LLM-Guided Explanation Generation for Knowledge Graph Embedding-Based Node Classification

Anonymous ACL submission

Abstract

Knowledge graph (KG) embedding models have achieved remarkable success in various tasks, particularly in node classification. However, their decision-making processes remain opaque, limiting interpretability and trustworthiness. To address this challenge, we propose the first method for explaining KG embedding-based node classification. It integrates large language models (LLMs) to generate both graph-structured and textual explanations, offering deeper insights into model reasoning. Specifically, we train a proxy model to approximate the behavior of the original KG embedding model. Leveraging the distilled knowledge from this proxy model, an LLM is finetuned to identify and reason about critical relation path patterns that significantly influence predictions. Guided by the selected patterns and proxy model, we design an efficient searching algorithm to extract the final set of critical triples with LLM-generated reasoning. Experiments on three representative KG embedding models across multiple benchmark datasets demonstrates the effectiveness and generalization of our method in explaining KG embedding-based node classification.

1 Introduction

Knowledge graph (KG) embedding (Wang et al., 2017; Choudhary et al., 2021), by mapping discrete entities and relations into continuous vector space, has pioneered new pathways for semantic representation learning of structured knowledge, establishing itself as a core paradigm in natural language processing (NLP). This embedding paradigm achieves deep integration of knowledge-driven neural networks in scenarios such as semantic search and intelligent question answering by preserving the topological structure and semantic relations of KGs. Particularly in node classification tasks, KG embedding significantly enhances the accuracy of fine-grained entity type inference by fusing graph structural information from entity neighborhoods with semantic relations, which holds critical significance for constructing vertical applications like precision medical diagnostic systems and financial risk prediction models.

Despite the superior predictive performance of KG embedding models, their internal mechanisms remain plagued by a "black-box" dilemma: mainstream embedding methods based on geometric transformations (Bordes et al., 2013; Wang et al., 2014), semantic matching (Yang et al., 2015; Trouillon et al., 2016), or graph neural networks (Schlichtkrull et al., 2018; Trouillon et al., 2016) often rely on implicit vector operations for decision-making, rendering their reasoning processes opaque to human interpretation. This lack of explainability severely hinders the trustworthy deployment of KG embedding in high-stakes scenarios such as judicial decision-making and medical diagnosis. To address this bottleneck, researchers have begun developing post-hoc explanation frameworks for link prediction (Zhang et al., 2023; Ma et al., 2024) and entity alignment (Tian et al., 2024). While these approaches highlight the effectiveness of explanation generation through subgraph patterns or path reasoning, their direct application to node classification is hindered by several critical challenges that must be addressed.

First, due to the fundamental differences in task objectives compared to link prediction and entity alignment, existing methods have not explored how to define a principled process for identifying the most critical subset of triples for node classification among numerous candidates. Second, unlike link prediction, which benefits from explicit source-to-target entity paths, or entity alignment, which leverages aligned neighborhood structures, node classification lacks such heuristic anchors. This absence makes it more challenging to discern meaningful relation patterns that directly contribute to classification decisions. Last, existing methods

predominantly rely on deep learning techniques or heuristic search strategies, which often result in limited interpretability. The generated explanations may lack practical significance or fail to align with human comprehension, further restricting their usability in real-world applications.

To tackle these challenges, the emergence of large language models (LLMs) presents a promising solution. With their advanced text comprehension and reasoning capabilities, LLMs can filter, refine, and contextualize extracted relation patterns, ensuring that the generated explanations are structurally coherent, semantically meaningful, and easily interpretable by humans. Motivated by this potential, we propose LLM-ExKG, a novel method that integrates LLMs into the explanation generation pipeline for KG embedding-based node classification. To generate human-readable and model-faithful explanation, we train an LLM that combines both general world knowledge and distilled knowledge from the original model. This enables the LLM to accurately identify and reason about critical relation patterns. Building upon these identified patterns, we further design an efficient searching algorithm that simulates the decision-making process of the original model, enabling precise identification of the critical triples with LLM-generated reasoning as final explanation results.

In summary, our main contributions include:

- We provide a definition of explanation for KG embedding-based node classification and incorporate LLMs into the explainability framework for KG embedding models.
- We propose LLM-ExKG, the first method for explaining KG embedding-based node classification, offering both interpretable graph-structured and textual explanations.
- We leverage LLM-ExKG to explain three representative KG embedding models and assess the performance on four datasets. The experimental results show the effectiveness and generalization of LLM-ExKG.

2 Related Work

KG embedding aims to map entities and relations in a KG into continuous vector space, enabling efficient computation and reasoning. A key application of KG embedding is node classification, which leverages embeddings capture semantic and structural information to improve categorization. Existing KG embedding models can be categorized into three classes: translation-based models (Bordes et al., 2013; Wang et al., 2014), which represent relations as vector translations; semantic matching-based models (Yang et al., 2015; Trouillon et al., 2016), which use matrix or tensor factorization to capture relational patterns; and GCN-based models (Schlichtkrull et al., 2018; Vashishth et al., 2020), which leverage graph convolution to incorporate neighborhood information. These models can be integrated with classifiers to support node classification. Regardless of model types, our method is able to generate effective explanations.

