KGDM: A Diffusion Model to Capture Multiple Relation Semantics for Knowledge Graph Embedding

Xiao Long[1], Liansheng Zhuang[1]*, Aodi Li[1], Jiuchang Wei[1], Houqiang Li[1], Shafei Wang[2]

[1]University of Science and Technology of China, Hefei 230026, China
[2]Peng Cheng Laboratory, Shenzhen 518000, China
longxiao1997@mail.ustc.edu.cn; lszhuang@ustc.edu.cn

Abstract

Knowledge graph embedding (KGE) is an efficient and scalable method for knowledge graph completion. However, most existing KGE methods suffer from the challenge of multiple relation semantics, which often degrades their performance. This is because most KGE methods learn fixed continuous vectors for entities (relations) and make deterministic entity predictions to complete the knowledge graph, which hardly captures multiple relation semantics. To tackle this issue, previous works try to learn complex probabilistic embeddings instead of fixed embeddings but suffer from heavy computational complexity. In contrast, this paper proposes a simple yet efficient framework namely the Knowledge Graph Diffusion Model (KGDM) to capture the multiple relation semantics in prediction. Its key idea is to cast the problem of entity prediction into conditional entity generation. Specifically, KGDM estimates the probabilistic distribution of target entities in prediction through Denoising Diffusion Probabilistic Models (DDPM). To bridge the gap between continuous diffusion models and discrete KGs, two learnable embedding functions are defined to map entities and relation to continuous vectors. To consider connectivity patterns of KGs, a Conditional Entity Denoiser model is introduced to generate target entities conditioned on given entities and relations. Extensive experiments demonstrate that KGDM significantly outperforms existing state-of-the-art methods in three benchmark datasets.

Introduction

Knowledge graphs (KGs) are a type of multi-relational graph that stores factual knowledge in the real world. Benefitting from their efficiency in storing and representing factual knowledge, KGs are essential for many applications such as question answering (Hao et al. 2017), information retrieval (Xiong, Power, and Callan 2017), recommender systems (Zhang et al. 2016), and natural language processing (Yang, Yang, and Cohen 2017). Usually, a KG consists of enormous factual triplets, where each triplet (h, r, t) includes a head entity h, a relation r, and a tail entity t. Due to the complexity of the real world, KGs are often incomplete, which restricts their applications in downstream tasks. Therefore, knowledge graph completion (KGC) is proposed to complete missing facts by inferring from existing ones. Generally, the KGC task is to make the entity prediction.

Knowledge graph embedding (KGE) is a promising approach to predicting the missing facts. It learns the embeddings of entities and relations of KGs in a low-dimensional vector space, where the embeddings are required to preserve the semantic meaning and relational structure. Despite the success of KGE models, most of them make deterministic entity predictions in the vector space. So, they can only capture specific semantics of relations and lack the capability to deal with relations that contain multiple semantics. Due to the ambiguity of knowledge in KGs, a relation often has multiple semantics revealed by the entity pairs in one-to-many and many-to-many forms. For example, the relation LocationContains has multiple latent semantics: city-related as (US, LocationContains, New York), college-related as (US, LocationContains, Yale University), and company-related as (US, LocationContains, Google). As shown in Table 1, the existence of multiple relation semantics is quite common in knowledge graphs and significantly degrades the performance of KGE methods. For example, RotatE (Sun et al. 2019) and SEA (Gregucci et al. 2023) have impressive performances on 1-1 and N-1 types, but they perform very poorly on 1-N and N-N types. Additionally, although TransH (Wang et al. 2014) and TransG (Xiao et al. 2015) perform well on the FB15k dataset in tackling the issue of multiple relation semantics, they still exhibit poor performance on the more comprehensive FB15k-237 dataset (excluding inverse relations and having a higher entity-relation ratio (Dettmers et al. 2018)) in the 1-N and N-N types.

Relation Type &Proportion	Methods			
	TransH	TransG	RotatE	SEA
1-1 (1.57%)	0.492	0.489	0.487	0.493
1-N (18.60%)	**0.077**	**0.078**	**0.081**	**0.087**
N-1 (4.60%)	0.449	0.458	0.467	0.470
N-N (75.23%)	**0.219**	**0.228**	**0.237**	**0.242**
All types (100%)	0.287	0.301	0.338	0.360

Table 1: MRR scores on FB15k-237 by relation types.

To address the above issue, some methods focus on learning probabilistic embeddings, where the predictions represent

*Corresponding author.
Copyright © 2024, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

a specific distribution rather than a deterministic value. For example, (Xiao et al. 2015; He et al. 2015) and (Feng et al. 2021) proposed density-based embeddings, where each entity/relation is represented by a multi-dimensional Gaussian distribution and a mixture of Gaussian distributions respectively. They use the mean vector μ to indicate its position and the covariance metric Σ to represent the corresponding uncertainty. The prediction results are no longer deterministic values but a distribution, which allows for the demonstration of multiple semantic relations and diversity of target entities. However, learning probabilistic embeddings often requires calculating the inverse of the large matrix and solving a linear system, which demands huge memory and is time-consuming. How to efficiently capture the multiple relation semantics with KG embeddings is still a challenging problem.

