Few-Shot Temporal Knowledge Graph Completion via Frequency-Based Relation Encoding and Conditional Diffusion

Lai Xu, Jibing Wu, Yanmin Li, Liwei Qian, Hang Zhang, and Lihua Liu

Laboratory for Big Data and Decision

National University of Defense Technology

Changsha, China

{xulai, wujibing, yanminli, qianliwei17, zhanghang13}@nudt.edu.cn, gogonudt@126.com

***Abstract*—Few-shot temporal knowledge graph completion (FTKGC) has emerged as an important task for predicting missing facts in temporal knowledge graphs (TKGs) with limited training instances. However, existing methods face two major challenges: (1) They mainly focus on capturing the temporal evolution of entities, overlooking the heterogeneous patterns inherent in relations, which are characterized by a combination of short-term fluctuation and long-term stability; (2) The data sparsity problem in few-shot settings leads to inadequately optimized entity representations, consequently undermining the reliability of prediction on query quadruples. To this end, we propose frequency-based relation encoding and conditional diffusion (FRECD) for FTKGC. Specifically, FRECD presents a novel relation encoding strategy, which integrates the fast Fourier transform (FFT) to decompose relation embeddings and performs further modulation with temporal information. Regarding insufficiently informative entity representations, FRECD leverages a conditional denoising network built upon the diffusion transformer (DiT) to perform generative feature enhancement. Comprehensive experiments are carried out on three benchmark datasets to demonstrate the superiority of our model. The code is available at https://github.com/caker2077/FRECD.**

***Index Terms*—Temporal knowledge graph completion, few-shot learning, frequency decomposition, conditional diffusion**

I. Introduction

As a structured form of knowledge representation, knowledge graphs (KGs) have demonstrated powerful capabilities in numerous downstream tasks, such as information retrieval [1] and question answering [2]. In order to capture real-world dynamics, temporal knowledge graphs (TKGs) constrain static facts in KGs with temporal information, forming quadruples (s, r, o, t). Due to the incompleteness of TKGs, various temporal knowledge graph completion (TKGC) methods learn latent patterns from existing facts to predict the missing ones, such as answering queries like $(Tim\ Cook, CEO, ?, 2024)$. However, a significant portion of relations in TKGs exhibit long-tail characteristics [3], failing to provide sufficient temporal facts for training. Consequently, most existing TKGC methods perform poorly on entity prediction for few-shot relations.

Given the challenge of data sparsity, few-shot temporal knowledge graph completion (FTKGC) methods [4], [5] introduce few-shot learning and have made progress. However, when modeling time-dependent relations, previous works often overlook the intrinsic heterogeneity between short- and long-term relational patterns. For example, considering the career of famous footballer David Beckham, $(Beckham, plays\ for, AC\ Milan, 2009)$ refers to his short-term loan to the club. This represents a temporary arrangement that lasts only until the loan period ends. In contrast, $(Beckham, plays\ for, Manchester\ United, 2003)$ corresponds to his continuous service from 1992 to 2003, which is a long-term commitment in his professional career. Noticeably, ignoring this heterogeneity makes it difficult to precisely model the relation dynamics. Moreover, existing methods rely on aggregating neighborhood information in TKGs to enrich entity representations, yet this strategy cannot offer adequate correlations among task entities under few-shot constraints. Although recursive aggregation can implicitly incorporate broader context, it also introduces excessive noise from irrelevant connections.

To address the above issues, we propose a novel model called few-shot temporal knowledge graph completion via frequency-based relation encoding and conditional diffusion (FRECD). First, we utilize the fast Fourier transform (FFT) [6] to decompose relation embeddings into low- and high-frequency components, representing long-term stability and short-term fluctuation, respectively. These components are then modulated using time embeddings and weighted for fusion. To capture neighborhood information, a neighborhood aggregator explicitly integrates the reconstructed relation embeddings into entity embeddings. Second, we design a conditional denoising network based on the diffusion transformer (DiT) to generate features, which leverages the U-Net structure for multi-scale feature extraction. Furthermore, we employ a hybrid attention pooler to refine the generated features and obtain latent representations to enrich entity embeddings, compensating for the lack of training instances. Experiments show that our model achieves superior performance compared to existing baselines on three FTKGC datasets.

The main contributions of this paper are summarized as follows:

- A novel frequency-based relation encoding method is proposed, which decomposes relation embeddings via

FFT and modulates the components with timestamps to capture short- and long-term patterns of relations.
- A DiT-based conditional denoising network is designed to perform generative feature enhancement for enriching entity embeddings.
- The experimental results confirm that our model outperforms existing methods on three benchmark datasets.

II. Related Work

A. Temporal Knowledge Graph Completion Methods

Existing TKGC methods can be broadly divided into four categories. (1) Transformation-based methods utilize temporal information to perform transformations on entity and relation embeddings, e.g., translation in TTransE [7]. To further enrich the expressiveness, some later studies introduce other geometric operations such as rotation in RotateQVS [8]. (2) Complex-based methods such as TNTComplEx [9] and TCompoundE [10], map TKGs into complex spaces to capture the evolution of facts over time. (3) Decomposition-based methods represent the TKG structure as a fourth-order tensor, with each element of a quadruple corresponding to a specific tensor mode. For example, TuckER-FA [11] is based on Tucker decomposition, while CEC-BD [12] employs block term decomposition. (4) Deep learning-based methods including TeMP [13] and SANe [14], adopt deep neural networks to model temporal dynamics. Such methods suffer from limited interpretability and stability due to their black-box nature.

B. Few-shot Knowledge Graph Completion Methods

Previous research on few-shot knowledge graph completion (FKGC) can be broadly divided into two categories. Metric learning-based methods calculate the similarity between the support and query triples in a metric space. For instance, GMatching [15] utilizes graph information and an LSTM-based matching metric to generate entity representations. FSRL [16] extends GMatching by implementing attentive encoding with fixed weights. SAFER [17] integrates support information through subgraph adaptation and filters out spurious data. However, it fails to effectively leverage global information in the support graph. To consider information in negative triples, RECDAP [18] models separate distributions for positive and negative samples through a diffusion model.

In contrast, meta-learning-based methods adapt model parameters to unseen relations using the support set. For example, MetaR [19] proposes a relational meta-learner that extracts common features of support entities for relation representations. GANA [20] learns few-shot relations and mitigates neighbor noise with a gated attention aggregation method. To address the difficulty in estimating uncertainty and handling complex relations, NP-FKGC [21] integrates neural processes with normalizing flows to model relational distributions.

