zrLLM: Zero-Shot Relational Learning on Temporal Knowledge Graphs with Large Language Models

Anonymous ACL submission

Abstract

Modeling evolving knowledge over temporal knowledge graphs (TKGs) has become a heated topic. Various methods have been proposed to forecast links on TKGs. Most of them are embedding-based, where hidden representations are learned to represent knowledge graph (KG) entities and relations based on the observed graph contexts. Although these methods show strong performance on traditional TKG forecasting (TKGF) benchmarks, they face a strong challenge in modeling the unseen zero-shot relations that have no prior graph context. In this paper, we try to mitigate this problem as follows. We first input the text descriptions of KG relations into large language models (LLMs) for generating relation representations, and then introduce them into embedding-based TKGF methods. LLM-empowered representations can capture the semantic information in the relation descriptions. This makes the relations, whether seen or unseen, with similar semantic meanings stay close in the embedding space, enabling TKGF models to recognize zero-shot relations even without any observed graph context. Experimental results show that our approach helps TKGF models to achieve much better performance in forecasting the facts with previously unseen relations, while still maintaining their ability in link forecasting regarding seen relations.

1 Introduction

Knowledge graphs (KGs) represent world knowledge with a collection of facts in the form of (s, r, o) triples, where in each fact, s, o are the subject and object entities and r is the relation between them. Temporal knowledge graphs (TKGs) are introduced by further specifying the time validity. Each TKG fact is denoted as a quadruple (s, r, o, t), where t (a timestamp or a time period) provides temporal constraints. Since world knowledge is ever-evolving, TKGs are more expressive in representing dynamic factual information.

In recent years, there has been an increasing number of works paying attention to forecasting future facts in TKGs, i.e., TKG forecasting (TKGF) or TKG extrapolated link prediction (LP). Most of them are embedding-based, where entity and relation representations are learned with the help of the observed graph contexts. Although traditional embedding-based TKGF methods show impressive performance on current benchmarks, they share a common limitation. In these works, models are trained on the TKG facts regarding a set of relations $\mathcal{R}$, and they are only expected to be evaluated on the facts containing the relations in $\mathcal{R}$. They cannot handle any zero-shot (ZS) unseen relation $r \notin \mathcal{R}$ because no graph context regarding unseen relations exists in the training data and thus no reasonable relation representations can be learned. In the forecasting scenario, as time flows, new knowledge is constantly introduced into a TKG, making it expand in size. This increases the chance of encountering newly-emerged relations, and therefore, it is meaningful to improve embedding-based TKGF methods to be more adaptive to ZS relations.

With the increasing scale of pre-trained language models (LMs), LMs become large LMs (LLMs). Recent studies find that LLMs have shown emerging abilities in various aspects (Wei et al., 2022) and can be taken as strong semantic knowledge bases (KBs) (Petroni et al., 2019). Inspired by this, we try to enhance the performance of embedding-based TKGF models over ZS relations with an approach consisting of the following three steps: (1) Based on the relation text descriptions provided in TKG datasets, we first use an LLM to produce an enriched relation description (ERD) with more details for each KG relation (Sec. 3.1). (1) We then generate the relation representations by leveraging another LLM, i.e., T5-11B (Raffel et al., 2020). We input ERDs into T5's encoder and transform its output into relation representations of TKGF models (Sec. 3.1). (3) We design a relation history learner (RHL) to capture historical relation patterns, where we leverage LLM-empowered relation representations to better reason over ZS relations (Sec. 3.2). With these steps, we align the natural language space provided by LLMs to the embedding space of TKGF models, rather than letting models learn relation representations solely from observed graph contexts. Even without any observed associated facts, ZS relations can be represented with LLM-empowered representations that contain semantic information. We term our approach as zrLLM since it is used to enhance **ZS** **r**elational learning on TKGF models by using **LLM**s.