Instead of learning complex probabilistic embeddings, this paper proposes to model the multiple relation semantics in prediction for KGE methods. Our key idea is to cast entity prediction into the task of conditional entity generation. That is, we generate the predicted entities conditioned on the given entities and relations in the vector space. To this end, this paper proposes a novel conditional diffusion model, namely the Knowledge Graph Diffusion Model (KGDM) to estimate the probabilistic distribution of target entities in prediction. To capture connectivity patterns of KGs, this paper introduces a Conditional Entity Denoiser module to generate target entities conditioned on given entities and relations. A major obstacle to knowledge graph diffusion is that diffusion processes typically operate in continuous space, while entities and relations are inherently discrete. Though there are some discrete diffusion-like approaches (Dhariwal and Nichol 2021), they cannot utilize the guidance, which drastically impresses the diffusion model's sample quality. To address this gap, we conduct diffusion directly in a vector space and define two learnable embedding functions: EMB^e, and EMB^r that map entities and relations to vectors. Our contributions can be summarized as follows:

- We propose a simple yet efficient framework namely KGDM to capture the multiple relation semantics in prediction for KGE methods. It directly learns the probabilistic distribution of target entities through Denoising Diffusion Probabilistic Models (DDPM) and casts the entity prediction task into the conditional fact generation task. To the best of our knowledge, KGDM is the first attempt to explore the potential of diffusion models in knowledge graph reasoning.
- A Conditional Entity Denoiser module is proposed to generate a target entity conditioned on given entities and relations. Furthermore, two embedding functions for entities and relations are defined to bridge the gap between continuous diffusion models and discrete KGs.
- Extensive experiments on four benchmark datasets demonstrate that KGDM achieves superior performances in KGC tasks and significantly outperforms all types of state-of-the-art methods on three datasets. In FB15k-237, KGDM achieves up to a 25% relative improvement over the state-of-the-art methods.

Related Work

KG embedding methods: KG embedding, which aims to encode entities and relations into a continuous vector space. The general intuition of these methods is to model and infer the connectivity patterns (i.e., symmetry/antisymmetry, inversion, and composition) in knowledge graphs according to the observed knowledge facts. Most KGE methods (Bordes et al. 2013; Sun et al. 2019; Yang et al. 2014; Cao et al. 2022) focus on defining a relation-dependent scoring function $f_r(\boldsymbol{h}, \boldsymbol{t})$ in the general or design space to model these patterns. For example, TransE (Bordes et al. 2013), which represents relations as translations, can model the inversion and composition patterns. RotatE (Sun et al. 2019), which represents entities as points in a complex space and relations as rotations, can model relation patterns including symmetry/antisymmetry, inversion, and composition. Meanwhile, some other works consider how to model the multiple relation semantics and diversity of target entities to improve the performance of embeddings. Translation-based model (Wang et al. 2014) models a relation as a hyperplane to tackle entity pairs in one-to-many form. However, they can not deal with the issue of multiple relation semantics. Density-based models (Xiao et al. 2015) and (He et al. 2015) also use Gaussian distributions to represent entities while drawing a mixture of Gaussian distributions for relations to represent the multiple relation semantics. However, these density-based embedding methods often make complex designs on embeddings, which often need to calculate the inverse of the large matrix and solve a linear system, which requires huge memory and is very time-consuming.

Diffusion Model: The diffusion model uses diffusion processes to model the generation and defines the sampling of data as the process of gradually denoising it from a complete Gaussian noise. The forward process gradually adds Gaussian noise to the data from a predefined noise schedule until the time step T. In recent years, the class of diffusion-based (or score-based) deep generative models has demonstrated its outstanding performance in modeling high-dimensional multi-modal distributions (Ho, Jain, and Abbeel 2020; Sohl-Dickstein et al. 2015), and the capability of generating high-quality and diverse samples (Dhariwal and Nichol 2021; Rombach et al. 2022) on several benchmark generation tasks in the field of computer vision (Dhariwal and Nichol 2021). Additionally, to handle discrete data, past works have studied text diffusion models on discrete state spaces, which defines a corruption process on discrete data (Austin et al. 2021). And some works (Li et al. 2022; Gong et al. 2022) focus on continuous diffusion models for text generation. The reverse process uses a neural backbone often implemented as a U-Net (Dhariwal and Nichol 2021; Ho, Jain, and Abbeel 2020) or transformer (Li et al. 2022; Gong et al. 2022) to parameterize the conditional distribution $p(x_{t-1}|x_t)$. However, knowledge graphs are often stored in the form of triplets (h, r, t), which is different from images or text. They have shorter lengths and less obvious long-range dependencies. So, in this work, we propose an MLP-based Conditional Entity Denoiser (CEDenoiser) for KG data to learn the reverse diffusion process.

Methodology

Problem Setup

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be an instance of a knowledge graph, where $\mathcal{V}$ is the set of nodes and $\mathcal{E}$ is the set of edges. Each edge e has a relation type $r \in \mathcal{R}$. We aim to predict the missing entities in $\mathcal{G}$, i.e., given an incomplete triplet $(h, r, ?)$, we aim to predict the missing tail entity t. Noticing that the problem $(?, r, t)$ is the same, this paper only discusses $(h, r, ?)$. The traditional KGE methods carry out entity prediction through a ranking procedure. Specifically, to predict the tail entity of an incomplete fact triple $(h, r, ?)$, it makes the prediction $\hat{\boldsymbol{t}}$ by relation-dependent score function $f_r(\boldsymbol{h})$ in vector space. Then ranking the distance between $\hat{\boldsymbol{t}}$ and each entity $\boldsymbol{t}$ in KGs to get the answer. While KGDM uses the Conditional Entity Denoiser to generate the predicted tail entity $\boldsymbol{X}^t$ conditioned on the given head entity $\boldsymbol{X}^h$ and relation $\boldsymbol{X}^r$ in vector space. And then, it does the same ranking process to get the final answer. The following sections will provide a detailed introduction to the architecture of KGDM and its training objectives.

The KGDM Architecture

Figure 1 illustrates the architecture of the KGDM. From a high-level perspective, KGDM can be divided into two stages: forward process and reverse process with conditional denoising. Specifically, KGDM learns to model the Markov transition from Gaussian distribution to the distribution of the target entities in vector space. In the inference stage, it uses the known entity and relation to guide the reverse process to generate target entities in vector space. Next, we will provide a detailed introduction to these two processes.