C. Few-shot Temporal Knowledge Graph Completion Methods

FTAG [22] is the first work for one-shot relation learning of TKGs, which encodes temporal dependencies within the neighborhood through self-attention. Given that timestamps play a pivotal role in learning entity representations as well as matching support and query quadruple, TFSC [4] introduces an attention-based neighbor encoder for temporal neighborhood aggregation, and generates time-aware entity pair representations through the Transformer encoder. Based on metric learning, TR-Match [23] dynamically captures local and global information to encode temporal relations. To mitigate the negative impact of noisy data, FTMF [5] introduces a fault-tolerance mechanism to alleviate the disturbance caused by erroneous samples in few-shot datasets. FTMI [24] formulates the path reasoning task within the Markov decision framework via reinforcement learning, which enhances the interpretability of predictions by generating explicit reasoning paths. Additionally, this model mitigates neighborhood noise based on the semantic relevance between direct and indirect neighbors. However, most existing studies focus on the temporal evolution of entities, neglecting semantic variations of relationships. Different from the studies mentioned above, our model achieves the disentanglement of relational patterns via a frequency-based relation encoding method.

III. Methodology

In this section, we introduce the proposed model in detail. The framework of FRECD is illustrated in Fig. 1.

A. Problem Formulation

A TKG is a collection of quadruples defined as $G = \{(s,r,o,t)\} \subseteq \mathcal{E} \times \mathcal{R} \times \mathcal{E} \times \mathcal{T}$, where $\mathcal{E}$, $\mathcal{R}$ and $\mathcal{T}$ represent the sets of entities, relations, and timestamps, respectively. In FTKGC, the relation set $\mathcal{R}$ is divided into a frequent relation set $\mathcal{R}_{freq}$ and a few-shot relation set $\mathcal{R}_{few}$ based on occurrence frequency, where $\mathcal{R}_{freq} \cap \mathcal{R}_{few} = \emptyset$. Correspondingly, G is divided into a background knowledge graph $G_{bg} = \{(s,r,o,t) \in G \mid r \in \mathcal{R}_{freq}\}$ for learning entity representations and $G_{few} = \{(s,r,o,t) \in G \mid r \in \mathcal{R}_{few}\}$ for few-shot task construction. Within a meta-learning framework, each few-shot task is centered around a specific relation $r \in \mathcal{R}_{few}$, composed of a support set $\mathcal{S}_r = \{(s_i, r, o_i, t_i)\}_{i=1}^{K}$ containing K quadruples for reference and a query set $\mathcal{Q}_r = \{(s_j, r, o_j, t_j)\}_{j=1}^{M}$ to predict missing entities.

B. Frequency-based Relation Encoding

Inspired by the effectiveness of Fourier analysis [25] and series decomposition [26], we characterize relation dynamics by two distinct patterns: low-frequency components reflecting long-term stability and high-frequency components capturing short-term fluctuation.

The initial relation embedding $\boldsymbol{r}$ is transformed by a projection matrix $\boldsymbol{W}_{\text{proj}}$ into $\boldsymbol{r}' = \boldsymbol{W}_{\text{proj}}\boldsymbol{r} = [\boldsymbol{r}^{\text{real}}; \boldsymbol{r}^{\text{imag}}]$, where $[;]$ denotes the concatenation operation. Subsequently, we treat the embedding dimensions as a discrete signal [27], and utilize FFT to convert $\boldsymbol{r}'$ from the time domain to the frequency domain, obtaining the frequency domain representation $\boldsymbol{F}$:

$$\boldsymbol{F} = \mathcal{F}(\boldsymbol{r}^{\text{real}} + i \cdot \boldsymbol{r}^{\text{imag}}), \tag{1}$$

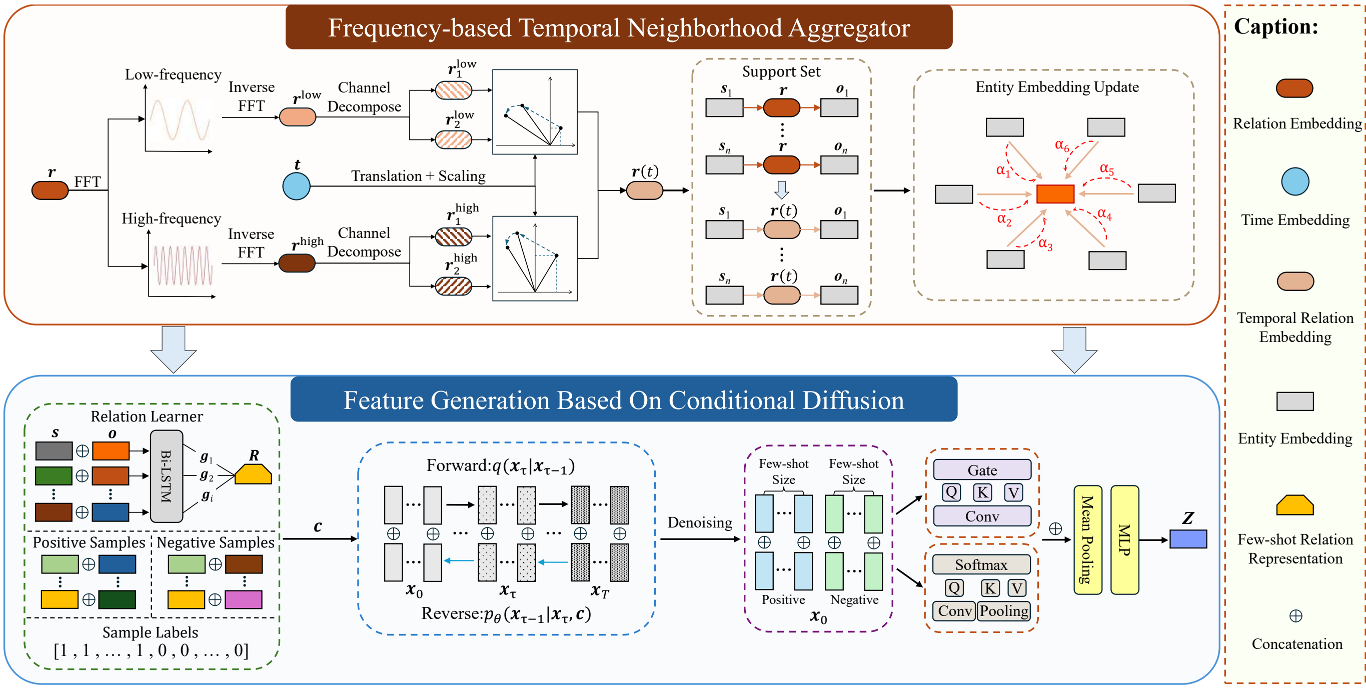


Fig. 1. The framework of FRECD. (1) Frequency-based Temporal Neighborhood Aggregator: a relation embedding $\boldsymbol{r}$ is decomposed via FFT, modulated by time, and fused into a temporal relation embedding $\boldsymbol{r}(t)$, which is then used to update entity embeddings in the neighborhood aggregator. (2) Feature Generation Based On Conditional Diffusion: a few-shot relation representation $\boldsymbol{R}$, learned from updated support entity pairs, is combined with the support samples and labels to form a condition vector $\boldsymbol{c}$. This condition guides the DiT to generate a feature $\boldsymbol{X}$, which is then refined by the hybrid attention pooler into a latent representation $\boldsymbol{Z}$ for TransE-based score.