We experiment zrLLM on seven recent embedding-based TKGF models and evaluate them on three new datasets constructed specifically for studying TKGF regarding ZS relations. Our contribution is three-folded: (1) To the best of our knowledge, this is the first work trying to study ZS relational learning in TKGF. (2) We design an LLM-empowered approach zrLLM and manage to enhance various recent embedding-based TKGF models in reasoning over ZS relations. (3) Experimental results show that zrLLM helps to substantially improve all considered TKGF models' abilities in forecasting the facts containing unseen ZS relations, while still maintaining their ability in link forecasting regarding seen relations.

2 Preliminaries

2.1 Related Work

Due to page limit, see App. J for more details.

Traditional TKG Forecasting Methods. Traditional TKGF methods are trained to forecast the facts containing the KG relations (and entities) seen in the training data, regardless of the case where ZS relations (or entities) appear as new knowledge arrives. These methods can be categorized into two types: embedding-based and rule-based. Embedding-based methods learn hidden representations of KG relations and entities, and perform link forecasting based on them. Most existing embedding-based methods, e.g., (Jin et al., 2020; Han et al., 2021b; Li et al., 2021b, 2022; Liu et al., 2023), learn evolutional entity and relation representations from the historical TKG information by jointly employing graph neural networks (Kipf and Welling, 2017) and recurrent neural structures, e.g., GRU (Cho et al., 2014). Some other approaches (Han et al., 2021a; Sun et al., 2021; Li et al., 2021a) start from each LP query[1] and traverse the temporal history in a TKG to search for the prediction answer. There also exist some methods, e.g., (Zhu et al., 2021; Xu et al., 2023b), that achieve forecasting based on the appearance of historical facts. Compared with embedding-based TKGF approaches, rule-based TKGF has still not been extensively explored. One popular rule-based TKGF method is TLogic (Liu et al., 2022). It extracts temporal logic rules from TKGs and uses a symbolic reasoning module for LP. Based on it, ALRE-IR (Mei et al., 2022) proposes an adaptive logical rule embedding model to encode temporal logical rules into rule representations. This makes ALRE-IR both a rule-based and an embedding-based method. Rule-based TKGF methods have strong ability in reasoning over ZS unseen entities connected by the seen relations, however, they are not able to handle unseen relations since the learned rules are strongly bounded by the observed relations.

Inductive Learning on TKGs. Inductive learning on TKGs refers to developing models that can handle the relations and entities unseen in the training data. Most of TKG inductive learning methods are based on few-shot learning (FSL), e.g., (Ding et al., 2022; Zhang et al., 2019; Ding et al., 2023b; Mirtaheri et al., 2021; Ding et al., 2023a,a; Ma et al., 2023). They first compute inductive representations of newly-emerged entities or relations based on K-associated facts (K is a small number, e.g., 1 or 3), and then use them to predict other facts regarding few-shot elements. One limitation of these works is that the inductive representations cannot be learned without the K-shot examples, making them hard to solve the ZS problems. Different from FSL methods, SST-BERT (Chen et al., 2023a) pre-trains a time-enhanced BERT (Devlin et al., 2019) and proves its inductive power over unseen entities but has not shown its ability in reasoning ZS relations. Another recent work MTKGE (Chen et al., 2023b) is able to concurrently deal with both unseen entities and relations. However, it requires a support graph containing a substantial number of data examples related to the unseen entities and relations, which is far from the ZS setting.

TKG Reasoning with Language Models. Recently, more and more works have introduced LMs into TKG reasoning. SST-BERT pre-trains an LM

[1] A TKG LP query is denoted as $(s, r, ?, t)$ (object prediction query) or $(?, r, o, t)$ (subject prediction query).

on a corpus of training TKGs for fact reasoning. ECOLA (Han et al., 2023) aligns facts with additional fact-related texts and enhances TKG reasoning with BERT-encoded language representations. PPT (Xu et al., 2023a) converts TKGF into the pre-trained LM masked token prediction task and finetunes a BERT for TKGF. Apart from them, one recent work (Lee et al., 2023) explores in-context learning (ICL) (Brown et al., 2020) with LLMs to predict future facts without finetuning. Another recent work GenTKG (Liao et al., 2023) finetunes Llama2-7B (Touvron et al., 2023), and let it directly generate the LP answer in TKGF.

