KnowFormer: Revisiting Transformers for Knowledge Graph Reasoning

Junnan Liu [1 2 *] Qianren Mao [1 †] Weifeng Jiang [3] Jianxin Li [1 2 †]

Abstract

Knowledge graph reasoning plays a vital role in various applications and has garnered considerable attention. Recently, path-based methods have achieved impressive performance. However, they may face limitations stemming from constraints in message-passing neural networks, such as missing paths and information over-squashing. In this paper, we revisit the application of transformers for knowledge graph reasoning to address the constraints faced by path-based methods and propose a novel method KnowFormer. KnowFormer utilizes a transformer architecture to perform reasoning on knowledge graphs from the message-passing perspective, rather than reasoning by textual information like previous pretrained language model based methods. Specifically, we define the attention computation based on the query prototype of knowledge graph reasoning, facilitating convenient construction and efficient optimization. To incorporate structural information into the self-attention mechanism, we introduce structure-aware modules to calculate query, key, and value respectively. Additionally, we present an efficient attention computation method for better scalability. Experimental results demonstrate the superior performance of KnowFormer compared to prominent baseline methods on both transductive and inductive benchmarks.

1. Introduction

Knowledge graphs (KGs) are structured knowledge bases that store known facts in the form of triplets (Hogan et al., 2022; Ji et al., 2022). Each triplet consists of a head entity, a relation, and a tail entity. However, real-world KGs often suffer from high incompleteness, making it challenging to retrieve the desired facts (Wang et al., 2017; Ji et al., 2022). Knowledge graph reasoning, which involves inferring new facts based on existing ones, is a fundamental and indispensable task in KGs with a wide range of applications (Qiu et al., 2019; Cao et al., 2019; Huang et al., 2019; Abujabal et al., 2018; Cheng et al., 2020; Zeng et al., 2022).

A substantial line of previous research has been dedicated to developing effective and robust methods for knowledge graph reasoning. Knowledge graph embedding (KGE) methods focus on embedding entities and relations into a low-dimensional space (Bordes et al., 2013; Sun et al., 2019; Yang et al., 2015; Trouillon et al., 2017; Li et al., 2022). Despite their impressive performance, these embedding-based methods struggle to generalize to inductive scenarios since they cannot leverage the local structures of knowledge graphs (Teru et al., 2020). As a result, researchers have turned to path-based methods to enhance reasoning performance and improve generalization. These methods learn pairwise representations from subgraphs (Teru et al., 2020; Mai et al., 2021; Wang et al., 2022a; Chamberlain et al., 2023) or relational paths (Zhu et al., 2021; Zhang & Yao, 2022; Zhang et al., 2023; Zhu et al., 2023) between entities which can be used to predict unseen facts. However, path-based methods can be limited by the missing paths (Franceschi et al., 2019) and over-squashing information (Alon & Yahav, 2021) as shown in Figure 1.

Recent works have leveraged the transformers for knowledge graph reasoning, drawing inspiration from the success of transformer-based models in various domains (Vaswani et al., 2017; Devlin et al., 2019; Dosovitskiy et al., 2021; Rives et al., 2021). Typically, these studies encode components of knowledge graphs using their text descriptions through pretrained language models (PLMs) and utilize either a discriminative paradigm (Lv et al., 2022; Wang et al., 2022b) or a generative paradigm (Xie et al., 2022; Saxena et al., 2022) to predict new facts. However, there are two main issues: (1) textual descriptions often contain ambiguity and require domain-specific knowledge for accurate encoding, *e.g.*, pretrained language models like BERT (Devlin et al., 2019), which are trained with commonsense knowledge, may not transfer well to domain-specific KGs; and (2) capturing structural information, which can be crucial for knowledge graph reasoning owing to the *low-rank*

[*]Work done during the internship at Zhongguancun Laboratory. [†]Corresponding authors. [1]Zhongguancun Laboratory, Beijing, P.R.China. [2]SCSE, Beihang University, Beijing, P.R.China. [3]SCSE, Nanyang Technological University, Singapore. Correspondence to: Junnan Liu <to.liujn@outlook.com>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

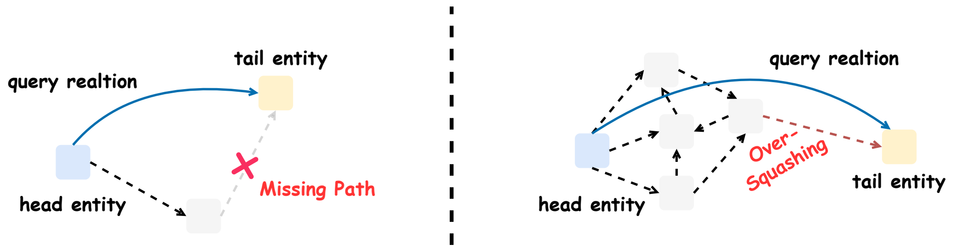


Figure 1. Path-based methods could be limited by the missing paths (Franceschi et al., 2019) and over-squashing information (Alon & Yahav, 2021).

assumption (Koren et al., 2009; Lacroix et al., 2018) is challenging, and some approaches address this by utilizing path and context encoding or pretraining on large-scale KGs as alternatives (Chen et al., 2021; Li et al., 2023).

In this paper, we aim to further investigate the potential benefits of applying the *transformer architecture* (Vaswani et al., 2017) to the task of knowledge graph reasoning. Contrary to previous approaches that utilize pretrained language models for encoding text descriptions, we leverage the attention mechanism within the transformer architecture to capture interactions between any pair of entities. Our principal contribution is the introduction of an expressive and scalable attention mechanism tailored specifically to knowledge graph reasoning. The resulting model, named **KnowFormer**, provides informative representations for knowledge graph reasoning in both transductive and inductive scenarios. It can address the limitations of vanilla path-based methods, such as the path missing and information over-squashing. Specifically:

- We redefine the self-attention mechanism, originally introduced by Vaswani et al. (2017), as a weighted aggregation of pairwise information for specific queries based on the plausibility of entity pairs as query prototypes. This redefinition enables us to perform attention computation on multi-relational KGs and reduce the modeling complexity.
- For constructing the attention mechanism, we introduce two modules that generate informative representations for query, key, and value, respectively. These modules are designed based on relational message passing neural networks, allowing us to consider structural information during attention calculation.
- To improve scalability, we adopt an instance-based similarity measure (Cui & Chen, 2022) to reduce the significant number of degrees of freedom and introduce an approximation method that maintains complexity linearly proportional to entities. Moreover, we provide theoretical guarantees for its stability and expressivity.
- We also provide empirical evidence of the effectiveness of our proposed KnowFormer on knowledge graph reasoning benchmarks in both transductive and inductive settings, surpassing the performance of salient baselines.

2. Related Work

We classify the related work on knowledge graph reasoning into three main paradigms: embedding-based methods, path-based methods, and transformer models.

Embedding-Based Methods. Embedding-based methods aim to learn distributed representations for entities and relations by preserving the triplets in the knowledge graph. Notable early methods include TransE (Bordes et al., 2013), DistMult (Yang et al., 2015) and ComplEX (Trouillon et al., 2017). Subsequent work has focused on improving the score function or embedding space of these methods to enhance the modeling of semantic patterns (Sun et al., 2019; Tang et al., 2020; Zhang et al., 2020; Li et al., 2022). In addition, some researchers have explored the application of neural networks in an encoder-decoder paradigm to obtain adaptive and robust embeddings. Representative methods include convolutional neural networks (Dettmers et al., 2018; Nguyen et al., 2018) and graph neural networks (Schlichtkrull et al., 2018; Vashishth et al., 2020; You et al., 2021). Although embedding methods have demonstrated promising performance in knowledge graph reasoning and scalability on large KGs, these embeddings are hard to generalize to unseen entities and relations, thus limiting their applicability in the inductive setting.

Path-Based Methods. Path-based methods in knowledge graph reasoning have their origins in traditional heuristic similarity approaches, which include measuring the weighted count of paths (Katz, 1953), random walk probability (Page, 1998), and shortest path length (Liben-Nowell & Kleinberg, 2007). In recent years, there have been proposals to employ neural networks to encode paths between entities, such as recurrent neural networks (Neelakantan et al., 2015), and aggregate these representations for reasoning. Another research direction focuses on learning probabilistic logical rules over KGs (Yang et al., 2017; Sadeghian et al., 2019) and utilizing these rules to assign weights to paths. More recent works have achieved state-of-the-art performance by incorporating well-designed message passing neural networks (Wang et al., 2021; Zhu et al., 2021; Zhang & Yao, 2022). These methods enhance expressivity through the use of *target distinguishable* initial function and *relational* message and aggregation functions. PathCon (Wang et al., 2021), for instance, introduces a relational message-passing framework for the KG completion task and achieves impressive results. However, prominent path-based methods still face limitations, including the recognized shortcomings of message-passing neural networks such as incompleteness (Franceschi et al., 2019) and over-squashing (Alon & Yahav, 2021).

Transformers for Knowledge Graph Reasoning. Currently, most research utilizes pretrained models based on transformer architecture to encode textual descriptions and leverage its knowledge and the semantic understanding ability for reasoning. For instance, KG-BERT (Yao et al., 2019) and PKGC (Lv et al., 2022) employ a pretrained language model encoder to represent KG triples with text descriptions, enabling the inference of new facts through a classification layer using a special token representation. In contrast, some studies explore a sequence-to-sequence approach for generating reasoning results (Xie et al., 2022; Saxena et al., 2022). This paradigm has received increasing attention with the emergence of large language models. To bridge the gap between unstructured textual descriptions and structured KGs, researchers have explored incorporating contextual information by sampling and encoding neighborhood triplets or paths (Xie et al., 2022; Chen et al., 2021; Li et al., 2023). However, these methods face limitations due to their heavy reliance on text description, including the lack of textual data, domain knowledge requirements, and the neglect of structural information in KGs. In this paper, we revisit how to perform knowledge graph reasoning leveraging the transformer architecture from the perspective of knowledge graph structure.

3. Preliminary

In this section, we introduce the background knowledge of knowledge graphs and knowledge graph reasoning. Due to the page limitations, more preliminaries about transformers can be found in the appendix.

Knowledge Graph. Typically, a knowledge graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, \mathcal{R}\}$ is a collection of triplets $\mathcal{E} = \{(h_i, r_i, t_i) \mid h_i, t_i \in \mathcal{V}, r_i \in \mathcal{R}\}$ consist a set of entities $\mathcal{V}$ and a set of relations $\mathcal{R}$. Each triplet is a relational edge from head entity h_i to tail entity t_i with the relation r_i. For ease of notation, we can also represent a fact as $r(u, v) \in \mathcal{E}$ where $u, v \in \mathcal{V}$ and $r \in \mathcal{R}$. Additionally, we define the neighborhood set of an entity $u \in \mathcal{V}$ relative to a relation $r \in \mathcal{R}$ as $\mathcal{N}_r(u) = \{v \mid r(u, v) \in \mathcal{E}\}$.

Knowledge Graph Reasoning. Given a knowledge graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, \mathcal{R}\}$, the goal of knowledge graph reasoning is to leverage existing facts to infer the missing elements of the query fact (h, r_q, t), where r_q is a query relation. Based on the type of missing elements, there are three sub-tasks: head reasoning to infer $(?, r_q, t)$, tail reasoning to infer $(h, r_q, ?)$, and relation reasoning to infer $(h, ?, t)$. This paper mainly focuses on the tail reasoning task, as the other tasks can be transformed into the same form.

4. Proposed Method

In this section, we introduce the construction of the KNOWFORMER. Specifically, we focus on how to create the proposed attention mechanism and integrate it into a transformer model.

4.1. Attention Computation of KNOWFORMER

We adopt a modified definition of self-attention, similar to the formulation used in Tsai et al. (2019) and Chen et al. (2022). Assuming that $\boldsymbol{X} \in \mathbb{R}^{n \times d}$ is input features, we have:

$$\mathrm{Attn}(\boldsymbol{x}_u) = \sum_{v \in \mathcal{V}} \frac{\kappa(f_q(\boldsymbol{x}_u), f_q(\boldsymbol{x}_v))}{\sum_{w \in \mathcal{V}} \kappa(f_q(\boldsymbol{x}_u), f_q(\boldsymbol{x}_w))} f_v(\boldsymbol{x}_v), \quad (1)$$

here, $f_q(\cdot)$ represents the query function, $f_v(\cdot)$ represents the value function, and $\kappa(\cdot, \cdot) : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}^+$ is a positive-definite kernel that measures the pairwise similarity.

In a knowledge graph, different relation types $r \in \mathcal{R}$ determine various modes of interaction between entity pairs. Consequently, a straightforward approach is to incorporate relation-specific attention to model these diverse interactions:

$$\mathrm{Attn}(\boldsymbol{x}_u) = \sum_{r \in \mathcal{R}} \sum_{u \in \mathcal{V}} \frac{\kappa(f_q^r(\boldsymbol{x}_u), f_q^r(\boldsymbol{x}_v))}{\sum_{w \in \mathcal{V}} \kappa(f_q^r(\boldsymbol{x}_u), f_q^r(\boldsymbol{x}_w))} f_v^r(\boldsymbol{x}_v), \quad (2)$$

where $f^r(.)$ means a relation-specific function. However, this approach will result in significant computational overhead ($\mathcal{O}(|\mathcal{R}| \cdot |\mathcal{V}|^2)$) and suboptimal optimization due to a high degree of freedom. Inspired by instance-based learning (Stanfill & Waltz, 1986; Cui & Chen, 2022), we consider modeling the pairwise interactions based on the plausibility of entity pairs as query prototypes.

Definition 4.1 (Query Prototype)**.** Given a relation $r \in \mathcal{R}$, entity $u \in \mathcal{V}$ and $v \in \mathcal{V}$ are the prototypes for relation r if $(u, r, ?) \wedge (v, r, ?) \neq \emptyset$.

