

SimDiff-DNS: Simplex Diffusion with Dynamic Noise Scheduling for Temporal Knowledge Graph Reasoning

Xiaoke Wang*, Hechun Yang*, Fu Zhang*, Yiwen Chi, Kai Li,
Hongzhi Chen, Yingsong Ning, Jiashun Peng, Jingwei Cheng, Shuangshuang Gu

School of Computer Science and Engineering,
Northeastern University, Shenyang 110819, China

Key Laboratory of Intelligent Computing in Medical Image, Ministry of Education,
Northeastern University, Shenyang 110819, China

{wangxk1, yanghc1}@mails.neu.edu.cn, zhangfu@neu.edu.cn

Abstract

Temporal knowledge graph reasoning (TKGR) is crucial for capturing dynamic relational dependencies and inferring evolving facts. However, existing embedding-based and diffusion-based methods primarily rely on Euclidean point embeddings and face two limitations: (1) sparse entities may suffer from embedding crowding and norm-sensitive scoring, especially under limited supervision in long-tail temporal knowledge graphs; (2) uniform noise schedules in diffusion models ignore temporal heterogeneity and instance-level prediction difficulty. To address these challenges, we propose SimDiff-DNS (**Simplex Diffusion with Dynamic Noise Scheduling**), a simplex-anchored logit diffusion framework for TKGR. *First*, we represent entity states as probability distributions on the simplex and obtain continuous embeddings as convex combinations of learnable entity basis vectors. This simplex-coordinate parameterization provides a probabilistic view of uncertainty and alleviates norm-sensitive ranking behavior for sparse entities. *Second*, we introduce a difficulty-aware dynamic noise scheduling strategy for the training corruption process, which adjusts the noise applied to target-entity logits according to a bounded difficulty score. Extensive experiments on three TKGR datasets show that SimDiff-DNS consistently improves over the diffusion-based baseline (improving MRR by 3.25 points on ICEWS14) and achieves competitive or state-of-the-art results on several metrics. Additional diagnostic analyses further demonstrate its benefits for long-tail entities, calibration, and ranking stability.

1 Introduction

Temporal knowledge graphs (TKGs) (Gottschalk and Elena, 2018; Zhao, 2021; Zhang et al., 2024a)

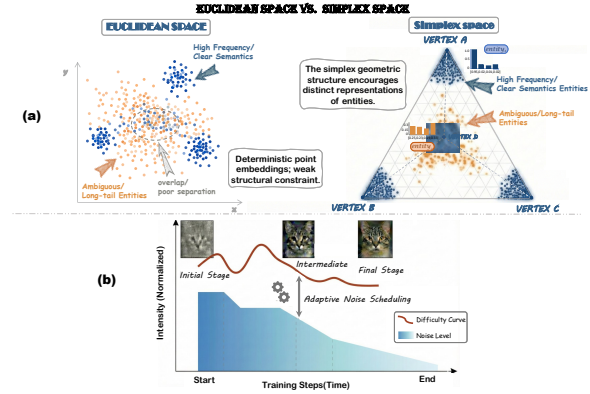


Figure 1: (a) The simplex representation of the entities. (b) Adaptive Dynamic Noise Scheduling.

are dynamic multirelational graphs represented in the form of quadruples (s, r, o, τ) , where s denotes the subject (*i.e.*, head entity), r denotes the relation, o denotes the object (*i.e.*, tail entity), and τ denotes the timestamp. The TKGR reasoning (TKGR) tasks are primarily divided into interpolation and extrapolation (Jin et al., 2020). The *interpolation* task (Xu et al., 2020; Xiong et al., 2024) involves inferring missing facts at past timestamps, while the *extrapolation* task (Xu et al., 2020; Sun et al., 2021; Liu et al., 2022) focuses on predicting future events. This study specifically addresses the extrapolation task of TKGs, as it can forecast future events and provide forward-looking insights for decision-making, offering substantial practical value in areas such as event prediction (Deng et al., 2020), trend analysis (Choudhury et al., 2020), and risk prediction (Jhee et al., 2025).

Existing TKGR methods mainly leverage geometric transformations or historical sequence encoders, and typically map entities and relations to Euclidean space (Wang et al., 2023a; Cai et al., 2023). For instance, Re-Temp (Wang et al., 2023b) extends graph neural network (GNN) encoders with explicit temporal embeddings and learns Euclidean-space representations for entities. CyGNet (Zhu

* These authors contributed equally to this work.

* Corresponding author.

et al., 2021) learns time-aware Euclidean embeddings via a copy-generation mechanism. Despite the promising results achieved by existing methods, there are still several key challenges: *On one hand*, sparse entities in TKGs often receive limited supervision, which makes Euclidean point embeddings vulnerable to embedding crowding and norm-sensitive global scoring; *on the other hand*, deterministic point embeddings provide limited tools for diagnosing prediction uncertainty in temporally evolving entity states. Recently, methods based on generative diffusion models have emerged. DiffuTKG (Cai et al., 2024) models temporal reasoning as a conditional sequence denoising process. However, it employs a uniform noise scheduling strategy, which overlooks the inherent heterogeneity and evolving fluctuations in temporal data.

In addition, in TKGs, there is often a severe *long-tail* distribution of entities, which leads to a significant drop in model performance when handling sparse tail entities. Existing geometric transformation models typically treat entities as points. Due to the lack of sufficient training samples, the embeddings of tail entities are difficult to update effectively, resulting in failure of the geometric transformations. On the other hand, although diffusion models excel at learning data distributions, during the reverse denoising process, they tend to overlook the weak feature signals of long-tail entities because these entities lack sample support, making it difficult to capture rare or sudden events.

To address these challenges, we propose SimDiff-DNS (**Simplex Diffusion with Dynamic Noise Scheduling**), a diffusion-based framework that anchors entity states in simplex coordinates. **First**, unlike Euclidean point embeddings, the simplex constraint models entities as sparse probability distributions. As shown in Figure 1(a), each entity state lies on the probability simplex, allowing the model to vary from sharp low-entropy distributions for predictable entities to more diffuse high-entropy distributions for ambiguous or sparse entities. The resulting continuous embedding is a convex combination of learnable entity basis vectors, which provides a more controlled representation than free Euclidean point vectors when basis norms are regularized. This structure encourages discriminative entity distributions and is especially useful for long-tail entities with limited supervision.

Furthermore, we introduce a difficulty-aware adaptive dynamic noise scheduling (DNS). As depicted in Figure 1(b), by quantifying historical

predictability, DNS adaptively adjusts noise levels according to the predicted difficulty of each entity. Specifically, the mechanism assigns lower noise to simple entities to refine decision boundaries, while injecting higher noise into hard-to-predict entities to promote robust structural learning. This difficulty-aware strategy significantly bolsters model resilience to data sparsity and enhances inductive reasoning on unseen facts.

Our contributions are summarized as follows:

- To the best of our knowledge, SimDiff-DNS is the first work to incorporate simplex-coordinate entity states into a diffusion framework for TKGR. By representing entity states as probability distributions and continuous embeddings as convex combinations of basis vectors, SimDiff-DNS reduces norm-sensitive ranking behavior and provides a natural uncertainty measure through entropy.
- We design a difficulty-aware dynamic noise scheduler for training-time logit corruption. Instead of uniform noise scheduling, DNS adaptively adjusts noise levels according to bounded prediction difficulty and uses clipped coefficients to guarantee a valid perturbation process.
- Experiments on three datasets show that SimDiff-DNS consistently improves over the diffusion-based baseline and achieves competitive performance against strong TKGR methods. Further long-tail, calibration, entropy-correlation, norm-stability, and ranking-stability analyses demonstrate the benefits of simplex-coordinate representations and difficulty-aware training-time corruption.

2 Related Work

2.1 Temporal Knowledge Graph Reasoning

Existing TKG reasoning (extrapolation) methods can be broadly divided into several categories: (1) **Embedding-based models** dynamically model temporal evolutionary patterns of entities and relationships using low-dimensional vectors, inferring missing facts through the similarity of historical embeddings. Representative models include CyGNet (Zhu et al., 2021), HIP (He et al., 2021), among others. (2) **GNN-based models** focus on uncovering structural evolution in TKGs, predicting dynamic associations by aggregating

neighborhood information through message passing, e.g., xERTE (Han et al., 2020) and RE-NET (Jin et al., 2020). Most embedding spaces of embedding-based and GNN-based models are Euclidean spaces. To address the intrinsic limitations of Euclidean space, non-Euclidean geometric embeddings attract attention in TKG interpolation task. IME (Wang et al., 2024a) models TKGs into multi-curvature spaces, including hyperspherical, hyperbolic, and Euclidean spaces. 5EL (Zhang et al., 2025) models TKGs into projective geometric space and utilizes Möbius transformations to capture complex topological structures. (3) **Rule-based models** focus on inductively deriving interpretable logical rules, such as TLogic (Liu et al., 2022) and TempValid (Huang et al., 2024). (4) **Language model-based models** treat entities and relations as semantic symbols, using generative models to predict missing facts, such as CoH (Xia et al., 2024) and GenTKG (Liao et al., 2024).

2.2 Diffusion-based TKGR

Diffusion models are a type of generative model that learn data distributions by gradually adding and removing noise, and they are commonly used for high-quality image and audio generation. Currently, some research has explored text diffusion models in discrete state spaces (Li et al., 2022; Reid et al., 2023; Gong et al., 2023). DiffuTKG (Cai et al., 2024) is the first model to introduce diffusion methods into TKG reasoning tasks, by introducing random noise to target entities and reconstructing the entities. However, DiffuTKG adopts a uniform noise scheduling strategy that is independent of the data, ignoring the inherent heterogeneity and evolving differences in time-series data. As a result, the model fails to adaptively adjust the diffusion process based on the prediction difficulty of samples.

Different from existing methods, in our work: (1) We model entity states as sparse probability distributions in simplex coordinates, which provides entropy-based uncertainty diagnostics beyond deterministic point embeddings; (2) unlike the uniform noise scheduling in DiffuTKG, we introduce a difficulty-aware Dynamic Noise Scheduler (DNS) that adaptively adjusts the perturbation intensity in the forward logit space for each training sample. By constraining the temporal reasoning difficulty, DNS directly modulates the instance-conditioned forward perturbation distribution; (3) by integrating simplex-coordinate representation with diffusion dynamics, our method mitigates the long-tail prob-

lem in TKGs and is further validated by frequency-stratified, calibration, entropy-correlation, norm-stability, and ranking-stability analyses.

3 Problem Formulation

A temporal knowledge graph (TKG) is defined as $\mathcal{G} = (\mathcal{E}, \mathcal{R}, \mathcal{T}, \mathcal{Q})$, where $\mathcal{E}$, $\mathcal{R}$, and $\mathcal{T}$ denote the sets of entities, relations, and timestamps, respectively, and $\mathcal{Q} = \{(e_s, r, e_o, \tau) \mid e_s, e_o \in \mathcal{E}, r \in \mathcal{R}, \tau \in \mathcal{T}\}$ is the set of timestamped quadruples. We further represent $\mathcal{G}$ as a sequence of snapshots ordered by time, $\mathcal{G} = \{\mathcal{G}_1, \mathcal{G}_2, \dots, \mathcal{G}_{|\mathcal{T}|}\}$. Given a query $(s_q, r_q, ?, \tau_q)$, the goal of TKG reasoning (extrapolation) task is to predict o_q from the historical snapshots $\{\mathcal{G}_1, \dots, \mathcal{G}_{\tau_q-1}\}$.

