2025 International Conference on Intelligent Computing
July 26-29, Ningbo, China
https://www.ic-icc.cn/2025/index.php

Dynamic Encoding Selection: Adaptive Mamba and LLM Fusion for Temporal Knowledge Graph Reasoning

Shuchong Wei[1,2][0000-0003-1449-0672] and Liangjun Zang[2(✉)][0000-0003-3483-521X]

[1] School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China
[2] Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
{weishuchong, zangliangjun}@iie.ac.cn

Abstract. This paper introduces Dynamic Encoding Selection (DES), a novel framework for temporal knowledge graph reasoning that adaptively fuses representations from state space models and large language models(LLMs). While recent advancements in sequence modeling have improved temporal pattern recognition, they often lack the semantic understanding necessary for comprehensive reasoning. Similarly, large language models possess rich semantic knowledge but struggle with structured temporal dependencies. Our approach leverages the complementary strengths of both paradigms—employing Mamba's state space architecture to efficiently capture sequential patterns with linear complexity, while utilizing LLMs' pre-trained knowledge for semantic understanding. The key innovation lies in our adaptive fusion mechanism, which dynamically selects between sequential, or fused representations for each query based on contextual factors such as entity connectivity, and query characteristics. This adaptive approach not only enhances prediction accuracy but also provides interpretable insights into temporal knowledge graph reasoning.

Keywords: Adaptive Representation Fusion, Dynamic Encoding Selection.

1 Introduction

Temporal knowledge graph reasoning has emerged as a critical capability for understanding and predicting evolving relationships in dynamic real-world data. Unlike static knowledge graphs, temporal knowledge graphs incorporate time as an essential dimension, capturing how facts and relationships change over different time periods. Despite significant advances in this domain, traditional approaches often struggle with capturing complex temporal dependencies, particularly when handling long-range interactions across irregular time intervals and diverse event types. This limitation significantly impacts performance on prediction tasks involving evolving multi-hop relationships and temporally distant but causally connected events.

Recent advances in sequence modeling have introduced the Mamba architecture [1], which offers unprecedented capabilities for handling long-range sequential patterns with linear computational complexity through its state space model design. Mamba represents a significant breakthrough in sequence modeling, employing selective state space models (SSMs) that combine the expressivity of recurrent architectures with the

parallelizability of transformers. Unlike conventional transformer-based approaches that suffer from quadratic complexity with sequence length, Mamba provides an efficient mechanism for modeling temporal dependencies across varying time scales with linear complexity, making it particularly well-suited for temporal knowledge graph reasoning tasks where efficiently processing sequences of historical facts is essential. However, while Mamba excels at capturing sequential patterns and long-range dependencies in temporal data, it lacks the rich semantic understanding inherent in LLMs. These LLMs have demonstrated remarkable capabilities in comprehending contextual relationships, conceptual similarities between entities and relations, and vast amounts of world knowledge that can be leveraged for reasoning tasks. LLMs can interpret the semantic meaning behind entity and relation descriptions, potentially identifying implicit connections not apparent from structural patterns alone. This semantic understanding is particularly valuable when dealing with sparse regions of knowledge graphs or newly introduced entities with limited historical information.

The fusion of Mamba's efficient sequential modeling with LLMs' semantic comprehension presents a compelling opportunity to address the multi-faceted challenges of temporal knowledge graph completion. Prior approaches have typically relied either on pure embedding-based methods [2–7] that struggle with interpretability or rule-based systems [8, 9] that lack flexibility. More recently, transformer-based architectures [10] have been applied to this domain, but they face inherent limitations in processing long sequences due to their quadratic complexity.

Unlike these existing methods, our proposal leverages the complementary strengths of both Mamba and LLM architectures. We introduce a novel Gumbel-Softmax mechanism to dynamically determine whether to prioritize sequential patterns from Mamba, or to utilize a fusion of both Mamba and LLM representations. This adaptive approach ensures that the model can flexibly emphasize the most informative representation based on the specific characteristics of each query, rather than applying a fixed fusion strategy across all scenarios. The selection mechanism operates in a fully differentiable manner, allowing end-to-end training while maintaining the ability to make discrete pathway choices during inference.

Overall, the main contributions of this paper are as follows:

1. We leverage the complementary strengths of Mamba's efficient sequential modeling and LLMs' semantic understanding to address the complex challenges in temporal knowledge graph reasoning.

2. We propose Dynamic Encoding Selection (DES), a novel framework that adaptively fuses representations based on contextual factors, enabling more accurate and interpretable predictions.

3. We conduct extensive experiments across multiple benchmark datasets that demonstrate significant performance improvements over state-of-the-art methods, with detailed analyses revealing how DES intelligently selects appropriate representations for different temporal reasoning scenarios.

2 Related Work

Temporal Knowledge Graph reasoning involves predicting missing elements within time-evolving knowledge graphs. Research in this domain has primarily concentrated on developing time-sensitive latent representations for entities and relations. These embedding-based approaches incorporate temporal dimensions and evaluate quadruple plausibility through specialized scoring functions.

Many methodologies extend traditional static knowledge graph models. For instance, TTransE [11] builds upon the TransE framework [12], while ChronoR [13] enhances RotatE [14] by representing timestamps as vector translations that function similarly to relations, positioning fact heads in proximity to their corresponding tails. TComplEx and TNTComplEx [15] expand on ComplEx [16] through fourth-order tensor decomposition of temporal knowledge graphs. Similarly, BoxTE [17] incorporates timestamp embeddings into the BoxE [18] architecture. These approaches typically associate temporal information directly with entities and relations based on static knowledge graph foundations. However, such time-independent techniques often fail to adequately capture the dynamic nature of evolving events.

Other research streams have developed time-specific entity representations with specialized functions, including diachronic embedding functions [19], time-rotating functions [20, 21], time-hyperplane functions [22–25], and non-linear embedding functions [26–30].

