

# T3A-LLM: A Two-Stage Temporal Knowledge Graph Alignment Method Enhanced by LLM

Tao Wang<sup>1</sup>, Runhao Zhao<sup>1</sup>, Jiuyang Tang<sup>1</sup>(✉), and Xiaolu Liu<sup>2</sup>

<sup>1</sup> Laboratory for Big Data and Decision, National University of Defense Technology, China

3359206357@qq.com, {runhaozhao, jiuyang\_tang}@nudt.edu.cn

<sup>2</sup> College of System Engineering, National University of Defense Technology, China  
lx1\_sunny@nudt.edu.cn

**Abstract.** Temporal knowledge graphs (TKGs) extend traditional knowledge graphs by incorporating temporal information to represent time-sensitive facts as quadruples (subject, relation, object, timestamp), enabling the modeling of dynamic real-world relationships that evolve over time. However, TKGs from different real-world sources are often incomplete and contain complementary information. Therefore, Temporal Entity Alignment (TEA) techniques are needed to integrate knowledge from multiple TKGs by identifying equivalent entities across different temporal knowledge graphs, thereby supporting the consolidation of knowledge from multiple sources. Although there has been extensive research on temporal entity alignment, existing approaches suffer from significant limitations: some methods fail to fully exploit rich semantic information and contextual background, while others that employ large language models incur prohibitively high computational costs. To address these challenges, we propose T3A-LLM, a two-stage triple-information alignment with large language model (LLM), a novel two-stage framework for TKG alignment that efficiently fuses structural, temporal, and semantic information. The first stage uses dual-feature encoding (relation, time) and graph matching for preliminary alignment to a top-n candidate set, reducing the search space and addressing LLM computational overhead. The second stage applies an LLM-Score mechanism for fine-grained semantic reasoning on the candidates, specifically designed to capture deep semantic relationships that traditional structural methods cannot handle. The final scores are obtained by fusing graph-based and LLM-based similarities. Experiments show that T3A-LLM significantly outperforms baselines, with ablation studies confirming the necessity of each component.

**Keywords:** Temporal Knowledge Graphs · Temporal Entity Alignment · LLM · Two-Stage Framework

## 1 Introduction

Temporal knowledge graphs (TKGs) are fundamental extensions of traditional knowledge graphs that incorporate temporal information to model dynamic real-world facts, powering applications in semantic search, recommendation systems,

and artificial intelligence [4–7]. TKGs represent knowledge as quadruples (subject, relation, object, timestamp), enabling the capture of time-sensitive relationships and evolving entity attributes that static knowledge graphs cannot adequately represent.

However, individual TKGs constructed from different sources are often incomplete and contain complementary information. This necessitates Temporal Entity Alignment (TEA) [6–10] - identifying equivalent entities across different temporal graphs while considering their temporal contexts and evolution patterns to create more comprehensive knowledge repositories [36, 37]. This becomes particularly complex with TKGs, as the temporal dimension introduces new challenges where methods must consider time-sensitive relationships and evolving entity attributes.

Existing temporal entity alignment research has explored various approaches to address these challenges. Methods like TEA-GNN integrate temporal information through temporal relational attention mechanisms [13], while TAlign leverages neighborhood distance awareness for temporal entity alignment [10, 12]. Other approaches include embedding-based methods for cross-lingual temporal knowledge graphs [5, 11], relation-enhanced models that utilize dual-encoder architectures [4, 6], and active learning strategies for improved alignment efficiency [9]. However, these approaches face common limitations: while they incorporate semantic information from entity attributes and contextual descriptions, the utilization of such semantic information remains insufficient, and they often struggle to effectively integrate multi-dimensional information sources for comprehensive alignment decisions.

The importance of semantic information in entity alignment has become increasingly recognized, as entity names, descriptions, and attributes provide crucial disambiguation signals that structural patterns alone cannot capture. Recent advances in LLMs have demonstrated remarkable capabilities in semantic understanding and reasoning, offering new opportunities for entity alignment tasks [38–40]. Recent studies have leveraged LLM for entity alignment due to their semantic understanding capabilities [2, 3, 20]. LLMEA integrates structural knowledge from KGs with semantic knowledge from LLMs through relation-aware graph attention networks and multi-choice question answering [3]. ChatEA introduces a KG-Code translation module that converts graph structures into LLM-comprehensible formats through a two-stage strategy [2]. However, these existing approaches utilizing LLMs for alignment tasks remain in preliminary stages regarding the collaborative utilization of semantic, structural, and temporal information. Therefore, there is significant potential to develop more sophisticated frameworks that can effectively integrate these multi-dimensional information sources with LLM capabilities.

To address these limitations, we propose two-stage triple-information alignment with large language model (T3A-LLM), a novel two-stage framework that strategically combines the computational efficiency of GNNs with the semantic reasoning power of LLMs specifically designed for temporal knowledge graph alignment. Our approach employs a dual-feature encoding mechanism that sepa-

rately captures temporal dynamics and relational structures, followed by a graph matching stage that generates a probability matrix to identify top-n candidate pairs. Subsequently, an LLM-based scoring mechanism performs fine-grained semantic reasoning on these candidates, with final alignment decisions made through adaptive fusion of structural-temporal and semantic similarity scores.

The key contributions of this paper are as follows:

- **Two-Stage Alignment Framework:** We propose a computationally efficient approach that combines GNNs for preliminary alignment and LLMs for semantic refinement, balancing performance and scalability specifically for temporal knowledge graphs.
- **Multi-Dimensional Information Fusion:** The framework systematically integrates three critical types of information-temporal dynamics, structural relationships, and deep semantic understanding obtained through LLM-based reasoning and inference of original entity, attribute, and relation information-through dual-feature encoding and adaptive fusion techniques to enhance alignment accuracy in temporal settings.
- **Empirical Validation:** We conduct extensive experiments on both temporal-dominant and semantic-dominant datasets, demonstrating that T3A-LLM outperforms state-of-the-art baselines and existing LLM-based approaches.