4 Method

In Figure 2, we present the SimDiff-DNS framework, which begins by grounding entity representations in simplex coordinates (§4.1) to enforce sparsity-inducing probabilistic constraints. Building on this, we formulate a simplex-anchored logit diffusion process (§4.3), where Gaussian perturbations are applied in unconstrained logit coordinates and each state is projected to the simplex through softmax. The diffusion process is further augmented by a difficulty-aware adaptive noise scheduling strategy (§4.4) that dynamically modulates noise levels based on bounded instance hardness. Finally, we detail the sequence encoding and denoising network (§4.5), which leverages historical contexts to guide TKG reasoning (§4.6).

4.1 Simplex Manifold Embedding

We model entities as sparse probability distributions using simplex embeddings. According to Han et al. (2023), we map an arbitrary entity $e \in \{1, 2, \dots, |\mathcal{E}|\}$ to a simplex-based Logit Vector $\mathbf{s}_e \in \mathbb{R}^{|\mathcal{E}|}$:

$$\mathbf{s}_e[i] = \begin{cases} +k & \text{if } i = e, \\ -k & \text{otherwise,} \end{cases} \quad (1)$$

where $k > 0$ is a scaling hyperparameter controlling the peakiness of the initial distribution. We then derive the simplex probability vector $\mathbf{p}_e$ via:

$$\mathbf{p}_e = \text{softmax}(\mathbf{s}_e), \quad \text{s.t.} \quad \mathbf{p}_e \in \Delta^{|\mathcal{E}|-1}, \quad (2)$$

where $\Delta^{|\mathcal{E}|-1} = \{\mathbf{x} \in \mathbb{R}^{|\mathcal{E}|} \mid \mathbf{x} \geq 0, \|\mathbf{x}\|_1 = 1\}$ denotes the standard simplex.

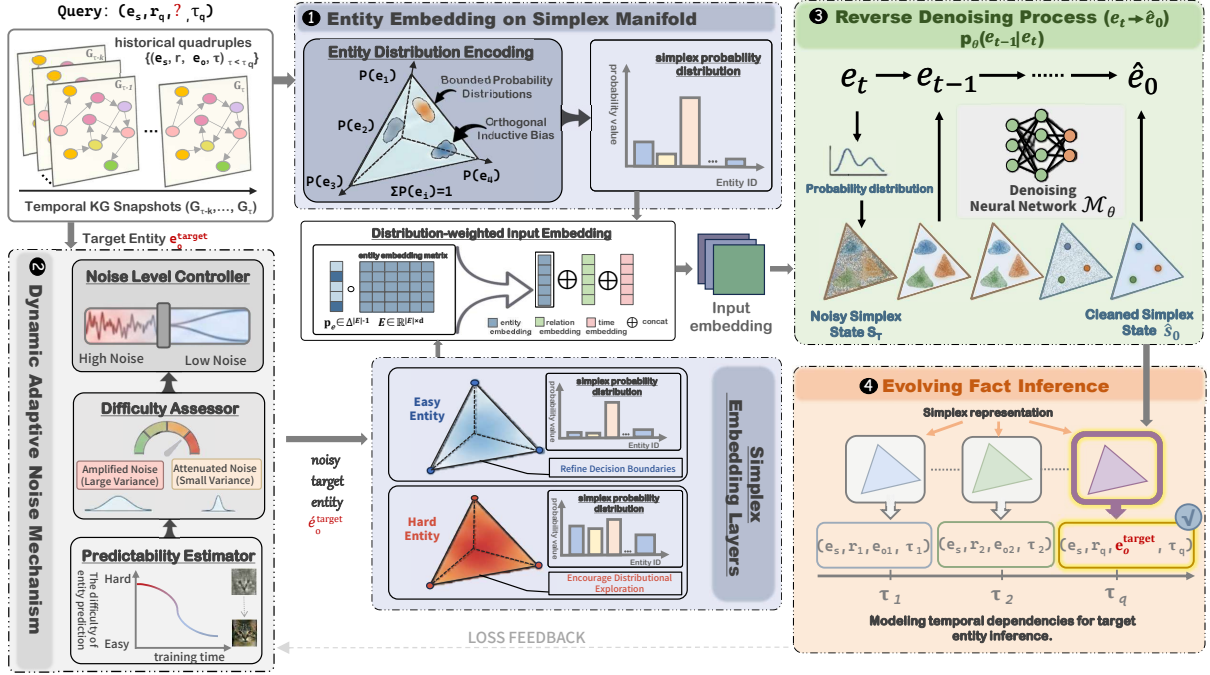


Figure 2: The overall framework of SimDiff-DNS. It comprises four primary components: (1) Simplex-coordinate Entity Embedding, which represents entity states as probability distributions and projects them into continuous embeddings; (2) Dynamic Adaptive Noise Mechanism, which estimates difficulty and adaptively modulates the noise injected into logit coordinates; (3) Reverse Denoising Process, designed to reconstruct the clean logit state $\hat{s}_0$ whose softmax projection yields a simplex-valued state; and (4) Inferred Evolving Facts, which predicts the target tail entity e_{target} based on historical quadruples.

To integrate semantic information, we project the discrete probability distribution $\mathbf{p}_e$ into a continuous latent space to derive the entity embedding $\mathbf{e}$ of the entity e :

$$\mathbf{e} = \mathbf{p}_e \mathbf{E}, \quad (3)$$

where $\mathbf{E} \in \mathbb{R}^{|\mathcal{E}| \times d}$ represents the learnable entity embedding matrix with hidden dimension d , resulting in $\mathbf{e} \in \mathbb{R}^d$.

4.2 Geometric Analysis of Simplex Space

By representing entity states as bounded probability distributions, simplex coordinates reduce norm-sensitive effects and provide an entropy-based view of prediction uncertainty, which is particularly useful for long-tail entities with limited supervision. We defer the detailed geometric analysis to Appendix A.

4.3 Simplex-Anchored Logit Diffusion

SimDiff-DNS performs diffusion in the unconstrained logit space rather than directly on the simplex. Each perturbed logit state $\mathbf{s}_t$ is mapped by

softmax to a valid simplex-valued entity state, inducing a logistic-normal distribution as discussed in Appendix H.3.

4.3.1 Forward Diffusion

The forward propagation perturbs the initial logit vector $\mathbf{s}_0$ through dynamically adaptive noise scheduling (see Section 4.4). Given an entity e , we first obtain its simplex-based logit vector $\mathbf{s}_0$ according to Eq. (1). Without DNS, the standard logit diffusion process is:

$$\mathbf{s}_t = \sqrt{\bar{\alpha}_t} \mathbf{s}_0 + \sqrt{1 - \bar{\alpha}_t} \boldsymbol{\epsilon}, \quad (4)$$

where, $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ denotes Gaussian noise, and $\bar{\alpha}_t = \prod_{i=1}^t (1 - \beta_i)$ is the cumulative noise coefficient. During training, DNS replaces the base coefficients in Eq. (4) with the sample-dependent clipped coefficients defined in Eq. (15). The noisy logit vector $\mathbf{s}_t$ is mapped by softmax to $\mathbf{p}_t \in \Delta^{|\mathcal{E}|-1}$, from which the continuous noisy entity embedding $\mathbf{e}_t$ is obtained via Eq. (3).

4.3.2 Reverse Denoising

The reverse process uses a conditional denoising network $\mathcal{M}_\theta$ to reconstruct the initial target-entity

logit state $\mathbf{s}_0$ from its noisy state $\mathbf{s}_t$. Given the noisy embedding $\mathbf{e}_t$ and historical context $\mathcal{H}$, the network predicts a denoised representation, which is projected to the entity vocabulary via $\mathbf{E}^T$:

$$\mathbf{p}_0^{\text{pred}} \approx \text{softmax}(\mathcal{M}_\theta(\mathbf{e}_t, t, \mathcal{H})\mathbf{E}^T). \quad (5)$$

To obtain the predicted clean logit state, we map the predicted simplex-valued distribution back to logit coordinates:

$$\hat{\mathbf{s}}_0 = \log(\mathbf{p}_0^{\text{pred}} + \delta), \quad (6)$$

where δ is a small constant for numerical stability.

For DNS-corrupted training states, the forward coefficients become sample-dependent. Let a_t and b_t denote the signal and noise coefficients at step t , where $a_t = \tilde{c}_t$ and $b_t = \tilde{\sigma}_t$ when DNS is used for training corruption. Under a fixed detached difficulty score, the corresponding Gaussian posterior can be written as follows. Define

$$\rho_t = \frac{a_t}{a_{t-1}}, \quad v_t = b_t^2 - \rho_t^2 b_{t-1}^2, \quad (7)$$

with $v_t > 0$, and the posterior variance

$$\tilde{v}_t = \left(\frac{1}{b_{t-1}^2} + \frac{\rho_t^2}{v_t} \right)^{-1}. \quad (8)$$

The posterior mean parameterized by the predicted clean state is

$$\tilde{\boldsymbol{\mu}}_t(\mathbf{s}_t, \hat{\mathbf{s}}_0) = \tilde{v}_t \left(\frac{a_{t-1}}{b_{t-1}^2} \hat{\mathbf{s}}_0 + \frac{\rho_t}{v_t} \mathbf{s}_t \right). \quad (9)$$

When $a_t = \sqrt{\bar{\alpha}_t}$ and $b_t = \sqrt{1 - \bar{\alpha}_t}$, this form reduces to the standard DDPM posterior. The inference procedure uses the fixed base schedule, as described in Section 4.6.2.

4.4 Difficulty-Aware Adaptive Dynamic Noise Scheduling

DNS adaptively corrupts target-entity logits during training by estimating entity-level prediction difficulty from model feedback. Unlike timestep-only schedules, it performs instance-level noise modulation using bounded temporal reasoning difficulty while preserving a valid clipped perturbation process.

4.4.1 Difficulty Assessment

We propose an evaluation mechanism that dynamically updates the prediction difficulty of entities based on the continuously evolving training feedback of the model. For a given query

tuple $(s_q, r_q, ?, \tau_q)$ in the training batch, we assess the prediction difficulty using the current model parameters and the historical sequence $\mathcal{H}$ (in Section 4.5.1). Specifically, we obtain the real-time output probability distribution $\mathbf{p} \in \Delta^{|\mathcal{E}|-1}$ by masking the target entity. We compute three bounded difficulty components:

$$u_1 = \frac{H(\mathbf{p})}{\log |\mathcal{E}|}, \quad u_3 = 1 - p_{\max}, \quad (10)$$

$$u_2 = \min\{-\log(p_{\text{target}} + \epsilon) / \log |\mathcal{E}|, 1\},$$

where $H(\mathbf{p}) = -\sum_i p_i \log p_i$ measures predictive uncertainty, p_{target} is the probability of the target entity, and $p_{\max}$ is the maximum predicted probability. The bounded difficulty score is:

$$\bar{\mathcal{W}}_{\text{diff}} = \text{clip}\left(\frac{\omega_1 u_1 + \omega_2 u_2 + \omega_3 u_3}{\omega_1 + \omega_2 + \omega_3 + \epsilon}, 0, 1\right), \quad (11)$$

$0 \leq \bar{\mathcal{W}}_{\text{diff}} \leq 1$, which prevents the adaptive noise coefficient from becoming unbounded.