Despite these advancements, current approaches generally lack explicit utilization of multi-hop structural information and recent temporal facts to enhance prediction accuracy. Some work attempts to address this limitation—TeMP [31] employs message-passing graph neural networks to derive structure-based entity representations at each timestamp, subsequently combining these representations across all timestamps using sequential encoders.

While embedding-based methods demonstrate reasonable link prediction performance, they suffer from limited interpretability. Addressing this concern, NeuSTIP [32] extracts temporal logic rules from temporal knowledge graphs to compute confidence scores for candidate answers in link and time interval prediction tasks. For example, LCGE [33] learns both time-sensitive and time-independent event representations while mining temporal rules to enhance explainability.

3 Problem Formulation

Within a temporal knowledge graph, each quadruple (s, p, o, t) is made up of distinct event components: s as the subject entity, p as the predicate (relation), o as the object entity, and t as the timestamp. Our main goal is to identify the missing component in these incomplete quadruples. We are particularly focused on predicting absent entities in cases such as $(s, p, ?, t)$ or $(?, p, o, t)$.

4 Method

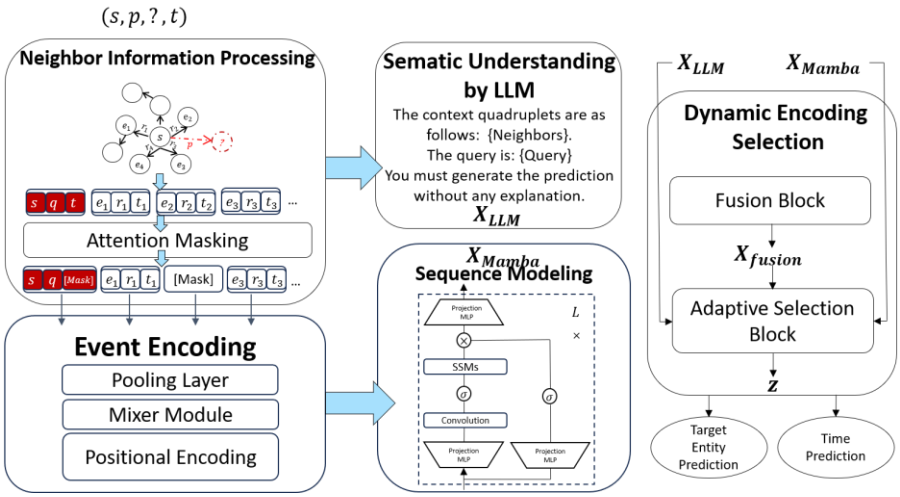

Fig. 1. Overall framework of the DES.

The method processes temporal knowledge graphs through a sophisticated multi-step approach. First, it retrieves and processes neighboring facts for each query entity, applying strategic attention masking. These ID-based representations are then transformed into dense embeddings and processed through an MLP-Mixer to capture inter-component dependencies. Next, a Mamba encoder models sequential relationships between events using state space models, which efficiently capture long-range dependencies with linear computational complexity. The approach then leverages a language model to extract semantic embeddings from the input context. Finally, an adaptive fusion mechanism dynamically integrates the sequential (Mamba) and semantic (LLM) representations using a Gumbel-Softmax selection strategy, allowing the model to optimally combine these complementary information sources for accurate event prediction. The overall algorithm could be illustrated in Algorithm 1 and **Fig. 1**.

4.1 Neighbor Information Processing

For each entity in a query, we retrieve a comprehensive set of neighboring facts from the temporal knowledge graph to provide rich contextual information. This process is crucial for understanding the entity's relationships and temporal patterns.

Neighbor Retrieval: Given an entity e_i in a query, we define its neighborhood as the set of all quadruples in the knowledge graph that involve e_i as either the subject or object:

$$N(e_i) = \{(e_i, r_j, e_j, t_j) \in \mathcal{G}\} \cup \{(e_k, r_k, e_i, t_k) \in \mathcal{G}\}$$

where $\mathcal{G}$ represents the temporal knowledge graph. This bidirectional neighborhood captures all temporal interactions of the entity, providing a comprehensive view of its relationships.

Neighbor Representation: Each neighboring fact is formally represented as a structured triplet of IDs—comprising the entity ID (e_j, identifying the neighboring entity), the relation ID (r_j, denoting the connection between entities), and the time ID (t_j, marking the interaction's timestamp). To handle the variable number of neighbors across entities, we define a maximum context size M and use attention masking strategies to focus only on valid neighbors. These IDs are stored in a context tensor $C_i \in R^{N_i \times 3}$ for each entity e_i, where N_i is the number of neighbors.

Attention Masking Strategies:

1. Uniform Sampling: During training, we employ uniform sampling to select neighbors when the number of neighbors exceeds the maximum context size:

$$\mathbf{M_{sample}} = \text{UniformSample}(\mathbf{C_i}, M)$$

where $M_{attention}$ is a binary mask where 1 indicates a retained neighbor and 0 indicates a discarded one. This sampling strategy ensures that our model can handle entities with varying neighborhood sizes and prevents bias toward entities with more connections.

2. Ground Truth Masking: During training, we mask out the ground truth facts from the neighborhood to prevent information leakage:

$$\mathbf{M_{gt}} = I\left[\left(e_j \neq e_{gt}\right) \vee \left(r_j \neq r_{gt}\right) \vee \left(t_j \neq t_{gt}\right)\right]$$

where e_{gt}, r_{gt}, and t_{gt} are the ground truth entity, relation, and time respectively. This ensures that our model learns to make predictions without directly accessing the answer.

3. Random Context Dropout: To improve model robustness, we randomly mask out some neighboring facts during training:

$$\mathbf{M_{dropout}} = \text{Bernoulli}\left(1 - p_{dropout}\right)$$

where $p_{dropout}$ is the context dropout probability. The final attention mask combines both sampling, ground truth masking and random dropout:

$$\mathbf{M_{attention}} = \mathbf{M_{sample}} \wedge \mathbf{M_{dropout}} \wedge \mathbf{M_{gt}}$$

This strategy forces the model to make predictions with incomplete context, improving its generalization to real-world scenarios where complete information may not be available.

