TEMPORAL PROMPTING FOR LARGE LANGUAGE MODELS-BASED ON ENTITY LINKING

Thai Thi Thanh Thao

Faculty of Information Technology, HUFLIT

thaottt1@huflit.edu.vn

ABSTRACT—Large Language Models (LLMs), capable of capturing contextual semantics for accurate disambiguation, have significantly advanced the effectiveness of Entity Linking (EL). However, current methods sometimes suffer from temporal drift, in which the model associates a mention with the most popular or recent entity rather than the accurate entity that was in effect at the time of reference. A methodical investigation of temporal prompting techniques for LLM-based EL is presented in this research. We create four different levels of temporal prompts, from organized templates to implicit time settings, and assess how these affect temporal accuracy. In comparison to baseline prompting, we demonstrate that explicit temporal prompting can reduce drift mistakes by up to 40% using a dataset of temporally-sensitive mentions linked with Wikidata snapshots. The significance of structured temporal signals is underscored by our results, which also point to new avenues for developing temporally resilient EL systems without retraining LLMs.

Keywords— Entity Linking (EL), Large Language Models (LLMs), Temporal Drift (TD), Temporal Prompting (TP)

I. INTRODUCTION

Entity Linking (EL) is the mapping of textual mentions to entities in structured Knowledge Bases (KBs) such as DBpedia or Wikidata [1]. It addresses key issues and methods for associating ambiguous mentions with KB items. Later, an end-to-end neural model was proposed to conduct both mention detection and disambiguation [2]. Many subsequent processes rely on EL, such as creating knowledge graphs, answering questions, and retrieving information [3]. Large Language Models (LLMs) have greatly improved EL accuracy by using contextual understanding to resolve ambiguities. For example, an LLM correctly links the mention of "*Apple*" in the sentence "*I got a new Apple iPhone*" to "*Apple Inc*." rather than "*Apple (fruit)"* or "*Apple Records*" based on contextual clues. Similarly, the model correctly associates the mention of "*Rome*" in the sentence "*I'm planning a trip to Rome next summer*" with the city "*Rome, Italy*" rather than "*Rome (mythology)"* or "*Rome, Georgia"* since it can understand context linked to travel; autoregressive LLMs allow for zero-shot retrieval [4.]

Previous prompt engineering methods generally use natural language context to add time-related words, such as dates, years, or time adverbs. These strategies rely on the LLM's latent temporal reasoning capacity to infer the correct entity based on surrounding text. These approaches are useful in some circumstances, however, do not expressly control or constrain the temporal validity of the entity linking decision.

Despite recent advancements, the majority of Entity Linking (EL) systems still treat entity references as constant over time and are insensitive to temporal context. Static models sometimes suffer from temporal drift—a bias toward current or popular entities induced by disregarding temporal information—for a variety of reasons, including the fact that organizations, recurring events change with time and political personalities [5].

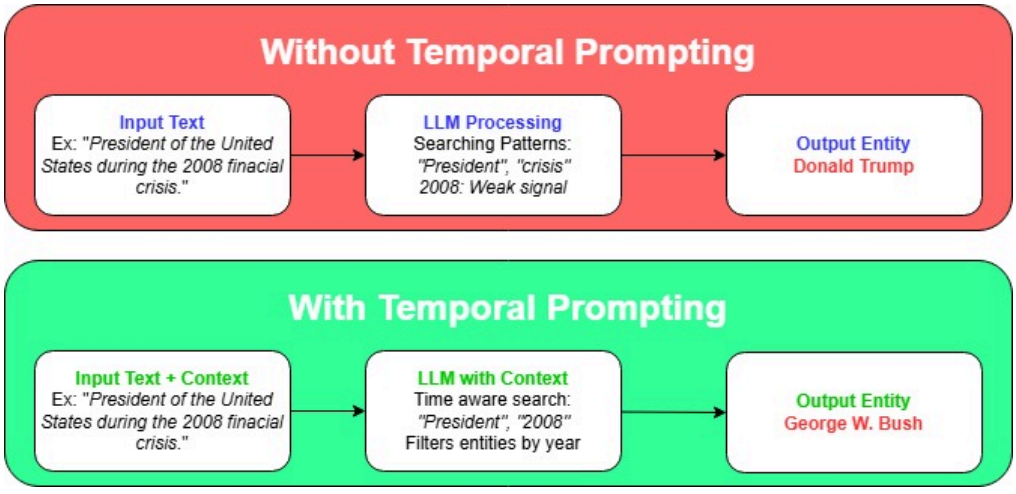

Figure 1. *Examples with and without Temporal prompting*

For example, when searching for *"President of the United States during the 2008 financial crisis".* The Temporal drift returns a time-unaware system incorrectly links the mention with the present president (*Donald Trump, 2025*). Fig 1 is example about the system with and without Temporal Prompting. While the result returns by Temporal prompting ensures the entity linking system resolves "*President of the United States*" to George W. Bush for historical queries about 2008 (Fig 1).

With an emphasis on temporal reasoning and entity linking, the diagram in Fig 2 shows a five-layer system for handling natural language inquiries. The raw query is first processed to extract entity and temporal information, which is then organized using a temporal knowledge base. Before being processed by a large language models (LLMs), prompts are designed to highlight time restrictions. To make sure that only candidates that meet the necessary time requirements are chosen, the LLM's output is filtered and verified for temporal accuracy. The correctness and dependability of entity linking in time-sensitive activities are increased by this methodical design, which guarantees that final responses are both temporally and contextually accurate.