Explanation generation aims to interpret the behavior of pre-trained models without modifying their architectures. Methods like LIME (Ribeiro et al., 2016) and SHAP (Lundberg and Lee, 2017) are widely applied across various machine learning models. These methods rely on feedback obtained from perturbing the original model, which can be difficult to acquire in certain KG embedding models, such as TransE. Other methods (Zhang et al., 2023; Ma et al., 2024; Tian et al., 2024) are specifically designed for KG embedding models, but they are primarily tailored for link prediction and entity alignment tasks. The former (Zhang et al., 2023; Ma et al., 2024) typically searches for paths leading to the target entity, while the latter (Tian et al., 2024) relies on heuristic reasoning based on entity alignment. It is challenging to transfer either method to node classification. Although certain GNN-based explanation methods (Ying et al., 2019; Yuan et al., 2021) can be adapted for node classification, many KG embedding models do not adhere to the GNN framework, limiting the applicability of these methods. More critically, most existing methods rely heavily on heuristic searching or deep learning-based models, raising concerns about their reliability. In contrast, our method is not only applicable to KG embedding models across various frameworks but also leverages LLMs to guide explanation generation in a more interpretable way, thereby improving trustworthiness.

3 Preliminaries

We define a KG as $\mathcal{G} = \{\mathcal{E}, \mathcal{R}, \mathcal{T}\}$, where $\mathcal{E}$ is the set of entities, literals, and concepts, and $\mathcal{R}$ is the set of relations. $\mathcal{T} \subseteq \mathcal{E} \times \mathcal{R} \times \mathcal{E}$ is the set of triples. Given a KG $\mathcal{G}$, node classification aims to learn a mapping function $f : \mathcal{E} \to \mathcal{C}$ that assigns a label from a pre-defined concept set $\mathcal{C}$ to each entity.

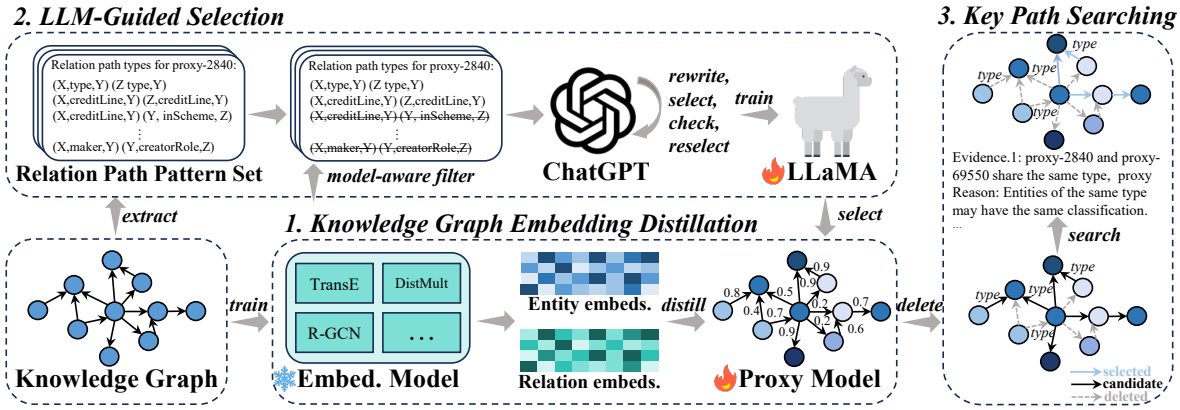

Figure 1: Framework of LLM-ExKG.

In this paper, we study post-hoc explanation generation for KG embedding-based node classification. Given an entity e predicted by the model to be of concept c, where $e \in \mathcal{E}$ and $c \in \mathcal{C}$, we define candidate triples $\mathcal{T}_e$ for generating explanations as those within k hops around entity e. A smaller k may lead to insufficient information, while a larger k can introduce excessive irrelevant triples. The objective of our studied problem is to identify the minimal subset $\mathcal{T}_e^* \subseteq \mathcal{T}_e$, where $\mathcal{T}_e^* \neq \emptyset$ such that, even after removing $\mathcal{T}_e - \mathcal{T}_e^*$, the model can still accurately predict e to be of c. Beyond the traditional studied graph-structured explanations, our method also generates corresponding textual explanations, making the reasoning process more interpretable.

4 Methodology

4.1 Overview

We propose LLM-ExKG, a novel method consisting of three key modules. (1) In the **KG embedding distillation** module, we design a lightweight proxy model to efficiently capture and inherit the key classification knowledge from the original model. (2) In the **LLM-guided selection** module, we train an LLM to identify important relation path patterns that are both practically meaningful and highly relevant to the prediction of the KG embedding model. (3) In the **key path searching** module, based on the proxy model, we simulate the decision-making process of the original KG embedding model to find the most relevant paths for classification within the relation path types selected by the LLM. The set of triples within these paths forms the graph-structured explanations. Additionally, leveraging the reasoning power of the LLM, we generate human-readable textual explanations, improving both interpretability and transparency.