4.2 Event Encoding

After retrieving and processing the neighboring facts, we transform the ID-based representations into dense embeddings:

$$\mathbf{E_{entity}} = \text{Embed}_{entity}(\mathbf{C_i}[:,0])$$

$$\mathbf{E_{relation}} = \text{Embed}_{relation}(\mathbf{C_i}[:,1])$$

$$\mathbf{E_{time}} = \text{Embed}_{time}(\mathbf{C_i}[:,2])$$

where $E_{entity} \in R^{N_i \times d}$, $E_{relation} \in R^{N_i \times d}$, $E_{time} \in R^{N_i \times d}$.

These embeddings are subsequently concatenated to form the initial representation of each neighboring fact:

$$\mathbf{X_{neighbors}} = \text{Stack}\left(\left[\mathbf{E_{entity}}, \mathbf{E_{relation}}, \mathbf{E_{time}}\right]\right)$$

where $X_{neighbors} \in R^{N_i \times 3 \times d}$.

The query fact undergoes analogous embedding and is prepended to the sequence of neighboring facts:

$$\mathbf{X_{query}} = \text{Stack}\left(\left[\mathbf{e_s}, \mathbf{e_p}, \mathbf{e_t}\right]\right)$$

$$\mathbf{X^{(0)}} = \text{Concat}\left(\left[\mathbf{X_{query}}, \mathbf{X_{neighbors}}\right]\right)$$

where $X_{query} \in R^{1 \times 3 \times d}$, $X^{(0)} \in R^{(N_i+1) \times 3 \times d}$.

This composite representation serves as input to our MLP-Mixer module, which processes the contextual information to generate entity-aware representations. Through this comprehensive neighbor information processing, our model effectively harnesses the rich structural and temporal context inherent in temporal knowledge graphs, thereby facilitating more precise predictions for missing facts.

To effectively capture inter-component dependencies within events, we employ an MLP-Mixer architecture, inspired by [34], to transform the embeddings of event components into a unified event representation. Specifically, for each event $E_i \in R^{3 \times d}$ in $X^{(0)}$(query or neighboring fact), it is represented as:

$$\mathbf{E_i} = \left[\mathbf{e_s}; \mathbf{e_p}; \mathbf{e_t}\right] + \mathbf{E_{pos}}$$

where e_s, e_p, and e_t denote the embeddings for entity, relation, and time respectively, while E_{pos} represents the role embedding to distinguish between different component types (subject entity, relation, object entity, and timestamp).

The MLP-Mixer processes these embeddings through two complementary mixing operations:

$$\mathbf{E_i^{channel}} = \mathbf{E_i} + W_2\sigma\left(W_1 \cdot Norm(\mathbf{E_i})\right)$$

$$\mathbf{E_i^{patch}} = \mathbf{E_i^{channel}} + W_4\sigma\left(W_3 \cdot Norm\left(\mathbf{E_i^{channel}}\right)\right)$$

The channel MLP operates along the feature dimension (columns), mapping each feature from R^3 to R^3 to capture correlations between identical features across different components. The patch MLP operates along the component dimension (rows), mapping each component from R^d to R^d to model interactions between distinct features within each component. The final representation of the event is derived via MLP-Pooling:

$$\mathbf{E_i^{event}} = W_5 \cdot Norm\left(\mathbf{E_i^{patch}}\right)$$

This methodology enables comprehensive modeling of event characteristics by capturing intricate dependencies between different components of the input. $X^{(1)} \in R^{(N_i+1)\times d}$ comprises the event embeddings $\{E_1^{event}, ..., E_{N_i+1}^{event}\}$, serving as the initial input for the subsequent sequence modeling module.

4.3 Sequence Modeling

We present a comprehensive formulation of event sequence processing through the Mamba encoder, which employs selective state space models to capture temporal relationships. Beginning with input event embeddings $X^{(L-1)} \in R^{N\times d}$ (where N is sequence length and d is embedding dimension), the encoder transforms these representations through a sequence of operations that can be summarized as:

$$\mathbf{X^L} = MambaEncoder\left(\mathbf{X^{(L-1)}}\right)$$

The processing pipeline initiates with RMS Normalization of the input embeddings, creating a normalized representation that stabilizes training:

$$\mathbf{H_{norm}} = RMSNorm(\mathbf{X^{L-1}})$$

where $\boldsymbol{H_{norm}} \in R^{N\times d}$.

Following normalization, the architecture bifurcates into parallel processing paths. The first path applies a linear projection followed by convolution and non-linear activation to extract local patterns:

$$\mathbf{H_{linear1}} = \mathbf{W_1 H_{norm}} + \mathbf{b_1}$$

$$\mathbf{H_{conv}} = Conv(\mathbf{H_{linear1}})$$

$$\mathbf{H_{SiLU1}} = SiLU(\mathbf{H_{conv}})$$

where $Conv$ is the convolution operation and $SiLU$ is the activation function.

This activated representation then passes through the SSM, which captures long-range dependencies through its recurrent structure:

$$\mathbf{H_{SSM}} = \mathrm{SSM}(\mathbf{H_{SiLU1}})$$

Concurrently, the second path applies an independent linear transformation and activation function to the normalized input, providing a complementary representation:

$$\mathbf{H_{linear2}} = \mathbf{W_2 H_{norm}} + \mathbf{b_2}$$

$$\mathbf{H_{SiLU2}} = SiLU(\mathbf{H_{linear2}})$$

The outputs from both paths undergo element-wise multiplication, creating a gating mechanism that adaptively controls information flow between the temporal and contextual features:

$$\mathbf{H}_{\times} = \mathbf{H_{SSM}} \odot \mathbf{H_{SiLU2}}$$

This multiplicative interaction is projected to the output space through a final linear transformation:

$$\mathbf{H_{final}} = \mathbf{W_3}\mathbf{H}_{\times} + \mathbf{b_3}$$

The process culminates with a residual connection that combines the transformed features with the original input, promoting gradient flow and representation stability:

$$\mathbf{X^L} = \mathbf{X^{L-1}} + \mathbf{H_{final}}$$

This entire processing block is applied L times in succession, with each iteration progressively refining the temporal representations and enhancing the model's capacity to capture intricate sequential patterns across diverse time scales.