4.4.2 Adaptive Noise Scheduling

We adjust the perturbation intensity based on the bounded difficulty score $\bar{\mathcal{W}}_{\text{diff}}$. For entities that are easy to predict, we assign a lower noise level to refine local decision boundaries. In contrast, for hard-to-predict entities, we inject stronger perturbations to promote broader semantic exploration during denoising.

We first compute the raw adaptive noise coefficient:

$$\hat{\sigma}_t = \sqrt{1 - \bar{\alpha}_t} \cdot (1 + \gamma \cdot \bar{\mathcal{W}}_{\text{diff}}), \quad (12)$$

where $\gamma \geq 0$ is the noise scaling factor. To guarantee numerical validity, we clip the coefficient:

$$\tilde{\sigma}_t = \text{clip}(\hat{\sigma}_t, \sigma_{\min}, 1 - \epsilon_\sigma), \quad (13)$$

where $0 < \sigma_{\min} < 1$ and $0 < \epsilon_\sigma < 1$. The signal component is then:

$$\tilde{c}_t = \sqrt{1 - \tilde{\sigma}_t^2}. \quad (14)$$

Therefore, $\tilde{c}_t^2 + \tilde{\sigma}_t^2 = 1$, ensuring a valid perturbation process. Utilizing the modified coefficients, we perturb the initial logit representation $\mathbf{s}_0$ to derive the state $\mathbf{s}_t$ at time step t :

$$\mathbf{s}_t = \tilde{c}_t \mathbf{s}_0 + \tilde{\sigma}_t \boldsymbol{\epsilon}, \quad (15)$$

where $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. Appendix H.5 proves that the clipped DNS coefficients are bounded and Appendix H.6 shows that difficulty monotonically reduces the signal-to-noise ratio in the unclipped region.

4.5 Sequence Encoding and Denoising Network

4.5.1 Sequence Construction

Unlike prior approaches such as DiffuTKG, which aggregate quadruple information via element-wise summation, we preserve the fine-grained temporal structure by constructing the input sequence $\mathcal{H}$ through chronological concatenation:

$$\mathcal{H} = [\mathbf{e}_{s_1}, \mathbf{r}_1, \mathbf{e}_{o_1}, \tau_1, \dots, \mathbf{e}_{s_n}, \mathbf{r}_n, \mathbf{e}_{o_n}, \tau_n], \quad (16)$$

where, n denotes the historical sequence length. Entity embeddings are obtained from the simplex-space mapping in Section 4.1. Relations are embedded as $\mathbf{r}_i = \mathbf{R}r_i$ with inverse relations included in $\mathbf{R} \in \mathbb{R}^{2|\mathcal{R}| \times d}$. We use relative time encoding $\tau_i = \mathbf{T}|t_q - \tau_i|$, where $\mathbf{T} \in \mathbb{R}^{|\mathcal{T}| \times d}$ is learnable and t_q is the query timestamp.

4.5.2 Transformer Denoising Network

We use a Transformer-based architecture to model conditional dependencies and perform denoising operations. We concatenate the historical sequence $\mathcal{H}$, query relation $\mathbf{r}_q$, query timestamp τ_q , and the noise target entity embedding $\mathbf{e}_t$ of the current diffusion step as the input:

$$\hat{\mathbf{e}}_0 = \text{Transformer}(\mathcal{H} \oplus \mathbf{r}_q \oplus \tau_q \oplus \mathbf{e}_t, \mathbf{M}), \quad (17)$$

where $\oplus$ denotes the concatenation operation, and $\mathbf{M}$ represents the attention mask. We extract the output vector corresponding to the position of the noisy entity $\mathbf{e}_t$ as the predicted denoised representation $\hat{\mathbf{e}}_0$.

4.6 Training and Inference

4.6.1 Training Objective

The training proceeds by sampling a time step $t \sim \mathcal{U}(1, T)$ and perturbing the target entity logit coordinate via DNS (Eq. (15)). The noised logit state is mapped to the simplex via softmax and then projected to the continuous embedding space. The model is optimized to reconstruct the initial state $\mathbf{s}_0$ conditioned on the historical sequence.

Global Entity Scoring. We project the predicted denoised embedding $\hat{\mathbf{e}}_0$ back into the entity vocabulary space to compute the plausibility scores for all candidate entities. Specifically, we employ a scaled dot-product formulation:

$$\mathbf{z} = \frac{\hat{\mathbf{e}}_0 \mathbf{E}^T}{\sqrt{d} \cdot \tau}, \quad (18)$$

where $\mathbf{z} \in \mathbb{R}^{|\mathcal{E}|}$ denotes the output logits, $\sqrt{d}$ serves as a scaling factor to maintain gradient stability. The temperature parameter τ regulates the sharpness of the probability distribution.

Loss Function. The model is optimized by minimizing the Cross-Entropy loss between the predicted distribution and the ground-truth entity o . This objective maximizes the log-likelihood of the correct entity:

$$\mathcal{L} = -\log \left(\frac{\exp(\mathbf{z}_o)}{\sum_{i=1}^{|\mathcal{E}|} \exp(\mathbf{z}_i)} \right), \quad (19)$$

where $\mathbf{z}_o$ corresponds to the logit score of the ground-truth entity.

4.6.2 Inference Procedure

We initialize the logit state with Gaussian noise $\mathbf{s}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. At each reverse step t , the denoising network estimates the clean state $\hat{\mathbf{s}}_0$ conditioned on the historical context $\mathcal{H}$ and the current noisy state $\mathbf{s}_t$. We compute the reverse mean using the standard posterior parameterization with the base coefficients $a_t = \sqrt{\bar{\alpha}_t}$ and $b_t = \sqrt{1 - \bar{\alpha}_t}$. Denote the resulting mean and variance by $\boldsymbol{\mu}_t^{\text{base}}$ and v_t^{base} , respectively. The previous state is sampled as

$$\mathbf{s}_{t-1} = \boldsymbol{\mu}_t^{\text{base}}(\mathbf{s}_t, \hat{\mathbf{s}}_0) + \sqrt{v_t^{\text{base}}} \boldsymbol{\epsilon}, \quad (20)$$

where $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. This iterative refinement continues until $t = 0$, after which candidate entities are ranked by the global scoring function.

5 Experiments

5.1 Experimental Setup

Datasets We evaluate SimDiff-DNS on three widely used datasets: ICEWS14 (García-Durán et al., 2018), ICEWS05-15 (García-Durán et al., 2018), and YAGO (Mahdisoltani et al., 2013).

Evaluation Metrics During testing, we perform experiments under time-aware filtering settings (Dong et al., 2023; Zhang et al., 2023) to filter out other correct entities. To evaluate model performance, we adopt standard evaluation metrics from the field, including Mean Reciprocal Rank (MRR), Hits@1, Hits@3, and Hits@10, where higher metric values indicate better performance.

Baseline Methods We compare SimDiff-DNS with representative state-of-the-art TKGR baselines, covering embedding-based, GNN-based,

Method	ICEWS14				ICEWS05-15				YAGO			
	MRR	H@1	H@3	H@10	MRR	H@1	H@3	H@10	MRR	H@1	H@3	H@10
CyGNet [†] (Zhu et al., 2021)	39.86	30.11	44.02	58.10	40.42	29.44	46.06	61.60	68.98	58.97	76.80	86.98
RE-NET [†] (Jin et al., 2020)	38.48	28.52	42.85	58.10	44.56	34.16	50.06	64.51	66.93	58.59	71.48	86.84
xERTE (Han et al., 2020)	40.79	32.70	45.67	57.30	46.62	37.84	52.31	63.92	53.62	48.53	58.42	60.53
REGCN [†] (Li et al., 2021)	42.48	31.90	47.73	62.85	48.10	37.48	53.92	68.56	82.30	<u>78.83</u>	84.27	88.58
ODE (Han et al., 2021)	26.25	17.30	29.07	44.18	42.86	32.72	48.14	62.34	62.50	58.77	64.73	68.63
TLogic (Liu et al., 2022)	42.53	33.20	47.61	60.29	46.94	36.16	53.24	67.21	78.76	74.31	83.38	83.72
TempValid (Huang et al., 2024)	45.78	35.50	51.34	65.06	50.31	39.46	56.71	70.55	79.72	74.64	84.78	85.73
ONSEP (Yu et al., 2024)	-	33.20	46.50	57.70	-	39.00	55.10	66.80	-	-	-	-
ChapTER (Peng et al., 2024)	33.80	-	38.00	52.70	33.10	-	36.90	52.50	14.80	11.50	14.40	20.20
STORE (Zhang et al., 2024b)	<u>48.77</u>	36.53	55.58	71.91	49.74	38.52	55.91	71.14	64.65	51.94	71.50	83.10
CoH (Xia et al., 2024)	43.94	33.07	49.64	64.90	49.71	38.01	56.40	71.25	-	-	-	-
GenTKG (Liao et al., 2024)	-	36.85	47.95	53.50	-	-	-	-	-	79.15	83.00	84.25
LLM-DA (Wang et al., 2024b)	47.10	<u>36.90</u>	52.60	67.10	52.10	41.60	<u>58.60</u>	72.80	-	-	-	-
DiffuTKG (Cai et al., 2024)	48.51	36.41	-	72.75	52.69	40.35	-	75.97	80.98 [‡]	74.25 [‡]	<u>85.63</u> [‡]	<u>89.41</u> [‡]
CognTKE (Chen et al., 2025)	46.06	36.49	51.11	64.49	<u>53.13</u>	<u>42.62</u>	59.42	72.70	-	-	-	-
MESH (Deng et al., 2025)	44.36	-	49.81	64.21	48.66	-	54.26	68.57	-	-	-	-
SimDiff-DNS(Ours)	51.76	39.31	<u>53.37</u>	76.87	53.73	43.58	57.48	75.98	<u>81.61</u>	76.30	86.33	89.81
<i>APG</i>	2.99 ↑	2.41 ↑	-2.21 ↓	4.12 ↑	0.60 ↑	0.96 ↑	-1.94 ↓	0.01 ↑	-0.69 ↓	-2.85 ↓	0.70 ↑	0.40 ↑
<i>RPG(%)</i>	6.13 ↑	6.53 ↑	-3.98 ↓	5.66 ↑	1.13 ↑	2.25 ↑	-3.26 ↓	0.01 ↑	-0.84 ↓	-3.60 ↓	0.82 ↑	0.45 ↑

Table 1: Performance (%) comparison of TKG reasoning (the extrapolation task) on three event-based TKG datasets (ICEWS14, ICEWS05-15, and YAGO). *APG* and *RPG* represent the absolute and relative performance differences of our model over the best-performing baselines, calculated as $APG = R_{ours} - R_{baseline}$ and $RPG = (R_{ours} - R_{baseline}) / R_{baseline}$, where R_{ours} and $R_{baseline}$ denote the results of our model and best-performing baselines, respectively. Best results are in bold, and the second best are underlined. The results marked with [†] are from Huang et al. (2024), the results marked with [‡] are reproduced by us, and other results are retrieved from the original papers.

rule-based, language-model-based, diffusion-based, and recent hybrid neural methods. The compared methods include CyGNet (Zhu et al., 2021), RE-NET (Jin et al., 2020), xERTE (Han et al., 2020), REGCN (Li et al., 2021), ODE (Han et al., 2021), TLogic (Liu et al., 2022), TempValid (Huang et al., 2024), ONSEP (Yu et al., 2024), ChapTER (Peng et al., 2024), STORE (Zhang et al., 2024b), CoH (Xia et al., 2024), LLM-DA (Wang et al., 2024b), GenTKG (Liao et al., 2024), CognTKE (Chen et al., 2025), MESH (Deng et al., 2025), and DiffuTKG (Cai et al., 2024). We provide the implementation details of SimDiff-DNS in Appendix B and introduce each baseline model in Appendix C.