4.2 KG Embedding Distillation

In this module, we aim to transfer the essential classification knowledge from the KG embedding model into a proxy model. This proxy model is designed to approximate the prediction logic of the original model while providing a coarse-grained estimation of triple contribution to the prediction. To achieve this, we employ a variant of GCN augmented with adaptive weights for each candidate triple. This choice leverages the GCN's ability to capture complex graph structures while incorporating adaptive weights to improve interpretability and more effectively capture contextual relevance. Additionally, we introduce a counterfactual mechanism to refine the proxy model's ability to discern the true causal impact of each triple, thereby enhancing its explanatory power.

Specifically, the GCN-based proxy model consists of k layers, with each layer aggregating information from neighboring triples to capture both structural and semantic features of the k-hop subgraph centered on the target entity. Here, k is a predefined parameter set according to the experimental configuration, as described in Section 3. Building on the insights from (Tan et al., 2022), we integrate both factual and counterfactual reasoning into each layer's computation, further strengthening the model's interpretability and robustness. The computation process at each layer of the proxy model is formulated as follows:

$$\begin{aligned}
\mathbf{h}_e^l &= \sum_{e_n} \sum_{r} \alpha_{(e,r,e_n)} f\left(\mathbf{h}_e^{(l-1)}, \mathbf{W}_r^l \mathbf{e}_r\right), \\
\tilde{\mathbf{h}}_e^l &= \sum_{e_n} \sum_{r} (1 - \alpha_{(e,r,e_n)}) f\left(\tilde{\mathbf{h}}_e^{(l-1)}, \mathbf{W}_r^l \mathbf{e}_r\right), \quad (1) \\
\alpha_{(e,r,e_n)} &= \text{sigmoid}\left(w_{(e,r,e_n)}\right),
\end{aligned}$$

where $\mathbf{h}_e^l$ and $\tilde{\mathbf{h}}_e^l$ represent the embeddings of en-

tity e at the l-th layer, corresponding to the factual and counterfactual reasoning results, respectively. $\mathbf{h}_e^l$ and $\tilde{\mathbf{h}}_e^l$ at layer $l = 0$ are initialized using the original KG embedding model's entity embeddings and $\mathbf{e}_r$ is derived from the KG embedding model's relation embeddings. e_n denotes the neighboring entities of e, and r represents the relation between e and e_n. $\mathbf{W}_r^l$ is the learnable transformation matrix for relation r at layer l, which projects relation embeddings into a suitable space for interaction with entity embeddings. $f(\cdot)$ represents the fusion operation between entity and relation information, which is implemented as an addition in our method. $w_{(e,r,e_n)}$ denotes the learnable parameter for triple (e, r, e_n), which is designed to capture the importance of each triple in the reasoning process.

Our objective is to minimize the distance between the factual embedding $\mathbf{h}_e^k$ and its corresponding entity embedding $\mathbf{e}_e$ in the KG embedding model, while simultaneously maximizing the dissimilarity between the counterfactual embedding $\tilde{\mathbf{h}}_e^k$ and $\mathbf{e}_e$. Inspired by (Sun et al., 2020), we employ the variant triple loss to optimize the proxy model. The formulation is as follows:

$$\mathcal{L}_{\text{proxy}} = \left\| \mathbf{h}_e^k - \mathbf{e}_e \right\| + \beta \left[\lambda - \left\| \tilde{\mathbf{h}}_e^k - \mathbf{e}_e \right\| \right]_+, \quad (2)$$

where $\| \cdot \|$ denotes the L_2-norm of the vector, $[\cdot]_+ = \max(0, \cdot)$, and λ, β are positive-valued hyperparameters. The first term encourages $\mathbf{h}_e^k$ to be close to $\mathbf{e}_e$, while the second term, governed by λ and β, imposes a margin-based constraint on $\tilde{\mathbf{h}}_e^k$, ensuring that its distance from $\mathbf{e}_e$ exceeds λ.

By this distillation process, the proxy model can emulate the decision-making of the original KG embedding model in factual reasoning and leverage the learned triple contributions to offer valuable insights for subsequent explanation generation.

4.3 LLM-Guided Selection

Due to the vast number of candidate triples, directly leveraging the proxy model for searching incurs substantial computational overhead. Moreover, the learned contributions of individual triples are often independent, making it challenging to identify the combinations of triples with practical significance. To this end, we strive to introduce commonsense knowledge from LLMs to ensure that the final explanation generation is not only faithful to the original model's prediction but also holds meaningful practical value. Since entity names in triples are typically encoded and lack inherent meaning, whereas relations contain semantic information relevant for classification, as illustrated in Figure 1, we aim to leverage the LLM to select the critical relation path patterns rather than specific relation paths. This approach not only enhances interpretability but also substantially reduces the candidate set, making the selection process more efficient. Building on the above consideration, in this module, we aim to integrate the knowledge from the original KG embedding model with the extensive commonsense reasoning capabilities of the LLM (e.g., ChatGPT) to construct a high-quality dataset. This dataset is used to train another LLM (e.g., LLaMA (Touvron et al., 2023)) capable of identifying relation path patterns that remain faithful to the original model's predictions while carrying real-world significance.