We extract the final position of the sequence as X_{Mamba} to serve as the comprehensive sequence representation.

4.4 Semantic Understanding by LLM

We use the specific prompt "The context quadruplets are as follows: {context} The query is: {query}. You must generate the prediction without any explanation." to include the query event and its neighbors into LLM to exploit the semantic understanding to promote event prediction. Then we extract the hidden state X_{LLM} corresponding to the final token of the input prompt to represent the semantic embedding. The "final token" refers to the last token in the input sequence before the model begins generating a response. At this position, the hidden state has processed the entire context and query information, making it an ideal representation that encapsulates the complete semantic understanding of the input. This approach leverages the inherent semantic understanding capabilities of LLM to capture the contextual relationships between events.

Furthermore, this method allows us to benefit from the LLM's pre-trained knowledge without requiring extensive fine-tuning for specific event prediction scenarios. The structured prompt format ensures consistent representation of the event context, while the extraction of hidden states provides a standardized approach to obtaining semantic embeddings that can be readily integrated into existing prediction frameworks.

4.5 Dynamic Encoding Selection

Fusion Block: Given semantic embeddings $X_{LLM} \in R^{d_2}$ and sequence embeddings $X_{Mamba} \in R^{d}$, the fusion process proceeds through the following sequence of operations:

First, the semantic embeddings are projected to match the dimensionality of the sequence space:

$$X'_{LLM} = W_{trans} X_{LLM}$$

where $W_{trans} \in R^{d \times d_2}$ is the transformation matrix of the linear projection.

Concurrently, the sequence embeddings undergo an alignment transformation:

$$X'_{Mamba} = W_{align} X_{Mamba}$$

where $W_{align} \in R^{d \times d}$ represents the alignment matrix.

The transformed representations are concatenated along the feature dimension:

$$X_{concat} = [X'_{Mamba}; X'_{LLM}] \in R^{2d}$$

This concatenated representation is then processed through a non-linear transformation using a bottleneck architecture:

$$X_{fusion} = W_{fuse2} \left(\sigma \left(W_{fuse1} X_{concat} \right) \right)$$

where $W_{fuse1} \in R^{10d \times 2d}$, $W_{fuse2} \in R^{d \times 10d}$, and σ denotes the SiLU activation function.

The resulting X_{fusion} representation effectively captures the complementary information from both modalities, enabling the model to leverage linguistic semantics and sequence dynamics simultaneously.

Adaptive Selection Block: We propose an adaptive fusion mechanism that dynamically selects either sequential event representations from Mamba, semantic representations from the LLM, or hybrid representations combining both sequential and semantic information.

This sophisticated approach enables the model to adaptively balance and synthesize complementary information streams for optimal predictive performance.

Given the two candidate representations:

- $X_{Mamba} \in R^{B \times d}$: The Mamba-encoded sequential representation
- $X_{fusion} \in R^{B \times d}$: The fused representation combining both modalities

where B is the batch size and d is the embedding dimension, we formulate our fusion mechanism as follows:

First, we compute the cosine similarity between these representations to generate selection logits:

$$\mathbf{s} = \cos\left(X_{Mamba}, X_{fusion}\right) = \frac{X_{Mamba} \cdot X_{fusion}}{\left|\left|X_{Mamba}\right|\right| \cdot \left|\left|X_{fusion}\right|\right|}$$

We then construct a bidirectional logit vector by concatenating s and its negation:

$$\mathbf{l} = [\mathbf{s}, -\mathbf{s}] \in R^{B \times 2}$$

The Gumbel-Softmax sampling procedure is applied to introduce beneficial stochasticity while maintaining differentiability:

$$\mathbf{g} = -\log(-\log(\mathbf{u})), \quad \mathbf{u} \sim \text{Uniform}(0,1)$$

$$\boldsymbol{p} = \text{softmax}\left(\frac{\boldsymbol{l} + \boldsymbol{g}}{\tau}\right)$$

where τ is the temperature parameter controlling the sharpness of the distribution.

For discrete selection during forward propagation, we employ the straight-through estimator:

$$\boldsymbol{y}_{\text{hard}} = \text{onehot}(\arg\max(\boldsymbol{p}))$$

$$\hat{\mathbf{y}} = \mathbf{y}_{\text{hard}} - \mathbf{p}.detach() + \mathbf{p}$$

The adaptive fusion is then performed using:

$$\mathbf{z} = \widehat{\mathbf{y}_{:,0}} \cdot X_{Mamba} + \widehat{\mathbf{y}_{:,1}} \cdot X_{fusion}$$

The differentiable nature of Gumbel-Softmax ensures that gradients flow through all pathways during training, facilitating effective learning of the selection policy.

The proposed mechanism enables dynamic switching between Mamba's sequential dynamics and a fused representation incorporating both LLM-derived semantic information and Mamba's processing outputs.

4.6 Training and Optimization

We compute similarities between this representation and candidate entities:

$$p(e_{gt}|q) = \text{Softmax}(\text{Sim}(\mathbf{z}, \mathcal{E}))$$

where Sim calculates similarity scores, q is the predicted entity with the highest score, and $p(e_{gt}|q)$ represents the likelihood of q being the target entity e_{gt}.

To enhance temporal understanding, we implement timestamp recovery by replacing query event timestamps with [MASK] tokens or random alternatives. The model then predicts the correct timestamp:

$$p(\tau_{gt}|q) = \text{Softmax}(\text{Sim}(\mathbf{z}, \mathcal{T}))$$

where $p(\tau_{gt}|q)$ indicates the probability of predicted timestamp matching the ground truth τ_{gt}.