Forward Process. To apply a continuous diffusion model to discrete entities. We define two learnable embedding functions: EMB^e and EMB^r, which both are linear layers. They maps each entity and relation to vectors $\boldsymbol{X}^e \in \mathbb{R}^e$, $\boldsymbol{X}^r \in \mathbb{R}^r$. Then, for the training triplet (h, r, t), given the ground-truth tail entity embedding $\boldsymbol{X}^t$ and its condition embedding $\boldsymbol{X}^h$ and $\boldsymbol{X}^r$. Let $q(\boldsymbol{X}^t|\boldsymbol{X}^h, \boldsymbol{X}^r)$ be the unknown target entities' distribution in vector space. First, KGDM defines the forward diffusion process $q(\boldsymbol{X}^t_{T_t}|\boldsymbol{X}^t_{T_t-1})$ which maps the embedding of tail entity to pure noise by gradually adding Gaussian noise at each time step $T_t = i$ until at diffusion step $T_t = T$. The forward transition is parametrized by:

$$q(\boldsymbol{X}^t_{T_t}|\boldsymbol{X}^t_{T_t-1}) = \mathcal{N}(\boldsymbol{X}^t_{T_t}; \sqrt{1-\beta_{T_t}}\boldsymbol{X}^t_{T_t-1}, \beta_{T_t}\boldsymbol{I}), \quad (1)$$

where $\{\beta_{T_t}\}_{T_t=1}^T$ are forward process variances.

Reverse Process with Conditional Denoising. In the second stage, KGDM defines the conditional reverse diffusion process $p(\boldsymbol{X}^t_{T_t-1}|\boldsymbol{X}^t_{T_t}, \boldsymbol{X}^h, \boldsymbol{X}^r)$ which performs iterative denoising from pure Gaussian noise to generate target entities in vector space conditioned on the known entity embedding $\boldsymbol{X}^h$ and relation embedding $\boldsymbol{X}^r$:

$$p_\theta(\boldsymbol{X}^t_{T_t-1}|\boldsymbol{X}^t_{T_t}, \boldsymbol{X}^h, \boldsymbol{X}^r) = \mathcal{N}(\boldsymbol{X}^t_{T_t-1}; \mu_\theta(\boldsymbol{X}^t_{T_t}, T_t, \boldsymbol{X}^h, \boldsymbol{X}^r), \sigma^2_{T_t}\boldsymbol{I}), \quad (2)$$

where σ_{T_t} is the constant variance following (Ho, Jain, and Abbeel 2020), μ_θ is the mean of the Gaussian distribution computed by a neural network, and θ is the parameters of the denoise model. As shown in (Ho, Jain, and Abbeel 2020), we can reparameterize the mean to make the neural network learn the added noise at time step T_t instead. In this way, μ_θ can be reparameterized as follows:

$$\mu_\theta(\boldsymbol{X}^t_{T_t}, T_t, \boldsymbol{X}^h, \boldsymbol{X}^r) = \frac{1}{\sqrt{\alpha_{T_t}}}(\boldsymbol{X}^t - \frac{\beta_{T_t}}{\sqrt{1-\bar{\alpha}_{T_t}}}\epsilon_\theta(\boldsymbol{X}^t_{T_t}, T_t, \boldsymbol{X}^h, \boldsymbol{X}^r)), \quad (3)$$

where T_t is the time step, $\{\beta_{T_t}\}_{T_t=1}^T$ are forward process variances, $\alpha_{T_t} = 1 - \beta_{T_t}$, and $\bar{\alpha}_{T_t} = \prod_{s-1}^{T_t} \alpha_s$. $\epsilon_\theta(\boldsymbol{X}^t_{T_t}, T_t, \boldsymbol{X}^h, \boldsymbol{X}^r)$ is the designed CEDenoiser to predict the added noise conditioned on known condition embeddings at time step T_t.

Conditional Entity Denoiser

Currently, the backbones of most existing denoising models are designed for image or text data. While knowledge graphs are often stored in the form of triplets (h, r, t), which have a short length and their long-range dependencies are not apparent. So, instead of using transformers, we propose a simple yet efficient MLP-based Conditional Entity Denoiser (CEDenoiser). We conduct ablation studies in the following sections to demonstrate that transformers are not suitable. The architecture of CEDenoiser is illustrated in Figure 1(b). Formally, the architecture of CEDenoiser can be described as follows:

$$\boldsymbol{X^c} = ScoringModule(\boldsymbol{X}^h, \boldsymbol{X}^r), \quad (4)$$

$$\boldsymbol{E} = CEDenoiserBlock(\boldsymbol{X}^{t'}, \boldsymbol{X}^{T_t}, \boldsymbol{X^c}), \quad (5)$$

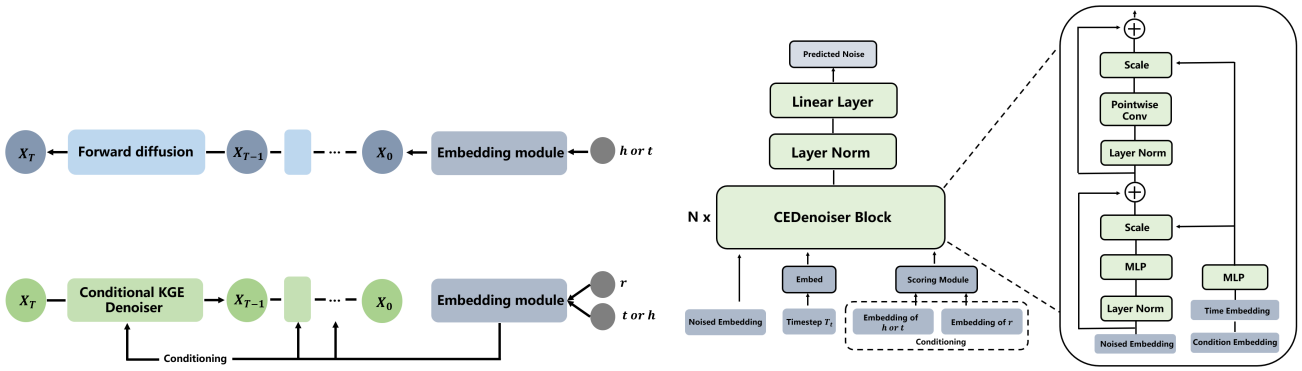
$$\boldsymbol{\epsilon} = LinearLayer(LN(\boldsymbol{E})), \quad (6)$$

where $\boldsymbol{X}^h$ and $\boldsymbol{X}^r$ are the embedding of entity h and relation r, $\boldsymbol{X^c}$ is the final condition embedding calculated by Scoring Module, and $\boldsymbol{X}^{t'}$ is the noised tail embedding. $\boldsymbol{X}^{T_t}$ denotes the timestep embedding at step T_t. $\boldsymbol{E}$ is intermediate feature calculated by the CEDenoiser block, and $\boldsymbol{\epsilon}$ is the noise predicted by CEDenoiser. Next, we will introduce the Scoring Module and the CEDenoiser block.