4.3.1 Data Construction

Relation path pattern extraction. We extract k-hop relation paths around the target entity and replace specific entities with variable names, retaining only relation information to construct generalized relation path patterns. For example, the path (proxy-2840, *type*, proxy) (proxy-69550, *type*, proxy) is transformed to (X, *type*, Y) (Z, *type*, Y).

Model-aware filter. Since we aim to ensure that the explanations remain faithful to the original KG embedding model, the selected relation path patterns should significantly contribute to the model's classification prediction. Fortunately, the proxy model trained in Section 4.2 is capable of estimating the contribution of triples. Building on this, we design a scoring mechanism for relation path patterns to derive the preference ranking of the original KG embedding model. Specifically, the score computation is formulated as follows:

$$\text{score}(p) = \max_{(t_1,t_2)\in\mathcal{S}_p} \left(\text{aggregate}\left(w_{t_1}, w_{t_2}\right)\right), \quad (3)$$

where p is a relation path pattern and $\mathcal{S}_p$ is the set of relation paths that conform to p. t_1, t_2 are two triples from relation path conforming to p, with w_{t_1}, w_{t_2} denoting their respective contribution values. Based on the computed scores, we rank the relation path patterns in descending order and select the top m ones as the filtered results, where m is a hyperparameter. In this way, we not only ensure that the selection results of the LLM align with the prediction preference of the original model but also narrow the candidate range, alleviating the challenge of LLMs in processing long text.

Practical relation path pattern selection. Given a set of filtered relation path patterns surrounding an entity, we first use an LLM (e.g., ChatGPT) to rewrite them into natural language expressions, enhancing human comprehension and enabling more effective subsequent processing by the LLM. For instance, the pattern $(X, \textit{type}, Y)$ $(Z, \textit{type}, Y)$ can be rewritten as "X and Z share the same type, Y". Next, we prompt the LLM to select key relation path patterns as evidence supporting the KG embedding model's classification and provide its reasoning. Through this approach, the LLM can provide reasonable relation path patterns along with corresponding rationale. Since its selection is based on the criterion of providing a valid rationale, the chosen patterns are more likely to align with the ground-truth decision-making logic, thereby enhancing the interpretability of the results. Nevertheless, the LLM may still produce occasional inaccurate results. Inspired by (He et al., 2024), to ensure the quality of the training data, we manually check and prompt the LLM to reselect the relation path patterns until the results are logically correct.

4.3.2 LLM Finetuning

We finetune a smaller open source LLM to integrate the collected knowledge, enabling efficient and precise reasoning for relation path pattern selection. Given an entity e, we construct the prompt $\mathcal{P}$, which includes the rewritten candidate relation path patterns surrounding e along with the query for selection and reasoning. The optimization is guided by the following loss function:

$$\mathcal{L}_{\text{LLM}} = -\sum_{i=1}^{N} \log \Pr\left[\, y_i \,|\, y_{<i}, \mathcal{P}(e) \,\right], \quad (4)$$

where N denotes the number of tokens in the selected relation path patterns and corresponding rationale. y_i $(i = 1, 2, \ldots, N)$ denotes the i-th token. $\Pr[\, y_i \,|\, y_{<i}, \mathcal{P}(e) \,]$ indicates the probability of generating y_i using the LLM, conditioned on the prompt $\mathcal{P}(e)$ and the previously generated tokens.

4.4 Key Path Searching

After obtaining the relation path patterns selected by the LLM, we design an efficient searching algorithm based on the trained proxy model to identify key relation paths, ultimately constructing the final explanations. Given a target entity, the searching process is presented in Algorithm 1. The process begins by initializing the result set $\mathcal{K}$ as empty

Algorithm 1: Greedy search for key paths

Input: Target label c; Proxy model $\mathcal{M}_p$; Set of relation path patterns $\mathcal{S}_{\text{pattern}}$; Dictionary of relation path patterns to relation path set $\mathcal{D}_p$.
Output: Set of key paths $\mathcal{K}$.

```
1  K ← set();
2  foreach p_pattern ∈ S_pattern do
3      L_path ← D_p[p_pattern].sort();
4      G_sub ← set();
5      foreach p_path ∈ L_path do
6          add path p_path in G_sub and K;
7          h ← M_p(G_sub);
8          get prediction score s of label c with h;
9          if s ≥ γ then
10             break;
11 return K;
```