where $\mathcal{F}(\cdot)$ denotes one-dimensional FFT. To separate low- and high-frequency components within $\boldsymbol{F}$, we employ frequency masks based on a pre-defined factor γ. The low-frequency mask is defined as:

$$\boldsymbol{M}^{\text{low}} = \exp(-\gamma \cdot |\boldsymbol{k}|), \tag{2}$$

where γ controls the degree of preservation for low-frequency components, and $\boldsymbol{k}$ is the absolute values of the frequencies. The high-frequency mask is calculated by subtracting $\boldsymbol{M}^{\text{low}}$ from an all-ones matrix:

$$\boldsymbol{M}^{\text{high}} = \boldsymbol{1} - \boldsymbol{M}^{\text{low}}. \tag{3}$$

With the frequency masks, we decompose $\boldsymbol{F}$ and convert frequency components back to the time domain through inverse FFT, obtaining the low- and high-frequency relation embeddings, denoted as $\boldsymbol{r}^{\text{low}}$ and $\boldsymbol{r}^{\text{high}}$:

$$\boldsymbol{r}^{\text{low}} = \mathcal{F}^{-1}(\boldsymbol{F} \cdot \boldsymbol{M}^{\text{low}}), \quad \boldsymbol{r}^{\text{high}} = \mathcal{F}^{-1}(\boldsymbol{F} \cdot \boldsymbol{M}^{\text{high}}), \tag{4}$$

where $\mathcal{F}^{-1}(\cdot)$ denotes one-dimensional inverse FFT.

To capture temporal dynamics, the discrete timestamp t is mapped to the time embedding $\boldsymbol{t}$ via a learnable matrix $\boldsymbol{E}$, and then partitioned into $\boldsymbol{t}_1$ and $\boldsymbol{t}_2$:

$$\boldsymbol{t} = \boldsymbol{E}(t) = [\boldsymbol{t}_1; \boldsymbol{t}_2]. \tag{5}$$

Following TCompoundE [10], we adopt a compound geometric transformation, i.e., translation and scaling for temporal modulation. In particular, a dual-channel modulation strategy is employed to achieve a fine-grained interaction between time and relation embeddings. To this end, we decompose $\boldsymbol{r}^{\text{low}}$ and $\boldsymbol{r}^{\text{high}}$ into two sub-vectors each, denoted as $\boldsymbol{r}^{\text{low}} = [\boldsymbol{r}_1^{\text{low}}; \boldsymbol{r}_2^{\text{low}}]$ and $\boldsymbol{r}^{\text{high}} = [\boldsymbol{r}_1^{\text{high}}; \boldsymbol{r}_2^{\text{high}}]$. The time smoothing terms $\bar{\boldsymbol{t}}_1$ and $\bar{\boldsymbol{t}}_2$ are formulated as:

$$\bar{\boldsymbol{t}}_1 = \frac{1}{d}\sum_{j=1}^{d} \boldsymbol{t}_{1,j}, \quad \bar{\boldsymbol{t}}_2 = \frac{1}{d}\sum_{j=1}^{d} \boldsymbol{t}_{2,j}. \tag{6}$$

The modulated relation embeddings for both frequency components, $\boldsymbol{f}^{\text{low}}$ and $\boldsymbol{f}^{\text{high}}$, are formulated as:

$$\boldsymbol{f}^{\text{low}} = [(\boldsymbol{r}_1^{\text{low}} + \bar{\boldsymbol{t}}_1) \odot \boldsymbol{t}_1; (\boldsymbol{r}_2^{\text{low}} + \bar{\boldsymbol{t}}_2) \odot \boldsymbol{t}_2], \tag{7}$$

$$\boldsymbol{f}^{\text{high}} = [(\boldsymbol{r}_1^{\text{high}} + \bar{\boldsymbol{t}}_1) \odot \boldsymbol{t}_1; (\boldsymbol{r}_2^{\text{high}} + \bar{\boldsymbol{t}}_2) \odot \boldsymbol{t}_2], \tag{8}$$

where $\odot$ represents the Hadamard product. Intuitively, time smoothing terms apply a global translation to the sub-vectors, while the multiplicative operation performs vector scaling. This combination facilitates the capture of time-relation interaction patterns. Finally, we perform a weighted fusion of $\boldsymbol{f}^{\text{low}}$ and $\boldsymbol{f}^{\text{high}}$ to obtain the temporal relation embedding $\boldsymbol{r}(t)$:

$$\boldsymbol{r}(t) = \lambda \cdot \boldsymbol{f}^{\text{low}} + (1 - \lambda) \cdot \boldsymbol{f}^{\text{high}}, \tag{9}$$

where $\lambda \in [0, 1]$ controls the contribution of different frequency components. Notably, since $\boldsymbol{r}^{\text{low}}$ and $\boldsymbol{r}^{\text{high}}$ are modulated by translation and scaling prior to fusion, this recombination avoids linear cancellation.

C. Neighborhood Aggregator

In this work, we employ the GNN architecture following NP-FKGC [21]. This module adopts a message-passing mechanism, where entities aggregate 2-hop neighborhood information to update their embeddings. Specifically, for an edge (u,v) in TKG, a message is generated as follows:

$$\boldsymbol{m}_{uv}^{(l)} = \boldsymbol{W}_r \left[\boldsymbol{u}^{(l)}; \boldsymbol{r}_{uv}; \boldsymbol{v}^{(l)}\right], \tag{10}$$

where $\boldsymbol{W}_r$ is a relation-specific matrix, $\boldsymbol{u}^{(l)}$ and $\boldsymbol{v}^{(l)}$ are the embeddings of u and v after l iterations of message passing, and $\boldsymbol{r}_{uv}$ is the temporal relation embedding $\boldsymbol{r}(t)$ in Eq. (9) between u and v. In this way, the messages explicitly integrate the results of frequency-based relation encoding. Subsequently, we compute the attention weight α_{uv} for each message to measure its importance:

$$\alpha_{uv} = \frac{\exp\left(\rho\left(\boldsymbol{W}_{\text{att}}^{\mathrm{T}}[\boldsymbol{v}^{(l)}; \boldsymbol{m}_{uv}^{(l)}]\right)\right)}{\sum_{k\in\mathcal{N}(v)} \exp\left(\rho\left(\boldsymbol{W}_{\text{att}}^{\mathrm{T}}[\boldsymbol{v}^{(l)}; \boldsymbol{m}_{kv}^{(l)}]\right)\right)}, \tag{11}$$

where $\boldsymbol{W}_{\text{att}}$ is a attention weight matrix, ρ represents the LeakyReLU activation function, and $\mathcal{N}(v)$ denotes the neighbors of v. The updated embedding of v is defined as:

$$\boldsymbol{v}^{(l+1)} = \sum_{k\in\mathcal{N}(v)} \alpha_{kv}\boldsymbol{m}_{kv}^{(l)} + \boldsymbol{S}^{(l+1)}\boldsymbol{v}^{(l)}, \tag{12}$$

where $\boldsymbol{S}^{(l+1)}$ is the self-loop transformation matrix for the $(l+1)$-th message passing layer. By stacking two message passing layers, we obtain the updated entity embeddings $\boldsymbol{s}$ and $\boldsymbol{o}$ for the quadruple (s,r,o,t).