Scoring Module. To generate the target tail entity from the noise in vector space, CEDenoiser uses the known embeddings $(\boldsymbol{X}^h, \boldsymbol{X}^r)$ to guide the generative process in DDPM. After the vectorization of conditions, most of the previous works (Li et al. 2022; Ho and Salimans 2022) concatenate the different conditional embeddings simply and use them as the final control condition. However, for the triplets in KGs, the entities and relations that serve as conditions usually have rich patterns, which are not independent of each other. Inspired by the existing KGE methods on modeling the patterns among the entities and relations. By defining the relation-dependent score functions equation (7) and equation (8) from (Bordes et al. 2013; Sun et al. 2019), the Scoring Module can better capture the patterns among conditions and utilize them to guide the generation process:

$$ScoringModule(\boldsymbol{X}^h, \boldsymbol{X}^r) = \boldsymbol{X}^h + \boldsymbol{X}^r, \quad (7)$$

$$ScoringModule(\boldsymbol{X}^h, \boldsymbol{X}^r) = \boldsymbol{X}^h \circ \boldsymbol{X}^r, \quad (8)$$



(a) KGDM Architecture (b) Conditional Entity Denoiser

Figure 1: (a) Architecture of KGDM. It consists of a forward diffusion process and a reverse process modeled by a Conditional Entity Denoiser. (b) Architecture of Conditional Entity Denoiser (CEDenoiser).

where $\circ$ denotes the Hadamard product. Noticing that different score functions model different patterns among entities and relations. So, we use the equation (7) in FB15k-237 which contains a large number of composition patterns, and use equation (8) in WN18RR, Kinship, and UMLS which contains many symmetry patterns. In the following experiments, we conduct ablation studies to demonstrate that the performance of simply concatenate conditional embeddings is far inferior compared to that of using score functions.

CEDenoiser Block. Inspired by the success of the transformer encoder (Vaswani et al. 2017) in the e graph data domain (Hu et al. 2020), the CEDenoiser Block adopts a similar architecture. It consists of alternating layers of MLP. Layernorm(LN) is applied before every layer and we employ residual connections around each of the sub-layers (Wang et al. 2019). However, since the form of triplets (h, r, t) is simple, with a short length, and their long-range dependencies are not apparent. CEDenoiser employs simple MLP layers rather than multiheaded self-attention layers. Furthermore, to make full use of the conditional embeddings to guide generation, we regress dimension-wise scaling parameters α which are applied immediately prior to any residual connections (Peebles and Xie 2022) within the sub-layers as shown in Figure 1(b).

Training and Inference

Since negative sampling has been proven quite effective for both learning knowledge graph embeddings (Trouillon et al. 2016) and word embeddings (Mikolov et al. 2013). To train the model, we use a loss function similar to the negative sampling loss (Mikolov et al. 2013):

$$L = -log\ \sigma(\gamma - d_1(\boldsymbol{X}^t, Denoise(\boldsymbol{X}^t)) - \sum_{i=1}^{n} \frac{1}{k} log\ \sigma(d_1(\boldsymbol{X}^t, Denoise(\boldsymbol{X}^{t_i^{'}})) - \gamma), \quad (9)$$

where γ is a fixed margin, σ is the sigmoid function, and the d_1 is the $L1$ distance. The predicted noise and the final denoised results can convert to each other (Ho, Jain, and Abbeel 2020) by $Denoise(\boldsymbol{X}^t) = \frac{1}{\sqrt{\bar{\alpha}_{T_t}}} \boldsymbol{X}^t_{T_t} - \sqrt{\frac{1}{\bar{\alpha}_{T_t}} - 1}\ \epsilon_\theta(\boldsymbol{X}^t_{T_t}, T_t, \boldsymbol{X}^h, \boldsymbol{X}^r)$. And $(h, r, t_i^{'})$ is the i-th negative triplet for the tail entity prediction on positive sample (h, r, t). For the head prediction, we replace the corresponding $h_i^{'}$. During the inference stage, when predicting $(h, r, ?)$, KGDM uses the trained CEDenoiser and the corresponding trained conditions (known embedding of the entity $\boldsymbol{X}^h$ and relation $\boldsymbol{X}^r$) to perform iterative denoising from pure Gaussian noise to target entity $\boldsymbol{X}^t$ in vector space. The training and inference algorithms of KGDM are illustrated in Algorithm 1 and Algorithm 2 respectively.

Algorithm 1: Training Stage

Input: (h, r, t), $(h, r, t^{'})$;
Parameters: EMB^e, EMB^r, $CEDenoiser$: ϵ_θ;
repeat
  Calculate $\boldsymbol{X}^h$, $\boldsymbol{X}^r$, $\boldsymbol{X}^t$, $\boldsymbol{X}^{t^{'}}$ by EMB^e, EMB^r;
  $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$;
  $T_t \sim$ Uniform $(\{1, \cdots, T\})$;
  $\boldsymbol{X}^t_{T_t} = \sqrt{\bar{\alpha}_{T_t}} \boldsymbol{X}^t + \sqrt{1 - \bar{\alpha}_{T_t}} \boldsymbol{\epsilon}$;
  $Denoise(\boldsymbol{X}^t) =$
  $\frac{1}{\sqrt{\bar{\alpha}_{T_t}}} \boldsymbol{X}^t_{T_t} - \sqrt{\frac{1}{\bar{\alpha}_{T_t}} - 1}\ \epsilon_\theta(\boldsymbol{X}^t_{T_t}, T_t, \boldsymbol{X}^h, \boldsymbol{X}^r)$;
  Take the gradient descent step on:
  $L = -\log \sigma(\gamma - d_1(\boldsymbol{X}^t, Denoise(\boldsymbol{X}^t)) -$
  $\sum_{i=1}^{n} \frac{1}{k} \log \sigma(d_1(\boldsymbol{X}^t, Denoise(\boldsymbol{X}^{t_i^{'}})) - \gamma)$;
until *converged*;