Training utilizes cross-entropy loss for both tasks:

$$\mathcal{L} = -\sum_{(s,p,o,\tau)\in\mathcal{G}} \log\left(p(e_{gt}|q)\right) + \lambda \log\left(p(\tau_{gt}|q)\right)$$

where (s, p, o, τ) represents historical training events and λ weights the timestamp prediction task.

Algorithm 1 Dynamic Encoding Selection: Adaptive Mamba and LLM Fusion for Temporal Knowledge Graph Reasoning

Require:
Knowledge Graph $\mathcal{G}$, Query $q = (s, q, ?, t)$
Maximum context size M, Context dropout probability $p_{dropout}$
Ensure: Entity prediction $\hat{e_o}$
function $MainPipeline(q, \mathcal{G}, M, p_{dropout})$
$C_i \leftarrow NeighborInfoProcessing(e_s, \mathcal{G}, M, p_{dropout})$ ▷ Retrieve and mask neighbors
$X^{(1)} \leftarrow EventEncoding\{C_i, q\}$ ▷ Encode event via MLP-Mixer
$X_{Mamba} \leftarrow SequenceModeling\{X^{(1)}\}$ ▷ Encode event sequence with Mamba
$X_{LLM} \leftarrow SemanticUnderstanding\{q, C_i\}$ ▷ Extract semantic information
$X_{fusion} \leftarrow FusionBlock\{X_{Mamba}, X_{LLM}\}$ ▷ Create fusion representation
$z \leftarrow AdaptiveSelection\{X_{Mamba}, X_{fusion}\}$ ▷ Apply Gumbel-Softmax selection
$\hat{e_o} \leftarrow \arg\max_{e \in \mathcal{E}} \mathrm{Sim}(z, e)$ ▷ Select highest similarity entity
Return $\hat{e_o}$
end function
$\hat{e_o} \leftarrow MainPipeline\{q, \mathcal{G}, M, p_{dropout}\}$
Return $\hat{e_o}$

5 Experiments

Table 1. Statistics of the Experimental Datasets.

Dataset	$\lvert\mathcal{E}\rvert$	$\lvert\mathcal{R}\rvert$	$\lvert\mathcal{T}\rvert$	Time	N_{train}	N_{valid}	N_{test}
ICEWS14	7,128	230	365	24 hours	72,826	8,941	8,963
YAGO11k	10,623	10	73	1 year	16,406	2,050	2,051
Wikidata12	12,554	24	84	1 year	32,497	4,062	4,062

This section outlines the datasets, implementation details, and experimental results to validate the efficacy of our proposed framework.

5.1 Datasets and Evaluation Criteria

Following established practices in prior work [23, 25, 29, 35], we evaluate our framework on three benchmark datasets: ICEWS14, YAGO11k, and Wikidata12k. ICEWS14 is derived from the Integrated Crisis Early Warning System [36], comprising temporally annotated news events from 2014. YAGO11k and Wikidata12k are curated subsets of YAGO3 [37] and Wikidata [38], respectively, both featuring timestamped facts to ensure temporal generalizability. Each dataset is partitioned into training, validation, and test sets. **Table 1** encapsulates their statistical properties, including temporal

granularity. Performance is quantified using Mean Reciprocal Rank (MRR), the average reciprocal rank of the correct entity among predicted candidates, and Hits@k, the percentage of instances where the true entity appears within the top *k* ranked predictions.

5.2 Baseline Methods

We classify temporal knowledge graph reasoning techniques into five categories:

- **Time-Independent Methods**: These models incorporate timestamps as direct embeddings, treating them as translational offsets between entities, such as TimePlex [2].
- **Time-Rotating Function Methods**: Temporal embeddings are derived by rotating time-independent entity representations. Examples: TeRo [20], RotateQVS [21].
- **Time-Hyperplane Function Methods**: Entities and relations are projected onto temporally defined hyperplanes to maintain contextual coherence. Examples: TGeomE [24], TeLM [23], HyIE [25].
- **Non-Linear Function Methods**: These methods model the embeddings of entities, relations, and timestamps in their specific spaces and introduce neural networks (e.g., convolutional neural networks, graph neural networks, Transformer) to facilitate the information exchange and capture semantics. The representative studies are QDN [29], SANe [28], T-GAP [27], ECEformer [35].
- **Rule-Based Methods**: Leverage logical rules for inferring missing links, enhancing interpretability. Examples: NeuSTIP [32], LCGE [33].

5.3 Implementation Details

For model implementation convenience, we leveraged the LibKGE framework[1], which offers comprehensive implementations of training protocols, hyperparameter tuning methods, and evaluation metrics compatible with various models. We implemented DES using PyTorch, with the Mamba component adapted from the official State Space Models repository[1]. For LLM integration, we utilized the LLaMA-7B as pretrained transformer model. All experiments were conducted on NVIDIA A100 GPUs with 40GB memory. This approach ensures our results are reproducible and facilitates equitable comparisons with alternative approaches. We standardized the dimensionality of all input embeddings at 256, encompassing entities, relations, and temporal markers. The feed-forward neural network layers maintain a consistent hidden width of 1024 units across all modules.

[1] https://github.com/state-spaces/mamba

5.4 Experiment Results

Table 2. Comparison of various models on the YAGO11K, Wikidata12k and ICEWS14 datasets. The best results are highlighted in bold.