For instance, `Barack Obama` and `Joseph Biden` are the prototypes for relation `Nationality`, since we have $(\texttt{Barack Obama}, \texttt{Nationality}, ?) \wedge (\texttt{Joseph Biden}, \texttt{Nationality}, ?) = \{\texttt{USA}\}$. Based on Definition 4.1, given query $(h, r_q, ?)$, we propose to calculate attention scores between entity pairs referring to their plausibility as prototypes. Our motivation is that the greater the plausibility of two entities as prototypes, the more *similarity* they exhibit under the current query relation, leading to higher attention scores. In this way, for each query in the dataset, we do not need to consider the interaction of entities for all relations, reducing the complexity to $\mathcal{O}(|\mathcal{V}|^2)$. Our idea considers the plausibility of two entities u and v being prototypes to the query relation r_q and enables information propagation between them based on the plausibility. To achieve this, we propose to utilize the function $f_q(\cdot)$ to obtain informative representations of entities to distinguish query prototypes, the function $\kappa(\cdot,\cdot)$ to serve as the score function for measuring plausibility, and lastly, $f_v(\cdot)$ to generate propagated information. Next, we will provide a detailed explanation of constructing the attention mechanism by defining the functions $f_q(\cdot)$, $f_v(\cdot)$, and $\kappa(\cdot,\cdot)$.

Query Function. Query function $f_q(\cdot)$ is designed to provide informative representations, denoted as $\widetilde{\boldsymbol{z}}_u$, for each entity u to distinguish query prototypes effectively. A key insight is that considering the context of each entity on the knowledge graph (*i.e.*, its k-hop neighbor facts) is essential in determining query prototypes. For example, entities such as `Barack Obama` and `Joseph Biden` can serve as prototypes for the relation `Nationality` due to their common neighbor relation `President of`. We introduce Q-RMPNN, a relational message-passing neural network (Gilmer et al., 2017; Xu et al., 2019; Wang et al., 2021) designed to incorporate neighbor facts into the entity representations. Q-RMPNN consists of two stages: (1) each entity sends relational messages to its neighbors; (2) each entity aggregates the received relational messages and updates its representation. Drawing inspiration from knowledge graph embeddings, we generate a relational message $\boldsymbol{m}_{v|u,r}$ for each fact $r(u, v)$ by maximizing its continuous truth value (Wang et al., 2023):

$$\boldsymbol{m}_{v|u,r} = \arg\max_{\boldsymbol{w}\in\mathcal{D}} \phi(\boldsymbol{z}_u, \hat{\boldsymbol{r}}, \boldsymbol{w}) = g(\boldsymbol{z}_u, \hat{\boldsymbol{r}}), \tag{3}$$

where $\mathcal{D}$ indicates the range domain for the entity embeddings and $\boldsymbol{z}_u$ is the representation of entity u within $f_q(\cdot)$, $\hat{\boldsymbol{r}}$ is the representation of relation r conditioned on query relation r_q, $\phi(\cdot,\cdot,\cdot)$ is a score function and $g(\cdot,\cdot)$ is an estimation function. In this paper, we adopt the DistMult method (Yang et al., 2015) as the foundation for our implementation similar to Zhu et al. (2021), to say:

$$\begin{aligned} &\hat{\boldsymbol{r}} = \boldsymbol{R}[r_q]\cdot \boldsymbol{W}_r + \boldsymbol{b}_r,\ g(\boldsymbol{z}_u, \hat{\boldsymbol{r}}) = \boldsymbol{z}_u \odot \hat{\boldsymbol{r}}, \\ &\phi(\boldsymbol{z}_u, \hat{\boldsymbol{r}}, \boldsymbol{w}) = \langle g(\boldsymbol{z}_u, \hat{\boldsymbol{r}}), \boldsymbol{w}\rangle, \end{aligned} \tag{4}$$

where $\boldsymbol{R}$ is the input features of relation set $\mathcal{R}$, and $\boldsymbol{W}_r, \boldsymbol{b}_r$ is the relation-specific parameters. At the l-th layer of $f_q(\cdot)$, we fuse $\boldsymbol{z}_u^{(l)}$ for each $u \in \mathcal{V}$ by the summation of aggregated information from the $(l-1)$-th layer:

$$\begin{aligned} &\boldsymbol{z}_u^{(0)} \leftarrow [\boldsymbol{x}_u, \boldsymbol{\epsilon}], \\ &\boldsymbol{z}_u^{(l)} \leftarrow \Phi^{(l)}\left(\boldsymbol{\alpha}^{(l)}\cdot \boldsymbol{z}_u^{(l-1)} + \sum_{r(v,u)\in\mathcal{E}} \boldsymbol{m}_{u|v,r}^{(l-1)}\right), \end{aligned} \tag{5}$$

where $\boldsymbol{x}_u \in \mathbb{R}^d$ represents the input features, $\boldsymbol{\epsilon} \sim N(\boldsymbol{0}, \boldsymbol{I})$ denotes Gaussian noise to distinguish different source entities, $[\cdot,\cdot]$ indicates concatenate function, $\boldsymbol{\alpha}^{(l)}$ captures layer-specific parameters to retain the original information, and $\boldsymbol{\Phi}^{(l)}(\cdot)$ represents a layer-specific update function parameterized by an MLP network. After $\widetilde{L}$ layers of updates, we obtain the final representation $\widetilde{\boldsymbol{z}}_u$ given by $\widetilde{\boldsymbol{z}}_u = \boldsymbol{z}_u^{(\widetilde{L})}$. Through Q-RMPNN, $\widetilde{\boldsymbol{z}}_u$ is optimized to capture $\widetilde{L}$-hop neighbor facts, resulting in a structure-aware and knowledge-oriented representation.

Value Function. We leverage the value function to generate pairwise representations $\hat{\boldsymbol{z}}_u$ conditioned on the query relation r_q for the query $(h, r_q, ?)$. The structural information between node pairs is crucial in knowledge graph reasoning, which can be viewed as a link prediction task (Zhang et al., 2021). However, it is challenging for the attention mechanism to capture the structural information of the input graph explicitly. To address this, we design V-RMPNN to encode pairwise structural information into the value. Specifically, V-RMPNN is implemented as follows:

$$\begin{aligned} &\boldsymbol{z}_{u|h,r_q}^{(0)} \leftarrow [\boldsymbol{x}_u, \mathbb{I}_{u=h}\odot \boldsymbol{1}], \\ &\boldsymbol{z}_{u|h,r_q}^{(l)} \leftarrow \Psi^{(l)}\left(\boldsymbol{\beta}^{(l)}\cdot \boldsymbol{z}_{u|h,r_q}^{(l-1)} + \sum_{r(v,u)\in\mathcal{E}} \boldsymbol{m}_{u|v,r}^{(l-1)}\right), \end{aligned} \tag{6}$$

where $\boldsymbol{x}_u$, $\boldsymbol{m}_{u|v,r}^{(l-1)}$, $\boldsymbol{\beta}^{(l)}$, $[\cdot,\cdot]$ and $\boldsymbol{\Psi}^{(l)}(\cdot)$ are defined similarly to Q-RMPNN. The term $\mathbb{I}_{u=h}\odot \boldsymbol{1}$ represents head entity labeling features to enhance node representation, which is essential for the link prediction task (Zhang et al., 2021; You et al., 2021). After $\widehat{L}$ layers of updates, the final representation $\hat{\boldsymbol{z}}_u$ is obtained as $\hat{\boldsymbol{z}}_u = \boldsymbol{z}_{u|h,r_q}^{(\widehat{L})}$.

Kernel Function. The kernel function $\kappa(\widetilde{\boldsymbol{z}}_u, \widetilde{\boldsymbol{z}}_v)$ is employed to quantify pairwise similarity. One common choice is the exponential kernel specified as follows:

$$\kappa_{\text{exp}}(\widetilde{\boldsymbol{z}}_u, \widetilde{\boldsymbol{z}}_v) = \exp\left(\langle \widetilde{\boldsymbol{z}}_u \boldsymbol{W}_1, \widetilde{\boldsymbol{z}}_v \boldsymbol{W}_2\rangle/\sqrt{d}\right), \tag{7}$$

where $\boldsymbol{W}_1$ and $\boldsymbol{W}_2$ are the linear projection matrixes and the bias is omitted for convenience. However, the *dot-then-exponentiate operation* will lead to quadratic complexity

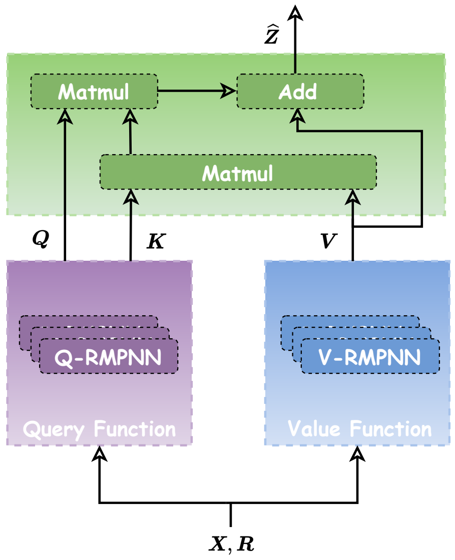


Figure 2. Overview of the proposed attention mechanism, which takes entity features X and R as input and outputs $\widehat{Z}$ for all $u \in \mathcal{V}$.

since we must calculate the inner product of the query matrix with the key matrix before the exponentiate function. In practice, we approximate the exponential function $\exp(\cdot)$ using the *first-order Taylor expansion* around zero (Li et al., 2020; Wu et al., 2023a; Dass et al., 2023), resulting in:

$$\kappa_{\text{exp}}(\tilde{z}_u, \tilde{z}_v) \approx 1 + \langle \tilde{z}_u W_1, \tilde{z}_v W_2 \rangle. \tag{8}$$

To ensure non-negativity of the attention scores, we further normalize the inputs by their Frobenius norm, leading to the kernel function implementation as follows:

$$\kappa(\tilde{z}_u, \tilde{z}_v) = 1 + \left\langle \frac{\tilde{z}_u W_1}{\|\tilde{z}_u W_1\|_{\mathcal{F}}}, \frac{\tilde{z}_v W_2}{\|\tilde{z}_v W_2\|_{\mathcal{F}}} \right\rangle. \tag{9}$$

Based on Equation (8) and Equation (9), we can initially compute the inner product of the key matrix with the value matrix, followed by calculating the inner product of the query matrix with the resultant matrix as shown in Figure 2, thereby reducing the complexity to linear.

Theorem 4.2 (Approximation Error for Exponential Kernel)**.** *For each $u, v \in \mathcal{V}$, the approximation error $\Delta = |\kappa(\tilde{z}_u, \tilde{z}_v) - \kappa_{exp}(\tilde{z}_u, \tilde{z}_v)|$ will be bounded by $\mathcal{O}(e^{\gamma}/2)$, where $\gamma \in (0, 1)$.*

We provide the proof in Appendix B. Theorem 4.2 demonstrates that the approximation error remains bounded regardless of the size of KGs, which implies Equation (8) is a stable estimation.

Summary. Now, we present the detailed computational process of the proposed attention mechanism, as illustrated in Figure 2. Assume $X \in \mathbb{R}^{|\mathcal{V}| \times d}$ is the input feature matrix of entities, $R \in \mathbb{R}^{|\mathcal{R}| \times d}$ is the input features matrix of relations, $f_q(\cdot)$ and $f_v(\cdot)$ are the query and value function, and $\kappa(\cdot, \cdot)$ is the kernel function. We firstly obtain $\widetilde{Z} = [f_q(x_u)]_{u \in \mathcal{V}}$ and $\widehat{Z} = [f_v(x_u)]_{u \in \mathcal{V}}$ by iteratively applying Equation (5) and Equation (6). Then the output feature matrix $\overline{Z}$ can be calculated as follows:

$$\begin{aligned}
Q &= \frac{\widetilde{Z} W_1}{\|\widetilde{Z} W_1\|_{\mathcal{F}}}, \quad K = \frac{\widetilde{Z} W_2}{\|\widetilde{Z} W_2\|_{\mathcal{F}}}, \quad V = \widehat{Z}, \\
D &= \text{diag}\left(\mathbf{1} + \frac{Q\left(K^T \mathbf{1}\right) + |\mathcal{V}|}{|\mathcal{V}|}\right), \\
\overline{Z} &= D^{-1}\left[V + \frac{\mathbf{1}^T V + Q\left(K^T V\right)}{|\mathcal{V}|}\right],
\end{aligned} \tag{10}$$

where we omit the bias and $\text{diag}(\cdot)$ means a transformation to diagonal matrix. We further show the overall procedure in Algorithm 1. We can observe that the time complexity of Equation (10) is linearly related to $|\mathcal{V}|$ and $|\mathcal{E}|$, which is scalable to massive KGs. We will discuss the details of complexity in Section 4.3.

4.2. Overall Architecture of KNOWFORMER

Based on the attention definition provided in Section 4.1, we complete the construction of our proposed model KNOWFORMER. The remaining components of KNOWFORMER adhere to the standard transformer architecture described in Vaswani et al. (2017). Each layer of KNOWFORMER comprises an attention function and a feedforward network. In addition, a skip-connection is applied after the attention function, and normalization layers are inserted before and after the feedforward network. Finally, we stack L layers to construct KNOWFORMER.