5.2 Main Results

The performance comparison of SimDiff-DNS against strong baselines on three TKG datasets is presented in Table 1. The results demonstrate that our proposed framework achieves the best or highly competitive performance across most metrics.

Specifically, compared to the *diffusion-based* baseline DiffuTKG, SimDiff-DNS achieves significant improvements. On the ICEWS14 dataset, our model surpasses DiffuTKG by 3.25% in MRR (reaching 51.76%) and 2.90% in Hits@1. Similarly, on the larger and more complex ICEWS05-15 dataset, SimDiff-DNS achieves an MRR of 53.73%,

outperforming DiffuTKG (52.69%) by nearly 1%.

Furthermore, compared to advanced generative methods like GenTKG and LLM-DA, SimDiff-DNS remains highly competitive. While LLM-based methods benefit from massive pre-trained knowledge, our method achieves comparable or better results with fewer parameters, highlighting the efficiency of our geometrically grounded diffusion process.

5.3 Ablation Study

To validate the contribution of each component, we conduct ablation studies in Table 2. First, replacing the simplex-coordinate embedding with Euclidean representations (**w/o Simplex**) leads to a notable MRR drop, indicating that simplex coordinates are effective for modeling sparse entity distributions and mitigating norm-sensitive ranking bias. Second, replacing DNS with a uniform schedule (**w/o DNS**) decreases MRR by **4.52%**, demonstrating that DNS better calibrates training dynamics by sharpening decision boundaries for predictable entities while encouraging broader exploration for challenging samples.

5.4 Analysis of Long-Tail Performance

To examine long-tail robustness, we group ICEWS14 test entities by their training-set frequency into high-, medium-, and low-frequency

Method	ICEWS14			
	MRR	H@1	H@3	H@10
SimDiff-DNS w/o <i>Simplex</i>	46.04	36.72	50.60	63.91
SimDiff-DNS w/o <i>DNS</i>	47.24	37.90	51.34	65.56
SimDiff-DNS	51.76	39.31	53.37	76.87

Table 2: Results (%) of the ablation studies on ICEWS14.

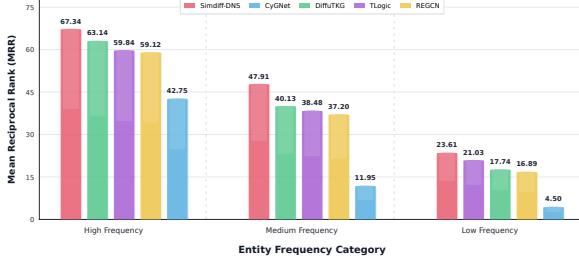


Figure 3: Fine-Grained Model Performance Analysis under Different Entity Frequencies on ICEWS14.

groups. As shown in Figure 3, all methods degrade as entity frequency decreases, while SimDiff-DNS consistently performs best. Additional long-tail diagnostics are summarized in Appendix E; hyper-parameter sensitivity is reported in Appendix F.

5.5 Inference Efficiency Analysis

As shown in Figure 4, reducing the number of denoising steps from 400 to 100 decreases the average latency from 45.0 ms to 12.1 ms, while MRR decreases moderately from 51.76% to 49.61%. When using only 50 steps, the latency further drops to 6.3 ms, with MRR decreasing to 48.90%. These results indicate that SimDiff-DNS can trade off accuracy and efficiency by adjusting the length of the reverse denoising trajectory.

5.6 Case Study

Appendix D shows that SimDiff-DNS assigns higher uncertainty and stronger DNS perturbation to ambiguous queries, while producing sharper candidate margins for easier queries.

5.7 Euclidean Regularization Controls

Table 3 compares Euclidean variants with L2 regularization, norm clipping, and label smoothing. These controls improve the Euclidean baseline but remain behind SimDiff-DNS, especially for long tail entities. Please refer to Appendix E for details.

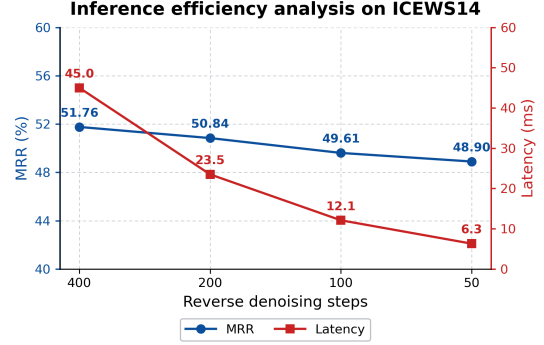


Figure 4: MRR–latency trade-off under different reverse denoising steps on ICEWS14.

Method	Overall				Long tail.	
	MRR	H@1	H@3	H@10	MRR	H@10
Euclidean	46.04	36.72	50.60	63.91	29.68	45.27
+ L2 Reg.	46.83	37.18	51.02	64.85	30.54	46.91
+ Norm Clip	47.06	37.31	51.29	65.17	31.08	47.74
+ Label Smooth.	47.41	37.65	51.57	65.63	31.36	48.12
SimDiff-DNS w/o DNS	47.24	37.90	51.34	65.56	31.72	48.36
SimDiff-DNS	51.76	39.31	53.37	76.87	37.92	57.86

Table 3: Comparison with Euclidean control variants on ICEWS14. L2 regularization, norm clipping, and label smoothing partially improve Euclidean embeddings but do not close the gap on low-frequency entities.

6 Conclusion

We propose SimDiff-DNS, a simplex-anchored logit diffusion framework for TKG reasoning. We represent entity states as simplex-valued probability distributions and project them into continuous embeddings through convex combinations of basis vectors. Furthermore, the proposed clipped difficulty-aware adaptive noise scheduling strikes a balance between boundary refinement and semantic exploration. Experiments show that SimDiff-DNS consistently improves over the diffusion-based baseline and achieves competitive results against strong TKGR methods, with particularly clear gains on ICEWS14 and in long-tail diagnostic settings.

Limitations

Despite its effectiveness, SimDiff-DNS presents certain limitations. First, the iterative softmax projections and reverse denoising steps introduce additional computational overhead and inference latency compared to non-diffusion Euclidean counterparts. Second, our diffusion process is implemented in logit coordinates and induces simplex-valued states through softmax, rather than perform-

ing intrinsic Riemannian Brownian motion directly on the simplex. Third, the history-dependent nature of our DNS strategy may exhibit a temporal lag when adapting to sudden semantic drifts triggered by bursty events, potentially affecting responsiveness to unexpected distributional shifts.

Acknowledgments

The authors sincerely thank the reviewers for their valuable comments, which improved the paper. The work is supported by the National Natural Science Foundation of China (62276057) and the Fundamental Research Funds for the Central Universities (No. N26GFZ009).

References

- Elizabeth Boschee, Jennifer Lautenschlager, Sean O’Brien, Steve Shellman, James Starz, and Michael Ward. 2015. [ICEWS Coded Event Data](#).
- Borui Cai, Yong Xiang, Longxiang Gao, He Zhang, Yunfeng Li, and Jianxin Li. 2023. Temporal knowledge graph completion: a survey. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence*, pages 6545–6553.
- Yuxiang Cai, Qiao Liu, Yanglei Gan, Changlin Li, Xueyi Liu, Run Lin, Da Luo, and JiayeYang JiayeYang. 2024. Predicting the unpredictable: uncertainty-aware reasoning over temporal knowledge graphs via diffusion process. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 5766–5778.
- Wei Chen, Yuting Wu, Shuhan Wu, Zhiyu Zhang, Mengqi Liao, Youfang Lin, and Huaiyu Wan. 2025. Cogntke: A cognitive temporal knowledge extrapolation framework. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 14815–14823.
- Nazim Choudhury, Fahim Faisal, and Matloob Khushi. 2020. Mining temporal evolution of knowledge graphs and genealogical features for literature-based discovery prediction. *Journal of Informetrics*, 14(3):101057.
- Songgaojun Deng, Huzefa Rangwala, and Yue Ning. 2020. Dynamic knowledge graph based multi-event forecasting. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 1585–1595.
- Yimin Deng, Yuxia Wu, Yejing Wang, Guoshuai Zhao, Li Zhu, Qidong Liu, Derong Xu, Zichuan Fu, Xian Wu, Yefeng Zheng, Xiangyu Zhao, and Xueming Qian. 2025. A multi-expert structural-semantic hybrid framework for unveiling historical patterns in temporal knowledge graphs. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 20553–20565.
- Hao Dong, Zhiyuan Ning, Pengyang Wang, Ziyue Qiao, Pengfei Wang, Yuanchun Zhou, and Yanjie Fu. 2023. Adaptive path-memory network for temporal knowledge graph reasoning. In *IJCAI*.
- Alberto García-Durán, Sebastijan Dumančić, and Mathias Niepert. 2018. Learning sequence encoders for temporal knowledge graph completion. *arXiv e-prints*, pages arXiv–1809.
- Shansan Gong, Mukai Li, Jiangtao Feng, Zhiyong Wu, and Lingpeng Kong. 2023. Diffuseq: Sequence to sequence text generation with diffusion models. In *International Conference on Learning Representations (ICLR 2023)(01/05/2023-05/05/2023, Kigali, Rwanda)*.
- Simon Gottschalk and Demidova Elena. 2018. Eventkg: A multilingual event-centric temporal knowledge graph. In *European semantic web conference*, pages 272–287. Springer.
- Xiaochuang Han, Sachin Kumar, and Yulia Tsvetkov. 2023. Ssd-lm: Semi-autoregressive simplex-based diffusion language model for text generation and modular control. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11575–11596.
- Zhen Han, Peng Chen, Yunpu Ma, and Volker Tresp. 2020. Explainable subgraph reasoning for forecasting on temporal knowledge graphs. In *International conference on learning representations*.
- Zhen Han, Zifeng Ding, Yunpu Ma, Yujia Gu, and Volker Tresp. 2021. Learning neural ordinary equations for forecasting future links on temporal knowledge graphs. In *Proceedings of the 2021 conference on empirical methods in natural language processing*, pages 8352–8364.
- Yongquan He, Peng Zhang, Luchen Liu, Qi Liang, Wenyuan Zhang, and Chuang Zhang. 2021. Hip network: Historical information passing network for extrapolation reasoning on temporal knowledge graph. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence*, pages 1915–1921.
- Rikui Huang, Wei Wei, Xiaoye Qu, Shengzhe Zhang, Danyang Chen, and Yu Cheng. 2024. Confidence is not timeless: Modeling temporal validity for rule-based temporal knowledge graph forecasting. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10783–10794.
- Jong Ho Jhee, Alberto Megina, Pacôme Constant Dit Beauflis, Matilde Karakachoff, Richard Redon, Alban Gaignard, and Adrien Coulet. 2025. Predicting clinical outcomes from patient care pathways represented with temporal knowledge graphs. *arXiv preprint arXiv:2502.21138*.