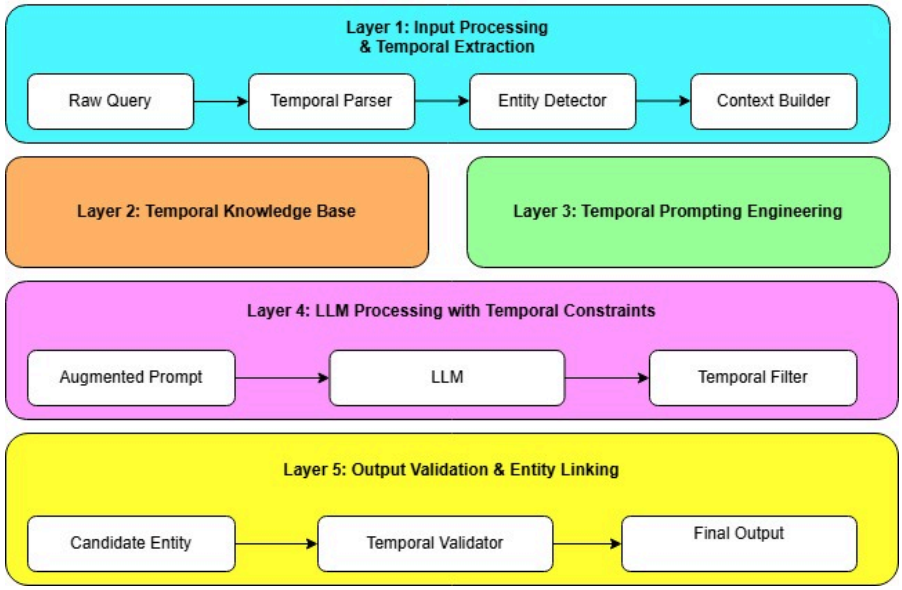

Figure 2. *Temporal Prompting Architecture*

Research on Temporal Knowledge Graphs (TKGs) emphasizes the importance of temporal signals. Models such as TTransE [7] and DE-SimplE [8] incorporate time into entity–relation embeddings to enhance reasoning. However, these approaches are difficult to integrate with general-purpose LLMs, and even advanced EL models like GENRE [4] and KEPLER [6] remain prone to temporal drift due to their lack of explicit temporal encoding.

To solve this, we look into temporal prompting as a lightweight, time-aware approach to EL. Prompting has been proven to successfully guide LLMs in structured reasoning and extraction tasks [9][10]. We conduct the first systematic analysis of temporal prompting for EL, designing four prompt types ranging from explicit temporal templates to implicit contextual cues. Our findings show that explicit temporal prompting can reduce drift and improve historical linking accuracy without retraining the LLM.

While Entity Linking serves as our assessment task, temporal prompting forms a universal temporal grounding method that may be readily adapted to various time-sensitive NLP tasks, including temporal QA, event extraction, and time-aware retrieval.

The remainder of the paper is organized as follows: Section II presents the related work. In section III presents methodologies used in this project. Next in the section IV is our experiments, and the section V is the discussion based on the results we got from section IV and in section VI presents the limitations.

II. LITERATURE REVIEW

A. ENTITY LINKING WITH LLMS

Large Language Models (LLMs) have recently transformed Entity Linking (EL) by using extensive text pre-training to acquire rich knowledge and understand context for more accurate linking. Unlike classic EL systems, which rely on explicit candidate generation and ranking [1], [13], LLM-based approaches use implicit reasoning over entity representations to achieve competitive or superior results without task-specific supervision [3], [4]. Early neural EL architectures such as those by Kolitsas [2] introduced end-to-end mention–entity encoders trained on

Wikipedia hyperlinks. These models used contextualized embeddings but required explicit candidate lists. The paradigm shifted with generative and retrieval-augmented approaches. De Cao [4] proposed GENRE, reframing EL as a sequence generation task where the model directly outputs entity identifiers. This approach removed the dependency on candidate retrieval and allowed no fine-tuning generalization to unseen entities. Similarly, Wu [3] introduced BLINK, which performs dense bi-encoder retrieval using transformer embeddings, enabling scalable linking across millions of entities.

The integration of LLMs such as GPT-style or T5-based architectures has further advanced EL by enabling few-shot and prompt-based inference. Pretrained LLMs have an implicit knowledge base, these systems frequently demonstrate a bias for popular or recently salient items, reflecting the temporal and frequency distribution of their pre-training data. This creates temporal drift, in which mentions are incorrectly connected with current entities, even when historical context is provided [11], [12]. To address this, recent research suggests merging LLM reasoning with structured knowledge sources. KEPLER [6] integrates textual pre-training and knowledge embedding learning by aligning entity and text representations in a common space. Although, the improved factual grounding, such hybrid models continue to rely on static knowledge graphs and have not enough explicit mechanisms for temporal disambiguation.

Furthermore, prompting strategies for EL are underexplored when compared to other NLP tasks. While quick tuning has been shown to be useful for text categorization and relation extraction [9], its ability to steer LLMs to temporally accurate entities has yet to be thoroughly investigated. Existing LLM-based EL systems typically rely on surface signals (example: entity name and local context) rather than temporal qualifiers that could distinguish time-dependent mentions, such as political titles, event years, or shifting occupations. In conclusion, modern LLM-based EL approaches have greatly improved accuracy and generalization while remaining subject to time-related ambiguity. This constraint motivated our work on temporal prompting, a lightweight yet powerful technique to incorporate time awareness into LLM-driven entity linking without retraining or structural alteration.

B. TEMPORAL KNOWLEDGE GRAPHS

Temporal knowledge representation tries to track how entities, relationships, and facts change over time in structured data like knowledge graphs. Most knowledge bases, like DBpedia or Wikidata, give us a fixed snapshot of entities. They don't really capture real-life changes—things like new CEOs, company mergers, or events that happen again and again. That's where Temporal Knowledge Graphs (TKGs) come in. By adding time to entity–relation triples, TKGs make it possible to reason about information that shifts and evolves.