(Line 1). For each relation path pattern p_{pattern} from LLM selection, we retrieve its associated paths from $\mathcal{D}_p$ (Lines 2–3). These paths are sorted based on the values computed by the function $\text{aggregate}(\cdot)$ mentioned in Eq. (3) (Line 3). Next, we initialize an empty set $\mathcal{G}_{\text{sub}}$, designed to collect the most essential paths under current pattern (Line 4). We then iterate the path of the pattern in descending order (Line 5). In Lines 6–7, we add the path into $\mathcal{G}_{\text{sub}}$ and feed to the proxy model to generate an approximate representation $\mathbf{h}$ for $\mathcal{G}_{\text{sub}}$. Subsequently, we can input $\mathbf{h}$ to the classifier of the original model to obtain the prediction score s of target label c (Line 8). If s meets or exceeds a pre-defined threshold γ, the search terminates early for the current pattern, ensuring that the algorithm efficiently selects the most relevant paths while avoiding redundancy (Lines 9–10). Finally, we obtain $\mathcal{K}$, which collects all key paths under the path patterns selected by the LLM. The triples in these paths serves as the final graph-structured explanations. To improve clarity, we map entities to variables in the rewritten relation path patterns and LLM-generated reasoning for generating textual explanations, as illustrated in Figure 1.

Computation complexity. Since the number of selected relation path patterns P is constrained to a constant, the complexity primarily depends on the number of relation paths per pattern, N_p. The algorithm first sorts the relation paths, which takes $O(N_p \log N_p)$ time per pattern. Then, it iterates through the sorted paths, updating the key path set and computing representations, with early termination if a confidence threshold is met. In the worst case, the inner loop runs $O(N_p)$ times. Therefore, the overall complexity is $O(N_p \log N_p)$.

5 Experiments and Results

In this section, we conduct a detailed evaluation of LLM-ExKG. The source code is attached as supplementary materials.

5.1 Experiment Setup

Dataset. We evaluate LLM-ExKG on four widely used KG node classification datasets (Schlichtkrull et al., 2018): AIFB, MUTAG, BGS, and AM. These datasets span multiple domains and present diverse structural and semantic challenges, enabling a thorough assessment of LLM-ExKG's effectiveness. We follow the experimental setup of (Schlichtkrull et al., 2018), including the removal of label-leaking relations. Table 1 lists the dataset statistics.

KG embedding models. LLM-ExKG is capable of generating explanations for any KG embedding-based node classification models. To validate its generality, we conduct experiments on three well-known models: TransE (Bordes et al., 2013), DistMult (Yang et al., 2015), and R-GCN (Schlichtkrull et al., 2018), each belonging to a representative category, as outlined in Section 2.

Explanation methods. As far as we know, no research has specifically investigated explanation generation for KG embedding-based node classification. Thus, we adapt existing methods from other tasks to establish three baselines for comparison. Appendix A describes the implementation details.

- **LIME-NC** is a method built upon LIME (Ribeiro et al., 2016). In LIME-NC, we design a feedback generator $\mathcal{M}_g$ based on a GCN with k layers to efficiently obtain perturbed feedback results. We perturb the triples within k hops and feed them into $\mathcal{M}_g$ to obtain the feedback results. Subsequently, the perturbed dataset is input into the LIME framework to produce the final explanation results.

- **SHAP-NC** is based on SHAP (Lundberg and Lee, 2017). It follows the same approach as LIME-NC to generate the perturbed dataset. To efficiently obtain the explanation results, it employs the KernelSHAP (Lundberg and Lee, 2017) to process the perturbed dataset.

- **KGExplainer-NC** is an adaptation of KGExplainer (Ma et al., 2024), the current state-of-the-art method for explaining KG embedding-based link prediction. Since there are no target entities to reach in node classification, we modify the termination condition of the original method's greedy search to be based on the number of triples that meet the specified limit.

Datasets	Entities	Relations	Edges	Classes
AIFB	8,285	45	29,043	4
MUTAG	23,644	23	74,227	2
BGS	333,843	103	916,199	2
AM	1,666,764	133	5,988,321	11

Table 1: Statistics of the datasets

Evaluation metrics. Since our goal is to identify the most essential triples for the predicted classification, as defined in Section 3, we assess the explanation results using two key metrics: fidelity and sparsity (Tian et al., 2024). To compute them, we extract the entities $\mathcal{E}_r$ that are correctly classified by the original model, ensuring a prediction accuracy of 100%. Let $\mathcal{T}$ denote the candidate triples for these entities and $\mathcal{T}^*$ be the explanation results. We remove $\mathcal{T} - \mathcal{T}^*$ from the dataset and retrain the model on the modified dataset. Fidelity is defined as the ratio of entities in $\mathcal{E}_r$ that remain correctly classified. Sparsity is computed as $\frac{\mathcal{T}^*}{\mathcal{T}}$.