D. Relation Learner

Following previous studies [20], [21], we employ an attention-based BiLSTM to learn the few-shot relation representations. Given the updated head and tail entity pairs $[\boldsymbol{s}_i; \boldsymbol{o}_i]_{i=1}^{K}$ from support quadruples as the input sequence, the relation representation $\boldsymbol{R}$ is computed as:

$$\boldsymbol{R} = \sum_{i=1}^{K} \frac{\exp(\boldsymbol{g}_i^{\mathrm{T}}\boldsymbol{q})}{\sum_{j=1}^{K}\exp(\boldsymbol{g}_j^{\mathrm{T}}\boldsymbol{q})}\boldsymbol{g}_i, \tag{13}$$

where K is the number of support quadruples, $\boldsymbol{q}$ and $\boldsymbol{g}_i$ denote the final cell state and the hidden state at the i-th timestep, respectively.

E. Conditional Denoising Network

The DiT-based conditional denoising network takes the timestep τ, noisy embedding $\boldsymbol{x}_\tau$, and condition vector $\boldsymbol{c}$ as input. The condition is composed of the relation representation $\boldsymbol{R}$ described in Eq. (13), the original embedding $\boldsymbol{x}_0$, and labels indicating positive/negative samples in $\boldsymbol{x}_0$. Specifically, $\boldsymbol{x}_0$ is constructed by concatenating the updated head and tail entity embeddings for both positive and negative samples from the support set. The diffusion timestep τ is encoded via the sinusoidal positional encoding and processed by a multilayer perceptron (MLP) to generate the timestep embedding $\boldsymbol{TE}(\tau)$.

To effectively guide the denoising process, $\boldsymbol{x}_\tau$ is concatenated with $\boldsymbol{c}$ and projected into the initial feature $\boldsymbol{h}_0$. Subsequently, $\boldsymbol{h}_0$ is processed by a stack of DiT blocks. Each block combines an attention mechanism with a feed-forward network (FFN) that employs learnable activation functions to flexibly adapt to the data distribution. Both components are modulated through AdaLN-Zero [28]:

$$(\boldsymbol{a}_1, \boldsymbol{b}_1, \boldsymbol{w}_1, \boldsymbol{a}_2, \boldsymbol{b}_2, \boldsymbol{w}_2) = \text{AdaLN-Zero}(\boldsymbol{TE}(\tau)). \tag{14}$$

In the $(m+1)$-th block, the feature h_m is updated as follows:

$$\boldsymbol{h}'_m = \boldsymbol{h}_m + \boldsymbol{w}_1 \odot \text{Att}((1+\boldsymbol{a}_1)\odot \text{LN}(\boldsymbol{h}_m) + \boldsymbol{b}_1), \tag{15}$$

$$\boldsymbol{h}_{m+1} = \boldsymbol{h}'_m + \boldsymbol{w}_2 \odot \text{FFN}((1+\boldsymbol{a}_2)\odot \text{LN}(\boldsymbol{h}'_m) + \boldsymbol{b}_2), \tag{16}$$

where LN$(\cdot)$ denotes layer normalization.

To achieve multi-scale feature extraction, we organize the DiT blocks in the U-Net structure, incorporating downsampling via patch merging and upsampling combined with skip connections. Furthermore, we adopt a stage-specific attention strategy to balance computational efficiency and expressiveness of features. Specifically, DiT blocks at the high-resolution input and output stages employ shifted window attention [29] to reduce complexity while preserving local feature details. In contrast, the encoder, bottleneck, and decoder layers work in the downsampled feature space, utilizing global attention to aggregate context and model long-range dependencies. With the predicted noise $\boldsymbol{\epsilon}_\theta$ output by this network, we obtain the generated feature $\boldsymbol{X}$ through the denoising process.

F. Hybrid Attention Pooler

Applying convolutions over embedding dimensions has been proven an effective strategy to capture feature interactions in representation learning [30]. Given the generated feature $\boldsymbol{X}$, this module processes it through two different attention branches. The local branch generates queries $\boldsymbol{Q}_{\text{local}}$, keys $\boldsymbol{K}_{\text{local}}$, and values $\boldsymbol{V}_{\text{local}}$ through convolution operations, and employs a gating mechanism to extract fine-grained details:

$$\boldsymbol{Z}_{\text{local}} = \text{Tanh}(\Phi(\boldsymbol{Q}_{\text{local}} \odot \boldsymbol{K}_{\text{local}})) \odot \boldsymbol{V}_{\text{local}}, \tag{17}$$

where $\Phi(\cdot)$ is a mapping function used to generate the gating weights, and Tanh$(\cdot)$ is the activation function. The global branch also generates queries $\boldsymbol{Q}_{\text{global}}$ by convolution, while applying pooling to mitigate noise. This branch employs the scaled dot-product attention to extract global information:

$$\boldsymbol{K}_{\text{global}}, \boldsymbol{V}_{\text{global}} = \boldsymbol{W}_{\text{global}}(\text{AvgPool}(\boldsymbol{X})), \tag{18}$$

$$\boldsymbol{Z}_{\text{global}} = \text{Softmax}\left(\frac{\boldsymbol{Q}_{\text{global}}\boldsymbol{K}_{\text{global}}^{\mathrm{T}}}{\sqrt{d_k}}\right)\boldsymbol{V}_{\text{global}}, \tag{19}$$

where $\boldsymbol{W}_{\text{global}}(\cdot)$ is a learnable projection matrix, AvgPool$(\cdot)$ denotes window average pooling, and d_k is the scaling factor. Subsequently, $\boldsymbol{Z}_{\text{local}}$ and $\boldsymbol{Z}_{\text{global}}$ are concatenated, followed by global average pooling and an MLP to generate the latent representation $\boldsymbol{Z}$.