Although previous works have shown success of LMs in TKG reasoning, they have limitations: (1) None of them has studied whether LMs, in particular LLMs, can be used to better reason ZS relations. (2) By only using ICL, LLMs are beaten by traditional TKGF methods in performance (Lee et al., 2023). The performance can be greatly improved by finetuning LLMs (Liao et al., 2023), but finetuning LLMs requires huge computational resources. (3) Since LMs are pre-trained with a huge corpus originating from diverse information sources, it is inevitable that they have already seen the world knowledge before they are used to solve TKG reasoning tasks. Most popular TKGF benchmarks are constructed with the facts before 2020 (ICEWS14/18/05-15 (Jin et al., 2020)). The facts inside are based on the world knowledge before 2019, which means LMs might have encountered them in their training corpus, posing a threat of information leak to the LM-driven TKG reasoning models. To this end, we (1) draw attention to studying the impact of LLMs on ZS relational learning in TKGs; (2) make a compromise between performance and computational efficiency by not finetuning LMs or LLMs but adapting the LLM-provided semantic information to non-LM-based TKGF methods; (3) construct new benchmarks whose facts are all happening from 2021 to 2023, which avoids the threat of information leak when we utilize T5-11B that was released in 2020.

2.2 Definitions and Task Formulation

Definition 1 (TKG). Let $\mathcal{E}$, $\mathcal{R}$, $\mathcal{T}$ denote a set of entities, relations and timestamps, respectively. A TKG $\mathcal{G} = \{(s, r, o, t)\} \subseteq \mathcal{E} \times \mathcal{R} \times \mathcal{E} \times \mathcal{T}$ is a set of temporal facts where each fact is represented with a fact quadruple (s, r, o, t).

Definition 2 (TKG Forecasting). Assume we have a ground truth TKG $\mathcal{G}_{gt}$ that contains all the true facts. Given an LP query $(s_q, r_q, ?, t_q)$ (or $(o_q, r_q, ?, t_q)$), TKGF requires the models to predict the missing object o_q (or subject s_q) based on the facts observed before the query timestamp t_q, i.e., $\mathcal{O} = \{(s, r, o, t_i) \in \mathcal{G}_{gt} | t_i < t_q\}$.

Definition 3 (Zero-Shot TKG Forecasting). Assume we have a ground truth TKG $\mathcal{G}_{gt} \subseteq \mathcal{E} \times \mathcal{R} \times \mathcal{E} \times \mathcal{T}$, where $\mathcal{R}$ can be split into seen $\mathcal{R}_{se}$ and unseen $\mathcal{R}_{un}$ relations ($\mathcal{R} = \mathcal{R}_{se} \cup \mathcal{R}_{un}, \mathcal{R}_{se} \cap \mathcal{R}_{un} = \emptyset$). Given an LP query $(s_q, r_q, ?, t_q)$ (or $(o_q, r_q, ?, t_q)$) whose query relation $r_q \in \mathcal{R}_{un}$, models are asked to predict the missing object o_q (or subject s_q) based on the facts $\mathcal{O} = \{(s, r_i, o, t_i) \in \mathcal{G}_{gt} | t_i < t_q, r_i \in \mathcal{R}_{se}\}$ containing seen relations and happening before t_q.

3 zrLLM

zrLLM is coupled with TKGF models to enhance ZS ability. It uses GPT-3.5 to generate enriched relation descriptions (ERDs) based on the relation texts provided by TKG datasets. It further inputs the ERDs into the encoder of T5-11B and aligns its output to TKG embedding space. zrLLM also employs a relation history learner (RHL) to capture the temporal relation patterns based on the LLM-empowered relation representations. See Fig. 1 for illustration of zrLLM-enhanced TKGF models.