At the start of training, we initialize the entity feature matrix X and relation feature matrix R. Specifically, X is set to an all-zero vector, while R is randomly initialized and is learnable. We then iteratively calculate the entity representations as follows:

$$\begin{aligned}
A^{(l)} &= \text{LayerNorm}_1^{(l)}\left(X^{(l-1)} + \text{Attn}^{(l)}\left(X^{(l-1)}, R\right)\right), \\
X^{(l)} &= \text{LayerNorm}_2^{(l)}\left(A^{(l)} + \text{FFN}^{(l)}(A^{(l)})\right).
\end{aligned} \tag{11}$$

Lastly, we obtain the final representation of entities $X^{(L)}$. We update model parameters by optimizing a negative sampling loss (Mikolov et al., 2013; Sun et al., 2019) using Adam optimizer (Kingma & Ba, 2015). The loss function is defined for each training fact (h, r, t) as:

$$\mathcal{L} = -\log(\sigma(t|h, r)) - \sum_{t'} \log\left(1 - \sigma(t'|h, r)\right), \tag{12}$$

Algorithm 1 Attention Computation

```
input knowledge graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, \mathcal{R}\}$, query $(h, r_q, ?)$, entity features $\boldsymbol{X} = [\boldsymbol{x}_u]_{u\in\mathcal{V}}$, relation features $\boldsymbol{R} = [\boldsymbol{r}_p]_{p\in\mathbf{R}}$, and model parameters include $\widetilde{\boldsymbol{W}}, \tilde{\boldsymbol{b}}, \Phi, \boldsymbol{\alpha}, \widehat{\boldsymbol{W}}, \hat{\boldsymbol{b}}, \Psi, \boldsymbol{\beta}, \boldsymbol{W}_1$, and $\boldsymbol{W}_2$;
output output features $\overline{\boldsymbol{Z}}$;
1:  for $l = 0$ to $\widetilde{L}$ do
2:    if $l = 0$ then
3:      $\tilde{\boldsymbol{z}}_u^{(0)} \leftarrow [\boldsymbol{x}_u, \epsilon]$;
4:    else
5:      $\tilde{\boldsymbol{h}}_v \leftarrow \tilde{\boldsymbol{z}}_v^{(l-1)} \odot \left(\boldsymbol{r}_q \cdot \widetilde{\boldsymbol{W}}_r + \tilde{b}_r\right)$;
6:      $\tilde{\boldsymbol{m}}_{u|v,r}^{(l-1)} \leftarrow \arg\max_{\boldsymbol{w}\in\mathcal{D}} \left\langle \tilde{\boldsymbol{h}}_v, \boldsymbol{w} \right\rangle$;
7:      $\tilde{\boldsymbol{z}}_u^{(l)} \leftarrow \Phi^{(l)} \left(\boldsymbol{\alpha}^{(l)} \cdot \tilde{\boldsymbol{z}}_u^{(l-1)} + \sum_{r(v,u)\in\mathcal{E}} \tilde{\boldsymbol{m}}_{u|v,r}^{(l-1)}\right)$;
8:    end if
9:    ▷ Iteratively compute $\widetilde{\boldsymbol{Z}}$ according to Equation (5) taking $\boldsymbol{X}$ and $\boldsymbol{r}_q$ as input;
10: end for
11: for $l = 0$ to $\widehat{L}$ do
12:   if $l = 0$ then
13:     $\hat{\boldsymbol{z}}_u^{(0)} \leftarrow [\boldsymbol{x}_u, \mathbb{I}_{u=h} \odot \mathbf{1}]$;
14:   else
15:     $\hat{\boldsymbol{h}}_v \leftarrow \hat{\boldsymbol{z}}_v^{(l-1)} \odot \left(\boldsymbol{r}_q \cdot \widehat{\boldsymbol{W}}_r + \hat{b}_r\right)$;
16:     $\hat{\boldsymbol{m}}_{u|v,r}^{(l-1)} \leftarrow \arg\max_{\boldsymbol{w}\in\mathcal{D}} \left\langle \hat{\boldsymbol{h}}_v, \boldsymbol{w} \right\rangle$;
17:     $\hat{\boldsymbol{z}}_u^{(l)} \leftarrow \Psi^{(l)} \left(\boldsymbol{\beta}^{(l)} \cdot \hat{\boldsymbol{z}}_u^{(l-1)} + \sum_{r(v,u)\in\mathcal{E}} \hat{\boldsymbol{m}}_{u|v,r}^{(l-1)}\right)$;
18:   end if
19:   ▷ Iteratively compute $\widehat{\boldsymbol{Z}}$ according to Equation (6) taking $\boldsymbol{X}$, $\boldsymbol{r}_q$, and $h$ as input;
20: end for
21: Obtain $\boldsymbol{Q}$, $\boldsymbol{K}$ and $\boldsymbol{V}$ by $\boldsymbol{Q} \leftarrow \frac{\widetilde{\boldsymbol{Z}}\boldsymbol{W}_1}{\|\widetilde{\boldsymbol{Z}}\boldsymbol{W}_1\|_{\mathcal{F}}}$, $\boldsymbol{K} \leftarrow \frac{\widetilde{\boldsymbol{Z}}\boldsymbol{W}_2}{\|\widetilde{\boldsymbol{Z}}\boldsymbol{W}_2\|_{\mathcal{F}}}$ and $\boldsymbol{V} \leftarrow \widehat{\boldsymbol{Z}}$;
22: $\boldsymbol{D} \leftarrow \mathrm{diag}\left(\mathbf{1} + \frac{\boldsymbol{Q}(\boldsymbol{K}^T\mathbf{1}) + |\mathcal{V}|}{|\mathcal{V}|}\right)$;
23: $\overline{\boldsymbol{Z}} \leftarrow \boldsymbol{D}^{-1}\left[\boldsymbol{V} + \frac{\mathbf{1}^T\boldsymbol{V} + \boldsymbol{Q}(\boldsymbol{K}^T\boldsymbol{V})}{|\mathcal{V}|}\right]$; ▷ kernel function computation
```

where $\sigma(\cdot|h, r)$ represents the score of a candidate fact, computed by a multilayer perceptron (MLP) and a sigmoid function using the output feature $\boldsymbol{X}^{(L)}$ and $t' \in \mathcal{V} \setminus \{t\}$ indicates the negative samples.

4.3. Discussion

In this section, we provide a thorough discussion of the KNOWFORMER, focusing on the analysis of its time complexity and expressivity.

Time Complexity. Time complexity of our proposed model primarily depends on the attention function. We can break down the time complexity into the query function, the value function, and the kernel function. The query function is called L times, with each call taking $\mathcal{O}(\widetilde{L}(|\mathcal{E}|d + |\mathcal{V}|d^2))$. Similarly, the value function is called L times, with a single call taking $\mathcal{O}(\widehat{L}(|\mathcal{E}|d + |\mathcal{V}|d^2))$. Finally, the kernel function is called L times, with each call taking $\mathcal{O}(|\mathcal{V}|d^2)$. Therefore, the total complexity amounts to $\mathcal{O}(L(\widetilde{L}+\widehat{L})(|\mathcal{E}|d+|\mathcal{V}|d^2))$. The above conclusion shows that the time complexity of our proposed method is linearly related to the number of facts and entities in the knowledge graph, exhibiting better scalability.

Expressivity Analysis. Huang et al. (2023) introduced a variant of the Weisfeiler-Leman Test (Weisfeiler & Leman, 1968; Xu et al., 2019; Barceló et al., 2022) called the *Relational Asymmetric Local 2-WL Test* (RA−WL$_2$) to evaluate the expressive power of message-passing networks in the knowledge graph reasoning task. We formally demonstrate the expressivity of KNOWFORMER based on RA−WL$_2$. This is stated in the following theorem:

Theorem 4.3 (Expressivity of KNOWFORMER)**.** *Assuming the estimated graph by the attention layer of* KNOWFORMER *is $\tilde{\mathcal{E}}$, the attention layer of* KNOWFORMER *can achieve at least the same level of expressive ability as RA−WL$_2$ on the extended graph $\mathcal{E} \cup \tilde{\mathcal{E}}$.*

We provide proof in Appendix D. The estimated graph $\tilde{\mathcal{E}}$ can be viewed as a collection of facts of special relations determining the equivalence between entity pairs for a specific query. KNOWFORMER exhibits stronger expressive power due to its ability to propagate information on an extended graph, distinguishing it from vanilla path-based methods.

5. Experimental Results

In this section, we conduct empirical studies motivated by the following aspects:

- **Transductive Performance.** As a general knowledge graph reasoning task, we aim to demonstrate the performance of KNOWFORMER in transductive knowledge graph reasoning tasks.
- **Inductive Performance.** Our proposed model supports reasoning in inductive settings. How does KNOWFORMER perform in inductive knowledge graph reasoning tasks?
- **Ablation Study.** How important are the various components within our proposed framework? For instance, how does model performance change if we omit the query function?
- **Further Experiments.** We conduct additional experiments to further showcase the effectiveness of KNOWFORMER. For example, does KNOWFORMER perform

Method	FB15k-237			WN18RR			NELL-995			YAGO3-10		
	MRR	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10
TransE (Bordes et al., 2013)	0.330	23.2	52.6	0.222	1.4	52.8	0.507	42.4	64.8	0.510	41.3	68.1
DistMult (Yang et al., 2015)	0.358	26.4	55.0	0.455	41.0	54.4	0.510	43.8	63.6	0.566	49.1	70.4
RotatE (Sun et al., 2019)	0.337	24.1	53.3	0.477	42.8	57.1	0.508	44.8	60.8	0.495	40.2	67.0
HousE (Li et al., 2022)	0.361	26.6	55.1	0.511	46.5	60.2	0.519	45.8	61.8	0.571	49.1	$\underline{71.4}$
DRUM (Sadeghian et al., 2019)	0.343	25.5	51.6	0.486	42.5	58.6	0.532	46.0	$\underline{66.2}$	0.531	45.3	67.6
CompGCN (Vashishth et al., 2020)	0.355	26.4	53.5	0.479	44.3	54.6	0.463	38.3	59.6	0.421	39.2	57.7
RNNLogic (Qu et al., 2021)	0.344	25.2	53.0	0.483	44.6	55.8	0.516	46.3	57.8	0.554	50.9	62.2
NBFNet (Zhu et al., 2021)	0.415	32.1	$\underline{59.9}$	0.551	49.7	66.6	0.525	45.1	63.9	0.563	48.0	70.8
RED-GNN (Zhang & Yao, 2022)	0.374	28.3	55.8	0.533	48.5	62.4	0.543	47.6	65.1	0.556	48.3	68.9
A*Net (Zhu et al., 2023)	0.411	32.1	58.6	0.549	49.5	65.9	0.521	44.7	63.1	0.556	47.0	70.7
AdaProp (Zhang et al., 2023)	$\underline{0.417}$	$\underline{33.1}$	58.5	0.562	49.9	67.1	$\underline{0.554}$	$\underline{49.3}$	65.5	$\underline{0.573}$	$\underline{51.0}$	68.5
Ultra (Galkin et al., 2024)	0.368	-	56.4	0.480	-	61.4	0.509	-	66.0	0.557	-	71.0
HittER (Chen et al., 2021)	0.373	27.9	55.8	0.503	46.2	58.4	-	-	-	-	-	-
SimKGC (Wang et al., 2022b)	0.336	24.9	51.1	**0.666**	**58.5**	**80.0**	0.501	42.6	65.3	0.211	14.1	35.1
KGT5 (Saxena et al., 2022)	0.276	21.0	41.4	0.508	48.7	54.4	-	-	-	0.426	36.8	52.8
KnowFormer	**0.430**	**34.3**	**60.8**	$\underline{0.579}$	$\underline{52.8}$	$\underline{68.7}$	**0.566**	**50.2**	**67.5**	**0.615**	**54.7**	**73.4**

Table 1. Transductive knowledge graph reasoning performance for 4 different datasets. The best results are **boldfaced** and the second-best results are underlined. Our proposed model, KnowFormer, achieves SOTA performance in most cases marked by **tan**.

better for longer paths?

Our code is available at https://github.com/jnanliu/KnowFormer.

5.1. Transductive Performance

Datasets. We conduct experiments on four widely utilized transductive knowledge graph reasoning datasets: FB15k-237 (Toutanova & Chen, 2015), WN18RR (Dettmers et al., 2018), NELL-995 (Xiong et al., 2017) and YAGO3-10 (Mahdisoltani et al., 2015). For consistency, we utilize the same data splits as in prior studies (Zhu et al., 2021; Zhang et al., 2023).

Baselines. We compare KnowFormer to several prominent baselines, categorized into three classes:

- *Embedding-based methods*, including TransE (Bordes et al., 2013), DistMult (Yang et al., 2015), RotatE (Sun et al., 2019), and HousE (Li et al., 2022). These methods learn embeddings for entities and relations and perform reasoning based on distance or similarity.
- *Path-based methods*, including DRUM (Sadeghian et al., 2019), CompGCN (Vashishth et al., 2020), RNNLogic (Qu et al., 2021), NBFNet (Zhu et al., 2021), RED-GNN (Zhang & Yao, 2022), A*Net (Zhu et al., 2023), AdaProp (Zhang et al., 2023), and Ultra (Galkin et al., 2024). These methods conduct reasoning by utilizing the path information connecting entities.
- *Text-based methods*, including HittER (Chen et al., 2021), SimKGC (Wang et al., 2022b), and KGT5 (Saxena et al., 2022). These methods utilize textual information for reasoning in knowledge graphs.

Results and Analysis. We evaluate the performance of reasoning using standard metrics (Wang et al., 2017; Ji et al., 2022), namely MRR (↑), Hit@1 (↑), and Hit@10 (↑). The results on various datasets are presented in Table 1. Across all metrics, KnowFormer demonstrates a substantial performance advantage over the best baseline method in FB15k-237, NELL-995, and YAGO3-10. Particularly in the large-scale YAGO3-10 dataset, KnowFormer exhibits a significant performance advantage over the best baseline methods. This highlights the high effectiveness of KnowFormer in transductive knowledge graph reasoning. In the WN18RR dataset, KnowFormer achieves the second-best performance among all baselines. Notably, SimKGC (Wang et al., 2022b) performs exceptionally well in this dataset, which can be attributed to its ability to acquire knowledge from extensive pretrained data, allowing it to capture the semantic relationships in WN18RR as a subset of a comprehensive English lexical database. Conversely, methods like SimKGC that rely on pretrained language models tend to underperform in datasets that require domain-specific knowledge. In contrast, our proposed text-free method relies solely on the structure of the knowledge graphs, resulting in enhanced robustness and scalability.