5.2 Main Results

The main results are presented in Table 2. Our LLM-ExKG consistently outperforms all baselines across different datasets and models on both fidelity and sparsity, demonstrating its robustness, generality, and effectiveness. KGExplainer-NC achieves the second-best performance, as it also utilizes transfer learning to approximate the decision-making behavior of the original model. However, its effectiveness is constrained by reliance on a local perspective during the searching process, lacking guidance from higher-level relation semantics. In contrast, LLM-ExKG overcomes this limitation by integrating an LLM, which enriches the explanation process with deeper semantic understanding and broader contextual awareness. Moreover, LLM-ExKG also demonstrates higher efficiency compared to KGExplainer-NC, as reported in Appendix B. LIME-NC and SHAP-NC perform poorly due to the difficulty of sampling from numerous candidate triples, as well as their inability to capture the intricate relation structure of the KG, further limiting their explanatory power.

Beyond these overall trends, the experimental results reveal several additional insights. (1) The performance of explanation methods is influenced

Embedding models	Explanation methods	AIFB		MUTAGG		BGS		AM	
		Fidelity↑	Sparsity↓	Fidelity↑	Sparsity↓	Fidelity↑	Sparsity↓	Fidelity↑	Sparsity↓
TransE	LIME-NC	0.546	7.05	0.638	18.49	0.563	0.62	0.100	0.80
	SHAP-NC	0.682	7.07	0.702	18.49	0.625	0.24	0.150	0.85
	KGExplainer-NC	0.818	5.80	0.723	19.67	0.750	0.73	0.375	1.01
	LLM-ExKG	**1.000**	**4.80**	**0.872**	**17.56**	**0.813**	**0.17**	**0.925**	**0.73**
DistMult	LIME-NC	0.500	7.65	0.667	17.45	0.643	0.55	0.046	0.73
	SHAP-NC	0.375	7.58	0.711	16.07	0.623	0.27	0.136	0.77
	KGExplainer-NC	0.792	6.88	0.778	15.16	**0.786**	0.18	0.227	0.69
	LLM-ExKG	**0.917**	**5.90**	**0.956**	**14.34**	**0.786**	**0.12**	**0.591**	**0.61**
R-GCN	LIME-NC	0.571	6.60	0.780	12.70	0.750	1.14	0.390	0.69
	SHAP-NC	0.600	7.16	0.800	11.62	0.750	0.91	0.396	0.73
	KGExplainer-NC	0.886	4.15	0.780	12.57	0.792	0.89	0.475	0.80
	LLM-ExKG	**0.971**	**4.01**	**0.900**	**10.47**	**0.875**	**0.62**	**0.729**	**0.63**

Table 2: Explanation comparison on KG embedding-based node classification.

Explanation methods	AIFB		MUTAGG		BGS		AM	
	Fidelity↑	Sparsity↓	Fidelity↑	Sparsity↓	Fidelity↑	Sparsity↓	Fidelity↑	Sparsity↓
LLM-ExKG	**0.971**	**4.01**	**0.900**	**10.47**	**0.875**	**0.62**	**0.729**	**0.63**
LLM-ExKG w/o KGED	0.714	4.07	0.780	11.20	0.750	0.76	0.390	0.88
LLM-ExKG w/o LGS	0.857	4.24	0.800	12.40	0.792	0.68	0.650	0.69
LLM-ExKG w/o KPS	0.742	4.14	0.800	11.94	0.750	0.70	0.418	0.83

Table 3: Ablation study of LLM-ExKG based on R-GCN.

by dataset size. Larger datasets introduce more candidate triples, increasing search complexity and making it more challenging to extract meaningful explanations. (2) Models with stronger classification capability tend to yield less sparse explanations, as they can more effectively capture useful features, reducing dependence on excessive input information. (3) Models that rely on topological structures are generally easier to explain, as their decision-making process is more aligned with the inherent graph connectivity patterns, facilitating explanation methods to extract meaningful evidence.

5.3 Ablation Study

To further analyze the contributions of different components in LLM-ExKG, we conduct an ablation study by removing KG embedding distillation (KGED), LLM-guided selection (LGS), and key path searching (KPS). Table 3 shows that all modules play a role in improving performance. Notably, KGED has the most significant impact, as it transfers the original model's classification knowledge to subsequent modules, ensuring the fidelity of generated explanations. LGS highlights the value of incorporating LLMs into the explanation process. By leveraging its broad knowledge and contextual understanding, the LLM helps refine the selection

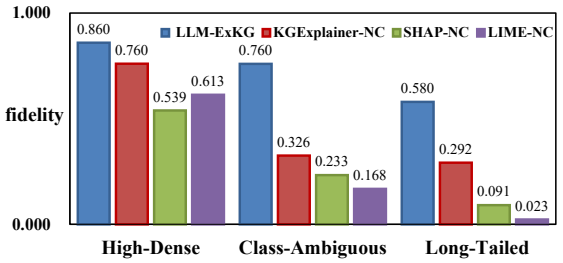

Figure 2: Comparison of explanation quality (fidelity) w.r.t. different entity types.

of relation path patterns, ensuring that the most relevant ones are prioritized for classification. This not only improves interpretability but also enhances the overall reliability of explanations. KPS further improves the explanation process by capturing the decision-making preferences of the original model. It effectively identifies paths most closely aligned with the correct classification, reinforcing the coherence between the generated explanations and the model's reasoning process.