Model	YAGO11k				Wikidata12k				ICEWS14			
	MRR	Hits@1	Hits@3	Hits@5	MRR	Hits@1	Hits@3	Hit@5	MRR	Hits@1	Hits@3	Hits@5
TimePlex	23.64	16.92	-	36.71	33.35	22.78	-	53.20	60.40	51.50	-	77.11
TeRo	18.70	12.10	19.70	31.90	29.90	19.80	32.90	50.70	56.20	46.80	62.10	73.20
RotateQVS	18.90	12.40	19.90	32.30	-	-	-	-	59.10	50.70	64.20	75.40
TeLM	19.10	12.90	19.40	32.10	33.20	23.10	36.00	54.20	62.50	54.50	67.30	77.40
TGeomE	19.50	13.00	19.60	32.60	33.30	23.20	36.20	54.60	62.90	54.60	68.00	78.00
HyIE	19.10	12.50	20.10	32.60	30.10	19.70	32.80	50.60	63.10	56.30	68.70	78.60
QDN	19.80	13.10	20.10	32.80	-	-	-	-	64.30	56.70	68.80	78.40
SANe	25.00	18.00	26.60	40.10	43.20	33.10	48.30	64.00	63.80	55.80	68.80	78.20
T-GAP	-	-	-	-	-	-	-	-	61.00	50.90	67.70	79.00
ECEformer	25.63	19.48	26.88	37.84	47.81	41.35	49.96	60.35	**71.70**	**67.31**	**73.60**	80.30
NeuSTIP	25.23	18.45	-	37.76	34.78	24.38	-	53.75	-	-	-	-
LCGE	-	-	-	-	42.90	30.40	49.50	**67.70**	66.70	58.80	71.40	**81.50**
DES	**29.79**	**23.15**	**31.56**	**42.86**	**48.14**	**41.38**	**50.36**	61.33	68.35	60.88	72.46	81.10

Table 2 presents the performance metrics of our proposed DES model compared with existing approaches across three benchmark datasets. The experimental results reveal DES's superior capability in handling diverse temporal knowledge graph characteristics across both Wikidata12k and YAGO11k datasets. On these benchmarks, DES demonstrates comprehensive improvements over existing approaches, establishing new state-of-the-art performance across all evaluation metrics. The model shows particularly strong advantages in complex temporal reasoning tasks, as evidenced by its superior performance in both precise entity ranking (Hits@1) and broader retrieval scenarios (Hits@3 and Hits@10).

However, the margin of improvement on ICEWS14 is relatively more modest compared to the substantial gains observed on YAGO11k and Wikidata12k. This suggests that temporal reasoning scenarios present unique challenges that require further methodological refinement.

5.5 Ablation Studies

We conducted an ablation study to evaluate each component's contribution to our DES framework. **Table 3** shows performance metrics when key components are progressively removed. Removing the adaptive selection mechanism reduced MRR from 29.79 to 28.17 (-1.62 points), with similar decreases in Hits metrics. This confirms the importance of dynamically selecting appropriate representations based on query context. Further removing the fusion mechanism caused additional performance drops, with MRR falling to 26.79 (-3.00 points from full model). This demonstrates the value of combining complementary information from both sequential and semantic sources. The most significant decline occurred when removing the Mamba architecture, resulting in an MRR of only 25.67 (-4.12 points from full model). This highlights Mamba's critical role in efficiently capturing long-range temporal dependencies with linear complexity. These results confirm that each component—Mamba's sequential modeling, the fusion mechanism, and adaptive selection—contributes uniquely to DES's effectiveness. The

full model's superior performance across all metrics validates our integrated approach for temporal knowledge graph reasoning.

Table 3. Ablation study of DES on the YAGO11k dataset.

Model Variant	MRR	Hits@1	Hits@3	Hits@10
DES(Full)	**29.79**	**23.15**	**31.56**	**42.86**
w/o Adaptive Selection	28.17	21.38	29.87	40.08
w/o Adaptive Selection + Fusion	26.79	20.20	28.31	39.56
w/o Adaptive Selection + Fusion + Mamba	25.67	18.78	27.50	37.85

5.6 Further Analysis

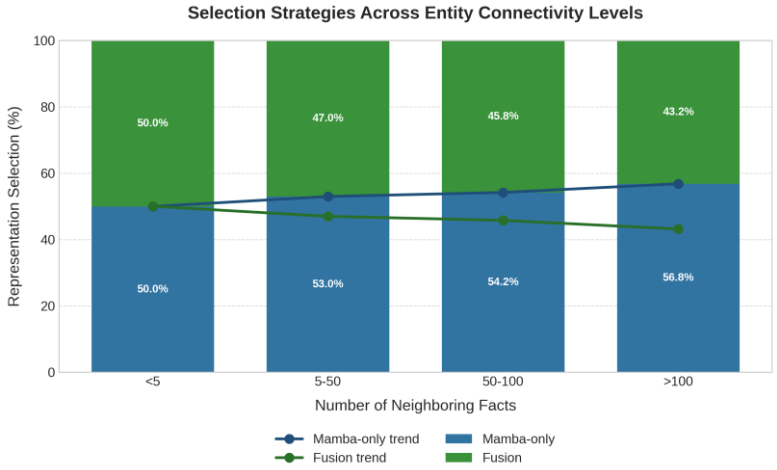

Fig. 2. Representation Selection Distribution Across Entity Connectivity Levels.

To gain deeper insights into how our DES model dynamically selects between different representation types across various scenarios, we conducted a comprehensive analysis of the adaptive selection mechanism on the YAGO11k dataset. We examined how the model distributes its attention across the two representation pathways (Mamba-only and fusion representation) under different conditions and query types.

Fig. 2 presents the representation selection distribution across different entity connectivity levels. The results reveal a clear pattern: as entity connectivity increases, the model's selection strategy systematically shifts. With sparsely connected entities (<5 neighboring facts), the model distributes attention equally between both pathways (50% each). However, as connectivity increases to highly connected entities (>100 neighboring facts), the model increasingly favors Mamba-only representations (56.8%) over fusion representations (43.2%).

This shift suggests that for entities with rich contextual information, the sequential modeling capabilities of Mamba become increasingly valuable, while the fusion pathway remains important but less dominant. This adaptive behavior highlights how our model dynamically leverages different representation types based on the structural characteristics of the knowledge graph, optimizing performance across varying connectivity scenarios.