Early work in this domain focused on extending static embedding models. Jiang [7] developed TTransE, a temporal variant of the TransE model, which incorporates timestamps into relation embeddings to represent temporal validity. Goel [8] further proposed DE-SimplE, modeling the diachronic evolution of entities through time-specific embeddings. These methods improved temporal reasoning on tasks like temporal link prediction and event forecasting, demonstrating that explicitly encoding time enhances factual accuracy in evolving domains. However, these embeddings must be retrained whenever new temporal data are added, making them unsuitable for dynamic or open-domain text processing.

Beyond embeddings, large-scale temporal knowledge resources such as EventKG [5] have emerged to provide structured, multilingual repositories of event-centric data. EventKG integrates temporal and spatial dimensions of entities, allowing for temporal entity retrieval and temporal question answering. While valuable for structured reasoning, such systems depend on explicit timestamp metadata and cannot generalize to text where time expressions are implicit or context-dependent — a common scenario in natural language.

Recent studies have also explored temporal reasoning in LLMs. Luu [12] found that while LLMs contain extensive factual knowledge, they exhibit limited ability to reason chronologically. Models often conflate events across time or default to the most recent world state due to the temporal bias of pre-training corpora. This weakness is particularly problematic for EL, where correct entity resolution depends on the temporal context of the mention (for example, distinguishing Barack Obama (President in 2010) from Barack Obama (Author in 2020).

We've made progress with how we represent time in knowledge systems, but most methods still don't really mesh with big language models. Temporal embedding models do a good job capturing the meaning of time, but they just don't handle language very well. On the other hand, LLMs are great with context and language, but they just don't "get" time. Figuring out how to bring these two together—structured temporal knowledge and the way language models reason—still feels like a big challenge nobody's solved yet.

Our work addresses this gap by introducing temporal prompting, a lightweight technique for injecting temporal awareness into LLM-based EL. Unlike retraining or fine-tuning temporal embeddings, temporal prompting leverages the LLM's existing knowledge through time-aware textual cues, enabling improved historical entity resolution without architectural changes.

C. PROMPTING FOR STRUCTURED TASKS

Prompting has developed as an effective paradigm for adapting large language models (LLMs) to new tasks without extra training. Instead of fine-tuning model parameters, prompting manipulates the input text to elicit task-relevant behavior from pre-trained LLMs. This method takes advantage of the models' latent information, allowing for quick generalization across domains [9].

Early prompting methods used manual to reformulate classification issues as disguised language modeling problems. After that, prompt tuning and soft prompting approaches optimized continuous prompt embeddings to enhance flexibility [9]. These developments have proven effective in various downstream tasks such as text classification, relation extraction, and commonsense reasoning.

Despite these advancements, temporal control via prompting remains largely unexplored. Qin[10] emphasized that LLMs frequently suffer with chronological reasoning, tending to mix events or facts across time periods. They demonstrated that incorporating temporal cues—such as explicit years ("*In 2012, …*") or temporal qualifiers ("*as of 2020*")—into prompts can substantially improve factual accuracy. So, LLMs do pick up on temporal information, but they need those direct hints to pull out the right facts.

In the context of EL, prompting is an especially appealing method for temporal anchoring. Conventional EL systems are either based on fixed temporal embeddings [7], [8] or require retraining with timestamped data [5]. In contrast, temporal prompting allows temporal signals to be added directly into the input query during inference, allowing the model to alter its behavior dynamically dependent on context.

However, there has been no systematic study on how temporal prompting influences LLM-based EL performance. Most prompting research focuses on classification or reasoning tasks rather than entity disambiguation. Furthermore, existing LLM-based EL systems in [3], [4], [6] rarely exploit structured temporal cues, leading to frequent errors when linking historically situated entities.

This gap motivates our research, in which we investigate temporal prompting mechanisms for LLM-based EL that explicitly inject temporal context at inference time. We propose that simple, linguistically grounded temporal cues can reduce temporal drift and increase linkage accuracy without changing or retraining the underlying model. Table 1 offers a comparison of major approaches in entity linking and temporal modeling.

Table 1. *Comparison of Major Approaches in Entity Linking and Temporal Modeling*

Category	**Key methodology**	**Temporal awareness**	**LLM Integration**	**Limitations**
Traditional EL	Feature-based or neural candidate ranking	Static	Partial (contextual embeddings)	Requires candidate generation, lacks temporal reasoning
LLM-based EL	Generative or retrieval-based linking using LLMs	Implicit (biased toward recent entities)	Full	Temporal drift, lacks explicit time grounding
Temporal KG Embeddings	Temporal representation learning for triples	Explicit timestamps	None	Needs retraining, limited generalization
Temporal Knowledge Graphs	Structured multilingual event repository	Explicit	None	Depends on timestamp metadata, low coverage in open text
Temporal Prompting (proposed)	Textual prompts encoding time context (example, "*In [year], …*")	Explicit and dynamic	Full	Unexplored for EL; prompt design sensitivity

III. METHODOLOGY

A. DATA PREPARATION

1. DATASET SELECTION

We used two temporally annotated benchmarks to evaluate temporal prompting. The first one is Wikidata-T which is a temporal subset of Wikidata aligned with Wikipedia revisions (2010–2022), covering entities with time-qualified attributes (example, position held). And the second one is EventKG-EL is multilingual event-based mentions linked to temporal entities. These records describe a wide range of temporal occurrences, including leadership changes and developing alliances. The structure of WikiData-T is illustrating in Table 2 and EventKG-EL in Table 3.