The remainder of this paper is organized as follows: Section 2 reviews related work. Section 3 details the proposed T3A-LLM framework. Section 4 presents experimental setups and results analysis. Finally, section 5 discusses limitations and future work.

## 2 Related Work

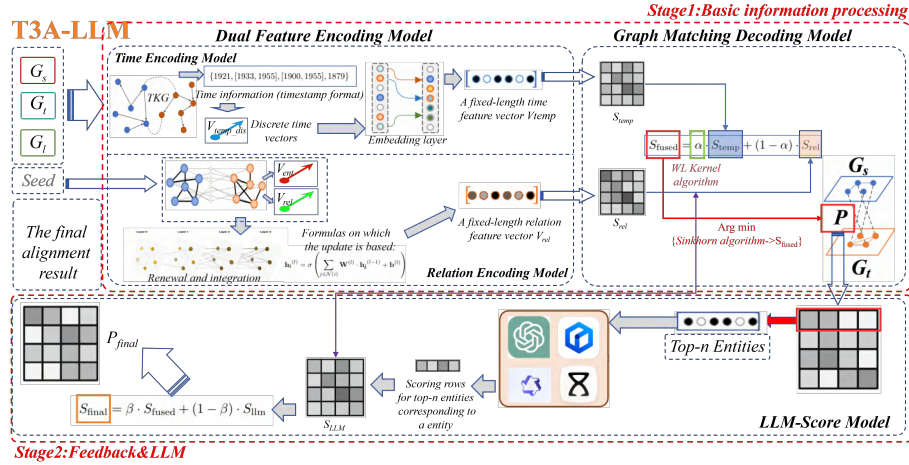
**Temporal Entity Alignment.** Recent temporal entity alignment research has explored diverse methodological approaches to address the challenges of aligning entities across temporal dimensions [36, 37]. Graph neural network-based methods such as TEA-GNN [13] combine GNN architectures with time-aware attention mechanisms to aggregate neighbor information and generate temporal-aware entity representations. Attention-based approaches like TREA integrate temporal and relational information through weighted neighbor aggregation [21], while methods such as TEA employ time-aware attention layers to automatically weight temporal and relational components [22]. Self-supervised learning techniques, exemplified by TSEA, enhance entity alignment through temporal knowledge completion without requiring manually annotated seeds [20]. Advanced modeling approaches have incorporated relation association and probabilistic calibration mechanisms [19], with some methods addressing uncertainty through fuzzy semantic modeling and global structure learning. However, these approaches primarily focus on structural and temporal embeddings while inadequately exploiting rich semantic information from entity attributes and contextual descriptions, and they lack unified frameworks for adaptively integrating and weighting multi-dimensional information sources (temporal, relational, and semantic) based on dataset-specific characteristics.

**LLM-Based Entity Alignment.** Recent advances in LLMs have opened new avenues for entity alignment through their semantic understanding and reasoning capabilities. LLM4EA combines active learning with label optimization, utilizing active selection strategies and label optimizers to refine LLM-generated pseudo-labels [2]. ChatEA transforms entity alignment into multi-round reasoning problems, combining LLM-generated labels with graph convolutional networks through joint optimization [3]. LLMEA integrates structural and semantic information by first generating candidate sets using graph structure, then employing LLMs for multi-round reasoning to filter candidates [20]. However, these LLM-based approaches face critical limitations in practical deployment: they rely heavily on intensive LLM inference without leveraging basic structural or temporal information for preliminary screening, resulting in prohibitive computational costs and processing time, particularly when temporal dimensions significantly expand the search space and computational complexity.

### 3 Method

#### 3.1 Model Overview

We propose a two-stage entity alignment framework that combines the advantages of GNNs and LLMs to optimize the entity alignment process using structured temporal, relational, and semantic information [1]. The framework consists of key modules, including a dual-feature encoding module (relation, time), a graph matching decoding module, and an LLM-Score module.



**Fig. 1.** T3A-LLM Framework

To reduce the computational burden of the LLM, we adopt a staged approach. Initially, the dual-feature encoding and graph matching modules perform a preliminary alignment based on structural and temporal information to compute a probability matrix  $P$ . Inspired by the work of Zhao et al. [32, 33], we select

a top-n candidate set for each entity based on the probability matrix  $P$ . Then, the LLM-Score module is applied to these candidates for fine-grained semantic reasoning and scoring based on the LLM’s semantic understanding capabilities. The final alignment decision integrates the scores from both stages. This approach enhances alignment accuracy by leveraging the LLM’s semantic power efficiently [34].

Formally, our framework, termed T3A-LLM (two-stage triple-information alignment with large language model), strategically combines the computational efficiency of GNNs with the semantic reasoning power of LLMs for temporal knowledge graph alignment. It employs a dual-feature encoding mechanism to capture temporal dynamics and relational structures, followed by an LLM-based scoring mechanism that performs fine-grained semantic reasoning with adaptive fusion of structural-temporal and semantic similarity scores.

Figure 1 illustrates the overall architecture and workflow of our T3A-LLM framework, showing the interaction between the dual-feature encoding stage and the LLM-based semantic reasoning stage.

### 3.2 Dual Feature Encoding Module

**Time Information Encoding Stage.** The goal of the Time Encoding Model is to capture the temporal features of entities through time-related information [1]. Temporal information is a natural label for entity alignment, but many previous studies have overlooked the value of time information [6, 19]. Different graphs may capture different events or attributes at different timestamps. The Time Encoding Model learns time-related features and embeds time information into the entity representation, thus providing richer contextual information.

*Input Time Information.* Time information typically exists in the form of timestamps or time intervals. For example, the entity "U.S. President" may have the following timestamps:

$$\{1789, [1861, 1865], [1933, 1945], 2021\} \quad (1)$$

Each timestamp represents the occurrence time of an event, while time intervals represent the duration of an event.

*Time Feature Vector Representation.* The Time Encoding Model converts this time information into a fixed-length time feature vector  $\mathbf{V}_{\text{temp}}$ . To achieve this, time information is first discretized (e.g., to the year level) and mapped to a dense vector space via an embedding layer. To capture the sequential nature and evolution patterns of an entity’s temporal context, we employ a Bidirectional Long Short-Term Memory network (Bi-LSTM) [1].