6 Conclusion

In this work, we presented a novel approach to temporal knowledge graph reasoning that harnesses the complementary strengths of Mamba's efficient sequential modeling and LLMs' semantic understanding. Our proposed Dynamic Encoding Selection framework adaptively fuses representations from these two paradigms based on contextual factors such as temporal patterns and entity relationships, enabling more accurate and interpretable predictions in temporal knowledge graphs.

The potential promising directions of future work include exploring more sophisticated fusion mechanisms between sequence models and language models to better capture the nuanced interplay between temporal dynamics and semantic relationships.

References

1. Gu, A., Dao, T.: Mamba: Linear-time sequence modeling with selective state spaces. arXiv preprint arXiv:2312.00752. (2023).
2. Jain, P., Rathi, S., Mausam, Chakrabarti, S.: Temporal Knowledge Base Completion: New Algorithms and Evaluation Protocols. In: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP). Association for Computational Linguistics, Stroudsburg, PA, USA (2020). https://doi.org/10.18653/v1/2020.emnlp-main.305.
3. Xu, C., Nayyeri, M., Alkhoury, F., Shariat Yazdi, H., Lehmann, J.: TeRo: A Time-aware Knowledge Graph Embedding via Temporal Rotation. In: Scott, D., Bel, N., and Zong, C. (eds.) Proceedings of the 28th International Conference on Computational Linguistics. pp. 1583–1593. International Committee on Computational Linguistics, Barcelona, Spain (Online) (2020). https://doi.org/10.18653/v1/2020.coling-main.139.
4. Chen, K., Wang, Y., Li, Y., Li, A.: RotateQVS: Representing Temporal Information as Rotations in Quaternion Vector Space for Temporal Knowledge Graph Completion. In: Muresan, S., Nakov, P., and Villavicencio, A. (eds.) Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). pp. 5843–5857. Association for Computational Linguistics, Dublin, Ireland (2022). https://doi.org/10.18653/v1/2022.acl-long.402.
5. Xu, C., Nayyeri, M., Chen, Y.-Y., Lehmann, J.: Geometric algebra based embeddings for static and temporal knowledge graph completion. IEEE Trans Knowl Data Eng. 35, 4838–4851 (2022).
6. Xu, C., Chen, Y.-Y., Nayyeri, M., Lehmann, J.: Temporal Knowledge Graph Completion using a Linear Temporal Regularizer and Multivector Embeddings. In: Toutanova, K., Rumshisky, A., Zettlemoyer, L., Hakkani-Tur, D., Beltagy, I., Bethard, S., Cotterell, R., Chakraborty, T., and Zhou, Y. (eds.) Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language

Technologies. pp. 2569–2578. Association for Computational Linguistics, Online (2021). https://doi.org/10.18653/v1/2021.naacl-main.202.
7. Zhang, S., Liang, X., Tang, H., Guan, Z.: Hybrid Interaction Temporal Knowledge Graph Embedding Based on Householder Transformations. In: Proceedings of the 31st ACM International Conference on Multimedia. pp. 8954–8962 (2023).
8. Singh, I., Kaur, N., Gaur, G., others: NeuSTIP: A Novel Neuro-Symbolic Model for Link and Time Prediction in Temporal Knowledge Graphs. arXiv preprint arXiv:2305.11301. (2023).
9. Niu, G., Li, B.: Logic and commonsense-guided temporal knowledge graph completion. In: Proceedings of the AAAI Conference on Artificial Intelligence. pp. 4569–4577 (2023).
10. Fang, Z., Lei, S.-L., Zhu, X., Yang, C., Zhang, S.-X., Yin, X.-C., Qin, J.: Transformer-based Reasoning for Learning Evolutionary Chain of Events on Temporal Knowledge Graph. arXiv preprint arXiv:2405.00352. (2024).
11. Leblay, J., Chekol, M.W.: Deriving validity time in knowledge graph. In: Companion proceedings of the the web conference 2018. pp. 1771–1776 (2018).
12. Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J., Yakhnenko, O.: Translating embeddings for modeling multi-relational data. Adv Neural Inf Process Syst. 26, (2013).
13. Sadeghian, A., Armandpour, M., Colas, A., Wang, D.Z.: Chronor: Rotation based temporal knowledge graph embedding. In: Proceedings of the AAAI conference on artificial intelligence. pp. 6471–6479 (2021).
14. Sun, Z., Deng, Z.-H., Nie, J.-Y., Tang, J.: Rotate: Knowledge graph embedding by relational rotation in complex space. arXiv preprint arXiv:1902.10197. (2019).
15. Lacroix, T., Obozinski, G., Usunier, N.: Tensor decompositions for temporal knowledge base completion. arXiv preprint arXiv:2004.04926. (2020).
16. Trouillon, T., Welbl, J., Riedel, S., Gaussier, É., Bouchard, G.: Complex embeddings for simple link prediction. In: International conference on machine learning. pp. 2071–2080 (2016).
17. Messner, J., Abboud, R., Ceylan, I.I.: Temporal knowledge graph completion using box embeddings. In: Proceedings of the AAAI Conference on Artificial Intelligence. pp. 7779–7787 (2022).
18. Abboud, R., Ceylan, I., Lukasiewicz, T., Salvatori, T.: Boxe: A box embedding model for knowledge base completion. Adv Neural Inf Process Syst. 33, 9649–9661 (2020).
19. Goel, R., Kazemi, S.M., Brubaker, M., Poupart, P.: Diachronic embedding for temporal knowledge graph completion. In: Proceedings of the AAAI conference on artificial intelligence. pp. 3988–3995 (2020).
20. Xu, C., Nayyeri, M., Alkhoury, F., Shariat Yazdi, H., Lehmann, J.: TeRo: A Time-aware Knowledge Graph Embedding via Temporal Rotation. In: Scott, D., Bel, N., and Zong, C. (eds.) Proceedings of the 28th International Conference on Computational Linguistics. pp. 1583–1593. International Committee on Computational Linguistics, Barcelona, Spain (Online) (2020). https://doi.org/10.18653/v1/2020.coling-main.139.
21. Chen, K., Wang, Y., Li, Y., Li, A.: RotateQVS: Representing Temporal Information as Rotations in Quaternion Vector Space for Temporal Knowledge Graph Completion. In: Muresan, S., Nakov, P., and Villavicencio, A. (eds.) Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). pp. 5843–5857. Association for Computational Linguistics, Dublin, Ireland (2022). https://doi.org/10.18653/v1/2022.acl-long.402.
22. Dasgupta, S.S., Ray, S.N., Talukdar, P.: HyTE: Hyperplane-based Temporally aware Knowledge Graph Embedding. In: Riloff, E., Chiang, D., Hockenmaier, J., and Tsujii, J.