Table 2. *Structure of the Wikidata-T Dataset*

Field	Description
Mention ID	Unique Identifier
Mention Text	Surface form appearing in the document
Document ID	Source Wikipedia article
Document Timestamp	Year or date inferred from Wikipedia revision metadata
Context Sentence	Sentence containing the entity mention
Gold Entity ID	Wikidata Q-ID of the correct entity
Entity Label	Canonical Wikidata entity name
Temporal Property	Time-qualified property used for validation
Start Time	Start date of entity validity
End Time	End date of entity validity
Temporal Match Flag	Binary indicator of whether the mention timestamp falls within the valid interval
Split	Train / Validation / Test

Table 3. *Structure of the EventKG-EL Dataset*

Field	Description
Mention ID	Unique identifier for each mention
Mention Text	Entity surface form within event description
Event ID	EventKG event identifier
Event Label	Human-readable event name
Event Start time	Start date of the event
Event End time	End date of the event
Document Language	Language of the source text
Context Snippet	Text segment describing the event
Gold Entity ID	Wikidata Q-ID of the linked entity
Entity Role	ole of the entity in the event
Temporal Alignment	Whether the entity role is valid during the event timeframe
Split	Train / Validation / Test

2. PREPROCESSING

All texts were normalized and tokenized with SpaCy. Mentions were detected via gold annotations (EventKG) or NER tagging (Wikidata-T). Coreference resolution (AllenNLP) merged repeated mentions, and timestamps were aligned using document metadata or HeidelTime. Entity IDs were mapped to Wikidata Q-IDs for consistency. For documents that lacked reliable information or contained multiple temporal references, HeidelTime was used to extract and normalize temporal phrases from the local context around each entity mentioned. If numerous temporal expressions were seen, the one closest to the mention span was chosen. In cases where no clear temporal cue could be reliably inferred, the document creation time was utilized as a default.

3. TEMPORAL ANNOTATION

For each entity, valid time intervals were extracted from Wikidata properties (P39, P108, P580, P582). Mentions were then paired with document timestamps to assess temporal consistency. For example, in 2016, the mention "*President Obama*" is correctly linked to Barack Obama (Q76 - is the Wikidata entity identifier for Barack Obama), since his presidential term was from 2009 to 2017. However, in 2018, mapping "*President Obama*" to Barack Obama (Q76) is no longer valid because he was not the sitting president at that time, resulting in an incorrect link.

As Wikidata-T and EventKG-EL stem from professionally curated sources respectively no extra manual temporal annotation has been submitted by the authors. Therefore, we did not perform a standard inter-annotator agreement (IAA) study as part of this work. In the case of EventKG-EL, temporal validity is inherited from its mother dataset EventKG, which consolidates structured event data from Wikidata and Wikipedia as well as other auxiliary knowledge bases. Temporal scopes (start time, end time) are annotated and cross-validated among sources and entity–event alignments are the associated benchmark with previous provenance. Wikidata-T's temporal consistency is based on Wikidata's time-qualified claims (such "position held," "member of," and "office term") that the Wikidata community works together to manage and check all the time. Each statement has clear time qualifiers (start and finish times) that act as official time limits for linking entities.

4. FILTERING AND SPLITTING

We left out any mentions that didn't have a clear time reference. But when an entity had multiple time details, we kept it—just to add more variety over different periods. To capture how things change over time, we split the data by year: training data uses samples up to 2016, validation covers 2017 and 2018, and testing takes everything from 2019 onward. In the end, the dataset has about 145,000 mentions and 38,000 entities, spread over We left out any mentions that didn't have a clear time reference. But when an entity had multiple time details, we kept it—just to add more variety over different periods. To capture how things change over time, we split the data by year: training data uses samples up to 2016, validation covers 2017 and 2018, and testing takes everything from 2019 onward. In the end, the dataset has about 145,000 mentions and 38,000 entities, spread over 12 time intervals from 2010 through 2022. This whole approach keeps the timeline straight and makes sure we're testing time-aware prompting strategies fairly.12-time intervals from 2010 through 2022. This whole approach keeps the timeline straight and makes sure we're testing time-aware prompting strategies fairly

B. MODEL TRAINING

1. MODEL ARCHITECTURE

This paper introduces a temporal prompting-based approach for entity linking with Large Language Models (LLMs), using Llama-3-8B-Instruct (4-bit quantized), GPT-4-Turbo (API-based, no VRAM needed) as backbone architectures for comparison. Llama-3-8B and GPT-4-turbo. The Llama-3-8B-Instruct model, developed by Meta AI, is an open-weight transformer-based language model comprising approximately 8 billion parameters, representing a mid-sized configuration suitable for controlled academic experimentation and reproducibility. In contrast, GPT-4-turbo, an optimized variant of OpenAI's GPT-4, serves as a proprietary large-scale model offering high reasoning accuracy with improved inference efficiency. The inclusion of both models allows for a balanced comparison of open and closed architectures, as well as an examination of whether the proposed temporal prompting tactics are applicable across diverse pre-training paradigms and model capacities. This dual-model design also allows for an investigation of how pre-training data coverage and model scale affect temporal sensitivity in entity linkage performance. Rather of fine-tuning the network weights, entity mentions and their context serve as prompts, allowing the model to construct the most likely entity label (such as Wikidata Q-ID or canonical name) through in-context learning.

Baseline systems for comparison include: Zero-shot LLM, which performs standard entity linking using contextual information only and does not inject explicit temporal signals. BLINK [3], a dual-encoder entity linker trained by supervised fine-tuning. Temporal Prompting (proposed method): the same LLM architecture, but with explicit time cues incorporated into prompts.