For a given entity, its sequence of timestamp embeddings is fed into the Bi-LSTM, which processes the sequence in both forward and backward directions. This bidirectional approach allows the model to capture both historical dependencies and future context. The final hidden states from both directions

are concatenated to form the final time feature vector  $\mathbf{V}_{\text{temp}}$ , which comprehensively represents the entity's temporal signature. This is crucial for distinguishing entities that are structurally similar but active in different time periods.

Consider the timestamps for the entity "U.S. President" in both graphs  $TKG_A$  and  $TKG_B$ . If the time patterns (e.g., similarities in time intervals or timestamps) between the two are highly similar, their time feature vectors will also be close to each other.

The final output of the Time Encoding Model is the time feature vector  $\mathbf{V}_{\text{temp}}$ , which serves as input to the subsequent graph matching decoding module.

**Relation Information Encoding Stage.** The goal of the Relation Encoding Model is to capture the structural relationship information between entities [1]. Each entity in the knowledge graph is connected to other entities through various relationships (e.g., "same type", "belongs to", "located in"). The Relation Encoding Model uses GNNs to capture these connection patterns, thereby generating the relational feature vectors of entities [10, 12].

*Input and Graph Structure Representation.* We consider the graph structure of entities and relationships. Suppose we have a knowledge graph containing several entities  $E = \{e_1, e_2, \dots, e_n\}$  and relationships  $R = \{r_1, r_2, \dots, r_m\}$  between these entities, where each relationship  $r_i$  connects two entities  $e_i$  and  $e_j$ . These entities are connected through relationships  $r$  to form an undirected or directed graph. For example, in a graph containing academic papers, the entities could be "papers" and "authors", with relationships such as "author writes paper" or "paper cites other paper".

*Relational Feature Vector Representation.* We aim to convert each entity  $e_i$  and relationship  $r_j$  into a vector that represents the relationship information of the entity. First, each entity  $e_i$  and relationship  $r_j$  are mapped to an initial embedding space, represented by the embedding vectors  $\mathbf{e}_i$  and  $\mathbf{r}_j$ . These embedding vectors are then updated and fused using GNNs. Note that in our design, the Relation Encoding Model focuses purely on the structural and relational patterns, while temporal dynamics are captured by the dedicated Time Encoding Model. These two distinct feature aspects,  $\mathbf{V}_{\text{rel}}$  and  $\mathbf{V}_{\text{temp}}$ , are then fused in the subsequent decoding stage. This modular separation allows each module to specialize in capturing its respective information type. Let  $\mathbf{h}_i^{(l)}$  represent the embedding of entity  $e_i$  at layer  $l$  of the GNN. The update rule is as follows:

$$\mathbf{h}_i^{(l)} = \sigma \left( \sum_{j \in \mathcal{N}(i)} \mathbf{W}^{(l)} \cdot \mathbf{h}_j^{(l-1)} + \mathbf{b}^{(l)} \right) \quad (2)$$

where  $\mathcal{N}(i)$  is the set of neighboring entities of  $e_i$ , and  $\mathbf{W}^{(l)}$  and  $\mathbf{b}^{(l)}$  are the weight matrix and bias term at layer  $l$ , respectively.  $\sigma(\cdot)$  is the nonlinear activation function. The updated embedding vector  $\mathbf{h}_i^{(L)}$  is the relational feature vector of entity  $e_i$  in the graph, which is used in subsequent matching processes.

*Example* For instance, in  $TKG_A$ , the entity "U.S. President" is connected to the entity "Nobel Prize" through the relation "received\_\_award". After processing by the Relation Encoding Model, the relational feature vectors of "U.S. President" and "Nobel Prize" will be updated. If these entities also have similar relations in other contexts, their feature vectors will tend to converge in high-dimensional space.

These feature vectors are used as input for the subsequent graph matching decoding module. Through this dual encoding approach, we can fully utilize both the structural and temporal information of entities, providing richer feature representations for cross-graph entity alignment [16, 17].

### 3.3 Graph Matching Decoding Module

Following DualMatch [1], we treat the entity alignment problem as a weighted graph matching problem. The Graph Matching Decoder module combines the relational and temporal information obtained from the dual-feature encoding module to determine which entities should be aligned by calculating the similarity and weighted fusion of the two major indices [1, 20].

**Similarity Calculation.** First, we calculate the similarity between all entities in  $TKG_A$  and  $TKG_B$ . We calculate the similarity based on the relational feature vector  $\mathbf{V}_{\text{rel}}$  and the temporal feature vector  $\mathbf{V}_{\text{temp}}$ .

*Relational Similarity.* Based on the  $\mathbf{V}_{\text{rel}}$  vector, we obtain a relational similarity matrix  $S_{\text{rel}}$ .

*Temporal Similarity.* Based on the  $\mathbf{V}_{\text{temp}}$  vector, we obtain a temporal similarity matrix  $S_{\text{temp}}$ .

The specific calculation method is as follows. For the relational feature vectors  $\mathbf{V}_{\text{rel}}(e_A)$  and  $\mathbf{V}_{\text{rel}}(e_B)$ , we use common similarity measures (such as cosine similarity or Euclidean distance) to compute the similarity. We use cosine similarity to measure the similarity between two vectors, defined by the following formula:

$$S_{\text{rel}}(e_A, e_B) = \frac{\mathbf{V}_{\text{rel}}(e_A) \cdot \mathbf{V}_{\text{rel}}(e_B)}{\|\mathbf{V}_{\text{rel}}(e_A)\| \|\mathbf{V}_{\text{rel}}(e_B)\|} \quad (3)$$

where  $\mathbf{V}_{\text{rel}}(e_A)$  and  $\mathbf{V}_{\text{rel}}(e_B)$  are the relational feature vectors of entities  $e_A$  and  $e_B$ , and  $\|\mathbf{V}_{\text{rel}}(e_A)\|$  and  $\|\mathbf{V}_{\text{rel}}(e_B)\|$  are their Euclidean norms.