3.1 Represent KG Relations with LLMs

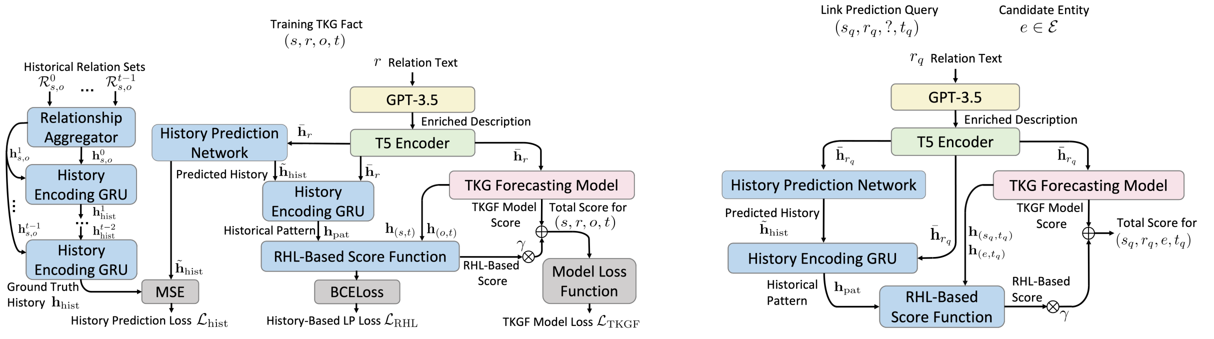
Generate Text Representations with ERDs. We generate text representations with T5-11B based on the textual descriptions of KG relations. Since the relation texts provided by TKG datasets are short and concise, we use GPT-3.5[2] to enrich them for more comprehensive semantics. Our prompt for description enrichment is depicted in Fig. 2. For each relation, we treat the combination of its relation text and LLM-generated explanation as its ERD. See Table 1 for two enrichment examples.

KG Relation Text	Enriched Relation Description
Engage in negotiation	Engage in negotiation: This indicates a willingness to participate in discussions or dialogues with the aim of reaching agreements or settlements on various issues.
Praise or endorse	Praise or endorse: This signifies a positive evaluation or approval of another entity's actions, policies, or behavior. It is a form of expressing support or admiration.

Table 1: Relation description enrichment examples.

We then input the ERDs into T5-11B. T5 is with an encoder-decoder architecture, where its encoder can be taken as a module that helps to understand the text input and the decoder is solely used for

[2]https://platform.openai.com/docs/model-index-for-researchers



(a) Training pipeline of zrLLM-enhanced model.

(b) Evaluation pipeline of zrLLM-enhanced model.

Figure 1: Illustration of zrLLM-enhanced TKGF models. RHL-related components are marked in blue. RHL works differently in training and evaluation. During training, since we know both entities (s, o in 1a) in the training fact, we can find the ground truth historical relations between them over time. We train a history prediction network (HPN) that aims to generate the relation history between two entities given their current relation (r). During evaluation, we directly use the trained HPN to infer the relation history. See Sec. 3 for details.

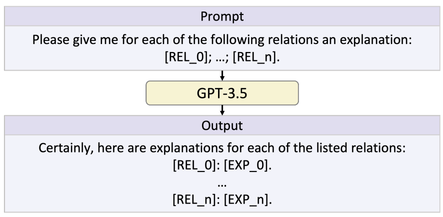


Figure 2: Prompting GPT-3.5 for ERDs. [REL_0], ..., [REL_n] are the dataset provided relation texts for a batch of n KG relations. [EXP_0], ..., [EXP_n] are the LLM-generated explanations. [REL:_0]: [EXP_0], ..., [REL:_n]: [EXP_n] are taken as ERDs.

text generation. We take the output of T5-11B's encoder, i.e., the hidden representations, for our downstream task. Note that although ERDs are produced by GPT-3.5 who is trained with the corpus until the end of 2021, the representations used for TKGF are generated only with T5-11B, preventing information leak. Also, through our prompt, GPT-3.5 does not know our underlying task of TKGF. We manually check the ERDs generated by GPT-3.5 and make sure that no factual information regarding entities after 2020 is included.