Experiment

Experiment Setup

Datasets: We select four typical KGC datasets for evaluation, including FB15k-237 (Toutanova and Chen 2015), WN18RR (Dettmers et al. 2018), Kinship and UMLS. For Kinship and UMLS, we use the datasets division in (Qu et al. 2020). Statistics of datasets can be found in the appendix.
Baselines: We compared with the four types of KGC methods following (Cui and Chen 2022). **Knowledge graph embed-**

Algorithm 2: Inference Stage

Input: Incomplete triplet $(h, r, ?)$;
Output: Predicted target entity embedding $\boldsymbol{X}_0^t$;
$\boldsymbol{X}_{T_t}^t \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$;
$\boldsymbol{X}^h \leftarrow EMB^e(h)$, $\boldsymbol{X}^r \leftarrow EMB^r(r)$;
for $T_t = T, \cdots, 1$ **do**
  $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $T_t > 1$, else $\mathbf{z} = \mathbf{0}$;
  $\boldsymbol{X}_{T_t-1}^t = \frac{1}{\sqrt{\alpha_{T_t}}}(\boldsymbol{X}_{T_t}^t - \frac{1-\alpha_{T_t}}{\sqrt{1-\bar{\alpha}_{T_t}}}\boldsymbol{\epsilon}_\theta(\boldsymbol{X}_{T_t}^t, T_t, \boldsymbol{X}^h, \boldsymbol{X}^r)) + \sigma_{T_t}\mathbf{z}$;
end
return $\boldsymbol{X}_0^t$

ding methods: TransE (Bordes et al. 2013), DualE (Cao et al. 2021), DistMult (Yang et al. 2014), ComplEx (Trouillon et al. 2016), ComplEx-N3 (Lacroix, Usunier, and Obozinski 2018), KG2GM (Feng et al. 2021), KG2E (He et al. 2015), TuckER (Balažević, Allen, and Hospedales 2019), ConvE (Dettmers et al. 2018), RotatE (Sun et al. 2019), HAKE (Zhang et al. 2020), ATTH (Chami et al. 2020), SEA (Gregucci et al. 2023), AnKGE-HAKE (Yao et al. 2023). **Path-based methods**: RNNLogic (Qu et al. 2020), NeuralLP(Yang, Yang, and Cohen 2017), DRUM (Sadeghian et al. 2019), PathRank (Lee et al. 2013), MINERVA (Das et al. 2017), and M-Walk(Shen et al. 2018). **Graph neural networks methods**: NBFNet (Zhu et al. 2021), COMPGCN (Vashishth et al. 2019) and RGCN (Schlichtkrull et al. 2018) and **Instance-based learning methods**: IBLE and CIBLE (Cui and Chen 2022)

Evaluation Protocols: For each test triplet (h, r, t), we construct two queries: $(h, r, ?)$ and $(?, r, t)$, with the answers t and h. The Mean Rank (MR), Mean Reciprocal Rank (MRR), and H@N are reported under the filtered setting (Sun et al. 2019), in line with previous research.

Implementation details: Our model is trained on 2 Nvidia Quadro RTX 8000 GPU. We describe the hyper-parameter, architectures, and more experimental details in the appendix. The source code of this paper can be obtained from https://github.com/key2long/KGDM.

Main Results

The main results are presented in Tables 2, and Table 3. We categorize the existing KGC methods into two main groups, non-embedding methods and embedding methods. The non-embedding methods are listed in the upper section of the table, while the embedding methods are listed in the lower section of the table. Our observations based on the results are as follows. First, compared to embedding methods, KGDM shows remarkable improvement across all metrics on four datasets. Specifically, it achieves a 15.8% (43.6% relative), 1.6% (3.2% relative), 13.8% (21.1% relative), and 11.8% (14.9% relative) increase in MRR scores over the best embedding models on the FB15k-237, WN18RR, Kinship, and UMLS datasets, respectively. Second, compared to non-embedding methods, KGDM achieves better results for all metrics on the FB15k-237, Kinship, and UMLS datasets. On the WN18RR, KGDM retains its superiority over other non-embedding methods except for the NBFNet. Specifically, KGDM achieves a 10.5% (25.3% relative), 6.1% (8.3% relative), and 6.7% (8.0% relative) increase in MRR scores over the best non-embedding models on the FB15k-237, Kinship, and UMLS datasets, respectively. In conclusion, these results illustrate that by modeling the multiple relation semantics and distribution of target entities in prediction, KGDM can improve the performance of embedding methods greatly and explore the potential of embedding methods in KGC tasks. Furthermore, we notice that the performance improvement of KGDM on the WN18RR is not significant. We analyze that it is caused by the higher entity-to-relation ratio (the number of entities/ the number of relations) on WN18RR(40943/11) than the other three datasets: FB15k-237 (14541/237), UMLS (135/46), and Kinship (104/25). The larger the entity-to-relation ratio implies more difficult to model the distribution of target entities.

Next, We break down the performance of KGDM by the categories of relations (Wang et al. 2014) in the FB15k-237. Table 4 shows the prediction MRR scores for each category (the results of tail pred mode can be found in the appendix). It is observed that KGDM shows a greater relative improvement in the 1-N and N-N types. Specifically, it achieves a 5.6% (11.2% relative), 14.8% (56.9% relative), 12.8% (21.3% relative), and 19.0% (40.9% relative) increase in MRR scores over the best embedding models on the 1-1, 1-N, N-1, N-N types. These results also illustrate that the multiple relation semantics in KGs often degrades the performance of KGE methods. KGDM can effectively alleviate this issue and perform better on 1-N and N-N types.