5.2. Inductive Performance

Inductive reasoning has become a prominent research topic (Hamilton et al., 2017; Teru et al., 2020) due to the ubiquitous occurrence of emergent entities in real-world applications (Zhang & Chen, 2020). In this part, we will show the performance of KnowFormer on the inductive knowl-

Method	v1			v2			v3			v4		
	MRR	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10
FB15k-237												
DRUM (Sadeghian et al., 2019)	0.333	24.7	47.4	0.395	28.4	59.5	0.402	30.8	57.1	0.410	30.9	59.3
NBFNet (Zhu et al., 2021)	0.442	33.5	57.4	0.514	42.1	68.5	0.476	38.4	63.7	0.453	36.0	62.7
RED-GNN (Zhang & Yao, 2022)	0.369	30.2	48.3	0.469	38.1	62.9	0.445	35.1	50.3	0.442	34.0	62.1
A*Net (Zhu et al., 2023)	0.457	**38.1**	58.9	0.510	41.9	67.2	0.476	38.9	62.9	0.466	36.5	64.5
AdaProp (Zhang et al., 2023)	0.310	19.1	55.1	0.471	37.2	65.9	0.471	37.7	63.7	0.454	35.3	63.8
InGram (Lee et al., 2023)	0.293	16.7	49.3	0.274	16.3	48.2	0.233	14.0	40.8	0.214	11.4	39.7
KnowFormer	**0.466**	37.8	**60.6**	**0.532**	**43.3**	**70.3**	**0.494**	**40.0**	**65.9**	**0.480**	**38.3**	**65.3**
WN18RR												
DRUM (Sadeghian et al., 2019)	0.666	61.3	77.7	0.646	59.5	74.7	0.380	33.0	47.7	0.627	58.6	70.2
NBFNet (Zhu et al., 2021)	0.741	69.5	**82.6**	0.704	65.1	79.8	0.452	39.2	56.8	0.641	60.8	69.4
RED-GNN (Zhang & Yao, 2022)	0.701	65.3	79.9	0.690	63.3	78.0	0.427	36.8	52.4	0.651	60.6	72.1
A*Net (Zhu et al., 2023)	0.727	68.2	81.0	0.704	64.9	80.3	0.441	38.6	54.4	0.661	**61.6**	74.3
AdaProp (Zhang et al., 2023)	0.733	66.8	80.6	**0.715**	64.2	**82.6**	**0.474**	39.6	**58.8**	**0.662**	61.1	**75.5**
InGram (Lee et al., 2023)	0.277	13.0	60.6	0.236	11.2	48.0	0.230	11.6	46.6	0.118	4.1	25.9
SimKGC (Wang et al., 2022b)	0.315	19.2	56.7	0.378	23.9	65.0	0.303	18.6	54.3	0.308	17.5	57.7
KnowFormer	**0.752**	**71.5**	81.9	0.709	**65.6**	81.7	0.467	**40.6**	57.1	0.646	60.9	72.7
NELL-995												
NBFNet (Zhu et al., 2021)	0.584	50.0	79.5	0.410	27.1	63.5	0.425	26.2	60.6	0.287	25.3	59.1
RED-GNN (Zhang & Yao, 2022)	0.637	52.2	86.6	0.419	31.9	60.1	0.436	34.5	59.4	0.363	25.9	**60.7**
AdaProp (Zhang et al., 2023)	0.644	52.2	88.6	0.452	34.4	65.2	0.435	33.7	61.8	0.366	24.7	**60.7**
InGram (Lee et al., 2023)	0.697	57.5	86.5	0.358	25.3	59.6	0.308	19.9	50.9	0.221	12.4	44.0
KnowFormer	**0.827**	**77.0**	**93.0**	**0.465**	**35.7**	**65.7**	**0.478**	**37.8**	**65.7**	**0.378**	**26.7**	59.8

Table 2. Inductive knowledge graph reasoning performance for 3 different datasets and 12 different versions. For each version, the best results are **boldfaced** and the second-best results are underlined. Our proposed model, KnowFormer is marked by **tan**.

edge graph reasoning task. Note that in this paper, we focus on the *semi-inductive* task, which has an invariant relation set. However, we believe that our method can be adapted to handle full-inductive tasks with slight modifications. We leave this aspect as future work to address.

Datasets. In line with Teru et al. (2020), we employ the same data divisions of FB15k-237 (Toutanova & Chen, 2015), WN18RR (Dettmers et al., 2018), and NELL-995 (Xiong et al., 2017). Each of these divisions comprises 4 versions, resulting in a total of 12 subsets. It is worth noting that each subset has a unique separation between the training and test sets. Specifically, the training and test sets within each subset have unique sets of entities while sharing the same set of relations.

Baselines. We compare KnowFormer with 7 baseline methods for inductive knowledge graph reasoning, including DRUM (Sadeghian et al., 2019), NBFNet (Zhu et al., 2021), RED-GNN (Zhang & Yao, 2022), A*Net (Zhu et al., 2023), AdaProp (Zhang et al., 2023), SimKGC (Wang et al., 2022b), and InGram (Lee et al., 2023). Note that Ultra (Galkin et al., 2024) is not included as a baseline model, since we consider Ultra to be a distinctive method based on the pre-training and fine-tuning paradigm, which can benefit from a large-scale amount of training data. However, our method still achieves comparable results to Ultra on some datasets. Please refer to the experimental results in Appendix F.1 for more details.

Results and Analysis. We also evaluate the performance by standard metrics (Wang et al., 2017; Ji et al., 2022), including MRR (↑), Hit@1 (↑), and Hit@10 (↑). Table 2 showcases the inductive performance of KnowFormer and baselines. We have observed that KnowFormer consistently achieves the highest performance on the majority of metrics across all versions of inductive datasets. Furthermore, our method also produces competitive results for the remaining metrics. Compared to transductive settings, KnowFormer demonstrates a relatively small performance gap. This can be attributed to the limited size of inductive datasets. Path-based methods can attain superior performance by utilizing shorter paths and avoiding excessive compression of information. However, KnowFormer maintains a performance advantage, indicating its effectiveness. Another noteworthy observation is the inadequate performance of SimKGC (Wang et al., 2022b) in inductive reasoning. This suggests a potential necessity for text-based and pretrained language model methods to have access to larger amounts of training data to converge

Method	MRR	H@1	H@10
	FB15k-237		
KNOWFORMER	**0.430**	**34.3**	**60.8**
w/o attention	0.417	32.8	58.8
w/o query function	0.422	33.0	59.2
w/o value function	0.367	30.7	48.7
RF-based kernel function	0.419	32.9	57.6
	UMLS		
KNOWFORMER	0.971	**95.8**	**99.8**
full-exponential kernel function	**0.974**	95.2	99.5

Table 3. Ablation study of KNOWFORMER and its variants on transductive FB15k-237 dataset and UMLS dataset. We mark the vanilla KNOWFORMER by **tan**. The best performance is **boldfaced**.

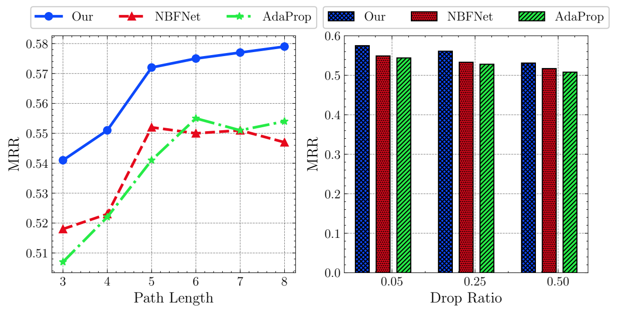


Figure 3. Experimental results on WN18RR. In the left chart, we evaluate the performance of all methods under different lengths of reasoning paths. In the right chart, we randomly drop some facts from the test data and report the performance of all methods.

and attain improved results.

5.3. Ablation Study

In this part, we aim to evaluate the effectiveness of the proposed attention mechanism in KNOWFORMER. We conduct comparisons against several variations, including: (1) removing attention, which results in KNOWFORMER degenerating into vanilla path-based methods such as NBFNet (Zhu et al., 2021); (2) excluding the query function; (3) excluding the value function; (4) substituting the kernel function with an approximation method based on random features (Sinha & Duchi, 2016; Liu et al., 2022; Wu et al., 2022); (5) substituting the kernel function with full-exponential kernel function.

The results of the ablation study on FB15k-237 are presented in Table 3. We can initially observe a decline in model performance when attention is removed, underscoring the importance of the proposed attention mechanism. The performance decrease caused by omitting the query function is relatively limited, suggesting that the input of the attention layer retains some structural information after the update of previous layers. Furthermore, omitting the value function leads to a significant decrease in model performance, as the explicit integration of structural information is crucial for reasoning tasks in knowledge graphs. In contrast, the pure transformer model faces difficulties in achieving this, highlighting the necessity of structural-aware modules. Lastly, a comparison with RF-based methods demonstrates the effectiveness of our kernel function.

To compare the performance between our proposed approximation kernel and the original full-exponential kernel, we conduct an additional ablation study on a small dataset UMLS (Kok & Domingos, 2007) in Table 6 due to the computation overhead. The minimal variation in the performance of both variants indicates the effectiveness of the approximate kernel. The full exponential kernel has a slight performance advantage, yet the quadratic computational complexity it brings is unacceptable in knowledge graph reasoning.

5.4. Further Analysis

In this section, we present additional experiments to demonstrate the superiority of KNOWFORMER over vanilla path-based methods (Zhu et al., 2021; Zhang & Yao, 2022). Specifically, we concentrate on two folds: (1) comparing methods in scenarios with gradually growing reasoning paths, and (2) comparing methods in scenarios with missing reasoning paths, where we randomly drop paths between pairs of entities in the test data. The experimental results on the WN18RR dataset are illustrated in Figure 3.

Performance w.r.t. Longer Paths. The individual performance of KNOWFORMER remained unaffected even as the reasoning path extended, validating its capacity to mitigate information over-squashing resulting from prolonged entity interactions. This finding is consistent with the results in Alon & Yahav (2021).

Performance w.r.t. Missing Paths. As the proportion of missing paths increases, KNOWFORMER exhibits stronger robustness compared to the baseline methods, which show a noticeable performance decline. This demonstrates that the presence of all-pair interactions in KNOWFORMER enables it to better handle missing paths.

6. Conclusion

This paper proposes a novel transformer-based method KNOWFORMER for knowledge graph reasoning. KNOWFORMER consists of an expressive and scalable attention mechanism. Specifically, we introduce message-passing neural network based query function and value function to obtain informative key and value representations. Additionally, we present an efficient attention computation method to enhance the scalability of KNOWFORMER. Experimental results show that KNOWFORMER outperforms salient baselines on both transductive and inductive benchmarks.

Acknowledgements

We appreciate all anonymous reviewers for their careful review and valuable comments. The authors of this paper are supported by the National Natural Science Foundation of China through grant No.62225202.

Impact Statement

This paper introduces KNOWFORMER, a novel method for knowledge graph reasoning. Knowledge graph reasoning is widely applied in various fields, including personalized recommendations, drug discovery, and enterprise-level knowledge management decisions. Each of these applications has the potential to generate significant societal impacts. Personalized recommendations, for instance, can enhance user experience, but also raise concerns about privacy breaches. For example, in a social platform, users can provide personalized friend recommendations based on their personal information, but this information is also easy to cause privacy disclosure. Similarly, drug discovery contributes to advancements in the field of biomedicine, but it also introduces risks associated with medical accidents. We encourage researchers to further investigate and address concerns related to privacy risks and errors that may arise in knowledge graph reasoning.

References

Abujabal, A., Roy, R. S., Yahya, M., and Weikum, G. Never-ending learning for open-domain question answering over knowledge bases. In *WWW*, pp. 1053–1062. ACM, 2018. 1

Alon, U. and Yahav, E. On the bottleneck of graph neural networks and its practical implications. In *ICLR*. OpenReview.net, 2021. 1, 2, 3, 9

Ba, L. J., Kiros, J. R., and Hinton, G. E. Layer normalization. *CoRR*, abs/1607.06450, 2016. 15

Barceló, P., Galkin, M., Morris, C., and Orth, M. A. R. Weisfeiler and leman go relational. In *LoG*, volume 198 of *Proceedings of Machine Learning Research*, pp. 46. PMLR, 2022. 6, 16

Bordes, A., Usunier, N., García-Durán, A., Weston, J., and Yakhnenko, O. Translating embeddings for modeling multi-relational data. In *NeurIPS*, pp. 2787–2795, 2013. 1, 2, 7, 19

Cao, Y., Wang, X., He, X., Hu, Z., and Chua, T. Unifying knowledge graph learning and recommendation: Towards a better understanding of user preferences. In *WWW*, pp. 151–161. ACM, 2019. 1

Chamberlain, B. P., Shirobokov, S., Rossi, E., Frasca, F., Markovich, T., Hammerla, N. Y., Bronstein, M. M., and Hansmire, M. Graph neural networks for link prediction with subgraph sketching. In *ICLR*. OpenReview.net, 2023. 1

Chen, D., O'Bray, L., and Borgwardt, K. M. Structure-aware transformer for graph representation learning. In *ICML*, volume 162 of *Proceedings of Machine Learning Research*, pp. 3469–3489. PMLR, 2022. 3, 15

Chen, S., Liu, X., Gao, J., Jiao, J., Zhang, R., and Ji, Y. Hitter: Hierarchical transformers for knowledge graph embeddings. In *EMNLP*, pp. 10395–10407. Association for Computational Linguistics, 2021. 2, 3, 7

Cheng, D., Yang, F., Wang, X., Zhang, Y., and Zhang, L. Knowledge graph-based event embedding framework for financial quantitative investments. In *SIGIR*, pp. 2221–2230. ACM, 2020. 1

Cui, W. and Chen, X. Instance-based learning for knowledge base completion. In *NeurIPS*, pp. 30744–30755, 2022. 2, 3

Dass, J., Wu, S., Shi, H., Li, C., Ye, Z., Wang, Z., and Lin, Y. Vitality: Unifying low-rank and sparse approximation for vision transformer acceleration with a linear taylor attention. In *HPCA*, pp. 415–428. IEEE, 2023. 5

Davidson, T. R., Falorsi, L., Cao, N. D., Kipf, T., and Tomczak, J. M. Hyperspherical variational auto-encoders. In *UAI*, pp. 856–865. AUAI Press, 2018. 21

Dettmers, T., Minervini, P., Stenetorp, P., and Riedel, S. Convolutional 2d knowledge graph embeddings. In *AAAI*, pp. 1811–1818. AAAI Press, 2018. 2, 7, 8