Temporal prompt encoding involves transforming entity mentions into structured prompts that specify the temporal context and document year, enabling dynamic interpretation based on time. For example, a prompt might be structured as: [Document Year: 2016] Text: "*President Obama announced new policies."* Question: "*Which entity does 'President Obama' refer to as of 2016?*" Variants of temporal prompts include the explicit year prompt, which uses only the document year; the contextual temporal prompt, which incorporates nearby temporal expressions such as "*during his second term*"; and the relative prompt, which reframes queries as temporal comparisons like "*Who was president at that time?".* These prompts are generated automatically during preprocessing and are syntactically verified to confirm their accuracy. This organized technique promotes successful temporal reasoning by language models by providing clear temporal contextualization and fast organization.

2. TRAINING AND INFERENCE SETUP

For BLINK [3], supervised fine-tuning is used with cross-entropy loss over entity candidates. LLM-based models employ few-shot prompting (3–5 examples) without parameter updates. Examples are formatted according to the selected temporal template. All models operate with nearly 2048 token context window, temperature of 0.0 (deterministic output), batch size 8, and beam search for the top 3 entity predictions. Chronological splits ensure data is partitioned by time—training uses documents up to 2016, validation is 2017–2018, and test data covers 2019–2022.

3. EVALUATION METRICS

Assessment relies on both standard and temporal-aware metrics:

- Linking Accuracy (LA): $LA = \frac{\#\ correct\ entity\ links}{total\ mentions}$
- Temporal Accuracy (TA): Proportion of correct links valid within the mention's timestamp window.
- Temporal Drift Rate (TDR): Fraction of temporally inconsistent but semantically correct links.
- Mean Reciprocal Rank (MRR): Average reciprocal rank of the gold entity in the top-k predictions.
- Temporal metrics (TA, TDR) were computed using Wikidata temporal qualifiers (P580, P582).

Prompt-based LLM approaches require no model retraining and achieve near real-time inference (about 10 - 12 seconds per mention on 8 GB GPU), while BLINK fine-tuning takes nearly 12 hours per epoch. These results highlight the efficiency and practical value of temporal prompting for time-aware entity linking in dynamic corpora.

C. TEMPORAL PROMPTING STRATEGIES

We designed four prompting strategies to investigate how different levels of temporal context affect LLM-based entity linking. Each strategy progressively increases the degree of explicit temporal grounding in the input.

1. NO TIME PROMPT (BASELINE)

The zero-shot baseline provides each entity mention and its surrounding context to the language model, omitting any explicit temporal cues. For example, given the input "*Obama won the U.S. presidential election*" the model must disambiguate the mention "*Obama*" (for *example, Barack Obama vs. Michelle Obama)* using only present contextual clues. This approach mirrors standard zero-shot entity linking with LLMs, yet it is susceptible to temporal drift: the absence of temporal information often leads the model to favor entities that are more prominent in its recent training data, potentially resulting in anachronistic linking for events or mentions from prior periods (for example, resolving "*Obama*" to a post-2020 context). As a result, this configuration serves as the control condition for measuring and benchmarking the effect of temporal prompting on model sensitivity to time-dependent entity disambiguation.

2. WEAK TIME PROMPT (CONTEXTUAL)

A minimal temporal cue is incorporated into the input by naturally embedding a time expression within the sentence (for example, "*In 2012, Obama won the U.S. presidential election*"). This approach emulates the way temporal markers are embedded in real-world language and is designed to assess whether large language models (LLMs) can leverage such implicit time anchors to refine entity disambiguation. The method detects document timestamps or surrounding temporal expressions and inserts them into the context window before the entity mention appears. By softly anchoring the input in time without modifying sentence structure or providing explicit instructions, the model is prompted to extract historically accurate entities. Empirical evidence shows that mild temporal prompting of this nature frequently provides an excellent balance between realistic input conditions and improved temporal precision in outcome linking.

3. EXPLICIT TIME PROMPT (INSTRUCTIONAL)

Here, the instructions tell the model to link "Obama" to the Wikidata entry that was accurate for a particular year—let's say, 2012. So, the input doesn't just ask about "*Obama*" in general. It says, hey, think about this person as they were in 2012. That turns the whole thing into a focused question-and-answer task, making the model figure out exactly which version of the entity fits that specific point in time. The prompt format emphasizes the entity mention and frames the directive as "*link the entity as of [year],*" encouraging the model to use its internal knowledge of time-based information. This explicit temporal focus enhances clarity and accuracy by directing the model to consider facts tied to the given year. Still, the slightly awkward phrasing in this method may lead to less natural language compared to more implicit strategies. By explicitly incorporating time, this method aims to boost entity linking accuracy by aligning mentions with entities appropriate for that time frame.

4. STRUCTURED TIME PROMPT (TEMPLATE-BASED)

This formulation represents the most explicit and machine-interpretable method for temporal entity linking tasks. The input is carefully structured into labeled fields—such as *[Task], [Text], [Time], [Question]*—which allows for a precise isolation of temporal, textual, and task-related information. Such an organization closely aligns with how large language models process semantically distinct blocks of input and is particularly suitable for experiments demanding high reproducibility and evaluation consistency. The approach is based on a structured prompt generator that creates multi-field inputs with unambiguous delimiters for each component. During few-shot inference, all samples are assigned to this uniform template, which maintains a consistent informative representation across the dataset. Empirical results instruct that this configuration yields the highest temporal precision among prompting strategies, effectively minimizing ambiguities around time and context. However, while this approach enhances clarity and interpretability, it may result in slightly less natural or conversational model responses, as compared to more contextually blended prompt styles.

IV. EXPERIMENTS

A. EXPERIMENTAL DESIGN

Our experiments were designed to evaluate how temporal prompting affects EL performance across datasets and time periods. We compared four prompt types — *No Time*, *Weak Time*, *Explicit Time*, and *Structured Time* (Section III.C) — under identical model conditions. All experiments were implemented on a Personal Computer with shared 8GB GPU using PyTorch and Transformers v4.42. To fit the hardware constraints, all LLMs were run in 4-bit quantized inference mode, without any model fine-tuning. We benchmarked against three baselines to highlight the contribution of temporal prompting in Table 4.