5.4 Explanation Quality w.r.t. Entity Types

To further analyze and compare the explanation quality of different methods, we evaluate the fidelity of node classification on three particularly challenging types of entities: high-density candi-

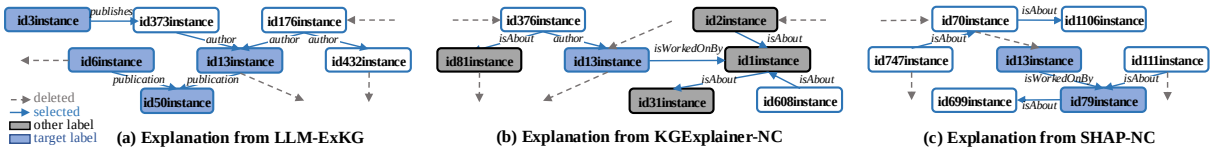

Figure 3: Case study of explaining R-GCN on entity "id13instance".

date entities, class-ambiguous entities, and long-tail entities. High-density candidate entities are those with an exceptionally large number of candidate triples, which we set to exceed 2,000. Class-ambiguous entities have neighboring entities that predominantly belong to other categories rather than the predicted one, increasing classification difficulty. Long-tail entities come from underrepresented categories with sparse training samples, often leading to lower predictive confidence and reduced explanation quality. We compute the average results across different datasets for each entity type when explaining R-GCN.

The findings are presented in Figure 2. LLM-ExKG consistently achieves the best performance across all three challenging entity types, demonstrating its robustness and adaptability in handling different complexities of node classification. It effectively reduces noise in high-density candidate entities, distinguishes subtle differences in class-ambiguous entities, and enhances generalization for long-tail entities. These results highlight the advantage of integrating model-specific knowledge distillation with LLM-driven guidance, ensuring both faithful and semantically rich explanations.

5.5 Case Study

We conduct a case study on the classification of entity "id13instance" to compare the explanations generated by LLM-ExKG, KGExplainer-NC and SHAP-NC. The results are shown in Figure 3. Notably, we observe that the LLM-guided selection effectively identifies path patterns strongly associated with the predicted entity's organization. For instance, $(Y, \textit{author}, X)$ $(Z, \textit{publishes}, Y)$ highlights the organization where the entity's work is published, while $(Y, \textit{author}, X)$ $(Y, \textit{author}, Z)$ and $(X, \textit{publication}, Y)$ $(Z, \textit{publication}, Y)$ connects entities closely linked to the organization through co-publication relation. Additionally, by extracting key paths, we identify entities that either belong to the same organization or have maintained close associations with it. In contrast, the identified subgraph patterns by KGExplainer-NC exhibit little relevance to the predicted organization. Moreover, due to the locality of the search, it overly explores a single entity belonging to a different category, leading to an incorrect prediction. In SHAP-NC, the inability to effectively model complex graph structures causes the generation of even disconnected subgraphs, compromising explanation quality.

Additionally, with the integration of the LLM, LLM-ExKG generates textual explanations, providing a more comprehensive and interpretable rationale for classification. For instance, given the path (id373instance, *author*, id13instance) (id3instance, *publishes*, id373instance), LLM-ExKG reformulates it as follows: "id373instance is authored by id3instance and published by id13instance." It further provides the reasoning: "the authorship by id13instance and publication by id13instance indicate a collaborative connection between id13instance and id3instance, suggesting they are associated through entity id373instance within the same organizational context." By translating structured paths into natural language with clear justifications, LLM-ExKG significantly enhances interpretability. This not only improves the readability of explanations but also helps users intuitively grasp the key factors influencing classification, effectively bridging the gap between structured graph reasoning and human understanding.

6 Conclusion

In this paper, we propose LLM-ExKG, the first method for explaining KG embedding-based node classification, leveraging LLMs to generate both graph-structured and textual explanations. LLM-ExKG first trains a proxy model to distill knowledge from the original KG embedding models. Then, an LLM is finetuned to identify and reason about critical relation path patterns. Finally, an efficient searching algorithm is employed to extract the final set of critical triples. These triples, alongside with the LLM-generated reasoning, form the final explanation results. Our experimental results demonstrate the effectiveness and generalization of LLM-ExKG in generating model-faithful and human-readable explanations.

Limitations

One limitation of the current method is that it is specifically designed for node classification tasks and does not yet extend to other KG embedding-based tasks, such as link prediction or entity alignment. However, we plan to extend LLM-ExKG to address these tasks in future work, exploring ways to generalize the method for broader applications in KG-related tasks. Another limitation of our method is its dependence on the current capabilities of LLMs. As LLMs continue to improve in reasoning, efficiency, and cost, future iterations of our method may generate even more accurate and interpretable explanations. This advancement could potentially eliminate the need for training a smaller LLM, allowing direct utilization of LLMs for explanation generation, thus simplifying the overall framework while enhancing performance.

References

Antoine Bordes, Nicolas Usunier, Alberto García-Durán, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *NIPS*, pages 2787–2795.