Align Text Representations to TKG Embedding Space. For each KG relation r, the T5-generated text representation is a parameter matrix $\bar{\mathbf{H}}_r \in \mathbb{R}^{L \times d_w}$. L is the length of the Transformers (Vaswani et al., 2017) in T5 and d_w is the embedding size of each word output from T5 encoder. The l^{th} row in $\bar{\mathbf{H}}_r$ is the T5 encoded hidden representation $\mathbf{w}_l \in \mathbb{R}^{d_w}$ of the l^{th} word in the enriched description. To align $\bar{\mathbf{H}}_r$ to an embedding-based TKGF model, we first use a multi-layer perceptron (MLP) to map each $\mathbf{w}_l$ to the dimension of the TKGF model's relation representation.

$$\mathbf{w}'_l = \text{MLP}(\mathbf{w}_l), \text{where } \mathbf{w}'_l \in \mathbb{R}^d. \tag{1}$$

Then we learn a representation of r's ERD $\bar{\mathbf{h}}_r$ using a GRU.

$$\begin{aligned} \bar{\mathbf{h}}_r^{(l)} &= \text{GRU}(\mathbf{w}'_l, \bar{\mathbf{h}}_r^{(l-1)});\ \bar{\mathbf{h}}_r^{(0)} = \mathbf{w}'_0, \\ \bar{\mathbf{h}}_r &= \bar{\mathbf{h}}_r^{(L-1)}. \end{aligned} \tag{2}$$

$l \in [1, L-1]$. $\bar{\mathbf{h}}_r$ contains semantic information from ERD, and therefore, we can view it as an LM-based relation representation. We substitute the relation representations of TKGF models with LM-based representations for semantics integration. Note that we fix the values of every $\bar{\mathbf{H}}_r$ to keep the LLM-provided semantic information intact. This is because we do not want the relation representations to lay excessive emphasis on the training data where ZS relations never appear. We want the models to maximally benefit from the semantic information for better generalization power. The textual descriptions of the relations with close meanings will show similar semantics. Since for each relation r, $\bar{\mathbf{H}}_r$ is generated based on r's ERD, the relations with close meanings will naturally lead to highly correlated text representations, building connections on top of the natural language space regardless of the observed TKG data.

3.2 Relation History Learner

As the relationship between two entities evolves through time, it follows certain temporal patterns. For example, the fact (*China*, *Sign formal agreement*, *Nicaragua*, 2022-01-10) happens after

(*China*, *Grant diplomatic recognition*, *Nicaragua*, 2022-01-04), implying that an agreement will be signed after showing diplomatic recognition. These temporal patterns are entity-agnostic and can reflect the dynamic relationship between any two entities over time. To this end, we develop RHL, aiming to capture such patterns. Assume we have a training fact (s, r, o, t), we search for the historical facts $\mathcal{G}_{s,o}^{<t}$ containing s and o before t, and group these facts according to their timestamps, i.e., $\mathcal{G}_{s,o}^{<t} = \{\mathcal{G}_{s,o}^{0}, ..., \mathcal{G}_{s,o}^{t-1}\}$. The searched facts with the same timestamp are put into the same group. For each group, we pick out the relations of all its facts and form a relation set, e.g., $\mathcal{R}_{s,o}^{0}$ is derived from $\mathcal{G}_{s,o}^{0}$. s and o's relationship at t_i $(t_i \in [0, t-1])$ is computed with an aggregator

$$\mathbf{h}_{s,o}^{t_i} = \textstyle\sum_m a_m \bar{\mathbf{h}}_{r_m};\ a_m = \mathrm{softmax}(\bar{\mathbf{h}}_{r_m}^{\top}\mathrm{MLP}_{\mathrm{agg}}(\bar{\mathbf{h}}_r)). \quad (3)$$