G. Training and Optimization

In FRECD, all task entity embeddings are generated through the neighborhood aggregator. In the forward process, we follow the settings of the Denoising Diffusion Probabilistic Model (DDPM) [31]. Starting from $\boldsymbol{x}_0$, Gaussian noise controlled by the variance schedule β_τ is progressively added through a Markov chain. In this way, the distribution of the noisy embedding $\boldsymbol{x}_\tau$ is formulated as follows:

$$q(\boldsymbol{x}_\tau|\boldsymbol{x}_{\tau-1}) = \mathcal{N}(\boldsymbol{x}_\tau; \sqrt{1-\beta_\tau}\boldsymbol{x}_{\tau-1}, \beta_\tau\mathbf{I}). \tag{20}$$

Conditioned on $\boldsymbol{c}$, the mean of the Gaussian distribution in the reverse process is formulated as:

$$\boldsymbol{\mu}_\theta(\boldsymbol{x}_\tau, \tau, \boldsymbol{c}) = \frac{1}{\sqrt{\alpha_\tau}}\left(\boldsymbol{x}_\tau - \frac{\beta_\tau}{\sqrt{1-\bar{\alpha}_\tau}}\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_\tau, \tau, \boldsymbol{c})\right), \tag{21}$$

where $\alpha_\tau = 1-\beta_\tau$, $\bar{\alpha}_\tau = \prod_{i=1}^{\tau}\alpha_i$ represents the proportion of the original signal at timestep τ, and $\boldsymbol{\epsilon}_\theta$ is the noise predicted by DiT. With Eq. (21), the denoising trajectory is denoted as:

$$p_\theta(\boldsymbol{x}_{\tau-1}|\boldsymbol{x}_\tau, \boldsymbol{c}) = \mathcal{N}(\boldsymbol{x}_{\tau-1}; \boldsymbol{\mu}_\theta(\boldsymbol{x}_\tau, \tau, \boldsymbol{c}), \boldsymbol{\Sigma}_\theta(\boldsymbol{x}_\tau, \tau, \boldsymbol{c})). \tag{22}$$

We obtain the generated feature $\boldsymbol{X}$ by sampling from the learned distribution $p_\theta(\boldsymbol{x}_0)$. Subsequently, the hybrid attention pooler refines it to generate the latent representation $\boldsymbol{Z}$. Based on TransE [32], for a query quadruple (s, r, o, t), the score is calculated as:

$$\boldsymbol{s}' = \boldsymbol{s} + \boldsymbol{W}_s\boldsymbol{Z}, \quad \boldsymbol{o}' = \boldsymbol{o} + \boldsymbol{W}_o\boldsymbol{Z}, \tag{23}$$

$$Score = -\left\|\boldsymbol{s}' + \boldsymbol{R} - \boldsymbol{o}'\right\|_2, \tag{24}$$

where W_s and W_o are linear projection layers for s and o, respectively.

We formulate the loss function to account for both entity discrimination and feature generation quality. Specifically, we employ margin ranking loss to ensure that the scores of positive samples are higher than those of negative samples by at least a margin δ:

$$\mathcal{L}_{\text{margin}} = \frac{1}{N}\sum_{i=1}^{N}\max(0, \delta - (Score_i^+ - Score_i^-)). \tag{25}$$

Furthermore, we use MSE loss [31] to minimize the distance between the predicted noise $\boldsymbol{\epsilon}_\theta$ and the added noise $\boldsymbol{\epsilon}$, thereby improving denoising ability. The diffusion loss is defined as:

$$\mathcal{L}_{\text{diff}} = \frac{1}{N}\sum_{i=1}^{N}[\|\boldsymbol{\epsilon}_i - \boldsymbol{\epsilon}_{\theta,i}(\boldsymbol{x}_\tau, \tau, \boldsymbol{c})\|_2^2]. \tag{26}$$

Combining Eq. (25) and Eq. (26), the total loss function is defined as $\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{margin}} + \mathcal{L}_{\text{diff}}$. The complete procedure of training and optimization is shown in Algorithm 1.

Algorithm 1 Training Process of FRECD

Input: Meta-training task set $\mathbb{T}_{mtr}$; Pre-trained TKG embeddings; background knowledge graph G_{bg}; initial model parameters Θ.

while not done **do**
  Shuffle the tasks in $\mathbb{T}_{mtr}$
  for $T_r \in \mathbb{T}_{mtr}$ **do**
    Sample the support set $\mathcal{S}_r$ and the query set $\mathcal{Q}_r$
    Generate the temporal relation embeddings $\boldsymbol{r}(t)$ by Eq. (1)-(9)
    Aggregate multi-hop neighborhood for entity embeddings by Eq. (10)-(12)
    Generate the few-shot relation representation $\boldsymbol{R}$ in $\mathcal{S}_r$ by Eq. (13)
    Generate the latent representation $\boldsymbol{Z}$ by Eq. (14)-(22)
    Calculate scores for $\mathcal{Q}_r$ by Eq. (23)-(24)
    Calculate the margin ranking loss $\mathcal{L}_{\text{margin}}$ and the diffusion loss $\mathcal{L}_{\text{diff}}$ by Eq. (25)-(26)
    Optimize Θ via the total loss $\mathcal{L}_{\text{total}}$
  end for
end while

TABLE I
STATISTICS OF THE EXPERIMENTAL DATASETS.

Dataset	# Entities	# Times	# Relations	# Tasks
ICEWS14-few	7,121	365	114	92/11/11
ICEWS05-15-few	10,471	4,017	99	81/9/9
ICEWS18-few	24,572	365	99	81/9/9

IV. EXPERIMENTS

A. Dataset

We evaluate FRECD on the few-shot subsets of the ICEWS14, ICEWS05-15 and ICEWS18 [33] TKGC datasets. For ICEWS14, relations with quadruple frequency between 20 and 200 are selected as task relations, forming the subset ICEWS14-few. A total of 114 task relations are divided into training, validation and test sets with a ratio of 92:11:11. For ICEWS05-15 and ICEWS18, relations with quadruple frequency between 50 and 500 are selected, forming the subsets ICEWS05-15-few and ICEWS18-few. Both subsets contain 99 task relations and follow an 81:9:9 ratio for the training, validation and test sets. The quadruples of the remaining relations serve as background information to provide initial embeddings for task entities and relations. The dataset statistics are shown in Table I.

B. Baselines

We compare our model with three categories of baselines:

- **TKGC methods:** We select TNTComplEx [9], TeLM [34], TeAST [35] and TeRDy [36] as TKGC baselines. Following GMatching [15], the complete training set for TKGC methods consists of quadruples from the background knowledge graph, the training quadruples, and the support quadruples within validation and test sets.
- **FKGC methods:** We select four FKGC baselines, including GANA [20], NP-FKGC [21], PARE [37] and ReCDAP [18]. Since such methods are restricted to static environment, we adapt the quadruples for these models by discarding the timestamps, i.e., neglecting t in (s, r, o, t).
- **FTKGC methods:** We select three FTKGC models, including FILT [38], TFSC [4] and TR-Match [23].