Shivani Choudhary, Tarun Luthra, Ashima Mittal, and Rajat Singh. 2021. A survey of knowledge graph embedding and their applications. *CoRR*.

Yinhan He, Zaiyi Zheng, Patrick Soga, Yaozhen Zhu, Yushun Dong, and Jundong Li. 2024. Explaining graph neural networks with large language models: A counterfactual perspective for molecular property prediction. *EMNLP Findings*.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *ICLR*.

Scott M Lundberg and Su-In Lee. 2017. A unified approach to interpreting model predictions. *NIPS*.

Tengfei Ma, Xiang Song, Wen Tao, Mufei Li, Jiani Zhang, Xiaoqin Pan, Jianxin Lin, Bosheng Song, and Xiangxiang Zeng. 2024. KGExplainer: Towards exploring connected subgraph explanations for knowledge graph completion. *CoRR*.

Marco Túlio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. "Why should I trust you?": Explaining the predictions of any classifier. In *KDD*, pages 1135–1144.

Michael Sejr Schlichtkrull, Thomas N. Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *ESWC*, pages 593–607.

Zequn Sun, Chengming Wang, Wei Hu, Muhao Chen, Jian Dai, Wei Zhang, and Yuzhong Qu. 2020. Knowledge graph alignment network with gated multi-hop neighborhood aggregation. In *AAAI*, pages 222–229.

Juntao Tan, Shijie Geng, Zuohui Fu, Yingqiang Ge, Shuyuan Xu, Yunqi Li, and Yongfeng Zhang. 2022. Learning and evaluating graph neural network explanations based on counterfactual and factual reasoning. In *WWW*, pages 1018–1027.

Xiaobin Tian, Zequn Sun, and Wei Hu. 2024. Generating explanations to understand and repair embedding-based entity alignment. In *ICDE*, pages 2205–2217.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton-Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurélien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv*, 2307.09288.

Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex embeddings for simple link prediction. In *ICML*.

Shikhar Vashishth, Soumya Sanyal, Vikram Nitin, and Partha P. Talukdar. 2020. Composition-based multi-relational graph convolutional networks. In *ICLR*.

Quan Wang, Zhendong Mao, Bin Wang, and Li Guo. 2017. Knowledge graph embedding: A survey of approaches and applications. *TKDE*, pages 2724–2743.

Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. Knowledge graph embedding by translating on hyperplanes. In *AAAI*, pages 1112–1119.

Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2015. Embedding entities and relations for learning and inference in knowledge bases. In *ICLR*.

Zhitao Ying, Dylan Bourgeois, Jiaxuan You, Marinka Zitnik, and Jure Leskovec. 2019. GNNExplainer: Generating explanations for graph neural networks. *NIPS*.

Hao Yuan, Haiyang Yu, Jie Wang, Kang Li, and Shuiwang Ji. 2021. On explainability of graph neural networks via subgraph explorations. In *ICML*, pages 12241–12252.

Shichang Zhang, Jiani Zhang, Xiang Song, Soji Adeshina, Da Zheng, Christos Faloutsos, and Yizhou Sun. 2023. PaGE-Link: Path-based graph neural network explanation for heterogeneous link prediction. In *WWW*, pages 3784–3793.

A Implementation Details

We run experiments on a workstation with two Intel Xeon Gold CPUs, an NVIDIA A800 GPU, and Ubuntu 18.04 LTS. The parameter k, which controls the range of candidate triples, is set to 2. The parameter β and λ are both set to 1. The parameter m for the model-aware filter is set to 50. The parameter P, representing the maximum number of relation path patterns selected by the LLM, is set between 10 and 20. The parameter γ is set between 0.9 and 1. We select LLaMA-2-7B[1] as the smaller LLM and employ LoRA (Hu et al., 2022) for parameter-efficient finetuning. The hyperparameters of LoRA are set to $r = 64$, alpha $= 32$, and dropout $= 0.1$.

B Comparison of Efficiency

We further conduct an efficiency comparison against the second-best method, KGExplainer-NC. The results shown in Figure 4 reveal that, despite utilizing an LLM for reasoning, LLM-ExKG consistently outperforms the second-best method in average running time across most datasets. Notably, this advantage becomes more pronounced as the dataset size increases, demonstrating the effectiveness of our method in selecting key relation path patterns and optimizing the search process.

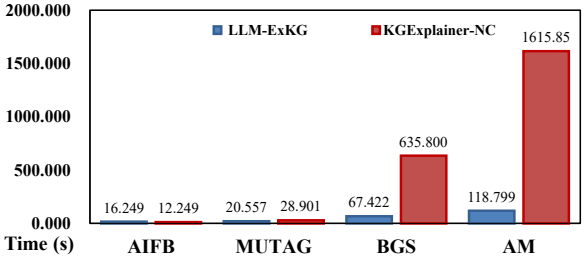

Figure 4: Comparison of average running time (s) between KGExplainer and LLM-ExKG in explaining R-GCN.

[1] https://huggingface.co/meta-llama/Llama-2-7b-hf