to the correcponding `progenitor`, the second one the lineage branch j, and the last one identifies the generation index within the lineage branch. We identify the `last-kid` entity of a `progenitor` `i` and lineage j as `i-j-lkid` and the correcponding `hobbie` entity with `i-j-hob`.

A.2 Data Generation for FTREE

The data generation process for FTREE is detailed in Alg. 2. In this context:

- We represent a `progenitor` entity as a single integer, denoted as `i`
- Descendant entities are identified by three integers, `i-j-k`, with the first integer corresponding to the `progenitor`, the second integer representing the lineage branch j, and the last integer indicating the generation index within the lineage branch.
- The `last-kid` entity of an `progenitor` `i` and lineage j is referred to as `i-j-lkid`, and the corresponding `hobbie` entity is identified as `i-j-hob`.

Algorithm 1 FRUNI generation process

Input: N_u number of universities; λ_f average number of friends per student; α_u probability of university collaboration; N_f number of universities that foster friendship
$\mathcal{V} = \emptyset$ ▷ Initialize the set of entites
$\mathcal{R} = \{\texttt{enrolls}, \texttt{friendOf}, \texttt{collabWith}\}$ ▷ Create the set of relations
$\mathcal{T} = \emptyset$ ▷ Initialize the set of triples
for $i \in \{1, \dots, N_u\}$ **do** ▷ Iterate over the number of universities
  $\mathcal{V} = \mathcal{V} \cup \{\texttt{i}\}$ ▷ Add `uni` entity
  $\mathcal{V} = \mathcal{V} \cup \{\texttt{i-1}, \texttt{i-2}\}$ ▷ Add `student` entities of `i`
  $\mathcal{T} = \mathcal{T} \cup \{\texttt{i}, \texttt{enrolls}, \texttt{i-j}\}_{j=1}^{2}$ ▷ Add triples connecting the uni to the students
  for $j \in \{1, 2\}$ **do**
    $K_{ij} \sim \text{Poisson}(\lambda_\text{f}) + 1$
    $\mathcal{V} = \mathcal{V} \cup \{\texttt{i-j-k}\}_{k=1}^{K_{ij}}$ ▷ Add K `friend` entities of `i-j`
    $\mathcal{T} = \mathcal{T} \cup \{\texttt{i-j}, \texttt{friendOf}, \texttt{i-j-k}\}_{k=1}^{K_{ij}}$
  end for
  if $i \leq N_f$ **then** ▷ University `i` fosters friendship, add triples
    $\mathcal{T} = \mathcal{T} \cup \{\texttt{i-1-k}, \texttt{friendOf}, \texttt{i-j-h} : k \in \{1, \dots, K_{i1}\}, h \in \{1, \dots, K_{i2}\}\}$
    $\mathcal{T} = \mathcal{T} \cup \{\texttt{i-j-h}, \texttt{friendOf}, \texttt{i-1-k} : k \in \{1, \dots, K_{i1}\}, h \in \{1, \dots, K_{i2}\}\}$
  end if
end for

$\mathcal{T} = \mathcal{T} \cup \{\texttt{i}, \texttt{collabWith}, \texttt{j} : u_{ij} < \alpha_u\}_{i,j=1}^{N_u}$ with $u_{ij} \sim \text{Uni}(0, 1)$
Output: $\mathcal{V}$, $\mathcal{R}$, $\mathcal{T}$

Algorithm 2 FTREE generation process

Input: N_t number of family trees; λ_b average number of branches per family tree; N_d number of different lengths of descendants
$\mathcal{V} = \emptyset$ ▷ Initialize the set of entites
$\mathcal{R} = \{\texttt{ancestorOf}\} \cup \{\texttt{sent-i}\}_{i=1}^{N_d}$ ▷ Create the set of relations
$\mathcal{T} = \emptyset$ ▷ Initialize the set of triples
for $i \in \{1, \dots, N_t\}$ **do** ▷ Iterate over the number of family trees
  $\mathcal{V} = \mathcal{V} \cup \{\texttt{i}\}$ ▷ Add `progenitor` entity
  $B_i \sim \text{Poisson}(\lambda_\text{b}) + 2$
  for $j \in \{1, \dots, B_i\}$ **do**
    $L_{ij} \sim \text{Uni}(\{1, \dots, \text{N}_\text{d}\})$
    $\mathcal{V} = \mathcal{V} \cup \{\texttt{i-j-k}\}_{k=1}^{L_{ij}}$ ▷ Add L_{ij} `kid` entities for family tree `i` and lineage j
    $\mathcal{T} = \mathcal{T} \cup \{\texttt{i}, \texttt{ancestorOf}, \texttt{i-j-1}\}$
    if $L_{ij} > 1$ **then**
      $\mathcal{T} = \mathcal{T} \cup \{\texttt{i-j-k}, \texttt{ancestorOf}, \texttt{i-j-h} : k \in \{1, \dots, L_{ij} - 1\}, h \in \{2, \dots, L_{ij}\}\}$
    end if
    $\mathcal{V} = \mathcal{V} \cup \{\texttt{i-j-lkid}\}$ ▷ Add `last-kid` entity
    $\mathcal{T} = \mathcal{T} \cup \{\texttt{i-j-L}_{ij}, \texttt{ancestorOf}, \texttt{i-j-lkid}\}$
    $\mathcal{V} = \mathcal{V} \cup \{\texttt{i-j-hob}\}$ ▷ Add `hobbie` entity
    $\mathcal{T} = \mathcal{T} \cup \{\texttt{i-j-L}_{ij}, \texttt{sent-L}_{ij}, \texttt{i-j-hob}\}$
  end for
end for
Output: $\mathcal{V}$, $\mathcal{R}$, $\mathcal{T}$