**Weighted Fusion.** Next, we perform weighted fusion of the two similarity matrices  $S_{\text{rel}}$  and  $S_{\text{temp}}$ . This fusion process is not a simple summation, but rather a weighted process that reflects the overall similarity between the two graphs in terms of structure and time. We use the Weisfeiler-Lehman (WL) graph kernel method to help determine the weights for this fusion [21]. Specifically, the WL graph kernel calculates the similarity between each pair of graphs and determines how to weight the temporal feature matrix  $S_{\text{temp}}$  and the relational feature matrix  $S_{\text{rel}}$ , thus generating the final weighted matrix  $S_{\text{fused}}$ .

The weighted fusion formula is as follows:

$$S_{\text{fused}} = \alpha \cdot S_{\text{temp}} + (1 - \alpha) \cdot S_{\text{rel}} \quad (4)$$

where the weight coefficient  $\alpha$  is automatically determined based on the WL method, reflecting the overall characteristics of the graph (such as time pattern differences or relational structure similarity).

**Graph Matching Optimization via Sinkhorn Algorithm.** After the weighted fusion, we can further solve the graph matching problem using the similarity matrix  $S_{\text{fused}}$ . We adopt optimization methods such as the Sinkhorn algorithm [23] to solve the weighted graph matching problem, thus generating an alignment probability matrix  $P$ , where  $P[i, j]$  represents the probability or confidence that entity  $e_i$  in  $TKG_A$  and entity  $e_j$  in  $TKG_B$  correspond to the same entity. This matrix  $P$  provides the basis for selecting the top- $n$  candidates for the LLM stage. Specifically, for each entity  $e_i \in TKG_A$ , we sort the probabilities in the corresponding row  $P[i, :]$  in descending order. The top  $n$  entities from  $TKG_B$  with the highest probabilities are then selected to form the candidate set for  $e_i$ . This screening step effectively narrows the search space from the entire set of entities in  $TKG_B$  down to a manageable size  $n$ , significantly improving the efficiency of the subsequent LLM-based refinement.


### 3.4 LLM-Score Module

To further enhance semantic understanding and improve alignment accuracy, we introduce a multiple-choice scoring mechanism that combines an LLM to score and infer the semantic relationships of candidate entities. We first obtain a top- $n$  candidate set through the feature extraction and initial screening steps described earlier (Section 3.3), where the value of  $n$  will be adjusted in experiments to observe its specific effect. Next, for these candidate entities, we prepare their relevant information (e.g., entity name, description, key neighbors, temporal context) and input it into the LLM for reasoning and scoring. By leveraging the LLM’s reasoning capabilities, we refine the entity alignment [34]. Finally, we perform a weighted fusion of the similarity scores obtained from the knowledge graph structure/time ( $S_{\text{fused}}$ ) and the LLM ( $S_{\text{llm}}$ ) to generate the final alignment score, and further compute the final probability matrix.

**LLM Multiple-Choice Scoring.** During the LLM’s multi-round reasoning process, the LLM will infer and make decisions based on the provided entity description, relationships, and contextual information [38–40]. Each reasoning round will assess the quality of the candidate entities and make alignment decisions. Given that LLMs often produce extreme scores (e.g., 0 or 1) in practical applications, we designed a multiple-choice mechanism to refine the scoring segments. The specific prompt template design is shown in Fig. 2.

By using this multiple-choice scoring approach, the LLM can more accurately judge the semantic matching between entities, while avoiding extreme scores (such as 0 or 1).





**Task:** Determine whether the following two entities match and give the most appropriate score based on their semantic relationship.

**-Entity 1:** The American Revolutionary War

**-Entity 2:** George Washington

**Background information:**

**Entity 1:** The American Revolutionary War

**Description:** A war in the 18th century in which the thirteen North American colonies fought to break free from British colonial rule.

**Relations:** occurred\_in -> North America, fought\_against -> British colonial rule


**Entity 2:** George Washington

**Description:** The first President of the United States who served as Commander-in-Chief of the Continental Army during the American Revolutionary War.

**Relations:** position\_held -> President of the United States (first), military\_rank -> Commander-in-Chief (Continental Army), participant\_in -> American Revolutionary War, nationality -> American

Based on the above background information, **select one of the following options** to represent the semantic matching degree between Entity 1 and Entity 2:

**A. Perfect match (1.0) B. High match (0.8) C. Moderate match (0.5) D. Low match (0.3) E. No match (0)**



**B. High match (0.8).** George Washington played a key leadership role in the American Revolutionary War. However, the war was a complex historical event involving many other factors and figures. So, it's not a perfect match but a high match.

**Fig. 2.** Sample Prompt Framework Design

**Similarity Fusion.** After the LLM reasoning process generates scores for each entity pair within the top-n candidate set, we fuse the graph-based similarity matrix  $S_{\text{fused}}$  calculated by the Graph Matching Decoder (for the specific pair) with the semantic similarity scores  $S_{\text{llm}}$  generated by the LLM, obtaining the final alignment score:

$$S_{\text{final}} = \beta \cdot S_{\text{fused}} + (1 - \beta) \cdot S_{\text{llm}} \quad (5)$$

where  $\beta$  is a weight coefficient determined through experiments, representing the combined weight of relational, temporal, and deep semantic understanding obtained through LLM-based reasoning and inference. Gradient experiments will be set up in the experiment to verify the effect of  $\beta$  on the alignment performance. This fusion is applied only to the top-n candidates considered by the LLM.

**Post-Processing After Comprehensive Scoring.** Based on the final comprehensive scores  $S_{\text{final}}$  for the top-n candidates (and potentially considering the original  $S_{\text{fused}}$  scores for others), we calculate the final probability matrix  $P$ . Alignment decisions are then made based on this final matrix, typically by selecting the highest probability match for each entity.

## 4 Experiments

### 4.1 Dataset Description

Our study utilizes two complementary temporal knowledge graph datasets that represent distinct alignment challenges [18].