- Woojeong Jin, Meng Qu, Xisen Jin, and Xiang Ren. 2020. Recurrent event network: Autoregressive structure inference over temporal knowledge graphs. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6669–6683.
- Xiang Li, John Thickstun, Ishaan Gulrajani, Percy S Liang, and Tatsunori B Hashimoto. 2022. Diffusion-lm improves controllable text generation. *Advances in neural information processing systems*, 35:4328–4343.
- Zixuan Li, Xiaolong Jin, Wei Li, Saiping Guan, Jiafeng Guo, Huawei Shen, Yuanzhuo Wang, and Xueqi Cheng. 2021. Temporal knowledge graph reasoning based on evolutionary representation learning. In *Proceedings of the 44th international ACM SIGIR conference on research and development in information retrieval*, pages 408–417.
- Ruotong Liao, Xu Jia, Yangzhe Li, Yunpu Ma, and Volker Tresp. 2024. Gentkg: Generative forecasting on temporal knowledge graph with large language models. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 4303–4317.
- Yushan Liu, Yunpu Ma, Marcel Hildebrandt, Mitchell Joblin, and Volker Tresp. 2022. Tlogic: Temporal logical rules for explainable link forecasting on temporal knowledge graphs. In *Proceedings of the AAAI conference on artificial intelligence*, volume 36, pages 4120–4127.
- Farzaneh Mahdisoltani, Joanna Biega, and Fabian M Suchanek. 2013. Yago3: A knowledge base from multilingual wikipeas. In *CIDR*.
- Miao Peng, Ben Liu, Wenjie Xu, Zihao Jiang, Jiahui Zhu, and Min Peng. 2024. Deja vu: Contrastive historical modeling with prefix-tuning for temporal knowledge graph reasoning. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 1178–1191.
- Machel Reid, Vincent Josua Hellendoorn, and Graham Neubig. 2023. Diffuser: Diffusion via edit-based reconstruction. In *The Eleventh International Conference on Learning Representations*.
- Haohai Sun, Jialun Zhong, Yunpu Ma, Zhen Han, and Kun He. 2021. Timetraveler: Reinforcement learning for temporal knowledge graph forecasting. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8306–8319.
- Jiapu Wang, Zheng Cui, Boyue Wang, Shirui Pan, Junbin Gao, Baocai Yin, and Wen Gao. 2024a. Ime: Integrating multi-curvature shared and specific embedding for temporal knowledge graph completion. In *Proceedings of the ACM Web Conference 2024*, pages 1954–1962.
- Jiapu Wang, Sun Kai, Linhao Luo, Wei Wei, Yongli Hu, Alan Wee-Chung Liew, Shirui Pan, and Baocai Yin. 2024b. Large language models-guided dynamic adaptation for temporal knowledge graph reasoning. *Advances in Neural Information Processing Systems*, 37:8384–8410.
- Jiapu Wang, Boyue Wang, Meikang Qiu, Shirui Pan, Bo Xiong, Heng Liu, Linhao Luo, Tengfei Liu, Yongli Hu, Baocai Yin, and Wen Gao. 2023a. A survey on temporal knowledge graph completion: Taxonomy, progress, and prospects. *arXiv preprint arXiv:2308.02457*.
- Kunze Wang, Caren Han, and Josiah Poon. 2023b. Re-temp: Relation-aware temporal representation learning for temporal knowledge graph completion. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 258–269.
- Yuwei Xia, Ding Wang, Qiang Liu, Liang Wang, Shu Wu, and Xiao-Yu Zhang. 2024. Chain-of-history reasoning for temporal knowledge graph forecasting. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 16144–16159.
- Siheng Xiong, Yuan Yang, Faramarz Fekri, and James Clayton Kerce. 2024. Tilp: Differentiable learning of temporal logical rules on knowledge graphs. *arXiv e-prints*, pages arXiv–2402.
- Chengjin Xu, Mojtaba Nayyeri, Fouad Alkhoury, Hamed Shariat Yazdi, and Jens Lehmann. 2020. Tero: A time-aware knowledge graph embedding via temporal rotation. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1583–1593.
- Xuanqing Yu, Wangtao Sun, Jingwei Li, Kang Liu, Chengbao Liu, and Jie Tan. 2024. Onsep: A novel online neural-symbolic framework for event prediction based on large language model. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 6335–6350.
- Fu Zhang, Jinghao Lin, and Jingwei Cheng. 2024a. Salmon: A structure-aware language model with logicity and densification strategy for temporal knowledge graph reasoning. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 8761–8774.
- Mengqi Zhang, Yuwei Xia, Qiang Liu, Shu Wu, and Liang Wang. 2023. Learning latent relations for temporal knowledge graph reasoning. In *ACL*.
- Sensen Zhang, Xun Liang, Simin Niu, Zhendong Niu, Bo Wu, Gengxin Hua, Long Wang, Zhenyu Guan, Hanyu Wang, Xuan Zhang, Zhiyu Li, and Yuefeng Ma. 2025. Integrating large language models and möbius group transformations for temporal knowledge graph embedding on the riemann sphere. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 13277–13285.
- Tianli Zhang, Tongya Zheng, Zhenbang Xiao, Zulong Chen, Liangyue Li, Zunlei Feng, Dongxiang

Zhang, and Mingli Song. 2024b. Language models-enhanced semantic topology representation learning for temporal knowledge graph extrapolation. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, pages 3227–3236.

Liang Zhao. 2021. Event prediction in the big data era: A systematic survey. *ACM Computing Surveys (CSUR)*, 54(5):1–37.

Cunchao Zhu, Muhao Chen, Changjun Fan, Guangquan Cheng, and Yan Zhang. 2021. Learning from history: Modeling temporal knowledge graphs with sequential copy-generation networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 4732–4740.

A Geometric Analysis of Simplex Space

Existing TKGR methods typically represent entities as unconstrained vectors in Euclidean space. For long-tail entities, since they appear only a few times in the training set, their representations can receive only limited supervision signals. As a result, they may be insufficiently separated from frequent entities or contextually similar entities. Meanwhile, dot-product-based global entity scoring is sensitive to embedding norms. Unconstrained Euclidean representations may therefore introduce unstable logit scales, further affecting the ranking results of long-tail entities.

In contrast, we constrain entity states in the simplex coordinate space. For an arbitrary entity e , its probability representation $\mathbf{p}_e$ satisfies $p_i \geq 0$ and $\sum_i p_i = 1$. This constraint provides a bounded probability coordinate: every state remains inside the simplex and cannot diverge in probability space. The final continuous embedding $\mathbf{p}_e \mathbf{E}$ lies in the convex hull of the entity basis vectors in $\mathbf{E}$; therefore, when basis row norms are controlled by standard regularization, the induced embedding is also norm-controlled.

Second, probability mass competes among candidate entities: increasing the probability assigned to one entity necessarily reduces the mass assigned to others. This competitive structure encourages sparse and discriminative entity distributions. Furthermore, simplex representations have a natural probabilistic interpretation, where entropy can be used to measure distributional uncertainty. This property is consistent with our DNS module, where normalized entropy is used as one component for estimating instance difficulty. We provide a formal analysis of simplex preservation, logistic-normal

diffusion states, convex-hull boundedness, and ranking stability in Appendix H.

B Implementation Details

Training configuration. We implement SimDiff-DNS in PyTorch and optimize all trainable parameters with AdamW. The learning rate is set to 1×10^{-3} . The learning rate is reduced by a factor of 0.5 if the validation MRR does not improve for 5 consecutive epochs. We select the checkpoint with the highest validation MRR and report the corresponding test performance.

Historical context construction. For each query $(s_q, r_q, ?, \tau_q)$, we construct the historical sequence using facts that occur before τ_q . We retain at most $L = 20$ historical quadruples associated with the query subject and relation context, ordered chronologically. If fewer than L historical facts are available, we use zero-padding and mask padded positions in the Transformer attention mask. Following common TKGR practice, we augment the graph with inverse relations, resulting in $2|\mathcal{R}|$ relation embeddings.

Model architecture. The entity and relation hidden dimension is $d = 200$. The Transformer denoising network has 2 layers, 4 attention heads, feed-forward dimension 800, and dropout rate 0.2. The temperature parameter in global entity scoring is set to $\tau = 1.0$. The simplex scaling factor is set to $k = 5.0$.

Diffusion configuration. The diffusion process uses $T = 400$ steps. The noise variance ranges from $\beta_{\text{start}} = 1 \times 10^{-4}$ to $\beta_{\text{end}} = 0.02$.

Dynamic noise scheduling. For DNS, the normalized entropy weight, target confidence loss weight, maximum confidence loss weight, and noise scaling factor are set to $\omega_1 = 0.3$, $\omega_2 = 0.5$, $\omega_3 = 0.2$, and $\gamma = 0.5$, respectively. For clipped DNS, we set $\sigma_{\text{min}} = 1 \times 10^{-4}$ and $\epsilon_\sigma = 1 \times 10^{-5}$. The difficulty score is computed online from the current model prediction distribution and detached from the computation graph when used to determine the adaptive noise coefficient.

Loss and candidate scoring. We use full softmax over all candidate entities rather than sampled negative entities. The training objective is cross-entropy over the entity vocabulary. During testing, we use time-aware filtered evaluation: other entities that form true facts with the same query

	N_{train}	N_{valid}	N_{test}	N_{ent}	N_{rel}	N_{time}	$Interval$
ICEWS14	74845	8514	7371	7128	230	365	24 hours
ICEWS05-15	368868	46302	46159	10488	251	4017	24 hours
YAGO	161540	19523	20026	10623	10	188	1 year

Table 4: Statistics of the datasets.

$(s_q, r_q, ?, \tau_q)$ are removed from the candidate set before computing ranks. MRR, Hits@1, Hits@3, and Hits@10 are computed after filtering.

Calibration and diagnostic metrics. For ECE, we use 10 equal-width confidence bins over $[0, 1]$. For NLL and Brier score, we compute probabilities over the filtered candidate distribution and evaluate the ground-truth entity probability. For entropy-correlation diagnostics, we compute normalized entropy

$$H(p) = -\frac{1}{\log |\mathcal{E}|} \sum_i p_i \log p_i.$$

We report Spearman correlation between $H(p)$ and $\log(1 + f)$, where f is the target entity training frequency, AUROC for detecting incorrect top-1 predictions using entropy as the risk score, and Spearman correlation between $H(p)$ and $\log(\text{rank})$.

Euclidean control variants. For Euclidean + L2 regularization, we tune the additional regularization strength over $\{10^{-6}, 10^{-5}, 10^{-4}, 10^{-3}\}$. For Euclidean + norm clipping, we clip each entity embedding row after every optimizer step and tune the clipping radius over $\{1, 2, 5, 10\}$. For Euclidean + label smoothing, we tune the smoothing coefficient over $\{0.05, 0.1, 0.2\}$. All control variants use the same Transformer architecture, optimizer, batch size, evaluation protocol, and random seeds as SimDiff-DNS.

Random seeds and hardware. We run each main experiment with three random seeds: 2023, 2024, and 2025. Unless otherwise specified, tables report the mean over three runs. All experiments are conducted on an NVIDIA A100 GPU.