Devlin, J., Chang, M., Lee, K., and Toutanova, K. BERT: pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*, pp. 4171–4186. Association for Computational Linguistics, 2019. 1

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., and Houlsby, N. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*. OpenReview.net, 2021. 1

Dwivedi, V. P. and Bresson, X. A generalization of transformer networks to graphs. *CoRR*, abs/2012.09699, 2020. 15

Franceschi, L., Niepert, M., Pontil, M., and He, X. Learning discrete structures for graph neural networks. In *ICML*, volume 97 of *Proceedings of Machine Learning Research*, pp. 1972–1982. PMLR, 2019. 1, 2, 3

Galkin, M., Yuan, X., Mostafa, H., Tang, J., and Zhu, Z. Towards foundation models for knowledge graph reasoning. In *ICLR*. OpenReview.net, 2024. 7, 8, 21

Gilmer, J., Schoenholz, S. S., Riley, P. F., Vinyals, O., and Dahl, G. E. Neural message passing for quantum chemistry. In *ICML*, volume 70 of *Proceedings of Machine Learning Research*, pp. 1263–1272. PMLR, 2017. 4

Glorot, X., Bordes, A., and Bengio, Y. Deep sparse rectifier neural networks. In *AISTATS*, volume 15 of *JMLR Proceedings*, pp. 315–323. JMLR.org, 2011. 20

Hamilton, W. L., Ying, Z., and Leskovec, J. Inductive representation learning on large graphs. In *NeurIPS*, pp. 1024–1034, 2017. 7

Hogan, A., Blomqvist, E., Cochez, M., d'Amato, C., de Melo, G., Gutierrez, C., Kirrane, S., Gayo, J. E. L., Navigli, R., Neumaier, S., Ngomo, A. N., Polleres, A., Rashid, S. M., Rula, A., Schmelzeisen, L., Sequeda, J. F., Staab, S., and Zimmermann, A. Knowledge graphs. *ACM Comput. Surv.*, 54(4):71:1–71:37, 2022. 1

Hornik, K. Approximation capabilities of multilayer feedforward networks. *Neural Networks*, 4(2):251–257, 1991. 18

Hornik, K., Stinchcombe, M. B., and White, H. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359–366, 1989. 18

Huang, X., Zhang, J., Li, D., and Li, P. Knowledge graph embedding based question answering. In *WSDM*, pp. 105–113. ACM, 2019. 1

Huang, X., Orth, M. R., Ceylan, I. I., and Barcelo, P. A theory of link prediction via relational weisfeiler-leman on knowledge graphs. In *NeurIPS*, 2023. 6, 17

Jeh, G. and Widom, J. Simrank: a measure of structural-context similarity. In *KDD*, pp. 538–543. ACM, 2002. 21

Ji, S., Pan, S., Cambria, E., Marttinen, P., and Yu, P. S. A survey on knowledge graphs: Representation, acquisition, and applications. *IEEE Trans. Neural Networks Learn. Syst.*, 33(2):494–514, 2022. 1, 7, 8

Katz, L. A new status index derived from sociometric analysis. *Psychometrika*, 18(1):39–43, 1953. 2, 21

Kim, J., Nguyen, D., Min, S., Cho, S., Lee, M., Lee, H., and Hong, S. Pure transformers are powerful graph learners. In *NeurIPS*, 2022. 15

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. In *ICLR (Poster)*, 2015. 5

Kipf, T. N. and Welling, M. Variational graph auto-encoders. *CoRR*, abs/1611.07308, 2016. 21

Kok, S. and Domingos, P. M. Statistical predicate invention. In *ICML*, volume 227 of *ACM International Conference Proceeding Series*, pp. 433–440. ACM, 2007. 9

Koren, Y., Bell, R. M., and Volinsky, C. Matrix factorization techniques for recommender systems. *Computer*, 42(8): 30–37, 2009. 2

Lacroix, T., Usunier, N., and Obozinski, G. Canonical tensor decomposition for knowledge base completion. In *ICML*, volume 80 of *Proceedings of Machine Learning Research*, pp. 2869–2878. PMLR, 2018. 2

Lee, J., Chung, C., and Whang, J. J. Ingram: Inductive knowledge graph embedding via relation graphs. In *ICML*, volume 202 of *Proceedings of Machine Learning Research*, pp. 18796–18809. PMLR, 2023. 8

Li, J., Wang, Q., and Mao, Z. Inductive relation prediction from relational paths and context with hierarchical transformers. In *ICASSP*, pp. 1–5. IEEE, 2023. 2, 3

Li, R., Su, J., Duan, C., and Zheng, S. Linear attention mechanism: An efficient attention for semantic segmentation. *CoRR*, abs/2007.14902, 2020. 5

Li, R., Zhao, J., Li, C., He, D., Wang, Y., Liu, Y., Sun, H., Wang, S., Deng, W., Shen, Y., Xie, X., and Zhang, Q. House: Knowledge graph embedding with householder parameterization. In *ICML*, volume 162 of *Proceedings of Machine Learning Research*, pp. 13209–13224. PMLR, 2022. 1, 2, 7

Liben-Nowell, D. and Kleinberg, J. M. The link-prediction problem for social networks. *J. Assoc. Inf. Sci. Technol.*, 58(7):1019–1031, 2007. 2

Liu, F., Huang, X., Chen, Y., and Suykens, J. A. K. Random features for kernel approximation: A survey on algorithms, theory, and beyond. *IEEE Trans. Pattern Anal. Mach. Intell.*, 44(10):7128–7148, 2022. 9

Lv, X., Lin, Y., Cao, Y., Hou, L., Li, J., Liu, Z., Li, P., and Zhou, J. Do pre-trained models benefit knowledge graph completion? A reliable evaluation and a reasonable approach. In *ACL (Findings)*, pp. 3570–3581. Association for Computational Linguistics, 2022. 1, 3

Ma, L., Lin, C., Lim, D., Romero-Soriano, A., Dokania, P. K., Coates, M., Torr, P. H. S., and Lim, S. Graph inductive biases in transformers without message passing. In *ICML*, volume 202 of *Proceedings of Machine Learning Research*, pp. 23321–23337. PMLR, 2023. 15

Mahdisoltani, F., Biega, J., and Suchanek, F. M. YAGO3: A knowledge base from multilingual wikipedias. In *CIDR*. www.cidrdb.org, 2015. 7

Mai, S., Zheng, S., Yang, Y., and Hu, H. Communicative message passing for inductive relation reasoning. In *AAAI*, pp. 4294–4302. AAAI Press, 2021. 1

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. Distributed representations of words and phrases and their compositionality. In *NeurIPS*, pp. 3111–3119, 2013. 5

Min, E., Chen, R., Bian, Y., Xu, T., Zhao, K., Huang, W., Zhao, P., Huang, J., Ananiadou, S., and Rong, Y. Transformer for graphs: An overview from architecture perspective. *CoRR*, abs/2202.08455, 2022. 15

Neelakantan, A., Roth, B., and McCallum, A. Compositional vector space models for knowledge base completion. In *ACL*, pp. 156–166. The Association for Computer Linguistics, 2015. 2

Nguyen, D. Q., Nguyen, T. D., Nguyen, D. Q., and Phung, D. Q. A novel embedding model for knowledge base completion based on convolutional neural network. In *NAACL-HLT*, pp. 327–333. Association for Computational Linguistics, 2018. 2

Page, L. The pagerank citation ranking: Bringing order to the web. technical report. *Stanford Digital Library Technologies Project*, 1998. 2, 21

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Köpf, A., Yang, E. Z., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S. Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*, pp. 8024–8035, 2019. 20

Perozzi, B., Al-Rfou, R., and Skiena, S. Deepwalk: online learning of social representations. In *KDD*, pp. 701–710. ACM, 2014. 21

Qiu, D., Zhang, Y., Feng, X., Liao, X., Jiang, W., Lyu, Y., Liu, K., and Zhao, J. Machine reading comprehension using structural knowledge graph-aware network. In *EMNLP/IJCNLP*, pp. 5895–5900. Association for Computational Linguistics, 2019. 1

Qu, M., Chen, J., Xhonneux, L. A. C., Bengio, Y., and Tang, J. Rnnlogic: Learning logic rules for reasoning on knowledge graphs. In *ICLR*. OpenReview.net, 2021. 7

Rampásek, L., Galkin, M., Dwivedi, V. P., Luu, A. T., Wolf, G., and Beaini, D. Recipe for a general, powerful, scalable graph transformer. In *NeurIPS*, 2022. 15

Rives, A., Meier, J., Sercu, T., Goyal, S., Lin, Z., Liu, J., Guo, D., Ott, M., Zitnick, C. L., Ma, J., and Fergus, R. Biological structure and function emerge from scaling unsupervised learning to 250 million protein sequences. *Proc. Natl. Acad. Sci. USA*, 118(15):e2016239118, 2021. 1

Sadeghian, A., Armandpour, M., Ding, P., and Wang, D. Z. DRUM: end-to-end differentiable rule mining on knowledge graphs. In *NeurIPS*, pp. 15321–15331, 2019. 2, 7, 8

Saxena, A., Kochsiek, A., and Gemulla, R. Sequence-to-sequence knowledge graph completion and question answering. In *ACL*, pp. 2814–2828. Association for Computational Linguistics, 2022. 1, 3, 7

Schlichtkrull, M. S., Kipf, T. N., Bloem, P., van den Berg, R., Titov, I., and Welling, M. Modeling relational data with graph convolutional networks. In *ESWC*, volume 10843 of *Lecture Notes in Computer Science*, pp. 593–607. Springer, 2018. 2

Sen, P., Namata, G., Bilgic, M., Getoor, L., Gallagher, B., and Eliassi-Rad, T. Collective classification in network data. *AI Mag.*, 29(3):93–106, 2008. 20

Sinha, A. and Duchi, J. C. Learning kernels with random features. In *NeurIPS*, pp. 1298–1306, 2016. 9

Stanfill, C. and Waltz, D. L. Toward memory-based reasoning. *Commun. ACM*, 29(12):1213–1228, 1986. 3

Sun, Z., Deng, Z., Nie, J., and Tang, J. Rotate: Knowledge graph embedding by relational rotation in complex space. In *ICLR (Poster)*. OpenReview.net, 2019. 1, 2, 5, 7

Tang, J., Qu, M., Wang, M., Zhang, M., Yan, J., and Mei, Q. LINE: large-scale information network embedding. In *WWW*, pp. 1067–1077. ACM, 2015. 21

Tang, Y., Huang, J., Wang, G., He, X., and Zhou, B. Orthogonal relation transforms with graph context modeling for knowledge graph embedding. In *ACL*, pp. 2713–2722. Association for Computational Linguistics, 2020. 2

Teru, K. K., Denis, E. G., and Hamilton, W. L. Inductive relation prediction by subgraph reasoning. In *ICML*, volume 119 of *Proceedings of Machine Learning Research*, pp. 9448–9457. PMLR, 2020. 1, 7, 8

Toutanova, K. and Chen, D. Observed versus latent features for knowledge base and text inference. In *CVSC*, pp. 57–66. Association for Computational Linguistics, 2015. 7, 8

Trouillon, T., Dance, C. R., Gaussier, É., Welbl, J., Riedel, S., and Bouchard, G. Knowledge graph completion via complex tensor factorization. *J. Mach. Learn. Res.*, 18: 130:1–130:38, 2017. 1, 2

Tsai, Y. H., Bai, S., Yamada, M., Morency, L., and Salakhutdinov, R. Transformer dissection: An unified understanding for transformer's attention via the lens of kernel. In *EMNLP/IJCNLP*, pp. 4343–4352. Association for Computational Linguistics, 2019. 3

Vashishth, S., Sanyal, S., Nitin, V., and Talukdar, P. P. Composition-based multi-relational graph convolutional networks. In *ICLR*. OpenReview.net, 2020. 2, 7

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. Attention is all you need. In *NeurIPS*, pp. 5998–6008, 2017. 1, 2, 5, 15

Wang, C., Zhou, X., Pan, S., Dong, L., Song, Z., and Sha, Y. Exploring relational semantics for inductive knowledge graph completion. In *AAAI*, pp. 4184–4192. AAAI Press, 2022a. 1

Wang, H., Ren, H., and Leskovec, J. Relational message passing for knowledge graph completion. In *KDD*, pp. 1697–1707. ACM, 2021. 3, 4

Wang, L., Zhao, W., Wei, Z., and Liu, J. Simkgc: Simple contrastive knowledge graph completion with pre-trained language models. In *ACL*, pp. 4281–4294. Association for Computational Linguistics, 2022b. 1, 7, 8

Wang, Q., Mao, Z., Wang, B., and Guo, L. Knowledge graph embedding: A survey of approaches and applications. *IEEE Trans. Knowl. Data Eng.*, 29(12):2724–2743, 2017. 1, 7, 8

Wang, Z., Song, Y., Wong, G. Y., and See, S. Logical message passing networks with one-hop inference on atomic formulas. In *ICLR*. OpenReview.net, 2023. 4

Weisfeiler, B. and Leman, A. The reduction of a graph to canonical form and the algebra which appears therein. *nti, Series*, 2(9):12–16, 1968. 6, 16

Wu, Q., Zhao, W., Li, Z., Wipf, D. P., and Yan, J. Nodeformer: A scalable graph structure learning transformer for node classification. In *NeurIPS*, 2022. 9, 15

Wu, Q., Yang, C., Zhao, W., He, Y., Wipf, D., and Yan, J. Difformer: Scalable (graph) transformers induced by energy constrained diffusion. In *ICLR*. OpenReview.net, 2023a. 5, 15

Wu, Q., Zhao, W., Yang, C., Zhang, H., Nie, F., Jiang, H., Bian, Y., and Yan, J. Simplifying and empowering transformers for large-graph representations. In *NeurIPS*, 2023b. 15

Wu, Z., Jain, P., Wright, M. A., Mirhoseini, A., Gonzalez, J. E., and Stoica, I. Representing long-range context for graph neural networks with global attention. In *NeurIPS*, pp. 13266–13279, 2021. 15