TABLE II
RESULTS OF ALL METHODS ON THREE FTKGC DATASETS.

Model	**ICEWS14-few**				**ICEWS05-15-few**				**ICEWS18-few**			
	MRR	Hits@1	Hits@5	Hits@10	MRR	Hits@1	Hits@5	Hits@10	MRR	Hits@1	Hits@5	Hits@10
TNTComplEx	0.218	0.130	0.317	0.402	0.097	0.045	0.138	0.199	0.138	0.070	0.195	0.283
TeLM	0.270	0.166	0.364	0.481	0.198	0.108	0.282	0.383	0.203	0.115	0.280	0.385
TeAST	0.292	0.181	0.412	0.503	0.261	0.165	0.359	0.454	0.182	0.090	0.275	0.378
TeRDy	0.348	0.234	0.481	0.577	0.234	0.131	0.337	0.445	0.241	0.142	0.342	0.440
GANA	0.242	0.121	0.378	0.472	0.220	0.084	0.364	0.487	0.115	0.047	0.192	0.255
NP-FKGC	<u>0.561</u>	<u>0.497</u>	<u>0.631</u>	<u>0.664</u>	<u>0.597</u>	<u>0.511</u>	<u>0.677</u>	<u>0.696</u>	0.439	0.387	0.495	0.537
PARE	0.260	0.163	0.369	0.452	0.266	0.152	0.374	0.508	0.174	0.102	0.238	0.310
ReCDAP	0.553	0.477	0.628	0.644	0.565	0.470	0.662	0.684	<u>0.460</u>	<u>0.392</u>	<u>0.534</u>	<u>0.559</u>
FILT	0.394	0.313	0.481	0.549	0.470	0.378	0.569	0.645	0.313	0.237	0.391	0.464
TFSC	0.250	0.166	0.333	0.421	0.265	0.155	0.378	0.499	0.138	0.069	0.205	0.271
TR-Match	0.321	0.233	0.414	0.501	0.282	0.180	0.392	0.486	0.215	0.137	0.281	0.372
FRECD	**0.593**	**0.519**	**0.662**	**0.680**	**0.638**	**0.550**	**0.724**	**0.734**	**0.489**	**0.431**	**0.557**	**0.578**

Note: The best results are highlighted in **bold**, and <u>underlined</u> values denote the second best.

C. Implementation

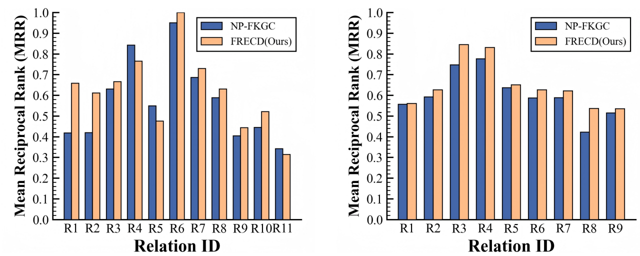
Our FRECD is conducted on a Linux server equipped with an NVIDIA A800 GPU. In this study, we utilize TransE [32] to obtain initial entity and relation embeddings from the background knowledge graphs of the three datasets. For hyperparameters, we fix the embedding dimension at 100, the margin at 1, and the learning rate at 1×10^{-4}. The batch size is set to 256 for both ICEWS14-few as well as ICEWS05-15-few, and 128 for ICEWS18-few. The neighborhood size is set to 2. The frequency masks factor γ is set to 10. The fusion weight λ for low- and high-frequency components is set to 0.5 for ICEWS14-few as well as ICEWS18-few, and 0.4 for ICEWS05-15-few. For the diffusion process, the total number of timestep is set to 100. The DiT is organized in a U-Net structure with 4 downsampling blocks, 4 upsampling blocks, and a central bottleneck block. The few-shot size K is set to 5 for all FKGC and FTKGC methods. For baselines, we use the optimal parameters provided in their original papers.

To evaluate the performance of all methods, we employ two standard metrics: Mean Reciprocal Rank (MRR) and Hits@N ($N = 1, 5, 10$).

D. Main Results

Table II presents the detailed performance comparison of all methods on the ICEWS14-few, ICEWS05-15-few and ICEWS18-few datasets. It is evident that our model outperforms all baselines in terms of MRR and Hits@1/5/10. Specifically, on the ICEWS14-few and ICEWS05-15-few datasets, our model surpasses the best baseline (i.e., NP-FKGC) by 5.7% and 6.9% in MRR, and by 4.4% and 7.6% in Hits@1, respectively. Furthermore, on the ICEWS18-few dataset, our model outperforms the best baseline (i.e., ReCDAP) by 6.3% in MRR and 9.9% in Hits@1.

The key advantage of FRECD is attributed to its effectiveness in capturing both long-term and short-term relational patterns, whereas other models such as NP-FKGC struggle to accurately capture this heterogeneity. Moreover, by utilizing the conditional denoising network, FRECD achieves adaptive multi-scale feature modeling and effectively performs generative feature enhancement on the entity embeddings.



(a) ICEWS14-few (b) ICEWS05-15-few

Fig. 2. Performance comparison across specific relations on ICEWS14-few and ICEWS05-15-few.

We also compare our model with the most competitive baseline, NP-FKGC, with respect to specific relations on the ICEWS14-few and ICEWS05-15-few datasets. As shown in Fig. 2, our model outperforms NP-FKGC with 8 out of 11 relations on ICEWS14-few, and extends this superiority to all 9 relations on ICEWS05-15-few. Experiments demonstrate that our model exhibits superior capabilities in learning quadruple representations across different relation types.

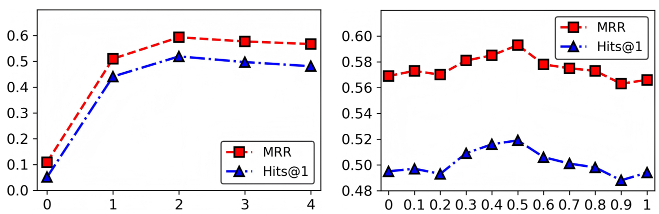
E. Ablation Study

We construct three variants of FRECD to better evaluate the contribution of each component to our model. First, w/o FRE refers to the removal of frequency-based relation encoding method for relation embeddings, just relying on static and semantically entangled relation representations. Next, w/o CD indicates that the conditional denoising network is replaced by the MLP used in NP-FKGC. Lastly, w/o AttPooler refers to applying only global average pooling to the generated features. The ablation results are shown in Table III.

Specifically, w/o FRE degrades the neighborhood aggregator into a static and frequency-agnostic entity encoder, leading to an evident performance drop. This demonstrates the importance of considering relational semantic heterogeneity and

TABLE III
ABLATION STUDY OF DIFFERENT VARIANTS. THE BEST RESULTS ARE DISPLAYED IN BOLD.