Ablation Study

Scoring module of the CEDenoiser. As mentioned above, to better leverage conditions to guide the generative process in conditional DDPM. We design a Scoring Module to deal with the embeddings of conditions rather than simply concatenating them. In this subsection, we analyze the necessity of the Scoring Module in the FB15k-237 dataset. We compare the results between using the Scoring Module and directly concatenating the conditions without using the Scoring Module. The contributions of the Scoring Module are summarized in Table 5. More details on the ablation study are provided in the appendix. It can be observed that without the Scoring Module, the performance of KGDM drops by about 15% on average, illustrating that the entities and relations in KGs are not independent of each other. Simply concatenating their embeddings cannot capture the patterns in the triplets. By defining the relation-dependent score functions, the Scoring Module can better utilize the conditional information in the triplets to guide the generation. The CEDenoiser can generate higher-quality answers that are more aligned with the probabilistic distribution of target entities.

MLP-based architecture of the CEDenoiser. Furthermore, we conduct an ablation study to prove that an MLP-based architecture is more suitable for KGs in conditional DDPM. We replace the MLP layers in the CEDenoiser block with two transformer layers. The results in Table 6 show that the performance of KGDM decreases by nearly 35% when using transformer layers in the CEDenoiser block, indicating that

Model	FB15k-237				WN18RR			
	MRR	H@1	H@3	H@10	MRR	H@1	H@3	H@10
Non-embedding methods								
PathRank	0.087	7.4	9.2	11.2	0.189	17.1	20.0	22.5
M-Walk	0.232	16.5	24.3	-	0.437	41.4	44.5	-
NeuralLP	0.237	17.3	25.9	36.1	0.381	36.8	38.6	40.8
DRUM	0.238	17.4	26.1	36.4	0.382	36.9	38.8	41.0
RNNLogic	0.344	25.2	38.0	53.0	0.483	44.6	49.7	55.8
RGCN	0.273	18.2	30.3	45.6	0.402	34.5	43.7	49.4
COMPGCN	0.355	26.4	39.0	53.5	0.479	44.3	49.4	54.6
NBFNet	<u>0.415</u>	<u>32.1</u>	<u>45.4</u>	<u>59.9</u>	**0.551**	**49.7**	**57.3**	**66.6**
IBLE	0.284	20.0	31.0	45.2	0.418	40.6	42.2	44.3
CIBLE	0.341	24.5	37.7	53.7	0.490	44.6	50.7	57.5
Embedding methods								
TransE	0.294	-	-	46.5	0.226	-	-	50.1
KG2E	0.108	4.2	11.5	24.9	0.054	0.70	7.70	13.3
ConvE	0.325	23.7	35.6	50.1	0.430	40.0	44.0	52.0
RotatE	0.338	24.1	37.5	53.3	0.476	42.8	49.2	57.1
HAKE	0.349	25.2	38.5	54.5	0.496	45.2	51.3	58.0
ATTH	0.348	25.2	38.4	54.0	0.486	44.3	49.9	57.3
DualE	0.365	26.8	40.0	55.9	0.492	44.4	51.3	58.4
KG2GM	0.168	9.0	18.2	32.5	0.087	0.46	18.2	29.6
SEA	0.360	26.4	39.8	54.9	0.500	45.4	51.8	59.1
AnKGE-HAKE	0.385	28.8	42.8	57.2	0.500	45.4	51.5	58.7
KGDM(Ours)	**0.520**	**42.3**	**56.6**	**70.8**	<u>0.516</u>	<u>45.7</u>	<u>51.9</u>	<u>59.3</u>

Table 2: Entity prediction results on FB15k-237 and WN18RR. The best results are in bold and the second best results are underlined.

Model	Kinship				UMLS			
	MRR	H@1	H@3	H@10	MRR	H@1	H@3	H@10
Non-embedding methods								
PathRank	0.369	27.2	41.6	67.3	0.197	14.8	21.4	25.2
NeuralLP	0.302	16.7	33.9	59.6	0.483	33.2	56.3	77.5
MINERVA	0.401	23.5	46.7	76.6	0.564	42.6	65.8	81.4
DRUM	0.334	18.3	37.8	67.5	0.548	35.8	69.9	85.4
RNNLogic	0.722	59.8	81.4	94.9	<u>0.842</u>	<u>77.2</u>	89.1	96.5
NBFNet	0.606	43.5	72.5	93.7	0.778	68.8	84.0	93.8
IBLE	0.615	45.9	71.7	92.8	0.816	71.7	<u>90.0</u>	96.1
CIBLE	<u>0.728</u>	<u>60.3</u>	<u>82.0</u>	<u>95.6</u>	0.831	74.9	89.7	<u>97.0</u>
Embedding methods								
DistMult	0.241	15.5	26.3	41.9	0.430	39.0	44.0	49.0
ComplEx	0.247	15.8	27.5	42.8	0.440	41.0	46.0	51.0
ComplEx-N3	0.605	43.7	71.0	92.1	0.791	68.9	87.3	95.7
TuckER	0.603	46.2	69.8	86.3	0.732	62.5	81.2	90.9
RotatE	0.651	50.4	75.5	93.2	0.744	63.6	82.2	93.9
KGDM(Ours)	**0.789**	**68.7**	**87.0**	**97.2**	**0.909**	**87.2**	**93.7**	**97.3**

Table 3: Entity prediction results on Kinship and UMLS. The best results are in bold and the second best results are underlined.

Model	Head Pred			
	1-1	1-N	N-1	N-N
TransE	0.498	0.079	0.455	0.224
TransG	0.489	0.078	0.458	0.228
RotatE	0.487	0.081	0.467	0.234
WGCN	0.422	0.093	0.454	0.261
COMPGCN	0.457	0.112	0.471	0.275
KGDM	**0.554**	**0.260**	**0.599**	**0.465**

Table 4: MRR scores by relation category in FB15k-237.

Model	FB15k-237		Kinship	
	MRR	H@1	MRR	H@1
KGDM w/o Scoring Module	0.449	35.0	0.635	47.9
KGDM w/ Scoring Module	**0.520**	**42.3**	**0.789**	**68.7**

Table 5: Ablation on Scoring Module of the CEDenoiser in FB15k-237 and Kinship.

the MLP layers are more suitable for modeling simple-form triplets in knowledge graphs.