(eds.) Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing. pp. 2001–2011. Association for Computational Linguistics, Brussels, Belgium (2018). https://doi.org/10.18653/v1/D18-1225.

23. Xu, C., Chen, Y.-Y., Nayyeri, M., Lehmann, J.: Temporal Knowledge Graph Completion using a Linear Temporal Regularizer and Multivector Embeddings. In: Toutanova, K., Rumshisky, A., Zettlemoyer, L., Hakkani-Tur, D., Beltagy, I., Bethard, S., Cotterell, R., Chakraborty, T., and Zhou, Y. (eds.) Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. pp. 2569–2578. Association for Computational Linguistics, Online (2021). https://doi.org/10.18653/v1/2021.naacl-main.202.
24. Xu, C., Nayyeri, M., Chen, Y.-Y., Lehmann, J.: Geometric algebra based embeddings for static and temporal knowledge graph completion. IEEE Trans Knowl Data Eng. 35, 4838–4851 (2022).
25. Zhang, S., Liang, X., Tang, H., Guan, Z.: Hybrid Interaction Temporal Knowledge Graph Embedding Based on Householder Transformations. In: Proceedings of the 31st ACM International Conference on Multimedia. pp. 8954–8962 (2023).
26. Han, Z., Chen, P., Ma, Y., Tresp, V.: DyERNIE: Dynamic Evolution of Riemannian Manifold Embeddings for Temporal Knowledge Graph Completion. In: Webber, B., Cohn, T., He, Y., and Liu, Y. (eds.) Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP). pp. 7301–7316. Association for Computational Linguistics, Online (2020). https://doi.org/10.18653/v1/2020.emnlp-main.593.
27. Jung, J., Jung, J., Kang, U.: Learning to walk across time for interpretable temporal knowledge graph completion. In: Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining. pp. 786–795 (2021).
28. Li, Y., Zhang, X., Zhang, B., Ren, H.: Each snapshot to each space: Space adaptation for temporal knowledge graph completion. In: International Semantic Web Conference. pp. 248–266 (2022).
29. Wang, J., Wang, B., Gao, J., Li, X., Hu, Y., Yin, B.: QDN: A Quadruplet Distributor Network for Temporal Knowledge Graph Completion. IEEE Trans Neural Netw Learn Syst. 1–13 (2023). https://doi.org/10.1109/TNNLS.2023.3274230.
30. Xu, C., Nayyeri, M., Alkhoury, F., Yazdi, H., Lehmann, J.: Temporal knowledge graph completion based on time series gaussian embedding. In: The Semantic Web–ISWC 2020: 19th International Semantic Web Conference, Athens, Greece, November 2–6, 2020, Proceedings, Part I 19. pp. 654–671 (2020).
31. Wu, J., Cao, M., Cheung, J.C.K., Hamilton, W.L.: TeMP: Temporal Message Passing for Temporal Knowledge Graph Completion. In: Webber, B., Cohn, T., He, Y., and Liu, Y. (eds.) Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP). pp. 5730–5746. Association for Computational Linguistics, Online (2020). https://doi.org/10.18653/v1/2020.emnlp-main.462.
32. Singh, I., Kaur, N., Gaur, G., others: NeuSTIP: A Novel Neuro-Symbolic Model for Link and Time Prediction in Temporal Knowledge Graphs. arXiv preprint arXiv:2305.11301. (2023).
33. Niu, G., Li, B.: Logic and commonsense-guided temporal knowledge graph completion. In: Proceedings of the AAAI Conference on Artificial Intelligence. pp. 4569–4577 (2023).
34. Tolstikhin, I.O., Houlsby, N., Kolesnikov, A., Beyer, L., Zhai, X., Unterthiner, T., Yung, J., Steiner, A., Keysers, D., Uszkoreit, J., others: Mlp-mixer: An all-mlp architecture for vision. Adv Neural Inf Process Syst. 34, 24261–24272 (2021).

35. Fang, Z., Lei, S.-L., Zhu, X., Yang, C., Zhang, S.-X., Yin, X.-C., Qin, J.: Transformer-based Reasoning for Learning Evolutionary Chain of Events on Temporal Knowledge Graph. arXiv preprint arXiv:2405.00352. (2024).
36. Boschee, E., Lautenschlager, J., O'Brien, S., Shellman, S., Starz, J., Ward, M.: ICEWS Coded Event Data, https://doi.org/10.7910/DVN/28075, (2015). https://doi.org/10.7910/DVN/28075.
37. Mahdisoltani, F., Biega, J., Suchanek, F.M.: YAGO3: A Knowledge Base from Multilingual Wikipedias. In: Conference on Innovative Data Systems Research (2014).
38. Erxleben, F., Günther, M., Krötzsch, M., Mendez, J., Vrandečić, D.: Introducing wikidata to the linked data web. In: The Semantic Web–ISWC 2014: 13th International Semantic Web Conference, Riva del Garda, Italy, October 19-23, 2014. Proceedings, Part I 13. pp. 50–65 (2014).