Variants	ICEWS14-few		ICEWS05-15-few		ICEWS18-few	
	MRR	H@1	MRR	H@1	MRR	H@1
FRECD	**0.593**	**0.519**	**0.638**	**0.550**	**0.489**	**0.431**
w/o FRE	0.557	0.506	0.556	0.459	0.407	0.369
w/o CD	0.568	0.492	0.607	0.516	0.465	0.398
w/o AttPooler	0.579	0.499	0.620	0.528	0.473	0.410



(a) *hop* (b) λ

Fig. 3. Parameter study w.r.t. *hop* and λ.

temporal modulation. Similarly, we observe a decrease in MRR and Hits@1 for the w/o CD and w/o AttPooler variants. Notably, the impact of removing the conditional denoising network is greater than that of directly applying global average pooling across all three datasets. This suggests that high-quality features are the basis for effective refinement.

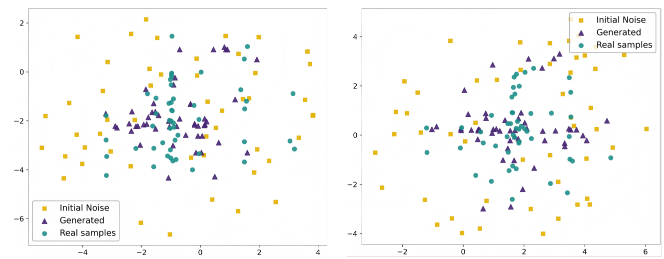
F. Parameter Analysis

We study the impact of parameters on the performance of FRECD on the ICEWS14-few dataset, with respect to the neighborhood size *hop*, and the fusion weight λ of low- and high-frequency components. As shown in Fig. 3a, we vary *hop* within the set $\{0, 1, 2, 3, 4\}$. Both MRR and Hit@1 peak in a 2-hop neighborhood, while dropping to the lowest values when *hop* is set to 0 (i.e., no neighborhood information is considered). The reason is that a smaller *hop* results in insufficiently informative entity embeddings, while a larger *hop* leads to excessive noise.

For λ, we vary its value from 0 (i.e., only high-frequency components) to 1 (i.e., only low-frequency components). As shown in Fig. 3b, FRECD achieves optimal performance at $\lambda = 0.5$. Furthermore, the performance exhibits a downward trend as λ approaches 0 or 1. This indicates that on the ICEWS14-few dataset, the model shows no obvious bias towards either frequency component.

G. Visualization Analysis

To further evaluate the effectiveness of our conditional denoising network, we apply t-SNE [39] to visualize the feature distributions on the ICEWS14-few and ICEWS05-15-few datasets, as shown in Fig. 4. The visualization includes three types of data, i.e., initial Gaussian noise (yellow squares), real entity embeddings from the support set (teal circles) and latent representations $\boldsymbol{Z}$ (purple triangles). We observe that the initial noise exhibits a disordered distribution. In contrast, the latent representations cluster around the real entity embeddings and gather closer to the center of the samples. This transition indicates that our model not only effectively removes the noise, but also generates high-quality latent representations.



(a) ICEWS14-few (b) ICEWS05-15-few

Fig. 4. Visualization of feature distributions on the ICEWS14-few and ICEWS05-15-few datasets.

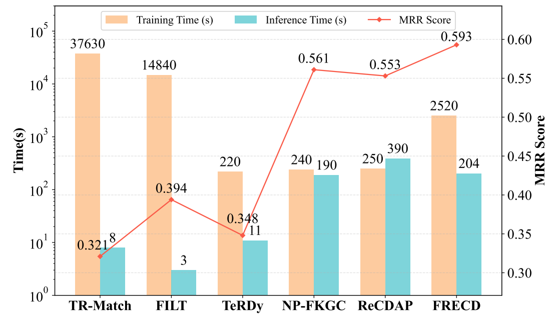


Fig. 5. Comparison of computational efficiency and MRR on ICEWS14-few.

H. Efficiency Analysis

We compare the computational efficiency of FRECD with that of baselines on ICEWS14-few. For a fair comparison, the experiments are conducted on an RTX 4090 GPU. Training time is measured until the best validation epoch is reached (excluding the time for evaluation steps), while inference time covers the entire test set. As shown in Fig. 5, FRECD (2,520s) trains significantly faster than FTKGC baselines such as TR-Match (37,630s) and FILT (14,840s). Although the DiT network requires more training time than static FKGC models (e.g., NP-FKGC), this acceptable extra cost is rewarded with a higher MRR. For inference, despite the iterative denoising process, FRECD (204s) is nearly on the same level as NP-FKGC (190s). While baselines like TR-Match and TeRDy require much less time for prediction, their performance remains notably inferior. Overall, FRECD achieves a balanced trade-off between computational efficiency and predictive performance.

V. Conclusion

In this paper, we propose a novel FRECD method for FTKGC. We integrate the frequency decomposition with the temporal modulation to model the distinct patterns of relation embeddings. Furthermore, a DiT-based conditional denoising network coupled with a hybrid attention pooler is introduced to enrich entity embeddings. Extensive experiments on three

FTKGC datasets demonstrate FRECD's superiority over existing baselines. Since frequency-based relation encoding is designed to capture complex and dynamic relation patterns, it may yield limited improvements on simple or static relations. Therefore, future work will focus on developing an encoding mechanism that dynamically adjusts to different relation types.

References

[1] Z. Liu, C. Xiong, M. Sun, and Z. Liu, "Entity-duet neural ranking: Understanding the role of knowledge graph semantics in neural information retrieval," *arXiv preprint arXiv:1805.07591*, 2018.

[2] Y. Zhang, H. Dai, Z. Kozareva, A. Smola, and L. Song, "Variational reasoning for question answering with knowledge graph," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 32, no. 1, 2018.

[3] W. Xiong, M. Yu, S. Chang, X. Guo, and W. Y. Wang, "One-shot relational learning for knowledge graphs," *arXiv preprint arXiv:1808.09040*, 2018.

[4] H. Zhang and L. Bai, "Few-shot link prediction for temporal knowledge graphs based on time-aware translation and attention mechanism," *Neural Networks*, vol. 161, pp. 371–381, 2023.

[5] L. Bai, M. Zhang, H. Zhang, and H. Zhang, "Ftmf: Few-shot temporal knowledge graph completion based on meta-optimization and fault-tolerant mechanism," *World Wide Web*, vol. 26, no. 3, pp. 1243–1270, 2023.

[6] J. W. Cooley, P. A. Lewis, and P. D. Welch, "The fast fourier transform and its applications," *IEEE Transactions on Education*, vol. 12, no. 1, pp. 27–34, 2007.