Datasets ICEWS14 and ICEWS05-15 are derived from the Integrated Crisis Early Warning System (ICEWS) (Boschee et al., 2015), which records political events that occurred in 2014 and from 2005 to 2015, respectively. YAGO (Mahdisoltani et al., 2013) is a knowledge base that includes temporal information. The statistics of these datasets are shown in Table 4.

C Baselines

The comparison of TKG reasoning models with our work is presented as follows:

CyGNet (Zhu et al., 2021) introduces time-aware replication generation, combining new facts with repeated pattern recognition to improve prediction accuracy.

RE-NET (Jin et al., 2020) combines event encoding and neighbor aggregation using an autoregressive architecture to sequentially reason about future facts.

xERTE (Han et al., 2020) is based on query subgraphs and integrates temporal attention and reverse update, taking into account both accuracy and interpretability.

REGCN (Li et al., 2021) combines relation-aware convolution with gated recurrence to dynamically model entity relations and fuse static attributes.

ODE (Han et al., 2021) extends multi-relation graph convolution to continuous time, integrates temporal structures, and models the dynamic formation and resolution of relationships.

TLogic (Liu et al., 2022) extracts logical rules based on temporal random walks, taking into account both temporal consistency and inductiveness, and supports rule migration and cross-set prediction.

TempValid (Huang et al., 2024) dynamically models rule confidence based on time functions and combines adversarial and time-aware negative sampling to improve learning efficiency.

ONSEP (Yu et al., 2024) integrates dynamic causal rule mining and dual history enhanced generation.

ChapTER (Peng et al., 2024) integrates contrastive learning and prefix tuning, and uses virtual time prefixes to achieve low-parameter fine-tuning and multi-scenario adaptation.

STORE (Zhang et al., 2024b) combines time-aware semantic sampling and virtual tokens, using multi-head attention to jointly optimize both semantic and topological representations of temporal knowledge graphs.

CoH (Xia et al., 2024) leverages higher-order historical information to enhance temporal reasoning capabilities in large language models.

LLM-DA (Wang et al., 2024b) utilizes large language models to extract temporal rules, enabling dynamic adaptation to ever-changing knowledge.

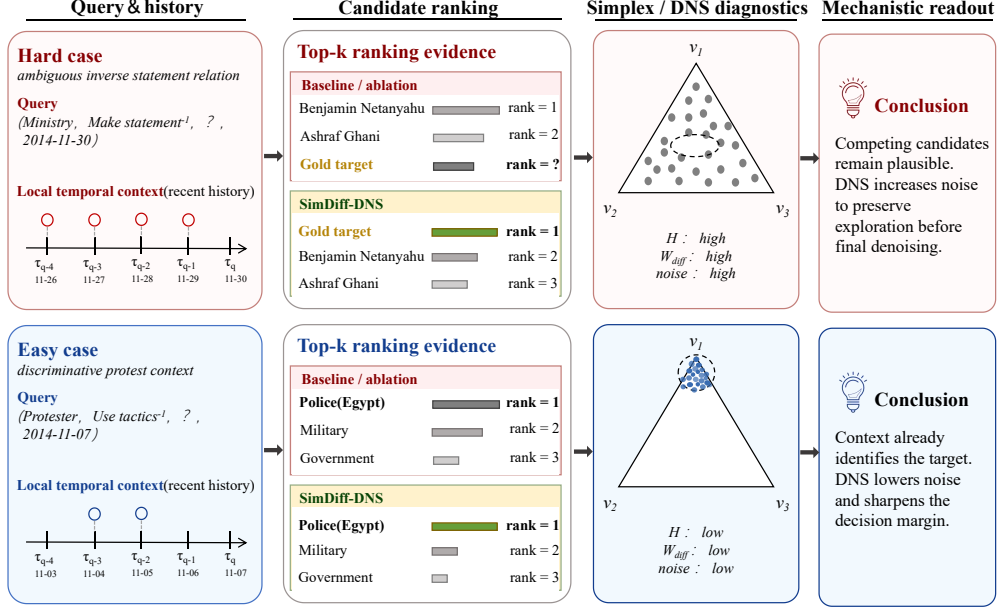


Figure 5: Qualitative comparison of hard and easy ICEWS14 queries using local context, top- k ranking, simplex uncertainty, and DNS diagnostics.

GenTKG (Liao et al., 2024) integrates temporal logic rule retrieval and few-shot instruction fine-tuning, connecting temporal knowledge graphs with large language models.

DiffuTKG (Cai et al., 2024) frames temporal knowledge graph reasoning as a denoising process for future fact sequences. It restores target facts using conditional sequence encoding and a Transformer-based denoiser, while applying uncertainty regularization to reduce prediction bias and handle rare or unseen facts.

CognTKE (Chen et al., 2025) introduces a temporal cognitive relation directed graph and performs dual-process-inspired global shallow reasoning and local deep reasoning, enabling interpretable extrapolation over both long-range historical relations and complex multi-hop temporal paths.

MESH (Deng et al., 2025) introduces a multi-expert structural-semantic hybrid framework to adaptively integrate graph-structural and semantic reasoning for different temporal event prediction scenarios.

D Case Study

Figure 5 compares two ICEWS14 examples. For the hard query (Ministry, Make statement⁻¹, ?, 2014-11-30), multiple candidates remain plausible, so SimDiff-DNS produces a dispersed simplex

distribution with higher entropy and assigns stronger DNS perturbation. For the easy query (Protester, Use tactics⁻¹, ?, 2014-11-07), the historical context more clearly supports Police (Egypt), yielding lower entropy, a sharper margin, and weaker perturbation.

E Additional Diagnostic and Component Analyses

This appendix provides additional analyses of long-tail behavior, calibration, representation spaces, and difficulty-aware noise scheduling. Unless otherwise stated, all experiments are conducted on ICEWS14 using the same training configuration and time-aware filtered evaluation protocol as in the main experiments. Ranking metrics are reported in percentage.

E.1 Frequency-Stratified and Representation Diagnostics

For each entity e , we define $f_{\text{train}}(e)$ as the number of times that e occurs as either a subject or an object in the training quadruples. We divide the entities into high-frequency ($f_{\text{train}}(e) \geq 201$), medium-frequency ($51 \leq f_{\text{train}}(e) \leq 200$), and low-frequency ($f_{\text{train}}(e) \leq 50$) groups.

Table 5 evaluates calibration across high-, medium-, and low-frequency entities. SimDiff-DNS achieves the lowest ECE, NLL, and Brier

score in every group, with the largest absolute improvements on low-frequency entities.

Table 6 complements the ranking results with uncertainty, stability, and neighborhood diagnostics. SimDiff-DNS exhibits the strongest entropy-difficulty alignment, the lowest norm-sensitive ranking variation, and the most coherent low-frequency neighborhood structure.

E.2 Representation-Space Controls and Simplex-DNS Interaction

We compare Euclidean, L2-regularized, unit-norm, Gaussian probabilistic, and simplex representations using the same denoising network and a uniform noise schedule. IME-style multi-curvature (Wang et al., 2024a) and 5EL-style projective (Zhang et al., 2025) representations are included as non-Euclidean controls. As shown in Table 7, the simplex representation achieves the best overall and long-tail performance among these controls. Relative to Euclidean + L2, it improves long-tail MRR and H@10 by 1.89 and 2.11 points, respectively.

Table 8 reports a controlled 2×2 comparison of representation space and noise schedule. The difference-in-differences interaction gains between Simplex and DNS are 4.37 points in overall MRR, 5.58 points in long-tail MRR, and 8.88 points in long-tail H@10, showing that the two components are complementary.

E.3 Ablation of DNS Difficulty Signals

DNS estimates sample difficulty from prediction entropy, target-entity confidence, and maximum prediction confidence. Table 9 compares the individual signals, two-signal combinations, and margin- and frequency-based alternatives. Full DNS achieves the best result on every metric, indicating that the three signals provide complementary information.

E.4 Comparison with Alternative Difficulty-Aware Strategies

We compare DNS with uniform noise, inverse DNS, frequency-aware schedules, margin-based DNS, and difficulty-based loss reweighting. Inverse DNS assigns weaker noise to difficult samples. Frequency-aware high-noise and low-noise assign stronger and weaker noise to low-frequency entities, respectively. Difficulty-based loss reweighting retains the uniform noise schedule and changes only the training loss weights.

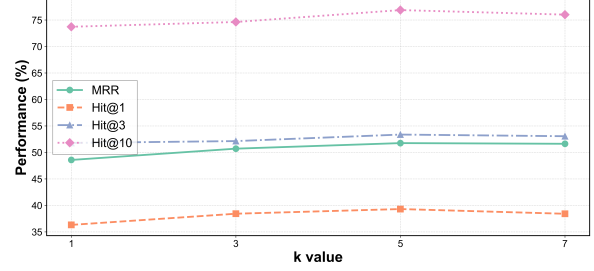


Figure 6: Sensitivity of the simplex scaling factor k on ICEWS14.

As shown in Table 10, the proposed DNS achieves the best results across all metrics. It improves MRR and H@10 over inverse DNS by 3.74 and 9.73 points, respectively, and exceeds difficulty-based loss reweighting by 2.13 points in MRR and 5.82 points in H@10. These results support the use of stronger but bounded training-time corruption for difficult samples.

F Sensitivity Analysis

To further examine the robustness of SimDiff-DNS, we conduct additional sensitivity analysis on the key hyperparameters, including the simplex scaling factor k , the weights used in difficulty estimation, and the adaptive noise scaling factor γ . All experiments are conducted under the same evaluation protocol as the main experiments. Since ICEWS14 sensitivity results are shown in Figure 6 and Table 12, Table 11 and Table 13 report additional results on ICEWS05-15 and YAGO.

F.1 Effect of the Simplex Scaling Factor k

The simplex scaling factor k controls the sharpness of the initial target distribution in Eq. (1). A small k produces a smoother distribution, while a large k makes the initial distribution closer to a one-hot vector. As shown in Table 11, SimDiff-DNS achieves the best performance when $k = 5.0$. When k is too small, the target distribution may not provide sufficiently discriminative supervision. When k is too large, the distribution becomes overly sharp, which may reduce the benefit of distributional denoising.

In DNS, the difficulty score is computed from multiple uncertainty-related terms, including normalized entropy and target confidence loss. We compare different weight configurations in Table 13. Using either normalized entropy or target confidence loss alone leads to inferior performance. The best results are obtained when both terms are

Method	High Freq.			Medium Freq.			Low Freq.		
	ECE↓	NLL↓	Brier↓	ECE↓	NLL↓	Brier↓	ECE↓	NLL↓	Brier↓
Euclidean	4.92	1.46	0.421	6.73	2.03	0.534	10.84	2.78	0.662
Euclidean + Label Smoothing	4.51	1.41	0.407	6.12	1.94	0.509	9.37	2.61	0.625
SimDiff-DNS w/o DNS	3.86	1.32	0.389	5.39	1.82	0.481	8.21	2.46	0.597
SimDiff-DNS	3.08	1.21	0.362	4.24	1.61	0.441	6.43	2.22	0.548

Table 5: Calibration analysis on ICEWS14. ECE is reported in percentage; NLL and Brier score are computed on the filtered candidate distribution.