$r_m \in \mathcal{R}_{s,o}^{t_i}$ denotes a relation bridging s and o at t_i. If $\mathcal{R}_{s,o}^{t_i} = \emptyset$, we set $\mathbf{h}_{s,o}^{t_i}$ to a dummy embedding $\mathbf{h}_{\mathrm{dum}}$. To capture the historical relation dynamics, we use another GRU, i.e., $\mathrm{GRU}_{\mathrm{RHL}}$.

$$\begin{aligned} &\mathbf{h}_{\mathrm{hist}}^{t_i} = \mathrm{GRU}_{\mathrm{RHL}}(\mathbf{h}_{s,o}^{t_i}, \mathbf{h}_{\mathrm{hist}}^{t_i-1});\ \mathbf{h}_{\mathrm{hist}}^{0} = \mathbf{h}_{s,o}^{0}, \\ &\mathbf{h}_{\mathrm{hist}} = \mathbf{h}_{\mathrm{hist}}^{t-1}. \end{aligned} \quad (4)$$

$\mathbf{h}_{\mathrm{hist}}$ is taken as the encoded relation history until $t-1$. Note that during evaluation, TKGF asks models to predict the missing object of each LP query $(s_q, r_q, ?, t_q)$, which means we do not know which two entities should be used for historical fact searching[3]. To solve this problem, during training, we train another history prediction network (HPN) that aims to directly infer the relation history given the training fact relation r.

$$\tilde{\mathbf{h}}_{\mathrm{hist}} = \alpha \mathrm{MLP}_{\mathrm{hist}}(\bar{\mathbf{h}}_r) + \bar{\mathbf{h}}_r. \quad (5)$$

Here, α is a hyperparameter scalar and $\mathrm{MLP}_{\mathrm{hist}}$ is an MLP. $\tilde{\mathbf{h}}_{\mathrm{hist}}$ is the predicted relation history given r. Since we want $\tilde{\mathbf{h}}_{\mathrm{hist}}$ to represent the ground truth relation history, we use a mean square error (MSE) loss to constrain it to be close to $\mathbf{h}_{\mathrm{hist}}$.

$$\mathcal{L}_{\mathrm{hist}} = \mathrm{MSE}(\tilde{\mathbf{h}}_{\mathrm{hist}}, \mathbf{h}_{\mathrm{hist}}). \quad (6)$$

In this way, during evaluation, we can directly use Eq. 5 to generate a meaningful $\tilde{\mathbf{h}}_{\mathrm{hist}}$ for further computation. Given $\tilde{\mathbf{h}}_{\mathrm{hist}}$, we do one more step in $\mathrm{GRU}_{\mathrm{RHL}}$ to capture the r-related relation pattern.

$$\mathbf{h}_{\mathrm{pat}} = \mathrm{GRU}_{\mathrm{RHL}}(\bar{\mathbf{h}}_r, \tilde{\mathbf{h}}_{\mathrm{hist}}). \quad (7)$$

[3] We can indeed couple s_q with every candidate entity $e \in \mathcal{E}$ and search for their historical facts. But it requires huge computational resources and greatly harms model's scalability.

$\mathbf{h}_{\mathrm{pat}}$ can be viewed as a hidden representation containing comprehensive information of temporal relation patterns. Inspired by TuckER (Balazevic et al., 2019), we compute an RHL-based score for the training target (s, r, o, t) as

$$\phi((s, r, o, t)) = \mathcal{W} \times_1 \mathbf{h}_{(s,t)} \times_2 \mathbf{h}_{\mathrm{pat}} \times_3 \mathbf{h}_{(o,t)}, \quad (8)$$

where $\mathcal{W} \in \mathbb{R}^{d\times d\times d}$ is a learnable core tensor and $\times_1, \times_2, \times_3$ are three operators indicating the tensor product in three different modes (details in (Balazevic et al., 2019)). $\mathbf{h}_{(s,t)}$ and $\mathbf{h}_{(o,t)}$ are the time-aware entity representations of s and o computed by TKGF model, respectively. RHL-based score can be viewed as measuring how much two entities match the relation pattern generated by the relation history. We couple this score with the score computed by the original TKGF model $\phi'((s, r, o, t))$ and use the total score for LP.

$$\phi_{\mathrm{total}}((s, r, o, t)) = \phi'((s, r, o, t)) + \gamma\phi((s, r, o, t)). \quad (9)$$

γ is a hyperparameter. RHL enables models to make decisions by additionally considering the temporal relation patterns. Note that patterns are captured with LLM-empowered relation representations that contain rich semantic information. This guarantees RHL to generalize well to ZS relations. See App. H for explanations.