**DICEWS:** The DICEWS dataset is derived from the Integrated Crisis Early Warning System (ICEWS) and records temporal political and military events from 2005 to 2015. It contains approximately 9,500 entities, 246 relations, and 300,000 quadruples with high structural symmetry. This creates a scenario where temporal information becomes the primary discriminative signal for entity alignment [9, 14].

**WY50K:** The WY50K dataset is constructed from Wikidata and YAGO, containing approximately 50,000 entities and over 200,000 quadruples covering general-knowledge entities. Due to its cross-knowledge-base nature, WY50K exhibits significant structural heterogeneity with lower structural overlap, introducing substantial semantic disambiguation challenges where entities require sophisticated semantic understanding for accurate alignment. This makes deep semantic understanding obtained through LLM-based reasoning and inference the dominant signal for successful alignment [11].

## 4.2 Evaluation Metrics

We assess the entity alignment performance using three standard metrics, which together provide a comprehensive view of the model’s accuracy, recall, and ranking robustness. The results of each indicator in the following experiment are the averages of 10 independent runs.

**Hits@1 (top-1 Hit Rate):** Measures the percentage of correct entities ranked as the top candidate. It serves as a strict metric for alignment precision.

**Hits@10 (top-10 Hit Rate):** Measures the percentage of correct entities ranked as the top candidate. It serves as a strict metric for alignment precision.

**Mean Reciprocal Rank (MRR):** Measures the percentage of correct entities ranked as the top candidate. It serves as a strict metric for alignment precision.

## 4.3 Comprehensive Performance Evaluation and Module Impact Analysis

**Comprehensive Evaluation of Entity Alignment Performance.** As shown in Table 1, our T3A-LLM framework consistently achieves state-of-the-art performance across all datasets and metrics. Notably, compared to the strongest time-aware baseline DualMatch, T3A-LLM secures a significant 2.0 percentage point gain in Hit@1 on the temporal-dominant DICEWS-1K dataset and a 1.3 point gain on the semantic-dominant WY50K-5K dataset. These results, confirmed to be statistically significant ( $p < 0.01$ ), underscore the framework’s superior ability to model temporal dynamics and resolve semantic ambiguities.

**Baselines.** We use multiple state-of-the-art EA methods as baselines following the settings in previous studies [1]. The baselines include time-unaware methods: *JAPE* [24], *AlignE* [25], *GCN-Align* [26], *MuGNN* [27], *MRAEA* [28], and *RREA* [29]; and time-aware methods: *TEA-GNN* [30], *TREA* [31], and *DualMatch* [1].

**Ablation Study.** The ablation studies (Table 2) validate the necessity of each component by revealing their dataset-specific contributions. On the temporal-dominant DICEWS-1K dataset, removing the time encoder (–Time) causes the most significant performance drop in Hit@1 (14.8%), whereas on the semantic-dominant WY50K-5K, removing the relation encoder (–Relation) is most detrimental (12.8% drop). This confirms that our model effectively adapts to the primary information signal of each dataset. The LLM module (–LLM) provides

a consistent performance boost, proving its crucial role in fine-grained semantic refinement.

**Table 1.** Entity Alignment Performance Comparison

Methods	DICEWS-1K			DICEWS-200			WY50K-5K			WY50K-1K		
	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10	MRR
<b>Time-Unaware Methods</b>												
JAPE	13.9	28.6	19.0	9.1	20.3	13.1	26.5	47.8	33.8	9.8	25.7	15.1
GCN-Align	19.7	45.1	28.3	15.8	35.1	22.4	50.3	70.2	57.3	20.9	38.6	27.2
AlignE	49.2	73.8	58.1	21.1	44.2	29.5	74.3	87.1	79.2	55.1	70.2	60.8
RREA	71.0	87.1	77.1	64.5	81.2	70.9	81.6	92.5	85.8	68.3	84.7	74.2
MRREA	66.8	85.7	73.6	46.2	71.9	55.5	79.3	90.1	83.9	61.1	78.8	67.6
MuGNN	51.2	78.1	60.7	35.9	57.1	40.5	75.0	87.8	79.8	57.3	72.1	62.1
<b>T3A-LLM</b>	<b>72.8</b>	<b>88.5</b>	<b>78.5</b>	<b>65.9</b>	<b>83.1</b>	<b>72.1</b>	<b>88.2</b>	<b>95.1</b>	<b>91.0</b>	<b>74.2</b>	<b>87.6</b>	<b>82.5</b>
<b>Time-Aware Methods</b>												
TREA	90.2	95.4	92.2	89.8	94.7	91.7	92.8	97.6	94.7	82.7	92.3	87.3
TEA-GNN	87.3	93.5	90.1	86.1	92.8	89.1	86.7	94.9	89.8	71.0	85.8	76.3
DualMatch	95.3	97.3	96.1	95.0	97.2	96.0	98.1	99.6	98.6	94.0	97.8	95.5
<b>T3A-LLM</b>	<b>97.3</b>	<b>98.6</b>	<b>97.8</b>	<b>96.8</b>	<b>98.5</b>	<b>97.6</b>	<b>99.4</b>	<b>100.0</b>	<b>99.6</b>	<b>96.2</b>	<b>99.1</b>	<b>97.5</b>

\* DICEWS-1K/200: DICEWS dataset with 1000/200 seed alignment pairs

\* WY50K-5K/1K: WY50K dataset with 5000/1000 seed alignment pairs

**Table 2.** Ablation Study of T3A-LLM(Performance Metrics in %)

Model Variant	DICEWS-1K			DICEWS-200			WY50K-5K			WY50K-1K		
	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10	MRR
<b>Original</b>	97.3	98.6	97.8	96.8	98.5	97.6	99.4	100.0	99.6	96.2	99.1	97.5
– <b>Relation</b>	90.8	95.9	93.1	89.2	95.4	92.1	86.6	94.2	89.9	84.1	93.1	88.3
– <b>Time</b>	82.5	92.1	86.6	81.8	91.8	86.2	96.2	99.3	97.7	93.1	97.6	95.2
– <b>LLM</b>	95.3	97.3	96.1	95.0	97.2	96.0	98.1	99.6	98.6	94.0	97.8	95.5

"–" indicates the removal of the corresponding module.