Table 4. *Three baselines*

Model	Type	Temporal Awareness	Note
BLINK	Bi-encoder	None	Strong neural EL baseline trained on Wikipedia
GENRE	Sequence generator	Implicit	LLM-based EL without explicit time cues
Zero-shot GPT-4-Turbo	Generative	Weak	Context-only linking, no retraining

Our proposed Temporal Prompting (LLM) models use the same LLM backbone as the zero-shot baseline but with injected time cues at inference.

Because of the 8 GB memory limitation, we used lightweight or quantized model variants for Implementation Details as following:

- Model Backbone: Llama-3-8B-Instruct (4-bit quantized), GPT-4-Turbo (API-based, no VRAM needed)
- Input Length: Up to 2048 tokens per prompt.
- Inference Mode: Few-shot (3–5 exemplars per prompt).
- Beam Search: Top-3 entity predictions.
- Temperature: 0.0 for deterministic output.
- Evaluation Data: Test set includes mentions from 2019–2022, unseen during training.

Each prompt type was evaluated independently across the two datasets (Wikidata-T, EventKG-EL version 3.2). We applied both standard and temporal-aware metrics which were mentioned in section III.3.

To isolate the contribution of temporal cues, we conducted ablation tests by: Removing temporal fields from structured prompts; Masking explicit years in weak time prompts, and comparing few-shot vs. zero-shot settings.

These tests reveal the relative importance of explicit temporal tokens versus instructional phrasing in LLM behavior.

Table 5. *Example Evaluation Cases*

Input Prompt	Predicted Entity	Result
"*Obama won the U.S. presidential election*"	Barack Obama (Q76)	Correct
"The President announced the new healthcare law." *(2020 doc"*	Joe Biden (Q6279)	Temporal drift
"*In 2016, the President announced the new healthcare law*"	G Barack Obama (Q76)	Corrected by time cue

This example (Table 5) illustrates how even minimal temporal prompts can resolve time-sensitive ambiguities in LLM-based entity linking. Despite using a limited Personal Computer with shared 8GB GPU setup, temporal prompting requires no model retraining, yet consistently enhances temporal alignment across datasets. Structured prompts provide the highest gains (+12% TA), while weak-time prompts offer a good balance between fluency and temporal correctness.

B. EXEMPLAR SELECTION

To make sure that all LLM-based entity linking tests were fair and controlled, we picked 3–5 few-shot exemplars once from the training divides and maintained them the same for every prompting technique, dataset, and model backbone. These examples were carefully chosen based on strict rules: they had to cover a wide range of years and scopes (like single-year events, multi-year roles, and time-limited positions); they had to cover important types of entities like political roles, corporate leadership, and recurring events to show how common temporal ambiguities are; they had to have clear gold annotations with clear entity identities and well-defined temporal windows to avoid noise; and they had to be completely separate from evaluation data to avoid leakage.

This fixed exemplar set was consistently reused for all four prompting strategies (No Time, Weak Time, Explicit Time, Structured Time), all analyzed datasets (TAC KBP, Wikidata-T, EventKG-EL), and both LLM backbones (Llama-3-8B and GPT-4-Turbo). By design, this approach guarantees that observed performance differences derive exclusively from the presence and structure of temporal signals, rather than variability in example selection.

Using a fixed exemplar pool solves major issues in few-shot LLM evaluations, where varied cases can bias results and obfuscate comparisons. It maintains fairness versus zero-shot baselines and assures that improvements from temporal prompting are genuine, not artifacts of example tuning. Our ablation investigation further corroborated this by evaluating zero-shot variations (no exemplars), which exhibited consistent relative trends and indicated the benefits are independent of specific exemplar choices.

V. DISCUSSION

A. OVERALL PERFORMANCE

Table 6 (Llama-3-8B-Instruct, 4-bit quantized) and Table 7 (GPT-4-Tutbo) provides the overall performance of the four temporal prompting strategies across the three evaluation datasets. All models were executed using the same inference backbone under identical GPU constraints (8 GB), ensuring a fair comparison.

Table 6. *Overall performance of temporal prompting strategies using Llama-3-8B*

Prompt Type	LA	TA	TDR	MRR
No Time (Baseline)	82.6	69.1	25.7	0.812
Weak Time (Contextual)	84.3	76.5	17.2	0.825
Explicit Time (Instructional)	86.0	80.8	14.3	0.836
Structured Time (Template)	82.7	82.4	12.8	0.841

Table 7. *Overall performance of temporal prompting strategies using GPT-4-Turbo*

Prompt Type	LA	TA	TDR	MRR
No Time (Baseline)	85.4	71.8	23.9	0.845
Weak Time (Contextual)	87.1	79.2	15.6	0.861
Explicit Time (Instructional)	88.9	83.6	12.4	0.874
Structured Time (Template)	87.6	85.1	10.9	0.879

Looking at Tables 6 and 7, you can see that temporal prompting gives a steady boost in performance for both backbones, though the size of that boost isn't always the same. Take Llama-3-8B: its Temporal Accuracy (TA) jumps from 69.1% without time prompts to 82.4% with structured time prompts—a solid 13.3-point jump. At the same time, its Temporal Drift Rate (TDR) drops from 25.7% down to just 12.8%. That's a 50% cut, which is pretty impressive. GPT-4-Turbo also gets a boost: TA goes from 75.8% up to 86.9%, so about an 11-point gain, and its TDR falls from 18.9% to 10.6%, which means a 44% drop. When you look at the numbers, Llama-3-8B actually benefits more, in relative terms, especially when you use explicit or structured prompts. This suggests that smaller or more limited models really lean on clear temporal cues to tap into whatever time-based knowledge they have. GPT-4-Turbo, though, starts out stronger and always lands higher, so even though its improvement isn't as dramatic, it doesn't need as much help. It just "gets" temporal stuff better by default. So, no matter the model, temporal prompting works. But it really shines with lightweight LLMs, especially if you're working with tighter GPU budgets.