Entropy and Error Diagnostics				Neighborhood Diagnostics		
Method	H-Freq. Corr.↓	Error AUROC↑	H-Rank Corr.↑	Method	Tail-Hub ↓	Tail Purity ↑
Euclidean	-0.31	0.68	0.42	Euclidean	41.8	33.6
Euclidean + Norm Clip	—	—	—	Euclidean + Norm Clip	36.7	38.9
SimDiff-DNS w/o DNS	-0.44	0.73	0.50	SimDiff-DNS w/o DNS	31.4	44.1
SimDiff-DNS	-0.53	0.79	0.58	SimDiff-DNS	24.6	51.8

Norm and Ranking Stability					
Method	Low Norm CV↓	Freq.-Norm Corr.↓	Low Margin↑	Flip Rate↓	Rank Var.↓
Euclidean	0.61	0.44	0.18	18.6	25.4
Euclidean + Norm Clip	0.42	0.29	0.22	15.1	20.7
SimDiff-DNS w/o DNS	0.34	0.23	0.28	12.4	16.9
SimDiff-DNS	0.29	0.18	0.34	9.7	12.8

Table 6: Compact representation diagnostics on ICEWS14. Entropy diagnostics measure the relations among uncertainty, frequency, prediction error, and rank. Stability diagnostics quantify norm-sensitive ranking behavior, while neighborhood diagnostics characterize low-frequency embedding structure. Flip rate, tail-hub overlap, and tail-neighbor purity are reported in percentage. “—” denotes an unreported diagnostic.

jointly considered, indicating that different aspects of prediction uncertainty provide complementary information for difficulty-aware noise scheduling.

F.2 Effect of the Adaptive Noise Scaling Factor γ

The hyperparameter γ controls the strength of difficulty-aware noise modulation in DNS. A smaller γ weakens the effect of adaptive noise scheduling, while a larger γ may introduce excessive perturbation for difficult samples. As shown in Table 14, $\gamma = 0.5$ achieves the best performance across the three datasets. This suggests that a moderate level of adaptive perturbation is more effective than either under-adjusted or over-amplified noise.

G Additional Results on YAGO

We provide additional comparison results on YAGO in Table 15. Compared with existing generative and diffusion-based baselines, SimDiff-DNS achieves competitive or better performance on most ranking metrics. In particular, SimDiff-DNS achieves strong Hits@3 and Hits@10 results, indicating that the proposed method is effective in

ranking the correct entity within the top candidate set.

H Mathematical Analysis of Simplex-Anchored Logit Diffusion

This appendix provides the mathematical details supporting the simplex-anchored logit diffusion process and the clipped dynamic noise scheduling (DNS) mechanism. We separate the analysis into four parts: simplex preservation, the distribution induced by logit-space diffusion, boundedness and stability of simplex-induced embeddings, and the adaptive posterior under sample-dependent DNS coefficients.

H.1 Notation and Scope

Let $N = |\mathcal{E}|$ denote the number of entities. For an entity e , the initial logit coordinate is $\mathbf{s}_0 \in \mathbb{R}^N$, and its simplex-valued state is

$$\mathbf{p}_0 = \text{softmax}(\mathbf{s}_0) \in \Delta^{N-1},$$

where

$$\Delta^{N-1} = \left\{ \mathbf{p} \in \mathbb{R}^N \mid p_i \geq 0, \sum_{i=1}^N p_i = 1 \right\}.$$

Representation	Overall MRR	Long-tail MRR	Long-tail H@10
Euclidean	45.89	29.06	44.65
Euclidean + L2	46.59	29.83	46.25
Unit-norm	46.62	30.08	46.45
Gaussian probabilistic	46.88	30.64	47.10
IME-style multi-curvature	31.82	20.14	34.72
5EL-style projective	28.67	18.08	31.56
Simplex	47.24	31.72	48.36

Table 7: Comparison of representation spaces on ICEWS14 under a uniform noise schedule. All values are reported in percentage.

Representation	Schedule	Overall MRR	Long-tail MRR	Long-tail H@10
Euclidean	Uniform	45.89	29.06	44.65
Euclidean	DNS	46.04	29.68	45.27
Simplex	Uniform	47.24	31.72	48.36
Simplex	DNS	51.76	37.92	57.86

Table 8: Interaction between representation space and noise schedule on ICEWS14. All values are reported in percentage.

The continuous entity representation is obtained by

$$\mathbf{e}_0 = \mathbf{p}_0 \mathbf{E},$$

where $\mathbf{E} \in \mathbb{R}^{N \times d}$ is the learnable entity basis matrix.

We emphasize that SimDiff-DNS does not inject Gaussian noise directly into the constrained simplex. Instead, it performs diffusion in the unconstrained logit coordinate $\mathbf{s}_t$ and maps the resulting state to the simplex via softmax:

$$\mathbf{p}_t = \text{softmax}(\mathbf{s}_t).$$

Therefore, the proposed process is a simplex-anchored logit diffusion process.

H.2 Simplex Preservation under Softmax Projection

Proposition 1. For any logit vector $\mathbf{s} \in \mathbb{R}^N$, the vector $\mathbf{p} = \text{softmax}(\mathbf{s})$ lies in the interior of the probability simplex Δ^{N-1} .

Proof. The i -th coordinate of $\mathbf{p}$ is

$$p_i = \frac{\exp(s_i)}{\sum_{j=1}^N \exp(s_j)}.$$

Since $\exp(s_i) > 0$ for all i , we have $p_i > 0$. Moreover,

$$\sum_{i=1}^N p_i = \sum_{i=1}^N \frac{\exp(s_i)}{\sum_{j=1}^N \exp(s_j)} = 1.$$

Thus, $\mathbf{p} \in \text{int}(\Delta^{N-1})$.

This proposition ensures that every noisy logit state $\mathbf{s}_t$ corresponds to a valid simplex-valued entity state $\mathbf{p}_t$.

H.3 Logistic-Normal Interpretation of the Forward Process

In the forward process, the logit coordinate is perturbed as

$$\mathbf{s}_t = a_t \mathbf{s}_0 + b_t \boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}),$$

where a_t and b_t are signal and noise coefficients satisfying $a_t^2 + b_t^2 = 1$. Therefore,

$$q(\mathbf{s}_t | \mathbf{s}_0) = \mathcal{N}(a_t \mathbf{s}_0, b_t^2 \mathbf{I}).$$

The corresponding simplex-valued state is

$$\mathbf{p}_t = \text{softmax}(\mathbf{s}_t).$$

Proposition 2. The random variable $\mathbf{p}_t = \text{softmax}(\mathbf{s}_t)$ follows a logistic-normal distribution over the simplex.

Proof. A logistic-normal random variable is obtained by applying the softmax transformation to a Gaussian random vector in logit coordinates. Since $\mathbf{s}_t | \mathbf{s}_0$ is Gaussian and $\mathbf{p}_t = \text{softmax}(\mathbf{s}_t)$, $\mathbf{p}_t$ is logistic-normal on Δ^{N-1} .

Equivalently, using the log-ratio coordinates with the N -th coordinate as reference,

$$u_i = \log \frac{p_{t,i}}{p_{t,N}} = s_{t,i} - s_{t,N}, \quad i = 1, \dots, N-1,$$

we see that $\mathbf{u}$ is Gaussian because it is a linear transformation of the Gaussian vector $\mathbf{s}_t$. Thus, the simplex-valued state $\mathbf{p}_t$ has a logistic-normal interpretation.

H.4 Bounded Convex-Hull Property of Simplex-Induced Embeddings

The simplex-valued entity embedding is

$$\mathbf{e}_t = \mathbf{p}_t \mathbf{E} = \sum_{i=1}^N p_{t,i} \mathbf{E}_i,$$

where $\mathbf{E}_i$ denotes the i -th row of the entity basis matrix $\mathbf{E}$.

Difficulty signal	MRR	H@1	H@10	Δ MRR
Uniform schedule (w/o DNS)	47.24	37.90	65.56	-4.52
Entropy only	44.31	34.12	63.82	-7.45
Target confidence only	45.06	34.52	65.43	-6.70
Maximum confidence only	43.58	33.21	63.37	-8.18
Entropy + maximum confidence	44.08	34.85	61.97	-7.68
Target + maximum confidence	45.23	35.63	63.19	-6.53
Margin-based difficulty	48.38	37.60	69.31	-3.38
Frequency-based difficulty	48.96	38.44	69.74	-2.80
Full DNS	51.76	39.31	76.87	0.00

Table 9: Ablation of DNS difficulty signals on ICEWS14. Δ MRR is the absolute MRR difference from Full DNS. All values are reported in percentage.

Strategy	MRR	H@1	H@10
Uniform noise	47.24	37.90	65.56
Inverse DNS	48.02	38.03	67.14
Frequency-aware high-noise	48.96	38.44	69.74
Frequency-aware low-noise	48.55	38.29	68.75
Margin-based DNS	48.38	37.60	69.31
Difficulty-based loss reweighting	49.63	38.80	71.05
Proposed DNS	51.76	39.31	76.87

Table 10: Comparison of DNS with alternative difficulty-aware strategies on ICEWS14. All values are reported in percentage.

k	ICEWS05-15	YAGO
3.0	50.06	77.78
5.0	53.73	81.61
7.0	52.96	80.09

Table 11: Sensitivity analysis of the simplex scaling factor k in terms of MRR (%).

Proposition 3. The embedding $\mathbf{e}_t$ lies in the convex hull of the entity basis vectors $\{\mathbf{E}_i\}_{i=1}^N$.

Proof. From Proposition 1, $p_{t,i} \geq 0$ and $\sum_i p_{t,i} = 1$. Therefore,

$$\mathbf{e}_t = \sum_{i=1}^N p_{t,i} \mathbf{E}_i$$

is a convex combination of the rows of $\mathbf{E}$. Hence,

$$\mathbf{e}_t \in \text{conv}\{\mathbf{E}_1, \dots, \mathbf{E}_N\}.$$

Corollary 1. If $\|\mathbf{E}_i\|_2 \leq R_E$ for all i , then

$$\|\mathbf{e}_t\|_2 \leq R_E.$$

Proof. By convexity of the norm,

$$\begin{aligned} \|\mathbf{e}_t\|_2 &= \left\| \sum_{i=1}^N p_{t,i} \mathbf{E}_i \right\|_2 \leq \sum_{i=1}^N p_{t,i} \|\mathbf{E}_i\|_2, \\ \sum_{i=1}^N p_{t,i} \|\mathbf{E}_i\|_2 &\leq \sum_{i=1}^N p_{t,i} R_E = R_E. \end{aligned}$$

Weights	ICEWS14			
$(\omega_1, \omega_2, \omega_3)$	MRR	H@1	H@3	H@10
(0.0, 0.8, 0.2)	45.23	35.63	50.15	63.19
(0.8, 0.0, 0.2)	44.08	34.85	48.49	61.97
(0.3, 0.5, 0.2)	51.76	39.31	53.37	76.87

Table 12: Sensitivity analysis of DNS difficulty weights on ICEWS14.

Weights			ICEWS05-15	YAGO
ω_1	ω_2	ω_3		
0.0	1.0	0.0	48.33	73.25
1.0	0.0	0.0	48.19	75.27
0.3	0.5	0.2	53.73	81.61

Table 13: Sensitivity analysis of the difficulty estimation weights in terms of MRR (%).

This result clarifies the role of simplex coordinates. The simplex constraint does not mean that the learnable basis matrix $\mathbf{E}$ is automatically bounded. Rather, it ensures that each entity state is represented as a convex combination of basis vectors. When the basis row norms are controlled by weight decay, norm clipping, or empirical training stability, the resulting simplex-induced embedding is also norm-controlled.