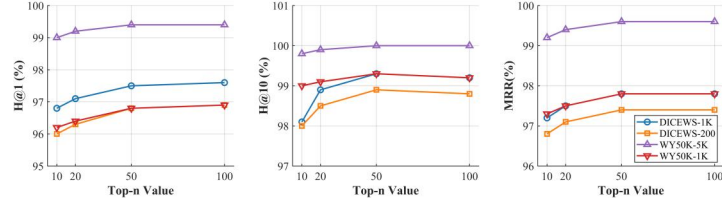
#### 4.4 Sensitivity Analysis

**Gradient Analysis of top-n.** We analyzed the impact of the candidate set size  $n$  (Table 3 and Fig. 3). Performance improves as  $n$  increases, with gains plateauing around  $n=50$ . This value strikes an optimal balance between maximizing recall for the LLM stage and maintaining computational efficiency, and was thus adopted for our experiments.

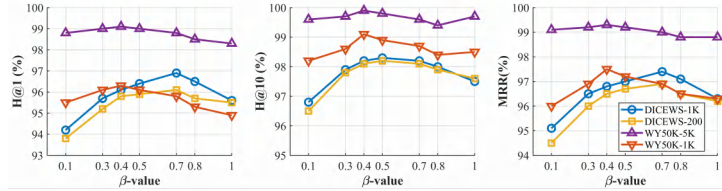
**Gradient Analysis of  $\beta$  Value in  $S_{\text{final}}$ .** The analysis of the fusion weight  $\beta$  (Table 4 and Fig. 4) reveals dataset-dependent optimal values. The temporal-dominant DICEWS datasets favor a higher  $\beta$  (0.7), giving more weight to structural/temporal features because of their high structural overlap and strong temporal signals. In contrast, the semantic-dominant WY50K datasets perform best with a lower  $\beta$  (0.4), highlighting the greater importance of LLM-based semantic scoring for resolving cross-knowledge-base heterogeneity and semantic ambiguities [15]. This adaptability validates our fusion strategy and demonstrates the framework’s ability to automatically adjust to different information dominance patterns.

**Table 3.** Impact of top-n Candidate Pool Size on Entity Alignment

top-n	DICEWS-1K			DICEWS-200			WY50K-5K			WY50K-1K		
	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10	MRR
n=10	97.1	98.4	97.6	96.5	98.3	97.4	99.2	99.9	99.5	96.0	98.9	97.3
n=20	97.4	98.7	97.9	96.8	98.6	97.7	99.5	100.0	99.7	96.3	99.2	97.6
n=50	<b>97.3</b>	<b>98.6</b>	<b>97.8</b>	<b>96.8</b>	<b>98.5</b>	<b>97.6</b>	<b>99.4</b>	<b>100.0</b>	<b>99.6</b>	<b>96.2</b>	<b>99.1</b>	<b>97.5</b>
n=100	97.2	98.5	97.7	96.7	98.4	97.5	99.3	100.0	99.5	96.1	99.0	97.4

**Fig. 3.** Gradient Analysis of top-n**Table 4.** Effect of LLM Fusion Weight  $\beta$  on Hybrid Scoring

$\beta$ Value	DICEWS-1K			DICEWS-200			WY50K-5K			WY50K-1K		
	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10	MRR
0.1	94.8	97.2	95.8	94.3	96.9	95.4	99.1	99.8	99.4	95.8	98.6	96.8
0.3	96.2	98.3	97.1	95.7	98.1	96.7	99.3	99.9	99.5	96.4	99.0	97.3
0.4	96.6	98.5	97.4	96.2	98.4	97.1	<b>99.4</b>	<b>100.0</b>	<b>99.6</b>	<b>96.2</b>	<b>99.1</b>	<b>97.5</b>
0.5	96.9	98.6	97.6	96.5	98.5	97.3	99.2	99.9	99.4	95.9	98.8	97.2
0.7	<b>97.3</b>	<b>98.6</b>	<b>97.8</b>	<b>96.8</b>	<b>98.5</b>	<b>97.6</b>	99.0	99.7	99.2	95.6	98.5	96.9
0.8	96.8	98.4	97.5	96.3	98.2	97.2	98.7	99.5	99.0	95.2	98.3	96.6
1.0	95.3	97.3	96.1	95.0	97.2	96.0	98.1	99.6	98.6	94.0	97.8	95.5

**Fig. 4.** Gradient Analysis of  $\beta$  Value in  $S_{final}$

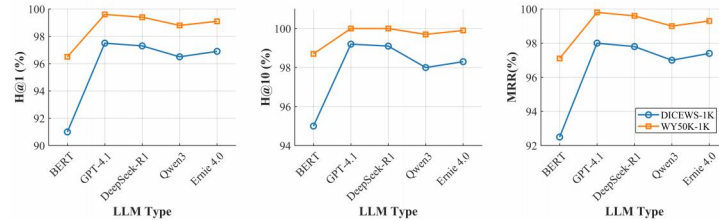
#### 4.5 Comparison of Performance Across Different LLM Architectures

To assess the impact of the LLM choice, we compared various architectures (Table 5 and Fig. 5). State-of-the-art models like GPT-4o and DeepSeek-R1 significantly outperform smaller models, achieving the highest scores on both temporal and semantic-dominant datasets. The results show a strong correlation between the LLM’s reasoning capability and the final alignment accuracy, confirming the value of using powerful models for the semantic refinement stage.

**Table 5.** Performance Comparison Across LLM Architectures

LLM Type	DICEWS			WY50K			Param. (B)
	H@1	H@10	MRR	H@1	H@10	MRR	
BERT	91.2	95.3	92.8	96.8	98.9	97.4	0.34
<b>GPT-4o</b>	97.3	98.6	97.8	99.4	100.0	99.6	~180
<b>DeepSeek-R1</b>	97.1	98.4	97.6	99.2	99.9	99.4	~67
Qwen3 (Alibaba)	96.8	98.2	97.3	99.0	99.8	99.2	110
Ernie 4.0 Turbo	97.0	98.3	97.5	99.1	99.9	99.3	~130

Note: DeepSeek-R1 refers to DeepSeek-R1-0528. Parameter counts are approximate. Best performers highlighted in bold.