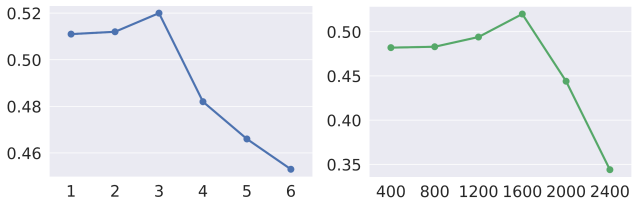
Hyper-parameters. Next, we conduct ablation experiments on hyper-parameters, including the number of the CEDenoiser block and the hidden size of the MLP layer in the CEDenoiser block. Figure 2 reports the results on FB15k-237 in terms of MRR scores. In Figure 2(a), we test the performance of different numbers of CEDenoiser blocks. It suggests that a shallow and proper number (3 in this case) is essential for KGDM. Because the inputs of the CEDenoiser are relatively simple, a very deep network may suffer from overfitting and cause performance drops. Figure 2(b) studies the influence of the hidden size of the MLP layer in the CEDenoiser block. We observe that the increase of hidden size helps to improve the performance of the model, but too large size (e.g., 2400) will harm the performance.

Case Study

Finally, we explore how KGDM performs better on triplets of the 1-N category. We provide an example from FB15k-237. The triplet to be predicted is (road running, /olympics/olympic_sport/country, ?). For this question, there are six countries as answers in the test dataset. We visualize the true entity embeddings and prediction embeddings calculated by the KGDM in Figure 3(a), and prediction results calculated by trained TransE in Figure 3(b). From this visualization, we can observe that the embedding predicted by TransE is a deterministic "point" in the vector space, which is calculated by $f_{\boldsymbol{r}}^{TransE}(\boldsymbol{t}) = \boldsymbol{h} + \boldsymbol{r}$. In fact, the embedding predicted by TransE can be seen as the "average" of the candidate answers, which neglects the diversity of target entities. So, this "average result" is usually far away from the truth entities. However, due to the KGDM models the distribution of target entities in prediction, the entities generated based on conditioned entities and relations are not unique and most of them are closer to true entities.

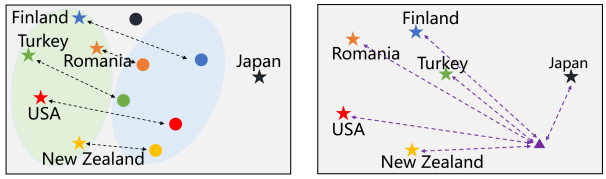
Model	FB15k-237		Kinship	
	MRR	H@1	MRR	H@1
KGDM w/ transformer-based	0.334	24.0	0.661	52.2
KGDM w/ MLP-based	**0.520**	**42.3**	**0.789**	**68.7**

Table 6: Ablation on MLP-based architecture of the CEDenoiser in FB15k-237 and Kinship.



(a) Num of CEDenoiser Blocks (b) Hidden Size

Figure 2: Hyper-parameters analysis on FB15k-237 (MRR).



(a) KGDM Predictions (b) TransE Predictions

Figure 3: T-SNE visualization to the predictions of (road running, /olympics/olympic_sport/country, ?) calculated by KGDM and TransE.

Conclusion

To better solve KGC tasks by considering the multiple relation semantics in KGs, this paper proposes a novel conditional diffusion model, namely the Knowledge Graph Diffusion Model (KGDM) that can model the multiple relation semantics in prediction for KGE methods. Different from existing methods of learning probabilistic embeddings, KGDM throws the entities prediction task into the conditional entities generation task and models the probabilistic distribution of target entities. Extensive experiments demonstrate that KGDM significantly outperforms existing state-of-the-art methods on benchmark datasets for KGC tasks.

Acknowledgements

This work was supported by the National Natural Science Foundation of China under Grant No.U20B2070, No.61976199, No.61836011 and No.72293573.

References

Austin, J.; Johnson, D. D.; Ho, J.; Tarlow, D.; and van den Berg, R. 2021. Structured denoising diffusion models in discrete state-spaces. *Advances in Neural Information Processing Systems*, 34: 17981–17993.

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

Bordes, A.; Usunier, N.; Garcia-Duran, A.; Weston, J.; and Yakhnenko, O. 2013. Translating embeddings for modeling multi-relational data. *Advances in neural information processing systems*, 26.

Cao, Z.; Xu, Q.; Yang, Z.; Cao, X.; and Huang, Q. 2021. Dual quaternion knowledge graph embeddings. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, 6894–6902.

Cao, Z.; Xu, Q.; Yang, Z.; Cao, X.; and Huang, Q. 2022. Geometry interaction knowledge graph embeddings. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, 5521–5529.

Chami, I.; Wolf, A.; Juan, D.-C.; Sala, F.; Ravi, S.; and Ré, C. 2020. Low-dimensional hyperbolic knowledge graph embeddings. *arXiv preprint arXiv:2005.00545*.

Cui, W.; and Chen, X. 2022. Instance-based Learning for Knowledge Base Completion. *Advances in Neural Information Processing Systems*, 35: 30744–30755.

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

Dettmers, T.; Minervini, P.; Stenetorp, P.; and Riedel, S. 2018. Convolutional 2d knowledge graph embeddings. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32.

Dhariwal, P.; and Nichol, A. 2021. Diffusion models beat gans on image synthesis. *Advances in Neural Information Processing Systems*, 34: 8780–8794.

Feng, W.; Zha, D.; Guo, X.; Dong, Y.; and He, Y. 2021. Representing Knowledge Graphs with Gaussian Mixture Embedding. In *Knowledge Science, Engineering and Management: 14th International Conference, KSEM 2021, Tokyo, Japan, August 14–16, 2021, Proceedings, Part I*, 166–178. Springer.