Xie, X., Zhang, N., Li, Z., Deng, S., Chen, H., Xiong, F., Chen, M., and Chen, H. From discrimination to generation: Knowledge graph completion with generative transformer. In *WWW (Companion Volume)*, pp. 162–165. ACM, 2022. 1, 3

Xiong, W., Hoang, T., and Wang, W. Y. Deeppath: A reinforcement learning method for knowledge graph reasoning. In *EMNLP*, pp. 564–573. Association for Computational Linguistics, 2017. 7, 8

Xu, K., Hu, W., Leskovec, J., and Jegelka, S. How powerful are graph neural networks? In *ICLR*. OpenReview.net, 2019. 4, 6, 16, 18

Yan, Z., Ma, T., Gao, L., Tang, Z., and Chen, C. Link prediction with persistent homology: An interactive view. In *ICML*, volume 139 of *Proceedings of Machine Learning Research*, pp. 11659–11669. PMLR, 2021. 21

Yang, B., Yih, W., He, X., Gao, J., and Deng, L. Embedding entities and relations for learning and inference in knowledge bases. In *ICLR (Poster)*, 2015. 1, 2, 4, 7

Yang, F., Yang, Z., and Cohen, W. W. Differentiable learning of logical rules for knowledge base reasoning. In *NeurIPS*, pp. 2319–2328, 2017. 2

Yao, L., Mao, C., and Luo, Y. KG-BERT: BERT for knowledge graph completion. *CoRR*, abs/1909.03193, 2019. 3

You, J., Selman, J. M. G., Ying, R., and Leskovec, J. Identity-aware graph neural networks. In *AAAI*, pp. 10737–10745. AAAI Press, 2021. 2, 4

Zeng, X., Tu, X., Liu, Y., Fu, X., and Su, Y. Toward better drug discovery with knowledge graph. *Curr. Opin. Struct. Biol.*, 72:114–126, 2022. 1

Zhang, M. and Chen, Y. Link prediction based on graph neural networks. In *NeurIPS*, pp. 5171–5181, 2018. 21

Zhang, M. and Chen, Y. Inductive matrix completion based on graph neural networks. In *ICLR*. OpenReview.net, 2020. 7

Zhang, M., Li, P., Xia, Y., Wang, K., and Jin, L. Labeling trick: A theory of using graph neural networks for multi-node representation learning. In *NeurIPS*, pp. 9061–9073, 2021. 4

Zhang, Y. and Yao, Q. Knowledge graph reasoning with relational digraph. In *WWW*, pp. 912–924. ACM, 2022. 1, 3, 7, 8, 9

Zhang, Y., Zhou, Z., Yao, Q., Chu, X., and Han, B. Adaprop: Learning adaptive propagation for graph neural network based knowledge graph reasoning. In *KDD*, pp. 3446–3457. ACM, 2023. 1, 7, 8

Zhang, Z., Cai, J., Zhang, Y., and Wang, J. Learning hierarchy-aware knowledge graph embeddings for link prediction. In *AAAI*, pp. 3065–3072. AAAI Press, 2020. 2

Zhu, Z., Zhang, Z., Xhonneux, L. A. C., and Tang, J. Neural bellman-ford networks: A general graph neural network framework for link prediction. In *NeurIPS*, pp. 29476–29490, 2021. 1, 3, 4, 7, 8, 9, 21

Zhu, Z., Yuan, X., Galkin, M., Xhonneux, S., Zhang, M., Gazeau, M., and Tang, J. A* net: A scalable path-based reasoning approach for knowledge graphs. In *NeurIPS*, 2023. 1, 7, 8

A. More Preliminaries & Backgrounds

Transformer Architecture. The Transformer architecture is originally introduced in Vaswani et al. (2017). A vanilla transformer block consists of two main modules: a self-attention module followed by a feed-forward neural network. In the self-attention module, the input feature matrix denoted as $\boldsymbol{X} \in \mathbb{R}^{n\times d}$ are projected to query $\boldsymbol{Q}$, key $\boldsymbol{K}$ and value $\boldsymbol{V}$ using linear projection matrices $\boldsymbol{W}_q \in \mathbb{R}^{d'\times n}$, $\boldsymbol{W}_k \in \mathbb{R}^{d'\times n}$ and $\boldsymbol{W}_v \in \mathbb{R}^{d'\times n}$. This is done by $\boldsymbol{Q} = \boldsymbol{X}\boldsymbol{W}_{query}$, $\boldsymbol{K} = \boldsymbol{X}\boldsymbol{W}_{key}$ and $\boldsymbol{V} = \boldsymbol{X}\boldsymbol{W}_{value}$, where the bias is omited. The self-attention computation is then given by:

$$\text{Self-Attention}(\boldsymbol{X}) = \text{Softmax}\left(\frac{\boldsymbol{Q}\boldsymbol{K}^T}{\sqrt{d'}}\right)\boldsymbol{V}. \tag{13}$$

In practice, it is common to utilize *multi-head* attention, where multiple instances of Equation (13) are concatenated. This approach has demonstrated effectiveness in capturing multi-view interactions. Subsequently, the output of the self-attention is combined with layer normalization (Ba et al., 2016) and a feedforward network (FFN) to form a transformer block.

Graph Transformers. Recently, transformer models (Vaswani et al., 2017) have gained popularity in graph learning due to their ability to capture complex relationships beyond those captured by regular graph neural networks (GNNs) differently. Injecting structural bias into the original attention mechanism is a key problem in graph transformers. Early work by Dwivedi & Bresson (2020) used Laplacian eigenvectors as positional encodings, and since then, several extensions and other models have been proposed (Min et al., 2022; Kim et al., 2022; Ma et al., 2023). Wu et al. (2021) propose a hybrid architecture that uses a stack of message-passing GNN layers followed by regular transformer layers. The most relevant work to this paper is SAT (Chen et al., 2022), which reformulates the self-attention mechanism as a smooth kernel and incorporates structural information by extracting a subgraph representation rooted at each node before attention computation. Computation during the process follows the equations displayed below:

$$\text{Self-Attention}(\boldsymbol{x}_u) = \sum_{v\in\mathcal{V}} \frac{\kappa(\boldsymbol{x}_u, \boldsymbol{x}_v)}{\sum_{w\in\mathcal{V}}\kappa(\boldsymbol{x}_u, \boldsymbol{x}_w)} f(\boldsymbol{x}_v), \forall u \in \mathcal{V}, \tag{14}$$

where $f(\boldsymbol{x}) = \boldsymbol{x}\boldsymbol{W}_{value} + b_{value}$, and the non-symmetric kernel κ is defined as:

$$\kappa(\boldsymbol{x}_u, \boldsymbol{x}_v) = \exp\left(\frac{\langle \text{GNN}(\boldsymbol{x}_u)\boldsymbol{W}_{query} + b_{query}, \text{GNN}(\boldsymbol{x}_v)\boldsymbol{W}_{key} + b_{key}\rangle}{\sqrt{d}}\right). \tag{15}$$

Moreover, other works focus on the development of scalable models, such as NodeFormer (Wu et al., 2022), GraphGPS (Rampásek et al., 2022), DIFFormer (Wu et al., 2023a), and SGFormer (Wu et al., 2023b).

B. Proof of Theorem 4.2

Proof. Recall that we have the following equations:

$$\begin{aligned} \kappa(\widetilde{\boldsymbol{z}}_u, \widetilde{\boldsymbol{z}}_v) &= 1 + \left\langle \frac{\widetilde{\boldsymbol{z}}_u\boldsymbol{W}_1}{\|\widetilde{\boldsymbol{z}}_u\boldsymbol{W}_1\|_{\mathcal{F}}}, \frac{\widetilde{\boldsymbol{z}}_v\boldsymbol{W}_2}{\|\widetilde{\boldsymbol{z}}_v\boldsymbol{W}_2\|_{\mathcal{F}}} \right\rangle, \\ \kappa_{\text{exp}}(\widetilde{\boldsymbol{z}}_u, \widetilde{\boldsymbol{z}}_v) &= \exp\left(\left\langle \frac{\widetilde{\boldsymbol{z}}_u\boldsymbol{W}_1}{\|\widetilde{\boldsymbol{z}}_u\boldsymbol{W}_1\|_{\mathcal{F}}}, \frac{\widetilde{\boldsymbol{z}}_v\boldsymbol{W}_2}{\|\widetilde{\boldsymbol{z}}_v\boldsymbol{W}_2\|_{\mathcal{F}}} \right\rangle\right), \end{aligned} \tag{16}$$

where we omit the bias. For convenience, we define $\boldsymbol{u} = \frac{\widetilde{\boldsymbol{z}}_u\boldsymbol{W}_1}{\|\widetilde{\boldsymbol{z}}_u\boldsymbol{W}_1\|_{\mathcal{F}}}$ and $\boldsymbol{v} = \frac{\widetilde{\boldsymbol{z}}_v\boldsymbol{W}_2}{\|\widetilde{\boldsymbol{z}}_v\boldsymbol{W}_2\|_{\mathcal{F}}}$, yielding:

$$\begin{aligned} \kappa(\widetilde{\boldsymbol{z}}_u, \widetilde{\boldsymbol{z}}_v) &= 1 + \langle \boldsymbol{u}, \boldsymbol{v}\rangle, \\ \kappa_{\text{exp}}(\widetilde{\boldsymbol{z}}_u, \widetilde{\boldsymbol{z}}_v) &= \exp\left(\langle \boldsymbol{u}, \boldsymbol{v}\rangle\right), \end{aligned} \tag{17}$$

where $\langle \boldsymbol{u}, \boldsymbol{v}\rangle \in [-1, 1]$. According to the Taylor formula, the expression for $\kappa_{\text{exp}}(\widetilde{\boldsymbol{z}}_u, \widetilde{\boldsymbol{z}}_v)$ is as follows:

$$\exp\left(\langle \boldsymbol{u}, \boldsymbol{v}\rangle\right) = \underbrace{1 + \langle \boldsymbol{u}, \boldsymbol{v}\rangle}_{\text{first-order Taylor expansion}} + \underbrace{\frac{\exp(\xi)}{2}\left(\langle \boldsymbol{u}, \boldsymbol{v}\rangle\right)^2}_{\text{Lagrange remainder term}}, \tag{18}$$

where $\xi \in (0, \langle \boldsymbol{u}, \boldsymbol{v} \rangle)$. In other words, we have:

$$|\kappa(\widetilde{\boldsymbol{z}}_u, \widetilde{\boldsymbol{z}}_v) - \kappa_{\exp}(\widetilde{\boldsymbol{z}}_u, \widetilde{\boldsymbol{z}}_v)| = \frac{\exp(\xi)}{2}(\langle \boldsymbol{u}, \boldsymbol{v} \rangle)^2. \tag{19}$$

Further, by taking $\xi = \gamma \cdot \langle \boldsymbol{u}, \boldsymbol{v} \rangle$, where $\gamma \in (0, 1)$, we can rewrite the Lagrange remainder term in Equation (18) as:

$$\frac{\exp(\xi)}{2}(\langle \boldsymbol{u}, \boldsymbol{v} \rangle)^2 = \frac{\exp(\gamma \cdot \langle \boldsymbol{u}, \boldsymbol{v} \rangle)}{2}(\langle \boldsymbol{u}, \boldsymbol{v} \rangle)^2. \tag{20}$$

Then we have:

$$\begin{aligned} |\kappa(\widetilde{\boldsymbol{z}}_u, \widetilde{\boldsymbol{z}}_v) - \kappa_{\exp}(\widetilde{\boldsymbol{z}}_u, \widetilde{\boldsymbol{z}}_v)| = \frac{\exp(\gamma \cdot \langle \boldsymbol{u}, \boldsymbol{v} \rangle)}{2}(\langle \boldsymbol{u}, \boldsymbol{v} \rangle)^2 &< \max(e^{-\gamma}/2, e^{\gamma}/2) \\ &< e^{\gamma}/2. \end{aligned} \tag{21}$$

$\square$

C. Details of Relational Asymmetric Local 2-WL Test

Firstly, we present the general form of the Weisfeiler-Lehman (WL) Test (Weisfeiler & Leman, 1968). Let $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, c\}$ denotes a graph where $\mathcal{V}$ represents the set of nodes, $\mathcal{E}$ represents the set of edges, and c represents the node coloring. The node coloring, also known as the feature mapping, is denoted as $c : \mathcal{V} \to \mathbb{R}^d$. The purpose of the Weisfeiler-Lehman ($\texttt{WL}$) test is to detect graph isomorphism.

Definition C.1 (Graph Isomorphism)**.** An *isomorphism* between a graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, c\}$ and $\mathcal{G}' = \{\mathcal{V}', \mathcal{E}', c'\}$ is a *bijection* $f : \mathcal{V} \to \mathcal{V}'$ that satisfies $c(v) = c'(f(v))$ for all $v \in \mathcal{V}$, and $(u, v) \in \mathcal{E}$ if and only if $(f(u), f(v)) \in \mathcal{E}'$ for all $u, v \in \mathcal{V}$.

Definition C.2 ($\texttt{WL}$ test)**.** Consider a graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, c\}$. The $\texttt{WL}$ test is defined as follows:

$$\begin{aligned} &\texttt{WL}^{(0)}(u) = c(u), \\ &\texttt{WL}^{(t+1)}(u) = \tau\left(\texttt{WL}^{(t)}(u), \{\!\{\texttt{WL}(v) \mid v \in \mathcal{N}(u)\}\!\}\right), \end{aligned} \tag{22}$$

where $\{\!\{\cdot\}\!\}$ denotes a multiset, $\mathcal{N}(u)$ means the neighborhood of node u, and τ maps the pair above injectively to a unique color that has not been used in previous iterations. Repeat the above steps T times until convergence.

In general, the $\texttt{WL}$ test can be used as a necessary but not sufficient condition for detecting graph isomorphism. Additionally, there exists the $\texttt{WL}_k$ ($k > 1$) test (Xu et al., 2019), which operates on k-tuples of nodes $\boldsymbol{u} \in \mathcal{V}^k$ and provides enhanced expressive power.