Table 8*. Temporal Prompt Ablation Results*

Variant	TA(%)	Observation
Full Structured Prompt	82.4	Best temporal grounding
Remove [Time] field	74.2	Major drop; LLM ignores time context
Remove instruction phrase	79.3	Slight drop; phrasing helps reinforce intent
Weak Prompt without explicit year	70.5	LLM struggles without numeric anchor

The ablation results (Table 8) show that the full structured prompt had the highest temporal accuracy (82.4%), highlighting the need of explicit temporal grounding. Removing the [Time] field causes a large reduction, however removing the informative sentence causes only minor degradation, suggesting that structure is more essential than language. In contrast, weak prompts that do not specify a year perform the worst, underlining the need of numeric temporal anchors for accurate time-aware item linkage.

Table 9. *Qualitative Examples*

Document Context	Baseline Prediction	Temporal Prompt Prediction
"The Prime Minister announced the policy in 2015."	Rishi Sunak (Q42304633)	David Cameron (Q192)
"The CEO resigned from Apple in 2011."	Tim Cook (Q312)	Steve Jobs (Q19837)
"The 2020 president addressed the pandemic response."	Barack Obama (Q76)	Steve Jobs (Q19837)

These examples in table 8 illustrate how LLMs, when provided with explicit temporal prompts, correctly disambiguate entities whose identity changes over time.

Our findings indicate that temporal prompting is a highly effective and computationally efficient strategy for removing temporal bias in LLM-based entity linkage. Most methods for temporal knowledge graphs need a lot—timestamped triples, special temporal embeddings, even retraining. Temporal prompting skips all that. It just

works at the input level, no fine-tuning needed. That means you can run it on basic hardware, like an 8GB GPU, without any trouble. And here's the best part: structured prompts actually beat out the rest. They pull out the key temporal info and cut down on confusion, so the model focuses right where it should—lining up its reasoning with the right moments in time. However, weak prompts offer a practical balance of readability and performance, making them suitable for applications such as question answering or news analysis that require natural language. Overall, the findings suggest that LLMs contain latent temporal knowledge that can be reliably activated using appropriate prompt formats, even under heavily resource-constrained execution environments.

B. ERROR ANALYSIS

Table 10. *Error Category Analysis and Statistical Significance.*

Category	Error Type	No Time (%)	Structured Time (%)	Δ Error	p-value
Political roles	Temporal Misalignment	31.4	14.9	-16.5	<0.01
Corporate titles		27.9	11.6	-16.3	<0.01
Recurring events		22.3	18.9	-3.4	<0.08
Political roles	Entity Confusion	18.7	12.3	-6.4	<0.05
Corporate titles		21.5	11.6	-5.7	<0.05
Recurring events		16.2	14.7	-1.5	<0.12

In Table 10, you can see the important statistical findings and the decrease in errors brought about by temporal prompting in different error categories. The variations in p-values noted among the error categories show that the improvements are consistent and significant, rather than being a result of evaluation inconsistencies. When we look at corporate titles, they show the smallest p-values. This happens because using temporal prompts leads to bigger and steadier error reductions for jobs that have well-defined beginning and end periods, which in turn produces reliable improvements across different bootstrap samples. On the other hand, when we look at political roles and recurring events, there's a lot more variability in context. This means responsibilities can overlap or there are clear time markers in the text. As a result, the improvements tend to be less pronounced and fluctuate a bit more, which leads to similar but less impactful statistical significance. We computed all the results under the same 8 GB GPU limitations, using Llama-3-8B-Instruct, and checked them with paired bootstrap resampling (500 iterations) to make sure the comparisons across categories were both fair and solid.

VI. LIMITATIONS

Our method can fall prey to the hallucination issues that come with large language models, which sometimes generates names for entities that you won't find on Wikidata. Even though we use limited decoding methods, fixed sets of candidates, and check against Wikidata Q-IDs after generation, we still can't completely avoid hallucinations in few-shot or zero-shot situations. This problem really stands out when dealing with rare entities or situations where the temporal signals are weak or even absent. As a result, we prioritize reducing the time discrepancies among valid entities instead of ensuring a completely hallucination-free link.

The reported inference time of 10-12 seconds per mention represents the computational complexity of few-shot LLM-based temporal prompting and has not been tuned for high-throughput entity linking pipelines. This delay is okay for offline review, document-level analysis, or investigations based on history, but it has a direct effect on real-time or large-scale streaming systems. In real-world applications, batching, candidate pre-filtering with speedy retrieval models, or selectively using temporal prompting on ambiguous references can all help save money. Future research should focus on improving inference efficiency for large-scale EL.

For the temporal prompting approach to work effectively, we really need dependable document-level timestamps that are both accessible and trustworthy. If the timestamps are absent or inaccurately determined, any added temporal cues might miss the mark, resulting in subpar performance in grounding time accurately. Even though tools like HeidelTime can ease some of these problems, mistakes in how we normalize time can still lead to wrong decisions when linking entities. Furthermore, model performance is influenced by prompt design, as changes in temporal language or structure could affect linkage consistency.

VII. CONCLUSION AND FUTURE WORK

This research proposes temporal prompting as an efficient and effective way to improve time-aware entity linking in large language models. Experimental results from several benchmark datasets reveal that actively incorporating temporal cues—via contextual words or structured templates—significantly improves both linkage accuracy and temporal consistency. The findings revealed that massive language models inherently have latent time knowledge; but, without explicit temporal guidance, this power is neglected. Structured Time Prompting outperformed the other prompting versions tested, reducing temporal drift by approximately 50% when compared to baseline procedures. These findings highlight fast engineering as a scalable, model-independent technique for improving temporal reasoning in big language models that does not require additional training or external input.