Gong, S.; Li, M.; Feng, J.; Wu, Z.; and Kong, L. 2022. Diffuseq: Sequence to sequence text generation with diffusion models. *arXiv preprint arXiv:2210.08933*.

Gregucci, C.; Nayyeri, M.; Hernández, D.; and Staab, S. 2023. Link prediction with attention applied on multiple knowledge graph embedding models. In *Proceedings of the ACM Web Conference 2023*, 2600–2610.

Hao, Y.; Zhang, Y.; Liu, K.; He, S.; Liu, Z.; Wu, H.; and Zhao, J. 2017. An end-to-end model for question answering over knowledge base with cross-attention combining global knowledge. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 221–231.

He, S.; Liu, K.; Ji, G.; and Zhao, J. 2015. Learning to represent knowledge graphs with gaussian embedding. In *Proceedings of the 24th ACM international on conference on information and knowledge management*, 623–632.

Ho, J.; Jain, A.; and Abbeel, P. 2020. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33: 6840–6851.

Ho, J.; and Salimans, T. 2022. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*.

Hu, Z.; Dong, Y.; Wang, K.; and Sun, Y. 2020. Heterogeneous graph transformer. In *Proceedings of the web conference 2020*, 2704–2710.

Lacroix, T.; Usunier, N.; and Obozinski, G. 2018. Canonical tensor decomposition for knowledge base completion. In *International Conference on Machine Learning*, 2863–2872. PMLR.

Lee, S.; Park, S.; Kahng, M.; and Lee, S.-g. 2013. PathRank: Ranking nodes on a heterogeneous graph for flexible hybrid recommender systems. *Expert Systems with Applications*, 40(2): 684–697.

Li, X.; Thickstun, J.; Gulrajani, I.; Liang, P. S.; and Hashimoto, T. B. 2022. Diffusion-lm improves controllable text generation. *Advances in Neural Information Processing Systems*, 35: 4328–4343.

Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G. S.; and Dean, J. 2013. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 26.

Peebles, W.; and Xie, S. 2022. Scalable Diffusion Models with Transformers. *arXiv preprint arXiv:2212.09748*.

Qu, M.; Chen, J.; Xhonneux, L.-P.; Bengio, Y.; and Tang, J. 2020. Rnnlogic: Learning logic rules for reasoning on knowledge graphs. *arXiv preprint arXiv:2010.04029*.

Rombach, R.; Blattmann, A.; Lorenz, D.; Esser, P.; and Ommer, B. 2022. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 10684–10695.

Sadeghian, A.; Armandpour, M.; Ding, P.; and Wang, D. Z. 2019. Drum: End-to-end differentiable rule mining on knowledge graphs. *Advances in Neural Information Processing Systems*, 32.

Schlichtkrull, M.; Kipf, T. N.; Bloem, P.; Van Den Berg, R.; Titov, I.; and Welling, M. 2018. Modeling relational data with graph convolutional networks. In *The Semantic Web: 15th International Conference, ESWC 2018, Heraklion, Crete, Greece, June 3–7, 2018, Proceedings 15*, 593–607. Springer.

Shen, Y.; Chen, J.; Huang, P.-S.; Guo, Y.; and Gao, J. 2018. M-walk: Learning to walk over graphs using monte carlo tree search. *Advances in Neural Information Processing Systems*, 31.

Sohl-Dickstein, J.; Weiss, E.; Maheswaranathan, N.; and Ganguli, S. 2015. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*, 2256–2265. PMLR.

Sun, Z.; Deng, Z.-H.; Nie, J.-Y.; and Tang, J. 2019. Rotate: Knowledge graph embedding by relational rotation in complex space. *arXiv preprint arXiv:1902.10197*.

Toutanova, K.; and Chen, D. 2015. Observed versus latent features for knowledge base and text inference. In *Proceedings of the 3rd workshop on continuous vector space models and their compositionality*, 57–66.

Trouillon, T.; Welbl, J.; Riedel, S.; Gaussier, É.; and Bouchard, G. 2016. Complex embeddings for simple link prediction. In *International conference on machine learning*, 2071–2080. PMLR.

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

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Wang, Q.; Li, B.; Xiao, T.; Zhu, J.; Li, C.; Wong, D. F.; and Chao, L. S. 2019. Learning deep transformer models for machine translation. *arXiv preprint arXiv:1906.01787*.

Wang, Z.; Zhang, J.; Feng, J.; and Chen, Z. 2014. Knowledge graph embedding by translating on hyperplanes. In *Proceedings of the AAAI conference on artificial intelligence*, volume 28.

Xiao, H.; Huang, M.; Hao, Y.; and Zhu, X. 2015. TransG: A generative mixture model for knowledge graph embedding. *arXiv preprint arXiv:1509.05488*.

Xiong, C.; Power, R.; and Callan, J. 2017. Explicit semantic ranking for academic search via knowledge graph embedding. In *Proceedings of the 26th international conference on world wide web*, 1271–1279.

Yang, B.; Yih, W.-t.; He, X.; Gao, J.; and Deng, L. 2014. Embedding entities and relations for learning and inference in knowledge bases. *arXiv preprint arXiv:1412.6575*.

Yang, F.; Yang, Z.; and Cohen, W. W. 2017. Differentiable learning of logical rules for knowledge base reasoning. *Advances in neural information processing systems*, 30.

Yao, Z.; Zhang, W.; Chen, M.; Huang, Y.; Yang, Y.; and Chen, H. 2023. Analogical inference enhanced knowledge graph embedding. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, 4801–4808.

Zhang, F.; Yuan, N. J.; Lian, D.; Xie, X.; and Ma, W.-Y. 2016. Collaborative knowledge base embedding for recommender systems. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, 353–362.

Zhang, Z.; Cai, J.; Zhang, Y.; and Wang, J. 2020. Learning hierarchy-aware knowledge graph embeddings for link prediction. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, 3065–3072.

Zhu, Z.; Zhang, Z.; Xhonneux, L.-P.; and Tang, J. 2021. Neural bellman-ford networks: A general graph neural network framework for link prediction. *Advances in Neural Information Processing Systems*, 34: 29476–29490.