**Fig. 5.** Comparison of Performance Across Different LLM Architectures

#### 4.6 Case Study

In the case study of the DICEWS dataset, regarding the alignment task for the entity "United\_States\_President" (USP-1024), existing methods such as Dual-Match [1], which rely solely on features like structure and time, incorrectly align it with "President\_of\_Argentina" (PA-789) (with a similarity score of 0.87). In contrast, the T3A-LLM framework adopts a two-stage processing approach: first, it generates a candidate set (including "President\_of\_Argentina", "POTUS", and "UN\_Secretary\_General") based on spatiotemporal embedding results; then, it leverages LLM for semantic reasoning to conduct a comprehensive evaluation of the candidate entities’ names (e.g., identifying "POTUS" as an abbreviation for "President of the United States"), descriptions (e.g., the differences

between the U.S. Electoral College system and Argentina’s universal suffrage system), as well as structural and temporal features. Eventually, it correctly matches "United\_States\_President" to "POTUS" (POTUS-456) with a 95% alignment probability. This verifies the crucial role of LLM’s semantic reasoning in breaking through the trap of structural similarity in traditional embedding methods, resolving semantic ambiguities, and integrating multi-dimensional information, thereby significantly improving the accuracy of entity alignment.

## 5 Conclusion and Future Discussion

Traditional TKG alignment methods struggle with semantic understanding and computational efficiency when dealing with temporal information and cross-domain heterogeneity. Current LLM-based approaches, while offering superior semantic reasoning capabilities, suffer from high computational costs and scalability limitations. To address these challenges, we propose T3A-LLM, a novel two-stage framework that synergistically combines GNNs and LLMs to achieve both computational efficiency and semantic precision. Our approach demonstrates significant performance improvements across temporal-dominant and semantic-dominant datasets, with experimental results showing consistent state-of-the-art performance and dataset-adaptive information fusion capabilities. This advancement is crucial for real-world applications requiring accurate entity alignment across diverse knowledge sources, such as knowledge integration systems and cross-domain information retrieval. However, our approach assumes relatively stable temporal patterns and may not handle rapidly evolving temporal dynamics. Future research directions include developing more sophisticated adaptive weighting mechanisms for dynamic temporal patterns and extending the framework to handle real-time knowledge graph evolution scenarios. Additionally, addressing scalability challenges in massive-scale temporal knowledge graphs with millions of entities and distributed processing architectures remains an important avenue for practical deployment in enterprise-level applications.

**Acknowledgments.** This work was partially supported by the National Natural Science Foundation of China (NSFC) under grant No. 62302513.

## References

1. Liu, X., Wu, J., Li, T., et al.: Unsupervised Entity Alignment for Temporal Knowledge Graphs. (2023)
2. Yang, L., Chen, H., Wang, X., et al.: Two Heads Are Better Than One: Integrating Knowledge from Knowledge Graphs and Large Language Models for Entity Alignment. (2024)
3. Jiang, X., Shen, Y., Shi, Z., et al.: Unlocking the Power of Large Language Models for Entity Alignment. (2024)
4. Jia, L., Song, D., Wang, H., et al.: Entity alignment for temporal knowledge graphs via adaptive graph networks. *Knowledge-Based Systems* **274**, 110631 (2023)

5. Bai, L., Li, N., Li, G., Zhang, Z., Zhu, L.: Embedding-Based Entity Alignment of Cross-Lingual Temporal Knowledge Graphs. *Neural Networks* **172**, 106143 (2024). <https://doi.org/10.1016/j.neunet.2024.106143>
6. Wang, Z., You, X., Lv, X.: A relation enhanced model for temporal knowledge graph alignment. *The Journal of Supercomputing* **80**(5), 1–23 (2024)
7. Fu, T., Zhou, et al.: Temporal knowledge completion enhanced self-supervised entity alignment. *Journal of Intelligent Information Systems* **63**(1), 43–62 (2025)
8. Song, X., Bai, L., Liu, R., Zhang, H.: Temporal Knowledge Graph Entity Alignment via Representation Learning. In: *Database Systems for Advanced Applications* (2022)
9. Zhou, J., Zeng, W., Xu, H., et al.: Active Temporal Knowledge Graph Alignment. *International Journal on Semantic Web & Information Systems* **19**(1), 1–17 (2023)
10. Zhu, L., Li, G., Bai, L.: Leveraging neighborhood distance awareness for entity alignment for temporal knowledge graphs. *Neural Networks* **185**, 107181 (2025)
11. Bai, L., Song, X., Zhu, L.: Joint Multi-Feature Information Entity Alignment for Cross-Lingual Temporal Knowledge Graph With BERT. *IEEE Transactions on Big Data* **11**(2), 345–358 (2025)
12. Zhu, L., Li, G., Bai, L.: Neighborhood-aware entity alignment for temporal knowledge graph. *International Journal of Machine Learning and Cybernetics* (2024)
13. Xu, C., Su, et al.: Time-aware Entity Alignment using Temporal Relational Attention. In: *31st ACM World Wide Web Conference, WWW 2022* (2022)
14. Yang, Y., Cao, et al.: An entity alignment approach coupling NGBoost and SHAP for constructing spatio-temporal evolution knowledge graph from historical atlases. *International Journal of Geographical Information Science* **38**(8), 1468–1485 (2024)
15. Zhu, L., Li, et al.: A Temporal Knowledge Graph Modeling Method Based on Temporal-RDF for Entity Alignment. In: *ACM-TURC '24: ACM Turing Award Celebration Conference* (2024)
16. Zhu, L., Li, N., Bai, L.: Embedding-based entity alignment between multi-source temporal knowledge graphs. *Engineering Applications of Artificial Intelligence* **133**, 108451 (2024)
17. Song, J., Bai, L., An, X., et al.: Unsupervised fuzzy temporal knowledge graph entity alignment via joint fuzzy semantics learning and global structure learning. *Neurocomputing* **617**, 129019 (2025)
18. Zeng, W., Zhou, J., Zhao, X.: Benchmarking Challenges for Temporal Knowledge Graph Alignment. In: *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management (CIKM '24)*, pp. 3103–3112. ACM, New York, NY, USA (2024). <https://doi.org/10.1145/3627673.3679784>
19. Jia, W., Ma, R., Yan, L., et al.: Time-aware structure matching for temporal knowledge graph alignment. *Data & Knowledge Engineering* **151**, 102300 (2024)
20. Liu, X., Wu, J., Li, T., et al.: Unsupervised Entity Alignment for Temporal Knowledge Graphs. (2023)
21. Cai, L., Mao, et al.: A Simple Temporal Information Matching Mechanism for Entity Alignment Between Temporal Knowledge Graphs. *arXiv* (2022)
22. Time-aware Graph Neural Networks for Entity Alignment between Temporal Knowledge Graphs. (2024)
23. Mao, X., Ma, M., Yuan, H., Zhu, J., Wang, Z., Xie, R., Wu, W., Lan, M.: An Effective and Efficient Entity Alignment Decoding Algorithm via Third-Order Tensor Isomorphism. In: *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 5888–5898 (2022)
24. Sun, Z., Hu, W., Li, C.: Cross-Lingual Entity Alignment via Joint Attribute-Preserving Embedding. In: *ISWC*, pp. 628–644 (2017)