H.5 Bounded Difficulty Score and Clipped DNS

The original difficulty score combines normalized entropy, target confidence loss, and maximum confidence loss. To ensure bounded adaptive coefficients, we use a normalized and clipped difficulty score.

For a predicted distribution $\mathbf{p}$, define

$$\begin{aligned} u_1 &= \frac{H(\mathbf{p})}{\log N}, \quad H(\mathbf{p}) = -\sum_{i=1}^N p_i \log p_i, \\ u_2 &= \min \left(\frac{-\log(p_y + \epsilon)}{\log N}, 1 \right), \end{aligned}$$

γ	ICEWS14	ICEWS05-15	YAGO
0.3	49.64	51.13	80.10
0.5	51.76	53.73	81.61
0.7	50.63	52.94	79.32

Table 14: Sensitivity analysis of the adaptive noise scaling factor γ in terms of MRR (%).

Model	MRR	Hits@1	Hits@3	Hits@10
TempValid	79.72	74.64	84.78	85.73
STORE	64.65	51.94	71.50	83.10
GenTKG	—	79.15	83.00	84.25
ChapTER	14.80	11.50	14.40	20.20
DiffuTKG	80.98	74.25	85.63	89.41
SimDiff-DNS	81.61	76.30	86.33	89.81

Table 15: Additional comparison results on YAGO. Results are reported in percentage.

and

$$u_3 = 1 - p_{\max}, \quad p_{\max} = \max_i p_i.$$

Since $0 \leq H(\mathbf{p}) \leq \log N$, $0 \leq u_1 \leq 1$. By construction, $0 \leq u_2 \leq 1$, and since $0 \leq p_{\max} \leq 1$, $0 \leq u_3 \leq 1$.

We define the bounded difficulty score as

$$\bar{\mathcal{W}}_{\text{diff}} = \text{clip} \left(\frac{\omega_1 u_1 + \omega_2 u_2 + \omega_3 u_3}{\omega_1 + \omega_2 + \omega_3 + \epsilon}, 0, 1 \right).$$

Thus,

$$0 \leq \bar{\mathcal{W}}_{\text{diff}} \leq 1.$$

Let the base noise coefficient be

$$b_t = \sqrt{1 - \bar{\alpha}_t}.$$

The raw adaptive noise coefficient is

$$\hat{b}_t = b_t(1 + \gamma \bar{\mathcal{W}}_{\text{diff}}),$$

where $\gamma \geq 0$. To avoid invalid coefficients, we clip it as

$$\tilde{b}_t = \text{clip} \left(\hat{b}_t, b_{\min}, 1 - \epsilon_\sigma \right),$$

where $0 < b_{\min} < 1$ and $0 < \epsilon_\sigma < 1$. The signal coefficient is then defined by

$$\tilde{a}_t = \sqrt{1 - \tilde{b}_t^2}.$$

Therefore,

$$\tilde{a}_t^2 + \tilde{b}_t^2 = 1,$$

and $\tilde{a}_t$ is always real-valued.

The DNS forward process becomes

$$\mathbf{s}_t = \tilde{a}_t \mathbf{s}_0 + \tilde{b}_t \boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}).$$

H.6 Monotonic SNR Control

The signal-to-noise ratio under DNS is

$$\text{SNR}_t(\bar{\mathcal{W}}_{\text{diff}}) = \frac{\tilde{a}_t^2}{\tilde{b}_t^2} = \frac{1 - \tilde{b}_t^2}{\tilde{b}_t^2}.$$

In the unclipped region,

$$\tilde{b}_t = b_t(1 + \gamma \bar{\mathcal{W}}_{\text{diff}}).$$

Since $b_t > 0$ and $\gamma \geq 0$,

$$\frac{\partial \tilde{b}_t}{\partial \bar{\mathcal{W}}_{\text{diff}}} = b_t \gamma \geq 0.$$

Moreover,

$$\text{SNR}_t = \frac{1}{\tilde{b}_t^2} - 1,$$

so

$$\frac{\partial \text{SNR}_t}{\partial \bar{\mathcal{W}}_{\text{diff}}} = -\frac{2}{\tilde{b}_t^3} \frac{\partial \tilde{b}_t}{\partial \bar{\mathcal{W}}_{\text{diff}}} \leq 0.$$

Thus, increasing difficulty monotonically decreases SNR in the unclipped region. In the clipped region, $\tilde{b}_t$ is constant, so the SNR is non-increasing. This supports the design choice that harder samples receive stronger perturbation and easier samples retain cleaner local geometry.

H.7 Adaptive Gaussian Posterior under DNS

Because DNS introduces sample-dependent coefficients, the reverse posterior should be written using the adaptive coefficients rather than only the standard DDPM coefficients.

For a fixed sample difficulty $\bar{\mathcal{W}}_{\text{diff}}$, define

$$q(\mathbf{s}_t \mid \mathbf{s}_0, \bar{\mathcal{W}}_{\text{diff}}) = \mathcal{N}(\tilde{a}_t \mathbf{s}_0, \tilde{b}_t^2 \mathbf{I}).$$

For notational simplicity, write

$$a_t = \tilde{a}_t, \quad b_t = \tilde{b}_t.$$

To obtain a Markov transition whose marginal matches $q(\mathbf{s}_t \mid \mathbf{s}_0)$, define

$$q(\mathbf{s}_t \mid \mathbf{s}_{t-1}, \bar{\mathcal{W}}_{\text{diff}}) = \mathcal{N}(\rho_t \mathbf{s}_{t-1}, v_t \mathbf{I}),$$

where

$$\rho_t = \frac{a_t}{a_{t-1}},$$

and

$$v_t = b_t^2 - \rho_t^2 b_{t-1}^2.$$

When the noise schedule is monotonic, $v_t \geq 0$.

The posterior is

$$q(\mathbf{s}_{t-1} \mid \mathbf{s}_t, \mathbf{s}_0, \bar{\mathcal{W}}_{\text{diff}}) \propto q(\mathbf{s}_t \mid \mathbf{s}_{t-1}, \bar{\mathcal{W}}_{\text{diff}}) q(\mathbf{s}_{t-1} \mid \mathbf{s}_0, \bar{\mathcal{W}}_{\text{diff}}).$$

Since both terms are Gaussian, the posterior is also Gaussian:

$$q(\mathbf{s}_{t-1} \mid \mathbf{s}_t, \mathbf{s}_0, \bar{\mathcal{W}}_{\text{diff}}) = \mathcal{N}(\tilde{\boldsymbol{\mu}}_t(\mathbf{s}_t, \mathbf{s}_0), \tilde{v}_t \mathbf{I}).$$

The posterior variance is

$$\tilde{v}_t = \left(\frac{1}{b_{t-1}^2} + \frac{\rho_t^2}{v_t} \right)^{-1}.$$

The posterior mean is

$$\tilde{\boldsymbol{\mu}}_t(\mathbf{s}_t, \mathbf{s}_0) = \tilde{v}_t \left(\frac{a_{t-1}}{b_{t-1}^2} \mathbf{s}_0 + \frac{\rho_t}{v_t} \mathbf{s}_t \right).$$

During reverse denoising, $\mathbf{s}_0$ is unknown and is replaced by the model prediction $\hat{\mathbf{s}}_0$:

$$\tilde{\boldsymbol{\mu}}_t(\mathbf{s}_t, \hat{\mathbf{s}}_0) = \tilde{v}_t \left(\frac{a_{t-1}}{b_{t-1}^2} \hat{\mathbf{s}}_0 + \frac{\rho_t}{v_t} \mathbf{s}_t \right).$$

The reverse sampling step is

$$\mathbf{s}_{t-1} = \tilde{\boldsymbol{\mu}}_t(\mathbf{s}_t, \hat{\mathbf{s}}_0) + \sqrt{\tilde{v}_t} \boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}).$$

When DNS is disabled and $a_t = \sqrt{\bar{\alpha}_t}$, $b_t = \sqrt{1 - \bar{\alpha}_t}$, the above posterior reduces to the standard DDPM posterior.

H.8 Ranking-Margin Stability

For a query with correct entity y and a competing entity j , the score margin is

$$m_{y,j} = z_y - z_j = \frac{\hat{\mathbf{e}}_0^\top (\mathbf{E}_y - \mathbf{E}_j)}{\sqrt{d}\tau}.$$

If the denoised representation changes from $\hat{\mathbf{e}}_0$ to $\hat{\mathbf{e}}_0 + \boldsymbol{\delta}$, the margin perturbation is

$$|\Delta m_{y,j}| = \left| \frac{\boldsymbol{\delta}^\top (\mathbf{E}_y - \mathbf{E}_j)}{\sqrt{d}\tau} \right|.$$

By Cauchy–Schwarz,

$$|\Delta m_{y,j}| \leq \frac{\|\boldsymbol{\delta}\|_2 \|\mathbf{E}_y - \mathbf{E}_j\|_2}{\sqrt{d}\tau}.$$

If $\|\mathbf{E}_i\|_2 \leq R_E$ for all i , then

$$\|\mathbf{E}_y - \mathbf{E}_j\|_2 \leq 2R_E,$$

and therefore

$$|\Delta m_{y,j}| \leq \frac{2R_E \|\boldsymbol{\delta}\|_2}{\sqrt{d}\tau}.$$

This bound shows that controlling embedding norms reduces the sensitivity of ranking margins to representation perturbations. The simplex-induced embedding further constrains entity states to the convex hull of basis vectors, which helps reduce unstable ranking behavior for sparse entities when the basis norms are controlled.

H.9 Long-Tail Robustness Interpretation

Let n_e denote the number of training occurrences of entity e . In unconstrained Euclidean embedding models, rare entities with small n_e receive fewer direct updates. This often leads to higher estimation variance, unstable vector norms, and smaller ranking margins.

The simplex-induced representation mitigates this issue in two ways. First, an entity state is represented through a probability vector $\mathbf{p}_e \in \Delta^{N-1}$, so updates are competitive across entity coordinates. Increasing the probability mass assigned to one candidate necessarily decreases the mass assigned to others. Second, the continuous representation $\mathbf{p}_e \mathbf{E}$ lies in the convex hull of the entity basis vectors, making the representation less prone to uncontrolled extrapolation than a free Euclidean point vector when basis norms are controlled.

DNS further complements this behavior. Since the target-confidence component satisfies

$$u_2 = \min \left(\frac{-\log(p_y + \epsilon)}{\log N}, 1 \right),$$

low-confidence samples obtain higher difficulty scores. Rare entities are more likely to have low target confidence during training, so they receive stronger perturbation and broader local exploration. This provides a mechanistic explanation for why the combination of simplex-induced representation and DNS improves long-tail temporal knowledge graph reasoning.

H.10 Summary

The above analysis supports the following claims. First, Gaussian diffusion in logit coordinates induces valid simplex-valued states through softmax projection. Second, the resulting simplex-valued states have a logistic-normal interpretation.

Third, simplex-induced embeddings lie in the convex hull of the entity basis vectors, which provides a more controlled representation than unconstrained free entity vectors when basis norms are bounded. Fourth, clipped DNS guarantees valid signal and noise coefficients and monotonically decreases SNR as sample difficulty increases. Finally, the adaptive posterior remains Gaussian and reduces to the standard DDPM posterior when DNS is disabled.