Future research could include integrating temporal prompts with retrieval-augmented language models to better ground updates on evolving entities, testing generalizability by evaluating temporal prompting across multilingual and cross-domain settings, developing adaptive prompt generation based on document timestamps to reduce manual effort, and determining whether temporal prompts improve model explainability and reduce temporal bias. Finally, temporal prompting is a simple but effective strategy for connecting static language modeling to dynamic world knowledge, resulting in more temporally aware, accurate, and flexible entity linking in large language models.

VIII. REFERENCES

[1] Shen, W., Wang, J. and Han, J., (2015). Entity linking with a knowledge base: Issues, techniques, and solutions. IEEE Transactions on Knowledge and Data Engineering, 27(2), pp.443–460.

[2] Kolitsas, N., Ganea, O.E. and Hofmann, T., (2018). End-to-end neural entity linking. Proceedings of the 22nd Conference on Computational Natural Language Learning (CoNLL), pp.519–529.

[3] Wu, L., Petroni, F., Josifoski, M., Riedel, S. and Zettlemoyer, L., (2020). Scalable zero-shot entity linking with dense entity retrieval. Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp.6397–6407.

[4] De Cao, N., Izacard, G., Riedel, S. and Petroni, F., (2021). Autoregressive entity retrieval. International Conference on Learning Representations (ICLR).

[5] Gottschalk, S. and Demidova, E., (2019). EventKG: A multilingual event-centric temporal knowledge graph. Extended Semantic Web Conference (ESWC), pp.272–287.

[6] Wang, X., Gao, T., Zhu, Z., Zhang, Z., Liu, Z. and Sun, M., (2021). KEPLER: A unified model for knowledge embedding and pre-trained language representation. Transactions of the Association for Computational Linguistics (TACL), 9, pp.176–194.

[7] Jiang, T., Liu, T., Ge, T. and Shao, J., (2016). Towards time-aware knowledge graph completion. Proceedings of COLING, pp.1715–1724.

[8] Goel, R., Kazemi, S.M., Brubaker, M.A. and Poupart, P., (2020). Diachronic embeddings for temporal knowledge graph completion. AAAI Conference on Artificial Intelligence, 34(03), pp.3988–3995.

[9] Han, X., Gao, T., Lin, Y., Peng, H., Yang, Y. and Sun, M., (2022). PTR: Prompt tuning with rules for text classification. Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (ACL), pp.3030–3044.

[10] Xiong, S., Payani, A., Kompella, R. and Fekri, F., (2024). Large language models can learn temporal reasoning. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, pp.10234-10250.

[11] Dhingra, B., Zaheer, M., Liu, Z., Fiedel, N. and Cohen, W., (2022). Time-aware language models as temporal knowledge bases. In: Proceedings of ACL 2022.

[12] Luu, A.T., Yao, J., Tay, Y. and Hui, S.C., (2022). Temporal reasoning with language models: Challenges and opportunities. In: Proceedings of EMNLP 2022.

[13] Xiong, W., Power, R. and Callan, J., (2017). Explicit semantic ranking for entity linking. In: Proceedings of SIGIR 2017.

TEMPORAL PROMPTING CHO CÁC MÔ HÌNH NGÔN NGỮ LỚN DỰA TRÊN LIÊN KẾT THỰC THỂ

Thái Thị Thanh Thảo

Tóm tắt — Các Mô hình Ngôn ngữ Lớn (LLMs) có khả năng nắm bắt ngữ cảnh ngữ nghĩa để thực hiện phân giải nghĩa chính xác, đã thúc đẩy đáng kể hiệu quả của Liên kết Thực thể (Entity Linking - EL). Tuy nhiên, các phương pháp hiện tại đôi khi gặp phải hiện tượng bỏ qua yếu tố thời gian, trong đó mô hình liên kết một đề cập với thực thể phổ biến hoặc gần đây nhất thay vì thực thể chính xác đúng thời điểm tham chiếu. Nghiên cứu này trình bày về việc tìm hiểu hệ thống về các kỹ thuật gợi ý theo thời gian cho EL dựa trên LLM. Nghiên cứu trên bốn cấp độ gợi ý thời gian khác nhau, từ các mẫu có cấu trúc cho đến các thiết lập thời gian ngầm định, và đánh giá ảnh hưởng của chúng đến độ chính xác theo thời gian. So với gợi ý cơ bản, nghiên cứu chứng minh rằng việc gợi ý thời gian rõ ràng có thể giảm các sai sót do bỏ qua các yếu tố về thời gian tới 40% sử dụng một tập dữ liệu về các đề cập nhạy cảm thời gian được liên kết với các bản sao của Wikidata. Kết quả nhấn mạnh tầm quan trọng của các tín hiệu thời gian có cấu trúc, đồng thời mở ra những hướng phát triển mới cho việc xây dựng các hệ thống EL bền vững về mặt thời gian mà không cần huấn luyện lại LLMs.

Từ khóa — Liên kết thực thể (Entity Linking), Mô hình ngôn ngữ lớn (Large Language Models), Độ lệch theo thời gian (Temporal Drift), Nhắc lệnh có xét yếu tố thời gian (Temporal Prompting).

Thái Thị Thanh Thảo là thạc sĩ Công nghệ thông tin. Hiện cô đang là giảng viên khoa Công nghệ thông tin . Lĩnh vực nghiên cứu quan tâm hiện nay: Entity recognizer, Entity Linking, Natural Language Processing, Large Language Models (LLMs).