25. Sun, Z., Hu, W., Zhang, Q., Qu, Y.: Bootstrapping Entity Alignment with Knowledge Graph Embedding. In: IJCAI, pp. 4396–4402 (2018)
26. Wang, Z., Lv, Q., Lan, X., Zhang, Y.: Cross-lingual Knowledge Graph Alignment via Graph Convolutional Networks. In: EMNLP, pp. 349–357 (2018)
27. Cao, Y., Liu, Z., Li, C., Liu, Z., Li, J., Chua, T.S.: Multi-Channel Graph Neural Network for Entity Alignment. In: ACL, pp. 1452–1461 (2019)
28. Mao, X., Wang, W., Xu, H., Lan, M., Wu, Y.: MRAEA: An Efficient and Robust Entity Alignment Approach for Cross-lingual Knowledge Graph. In: WSDM, pp. 420–428 (2020)
29. Mao, X., Wang, W., Xu, H., Wu, Y., Lan, M.: Relational Reflection Entity Alignment. In: CIKM, pp. 1095–1104 (2020)
30. Xu, C., Su, F., Lehmann, J.: Time-aware Graph Neural Network for Entity Alignment between Temporal Knowledge Graphs. In: EMNLP, pp. 8999–9010 (2021)
31. Xu, C., Su, F., Xiong, B., Lehmann, J.: Time-aware Entity Alignment using Temporal Relational Attention. In: WWW, pp. 788–797 (2022)
32. Zhao, R., Zeng, W., Zhang, W., Zhao, X., Tang, J., Chen, L.: Towards Temporal Knowledge Graph Alignment in the Wild. arXiv preprint arXiv:2507.14475 (2025)
33. Zhao, R., Zeng, W., Tang, J., Li, Y., Ye, G., Du, J., Zhao, X.: Towards Unsupervised Entity Alignment for Highly Heterogeneous Knowledge Graphs. In: 2025 IEEE 41st International Conference on Data Engineering (ICDE), pp. 3792–3806. IEEE Computer Society, Los Alamitos, CA, USA (2025). <https://doi.org/10.1109/ICDE65448.2025.00283>
34. Zhao, R., Tang, J., et al.: Zero-shot Knowledge Graph Question Generation via Multi-agent LLMs and Small Models Synthesis. In: Proceedings of the 33rd ACM International Conference on Information and Knowledge Management (CIKM '24). ACM, New York, NY, USA (2024)
35. Zeng, W., Zhang, S., Peng, H., Tan, Z., Xiao, W., Zhao, X.: IKGA: An Interactive Visualization Tool for Knowledge Graph Alignment. In: 2025 IEEE 41st International Conference on Data Engineering (ICDE) (Demo Track). IEEE Computer Society, Los Alamitos, CA, USA (2025)
36. Zeng, W., Zhou, J., Zhao, X.: Benchmarking Challenges for Temporal Knowledge Graph Alignment. In: Proceedings of the 33rd ACM International Conference on Information and Knowledge Management (CIKM '24), pp. 3103–3112. ACM, New York, NY, USA (2024)
37. Zhang, S., Zeng, W., Tan, Z., Xiao, W., Zhao, X.: M3: A Multi-Image Multi-Modal Entity Alignment Dataset. In: Proceedings of the 33rd ACM International Conference on Information and Knowledge Management (CIKM '24) (Resource Paper). ACM, New York, NY, USA (2024)
38. Luo, Y., An, R., Zou, B., Tang, Y., Liu, J., Zhang, S.: LLM as Dataset Analyst: Subpopulation Structure Discovery with Large Language Model. arXiv preprint arXiv:2405.02363 (2024). <https://doi.org/10.48550/arXiv.2405.02363>
39. An, R., Yang, S., Lu, M., Zhang, R., Zeng, K., Luo, Y., Cao, J., Liang, H., Chen, Y., She, Q., Zhang, S., Zhang, W.: MC-LLaVA: Multi-Concept Personalized Vision-Language Model. arXiv preprint arXiv:2411.11706 (2024). <https://doi.org/10.48550/arXiv.2503.18854>
40. Lin, W., Wei, X., An, R., Gao, P., Zou, B., Luo, Y., Huang, S., Zhang, S., Li, H.: Draw-and-Understand: Leveraging Visual Prompts to Enable MLLMs to Comprehend What You Want. arXiv preprint arXiv:2403.20271 (2024). <https://doi.org/10.48550/arXiv.2403.20271>