Now let us delve into the realm of knowledge graphs. Consider a knowledge graph denoted by $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, \mathcal{R}, c\}$, where $\mathcal{V}$ represents the set of entities, $\mathcal{R}$ represents the set of relations, $\mathcal{E}$ represents the set of facts, and c represents the entity coloring scheme. Similarly, we provide the definition for knowledge graph isomorphism.

Definition C.3 (Knowledge Graph Isomorphism)**.** An *isomorphism* between knowledge graphs $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, \mathcal{R}, c\}$ and $\mathcal{G}' = \{\mathcal{V}', \mathcal{E}', \mathcal{R}', c'\}$ is a *bijection* $f : \mathcal{V} \to \mathcal{V}'$ that satisfies $c(v) = c'(f(v))$ for all $v \in \mathcal{V}$, and $r(u, v) \in \mathcal{E}$ if and only if $r(f(u), f(v)) \in \mathcal{E}'$ for all $r \in \mathcal{R}$ and $u, v \in \mathcal{V}$.

The $\texttt{R-WL}$ test is a relational variant of the $\texttt{WL}$ test, proposed by Barceló et al. (2022). It is defined as follows.

Definition C.4 ($\texttt{R-WL}$ test)**.** Let $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, \mathcal{R}, c\}$ be a knowledge graph. The $\texttt{R-WL}$ test is defined as:

$$\begin{aligned} &\texttt{R-WL}^{(0)}(u) = c(u), \\ &\texttt{R-WL}^{(t+1)}(u) = \tau\left(\texttt{R-WL}^{(t)}(u), \{\!\{(\texttt{R-WL}(v), r) \mid v \in \mathcal{N}_r(u), r \in \mathcal{R}\}\!\}\right), \end{aligned} \tag{23}$$

where $\{\!\{\cdot\}\!\}$ denotes a multiset, $\mathcal{N}_r(u)$ refers to the neighborhood of node u corresponding to relation r, and τ injectively maps the aforementioned pair to a unique color not yet used in prior iterations.

However, the aforementioned tests are not suitable for quantifying the expressive power of methods employed in link prediction, as they require a measurement of a *binary variant*.

Definition C.5 (*binary variant* on knowledge graphs)**.** A *binary variant* on knowledge graphs is represented as a function ξ that associates each knowledge graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, \mathcal{R}, c\}$ with a function $\xi(\mathcal{G})$ defined on $\mathcal{V}^2$. This function satisfies the condition that for all knowledge graphs $\mathcal{G}$ and $\mathcal{G}'$, all isomorphisms f from $\mathcal{G}$ to $\mathcal{G}'$, and all 2-tuples of entities $\boldsymbol{u} \in \mathcal{V}^2$, it holds that $\xi(\mathcal{G})(\boldsymbol{u}) = \xi(\mathcal{G}')(f(\boldsymbol{u}))$.

To analyze the expressive power of knowledge graph reasoning methods, Huang et al. (2023) introduces the *relational asymmetric local 2-WL* test denoted by $\texttt{RA-WL}_2$. Specifically, $\texttt{RA-WL}_2$ is defined on a knowledge graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, \mathcal{R}, c, \eta\}$ with a *pairwise coloring* $\eta : \mathcal{V} \times \mathcal{V} \to \mathcal{D}$ that satisfies *target node distinguishability*, meaning $\eta(u, u) \neq \eta(u, v)$ for all $u \neq v \in \mathcal{V}$. Then $\texttt{RA-WL}_2$ is defined as:

Definition C.6 ($\texttt{RA-WL}_2$ test)**.** Let $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, \mathcal{R}, c\}$ be a knowledge graph. The $\texttt{RA-WL}_2$ test is defined as:

$$\begin{aligned}
&\texttt{RA-WL}_2^{(0)}(u, v) = \eta(u, v), \\
&\texttt{RA-WL}_2^{(t+1)}(u, v) = \tau\left(\texttt{RA-WL}_2^{(t)}(u, v), \{\!\{(\texttt{RA-WL}_2(u, w), r) \mid w \in \mathcal{N}_r(u), r \in \mathcal{R}\}\!\}\right),
\end{aligned} \tag{24}$$

where $\{\!\{\cdot\}\!\}$ denotes a multiset, $\mathcal{N}_r(u)$ means the neighborhood of node u corresponding to relation r, and τ maps the pair above injectively to a unique color that has not been used in previous iterations.

$\texttt{RA-WL}_2$ test allows us to characterize the power of methods in terms of their ability to distinguish pairs of entities on knowledge graphs.

D. Proof of Theorem 4.3

Proof. To prove Theorem 4.3, we begin by establishing the equivalence between the query function $f_q(\cdot)$ and $\texttt{RA-WL}_2$ on original knowledge graph $\mathcal{E}$. The proof is mainly based on the results in Huang et al. (2023). As stated in Huang et al. (2023), a conditional message-passing neural network for inferring the query $(e_h, r_q, ?)$ is defined as follows:

$$\begin{aligned}
&\boldsymbol{h}_{u|e_h,r_q}^{(0)} = \textsc{Init}(u|e_h, r_q), \\
&\boldsymbol{h}_{u|e_h,r_q}^{(t+1)} = \textsc{Upd}\left(\boldsymbol{h}_{u|e_h,r_q}^{(t)}, \textsc{Agg}\left(\left\{\!\!\left\{\textsc{Msg}\left(\boldsymbol{h}_{v|e_h,r_q}^{(t)}, \boldsymbol{w}_{r_q}\right) | v \in \mathcal{N}_r(u), r \in \mathcal{R}\right\}\!\!\right\}\right), \textsc{Read}\left(\left\{\!\!\left\{\boldsymbol{h}_{w|e_h,r_q}^{(t)} | w \in \mathcal{V}\right\}\!\!\right\}\right)\right),
\end{aligned} \tag{25}$$

where $\boldsymbol{h}_{u|e_h,r_q}^{(t)}$ represents a pairwise representation corresponding to $\boldsymbol{h}_{r_q}^{(t)}(e_h, u) : \mathcal{V} \times \mathcal{V} \to \mathbb{R}^d$. Here, Init, Upd, Agg, Read, and $\textsc{Msg}_r$ are differentiable functions responsible for *initialization*, *update*, *aggregation*, *global readout*, and *relation-specific message* computations, respectively. Based on Theorem 5.1 in Huang et al. (2023), it is suggested that a message-passing neural network with an architecture equivalent to Equation (25), featuring a *target node distinguishable* Init, exhibits the same expressivity as $\texttt{RA-WL}_2$.

Now we establish the equivalence of our proposed query function and Equation (25).

Lemma D.1. *The query function $f_q(\cdot)$ shares the same architecture as the conditional message-passing neural network defined in Equation* (25)*.*

Proof. From Equation (5), we derive the following:

$$\begin{aligned}
&\underbrace{\boldsymbol{z}_u^{(0)}}_{\boldsymbol{h}_{u|e_h,r_q}^{(0)}} \leftarrow \underbrace{[\boldsymbol{x}_u, \boldsymbol{\epsilon}]}_{\textsc{Init}}, \\
&\underbrace{\boldsymbol{z}_u^{(l)}}_{\boldsymbol{h}_{u|e_h,r_q}^{(t+1)}} \leftarrow \underbrace{\Phi\left(\boldsymbol{\alpha}^{(l-1)} \cdot \boldsymbol{z}_u^{(l-1)} + \underbrace{\sum_{r(v,u)\in\mathcal{E}} \underbrace{\boldsymbol{m}_{u|v,r}^{(l-1)}}_{\textsc{Msg}}}_{\textsc{Agg}}\right)}_{\textsc{Upd}},
\end{aligned} \tag{26}$$

where we have omitted the Read function in our query function. □

We will now demonstrate that the INIT function in our query function ensures *target node distinguishability*, which is defined as $\textsc{Init}(u|u, r_q) \neq \textsc{Init}(v|u, r_q)$ if $u \neq v$ for each $v \in \mathcal{V} \setminus \{u\}$.

Lemma D.2. *For each $v \in \mathcal{V} \setminus \{u\}$, there exists $\lim_{d\to\infty} P(\boldsymbol{z}_u^{(0)} = \boldsymbol{z}_v^{(0)}) = 0$, where $\boldsymbol{z}_u^{(0)}, \boldsymbol{z}_v^{(0)} \in \mathbb{R}^d$.*

Proof. For simplicity, we consider only the random component of $\boldsymbol{z}^{(0)} \in \mathbb{R}^d$, assuming $\boldsymbol{z}^{(0)}[i] \sim N(0, 1)$. Let $\boldsymbol{z}_u^{(0)} = [z_{u,0}, z_{u,1}, \ldots, z_{u,d}]$ and $\boldsymbol{z}_v^{(0)} = [z_{v,0}, z_{v,1}, \ldots, z_{v,d}]$. We can calculate:

$$\begin{aligned} P(\boldsymbol{z}_u^{(0)} = \boldsymbol{z}_v^{(0)}) &= \prod_{i=0}^{d} P(z_{u,i} = z_{v,i}) \\ &= \prod_{i=0}^{d} \frac{1}{2\pi} \exp\left(-z_{u,i}^2\right) \\ &\leq \prod_{i=0}^{d} \frac{1}{2\pi} \\ &= \left(\frac{1}{2\pi}\right)^d . \end{aligned} \tag{27}$$

Furthermore, using $\frac{1}{2\pi} < 1$, we find $\lim_{d\to\infty} P\left(\boldsymbol{z}_u^{(0)} = \boldsymbol{z}_v^{(0)}\right) = 0$. □

From Lemma D.2, it follows that our INIT function ensures *target node distinguishability* when employing a sufficiently large dimension d for random features. Based on Lemma D.1 and Lemma D.2, we can derive the following lemma:

Lemma D.3. *The proposed query function $f_q(\cdot)$ can be as expressive as $\texttt{RA-WL}_2$ on graph $\mathcal{E}$.*

Since the initialization in Equation (6) also satisfies the *target node distinguishability*. By employing a similar approach, we can also derive the following lemma.

Lemma D.4. *The proposed value function $f_v(\cdot)$ can be as expressive as $\texttt{RA-WL}_2$ on graph $\mathcal{E}$.*

To further showcase the expressivity of our proposed attention function on the estimated graph $\tilde{\mathcal{E}}$, we aim to establish the injectiveness of our attention function. The update process of KNOWFORMER's attention can be reformulated as:

$$\boldsymbol{z}_u = \phi\left(\xi \cdot \overline{\boldsymbol{z}}_u, f\left(\{\hat{\boldsymbol{z}}_v : v \in \mathcal{V}\}\right)\right), \tag{28}$$

where $\overline{\boldsymbol{z}}_u$ means the output of the value function, $\{\hat{\boldsymbol{z}}_v : v \in \mathcal{V}\}$ indicates the weighted values $\alpha_{uv}\overline{\boldsymbol{z}}_v$ where α_{uv} is the attention score and ξ is a constant factor which is equal to $|\mathcal{V}|$ here.

We denote the space of $\hat{\boldsymbol{z}}$ as $\mathcal{Z}$. Recall that the query function $f_q(\cdot)$ shares the same expressive capacity as $\texttt{RA-WL}_2$, enabling it to distinguish the diverse neighbor structures associated with each $u \in \mathcal{V}$. Consequently, for distinct u values, we can obtain diverse multisets $\{\hat{\boldsymbol{z}}_v : v \in \mathcal{V}\} \subset \mathcal{Z}$ which forms the estimated graph $\tilde{\mathcal{E}}$.

Further, we have the following theorem:

Lemma D.5. *Assuming $\mathcal{Z}$ is countable, the update process of* KNOWFORMER*'s attention is injective to each pair $(\overline{\boldsymbol{z}}_v, \{\hat{\boldsymbol{z}}_v : v \in \mathcal{V}\})$.*

Proof. To prove Lemma D.5, we express the update process as follows:

$$\boldsymbol{z}_u = g\left((1+\zeta) \cdot f(\overline{\boldsymbol{z}}_u) + \sum_{v\in\mathcal{V}} f(\hat{\boldsymbol{z}}_v)\right). \tag{29}$$

It has been shown in Lemma 5 and Corollary 6 in Xu et al. (2019) that there exists a function $f : \mathcal{Z} \to \mathbb{R}^d$ for which Equation (29) is unique for $(\overline{\boldsymbol{z}}_v, \{\hat{\boldsymbol{z}}_v : v \in \mathcal{V}\})$. The FFN following the attention can be employed to model $g \circ f$ based on the universal approximation theorem (Hornik et al., 1989; Hornik, 1991). Consequently, Lemma D.5 can be concluded. □

Lemma D.5 provides evidence for the injectiveness of our proposed attention function on the estimated graph $\tilde{\mathcal{E}}$. From the definition of $\texttt{RA-WL}_2$, we can conclude that the attention layer of KNOWFORMER is capable of attaining the same level of expressiveness as $\texttt{RA-WL}_2$, thereby validating Theorem 4.3. □

E. Experimental Details

E.1. Dataset Statistics

We conduct experiments on four transductive knowledge graph reasoning datasets, and the statistics of these datasets are summarized in Table 4. Additionally, we perform experiments on three inductive knowledge graph reasoning datasets, each of which contains four different splits. The statistics of the inductive datasets are summarized in Table 5.

Dataset	**#Relation**	**#Entity**	**#Triplet**		
			#Train	#Valid	#Test
FB15k-237	237	14,541	272,115	17,535	20,466
WN18RR	11	40,943	86,835	3,034	3,134
NELL-995	200	74,536	149,678	543	2,818
YAGO3-10	37	123,182	1,079,040	5,000	5,000

Table 4. Dataset Statistics for transductive knowledge graph reasoning datasets.