[7] J. Leblay and M. W. Chekol, "Deriving validity time in knowledge graph," in *Companion proceedings of the the web conference 2018*, 2018, pp. 1771–1776.

[8] K. Chen, Y. Wang, Y. Li, and A. Li, "Rotateqvs: Representing temporal information as rotations in quaternion vector space for temporal knowledge graph completion," *arXiv preprint arXiv:2203.07993*, 2022.

[9] T. Lacroix, G. Obozinski, and N. Usunier, "Tensor decompositions for temporal knowledge base completion," *arXiv preprint arXiv:2004.04926*, 2020.

[10] R. Ying, M. Hu, J. Wu, Y. Xie, X. Liu, Z. Wang, M. Jiang, H. Gao, L. Zhang, and R. Cheng, "Simple but effective compound geometric operations for temporal knowledge graph completion," *arXiv preprint arXiv:2408.06603*, 2024.

[11] L. Xiao, R. Zhang, Z. Chen, and J. Chen, "Tucker decomposition with frequency attention for temporal knowledge graph completion," in *Findings of the Association for Computational Linguistics: ACL 2023*, 2023, pp. 7286–7300.

[12] L. Yue, Y. Ren, Y. Zeng, J. Zhang, K. Zeng, J. Wan, and M. Zhou, "Complex expressional characterizations learning based on block decomposition for temporal knowledge graph completion," *Knowledge-Based Systems*, vol. 290, p. 111591, 2024.

[13] J. Wu, M. Cao, J. C. K. Cheung, and W. L. Hamilton, "Temp: Temporal message passing for temporal knowledge graph completion," *arXiv preprint arXiv:2010.03526*, 2020.

[14] Y. Li, X. Zhang, B. Zhang, F. Huang, X. Chen, M. Lu, and S. Ma, "Sane: Space adaptation network for temporal knowledge graph completion," *Information Sciences*, vol. 667, p. 120430, 2024.

[15] W. Xiong, M. Yu, S. Chang, X. Guo, and W. Y. Wang, "One-shot relational learning for knowledge graphs," *arXiv preprint arXiv:1808.09040*, 2018.

[16] C. Zhang, H. Yao, C. Huang, M. Jiang, Z. Li, and N. V. Chawla, "Few-shot knowledge graph completion," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 34, no. 03, 2020, pp. 3041–3048.

[17] H. Liu, S. Wang, C. Chen, and J. Li, "Few-shot knowledge graph relational reasoning via subgraph adaptation," *arXiv preprint arXiv:2406.15507*, 2024.

[18] J. Kim, C. Heo, and J. Jung, "Recdap: Relation-based conditional diffusion with attention pooling for few-shot knowledge graph completion," in *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2025, pp. 2848–2852.

[19] M. Chen, W. Zhang, W. Zhang, Q. Chen, and H. Chen, "Meta relational learning for few-shot link prediction in knowledge graphs," *arXiv preprint arXiv:1909.01515*, 2019.

[20] G. Niu, Y. Li, C. Tang, R. Geng, J. Dai, Q. Liu, H. Wang, J. Sun, F. Huang, and L. Si, "Relational learning with gated and attentive neighbor aggregator for few-shot knowledge graph completion," in *Proceedings of the 44th international ACM SIGIR conference on research and development in information retrieval*, 2021, pp. 213–222.

[21] L. Luo, Y.-F. Li, G. Haffari, and S. Pan, "Normalizing flow-based neural process for few-shot knowledge graph completion," in *Proceedings of the 46th international ACM SIGIR conference on research and development in information retrieval*, 2023, pp. 900–910.

[22] M. Mirtaheri, M. Rostami, X. Ren, F. Morstatter, and A. Galstyan, "One-shot learning for temporal knowledge graphs," *arXiv preprint arXiv:2010.12144*, 2020.

[23] X. Gong, J. Qin, H. Chai, Y. Ding, Y. Jia, and Q. Liao, "Temporal-relational matching network for few-shot temporal knowledge graph completion," in *International conference on database systems for advanced applications*. Springer, 2023, pp. 768–783.

[24] L. Bai, S. Han, and L. Zhu, "Multi-hop interpretable meta learning for few-shot temporal knowledge graph completion," *Neural Networks*, vol. 183, p. 106981, 2025.

[25] Z.-Q. J. Xu, Y. Zhang, T. Luo, Y. Xiao, and Z. Ma, "Frequency principle: Fourier analysis sheds light on deep neural networks," *arXiv preprint arXiv:1901.06523*, 2019.

[26] H. Wu, J. Xu, J. Wang, and M. Long, "Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting," *Advances in neural information processing systems*, vol. 34, pp. 22 419–22 430, 2021.

[27] J. Lee-Thorp, J. Ainslie, I. Eckstein, and S. Ontanon, "Fnet: Mixing tokens with fourier transforms," in *Proceedings of the 2022 Conference of the north American chapter of the Association for Computational Linguistics: human language technologies*, 2022, pp. 4296–4313.

[28] W. Peebles and S. Xie, "Scalable diffusion models with transformers," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2023, pp. 4195–4205.

[29] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 10 012–10 022.

[30] T. D. Nguyen, D. Q. Nguyen, D. Phung *et al.*, "A novel embedding model for knowledge base completion based on convolutional neural network," in *Proceedings of the 2018 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 2 (short papers)*, 2018, pp. 327–333.

[31] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," *Advances in neural information processing systems*, vol. 33, pp. 6840–6851, 2020.

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

[33] E. Boschee, J. Lautenschlager, S. O'Brien, S. Shellman, J. Starz, and M. Ward, "Icews coded event data," *Harvard Dataverse*, vol. 12, no. 2, 2015.

[34] C. Xu, Y.-Y. Chen, M. Nayyeri, and J. Lehmann, "Temporal knowledge graph completion using a linear temporal regularizer and multivector embeddings," in *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2021, pp. 2569–2578.

[35] J. Li, X. Su, and G. Gao, "Teast: Temporal knowledge graph embedding via archimedean spiral timeline," in *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2023, pp. 15 460–15 474.

[36] Z. Liu and C. Wang, "Terdy: Temporal relation dynamics through frequency decomposition for temporal knowledge graph completion," in *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2025, pp. 9611–9622.

[37] S. Yu, Y. Wang, Z. Wan, Y. Shen, Q. Zhang, and F. Xia, "Path-aware few-shot knowledge graph completion," *IEEE Transactions on Artificial Intelligence*, 2025.

[38] Z. Ding, J. Wu, B. He, Y. Ma, Z. Han, and V. Tresp, "Few-shot inductive learning on temporal knowledge graphs using concept-aware information," *arXiv preprint arXiv:2211.08169*, 2022.

[39] L. v. d. Maaten and G. Hinton, "Visualizing data using t-sne," *Journal of machine learning research*, vol. 9, no. Nov, pp. 2579–2605, 2008.