Dataset		**#Relation**	**Train**			**Validation**			**Test**		
			#Entity	#Query	#Fact	#Entity	#Query	#Fact	#Entity	#Query	#Fact
FB15k-237	v1	180	1,594	4,245	4,245	1,594	489	4,245	1,093	205	1,993
	v2	200	2,608	9,739	9,739	2,608	1,166	9,739	1,660	478	4,145
	v3	215	3,668	17,986	17,986	3,668	2,194	17,986	2,501	865	7,406
	v4	219	4,707	27,203	27,203	4,707	3,352	27,203	3,051	1,424	11,714
WN18RR	v1	9	2,746	5,410	5,410	2,746	630	5,410	922	188	1,618
	v2	10	6,954	15,262	15,262	6,954	1,838	15,262	2,757	441	4,011
	v3	11	12,078	25,901	25,901	12,078	3,097	25,901	5,084	605	6,327
	v4	9	3,861	7,940	7,940	3,861	934	7,940	7,084	1,429	12,334
NELL-995	v1	14	3,103	4,687	4,687	3,103	414	4,687	225	100	833
	v2	86	2,564	15,262	8,219	8,219	922	8,219	2,086	476	4,586
	v3	142	4,647	16,393	16,393	4,647	1,851	16,393	3,566	809	8,048
	v4	76	2,092	7,546	7,546	2,092	876	7,546	2,795	7,073	731

Table 5. Dataset Statistics for inductive knowledge graph reasoning datasets. In each split, one needs to infer #Query triplets based #Fact triplets.

E.2. Evaluation

For each dataset, we evaluate the performance of ranking each answer entity against all negative entities given a query $(h, r, ?)$. The evaluation metrics we use are the mean reciprocal rank (MRR) (Bordes et al., 2013) and Hits at n (H@n) (Bordes et al., 2013), which are computed as follows:

$$\begin{aligned} \text{MRR} &= \frac{1}{|\mathcal{A}|} \sum_{i=1}^{|A|} \frac{1}{rank_i}, \\ \text{Hits at } n &= \frac{1}{|\mathcal{A}|} \mathbb{I}[rank_i \leq k], \end{aligned} \tag{30}$$

where $\mathcal{A}$ indicates the answer entity set, $\mathbb{I}[\cdot]$ indicates the indicative function, and $rank_i$ indicates the rank of each answer entity.

E.3. Implement Details

Our model is comprised of three modules: the input layer, the attention layer, and the output layer. We present the details for each of them as follows.

- The input layer consists of the initialization of entity representations and relation representations. We initialize entity representations as all-zero vectors $\boldsymbol{X} = [0]^{|\mathcal{V}| \times d}$ and relation representations as randomly initialized learnable vectors $\boldsymbol{R} \in \mathbb{R}^{|\mathcal{R}| \times d}$.
- The attention layer is composed of the attention function which is followed by a skip-connection, an FFN, and two normalization layers before and after the FFN. The attention function consists of three sub-modules as described in Section 4.1. Specifically, each attention function feeds entity representations $\boldsymbol{X}^{(l-1)}$ and relation representation $\boldsymbol{R}$ and outputs the updated entity representation $\boldsymbol{X}^{(l)}$. The MLPs in the attention function are implemented as three-layer MLPs with ReLU (Glorot et al., 2011) activation. Additionally, We implement FFN as a two-layer MLP with ReLU (Glorot et al., 2011) activation.
- The output layer is a feed-forward layer for prediction, which maps the entity representations into the predicted scores. The scores will be used for ranking answer entities.

For each experiment, we employ a fixed random seed and run it multiple times to report the average performance.

E.4. Hyperparameters Setup

For each dataset, we performed hyperparameter tuning on the validation set. We considered different values for the learning rate (lr) from the set $\{1e-4, 5e-4, 1e-3, 5e-3\}$, weight decay ($wd$) from the set $\{0, 1e-6, 1e-5, 1e-4\}$, hidden dimension (d) from the set $\{16, 32, 64\}$, number of negative samples ($|[t']|$) from the set $\{2^6, 2^8, 2^{10}, 2^{12}, 2^{14}, 2^{16}\}$, number of layers for the query function ($\widetilde{L}$) from the set $\{1, 2, 3\}$, number of layers for the value function ($\widehat{L}$) from the set $\{1, 2, 3\}$, and number of layers for KNOWFORMER (L) from the set $\{1, 2, 3\}$.

E.5. Hardcore Configurations

We conduct all experiments with:

- Operating System: Ubuntu 22.04.3 LTS.
- CPU: Intel (R) Xeon (R) Platinum 8358 CPU @ 2.60GHz with 1TB DDR4 of Memory and Intel Xeon Gold 6148 CPU @ 2.40GHz with 384GB DDR4 of Memory.
- GPU: NVIDIA Tesla A100 SMX4 with 40GB of Memory and NVIDIA Tesla V100 SXM2 with 32GB of Memory.
- Software: CUDA 12.1, Python 3.9.14, PyTorch (Paszke et al., 2019) 2.1.0.

F. More Experimental Results

F.1. KNOWFORMER v.s. ULTRA

We present the experimental results of KNOWFORMER and ULTRA in Table 6. It can be observed that KNOWFORMER achieves comparable performance to ULTRA. Notably, ULTRA demonstrates state-of-the-art (SOTA) performance in certain datasets due to its utilization of extensive pretraining data. As mentioned previously, our method can be easily adapted to address full-inductive tasks with minor modifications, making it compatible with a pre-training and fine-tuning paradigm. We intend to explore this avenue in future work.

F.2. Performance of Homogeneous Graph Link Prediction

Although KNOWFORMER is primarily designed for knowledge graph reasoning, it can also be applied to general link prediction tasks. In this section, we present experimental results for the homogeneous graph link task using datasets including Cora, Citeseer, and PubMed (Sen et al., 2008). We follow the same settings (baselines and other experimental

Method	v1			v2			v3			v4		
	MRR	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10
WN18RR												
ULTRA (Galkin et al., 2024)	0.685	-	79.3	0.679	-	77.9	0.411	-	54.6	0.614	-	72.0
KNOWFORMER	**0.752**	**71.5**	**81.9**	**0.709**	**65.6**	**81.7**	**0.467**	**40.6**	**57.1**	**0.646**	**60.9**	**72.7**
NELL-995												
ULTRA (Galkin et al., 2024)	0.757	-	87.8	**0.575**	-	**76.1**	**0.563**	-	**75.5**	**0.469**	-	**73.3**
KNOWFORMER	**0.827**	**77.0**	**93.0**	0.465	35.7	65.7	0.478	37.8	65.7	0.378	26.7	59.8

Table 6. Inductive knowledge graph reasoning performance comparison between KNOWFORMER and ULTRA. The best results are **boldfaced**. Our proposed model, KNOWFORMER is marked by **tan** .

setup) as in Zhu et al. (2021). Table 7 summarizes the results of KNOWFORMER and baselines on three datasets. We can observe that KNOWFORMER surpasses the baselines on Cora and PubMed and achieves comparable performance on Citeseer, which demonstrates the effectiveness of KNOWFORMER on general graph link prediction tasks.

Method	Cora		Citeseer		PubMed	
	AUROC	AP	AUROC	AP	AUROC	AP
Katz Index (Katz, 1953)	0.834	0.889	0.768	0.810	0.757	0.856
Personalized PageRank (Page, 1998)	0.845	0.899	0.762	0.814	0.763	0.860
SimRank (Jeh & Widom, 2002)	0.838	0.888	0.755	0.805	0.743	0.829
DeepWalk (Perozzi et al., 2014)	0.831	0.850	0.805	0.836	0.844	0.841
LINE (Tang et al., 2015)	0.844	0.879	0.838	0.868	0.891	0.914
VGAE (Kipf & Welling, 2016)	0.914	0.926	0.908	0.920	0.944	0.947
S-VGAE (Davidson et al., 2018)	0.941	0.941	**0.947**	**0.952**	0.960	0.960
SEAL (Zhang & Chen, 2018)	0.933	0.942	0.905	0.924	0.978	0.979
TLC-GNN (Yan et al., 2021)	0.934	0.931	0.909	0.916	0.970	0.968
NBFNet (Zhu et al., 2021)	0.956	0.962	0.923	0.936	0.983	0.982
KNOWFORMER	**0.961**	**0.965**	0.941	0.950	**0.987**	**0.988**

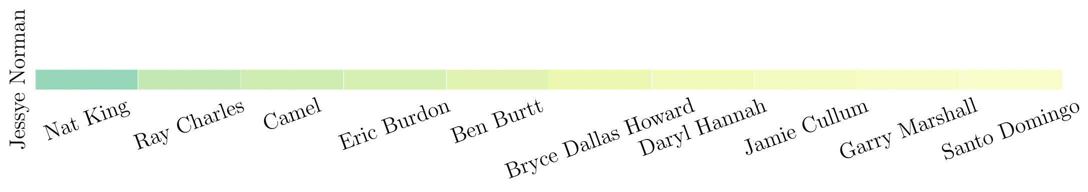
Table 7. Results of homogeneous graph link prediction task. The best results are **boldfaced**. Our proposed model, KNOWFORMER is marked by **tan** . Most baseline results are taken from original papers.

F.3. Impact of the Number of Layers

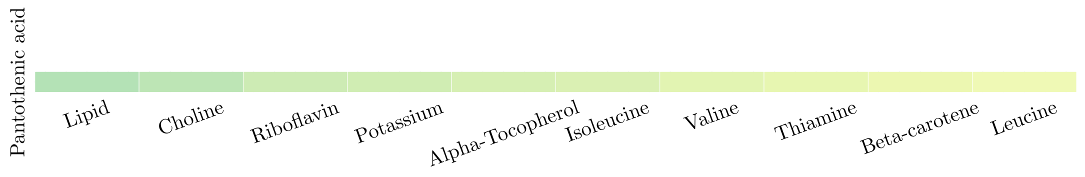
The number of layers, denoted as L, $\widetilde{L}$, and $\widehat{L}$, is a crucial hyperparameter in KNOWFORMER. To illustrate the influence of the number of layers, we present the experimental results on the transductive WN18RR dataset in Table 8. We can observe that both the number of attention layers and the number of value function layers play a crucial role in determining the reasoning performance of KNOWFORMER. One notable finding is that increasing the number of attention layers can yield significant performance improvements, even in scenarios where the value function layers are limited. This finding further reinforces the effectiveness of the attention layers proposed in this paper. Moreover, increasing the number of query function layers enhances the inference performance, particularly for shallow models (*e.g.*, limited number of attention layers and value function layers). This observation suggests that the query function is adept at effectively capturing structural information.

F.4. Case Study on KNOWFORMER's Interpretation

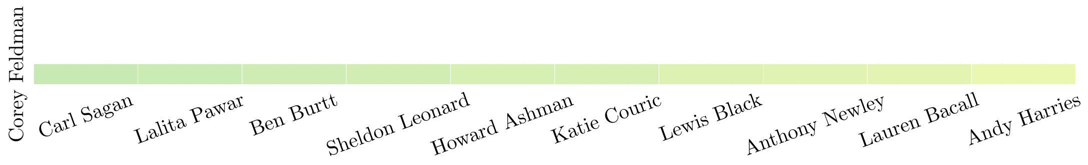
To gain a deeper understanding of the attention function of KNOWFORMER, we present visualizations of the attention matrix in Figure 4. These examples demonstrate that KNOWFORMER is capable of effectively capturing the inherent patterns among various entities, which contributes to the enhancement of its performance. For instance, when examining the test fact (`Egg`, `nutrition_fact`, ?), we observe that KNOWFORMER successfully distinguishes the nutrients of eggs, such as Lipid, Choline, and Riboflvain, aligning with human cognition.



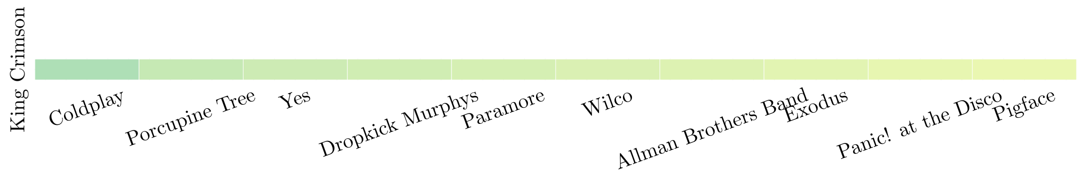
(a) (`Decca_Records`, `artist`, ?)



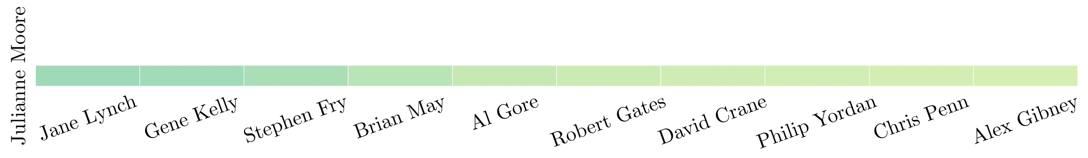
(b) (`Egg`, `nutrition_fact`, ?)



(c) (`Jewish_people`, `ethnicity`, ?)



(d) (`Piano`, `music_group`, ?)



(e) (`Bachelor's_degree`, `educational_degree`, ?)

Figure 4. Visualization of KnowFormer attention on FB15k-237 test set. We select the attention matrix corresponding to the answer entity for each test fact and visualize the top-10 entities, excluding the answer entity itself.

#Value Layers	**#Attention Layers=1**			**#Attention Layers=2**			**#Attention Layers=3**		
	MRR	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10
#Query Layers=1									
1	0.374	36.4	39.4	0.442	41.3	49.5	0.543	50.1	62.5
2	0.425	40.5	46.1	0.554	50.7	65.0	0.572	52.0	67.6
3	0.538	49.7	62.0	0.573	52.0	67.0	0.578	52.4	67.8
#Query Layers=2									
1	0.376	36.4	39.6	0.540	50.0	61.9	0.572	52.1	66.8
2	0.425	40.5	46.0	0.551	50.7	62.1	0.576	52.5	67.5
3	0.539	49.8	61.9	0.572	51.5	67.2	0.578	52.2	68.1
#Query Layers=3									
1	0.379	37.0	39.7	0.554	50.7	64.6	0.568	51.7	66.8
2	0.428	40.9	46.2	0.569	51.8	67.1	0.575	52.0	67.9
3	0.537	49.6	61.9	0.572	52.3	67.0	0.577	51.9	67.3

Table 8. Results of different numbers of layers on the WN18RR dataset, with each metric represented by a different color. The darkness or lightness of the color corresponds to the performance of the metric.