3.3 Parameter Learning and Evaluation

We let zrLLM be co-trained with TKGF model. Assume f is a TKGF model's loss function, e.g., cross-entropy, where f takes a fact quadruple's score computed by model's score function ϕ' and returns a loss for this fact. We input the quadruple score computed with Eq. 9 into f to let TKGF models better learn the parameters in RHL.

$$\mathcal{L}_{\mathrm{TKGF}} = \tfrac{1}{|\mathcal{G}_{\mathrm{train}}|} \textstyle\sum_{\lambda \in \mathcal{G}_{\mathrm{train}}} f(\phi_{\mathrm{total}}(\lambda)), \quad (10)$$

where λ denotes a fact quadruple $(s, r, o, t) \in \mathcal{G}_{\mathrm{train}}$ in the training set $\mathcal{G}_{\mathrm{train}}$. Besides, we also employ an additional binary cross-entropy loss $\mathcal{L}_{\mathrm{RHL}}$ directly on the RHL-based score

$$\begin{aligned} &\mathcal{L}_{\mathrm{RHL}} = \frac{1}{N}\sum_{\lambda}\sum_{e\in\mathcal{E}} \mathcal{L}_{\mathrm{RHL}}^{\lambda,e}; \\ &\mathcal{L}_{\mathrm{RHL}}^{\lambda,e} = -y_{\lambda'}\log(\phi(\lambda')) - (1-y_{\lambda'})\log(1-\phi(\lambda')). \end{aligned} \quad (11)$$

$N = |\mathcal{G}_{\mathrm{train}}| \times |\mathcal{E}|$. λ' is a perturbed fact by switching the object of λ to any $e \in \mathcal{E}$ and $y_{\lambda'}$ is its label. If $\lambda' \in \mathcal{G}_{\mathrm{train}}$, then $y_{\lambda'} = 1$, otherwise $y_{\lambda'} = 0$. Finally, we define the total loss $\mathcal{L}_{\mathrm{total}}$ as

$$\mathcal{L}_{\mathrm{total}} = \mathcal{L}_{\mathrm{TKGF}} + \mathcal{L}_{\mathrm{hist}} + \eta\mathcal{L}_{\mathrm{RHL}}. \quad (12)$$

η is a hyperparameter deciding $\mathcal{L}_{\text{RHL}}$'s magnitude. During evaluation, for each LP query $(s_q, r_q, ?, t_q)$, we compute scores $\{\phi_{\text{total}}((s_q, r_q, e, t_q))\} | e \in \mathcal{E}\}$ and take the entity with maximum score as the predicted answer. We provide algorithms of training and evaluation in App. A.

4 Experiments

We give details of our new ZS TKGF datasets in Sec. 4.1. In Sec. 4.3, we (1) do a comparative study to show how zrLLM improves TKGF models, (2) do ablation studies, (3) compare zrLLM with recent LM-enhanced TKGF models, and (4) do a case study to prove RHL's effectiveness.

Dataset	$\|\mathcal{E}\|$	$\|\mathcal{R}\|$	$\|\mathcal{T}_{\text{train}}\|$	$\|\mathcal{T}_{\text{eval}}\|$	$\|\mathcal{R}_{se}\|$	$\|\mathcal{R}_{un}\|$	$\|\mathcal{G}_{\text{train}}\|$	$\|\mathcal{G}_{\text{valid}}\|$	$\|\mathcal{G}_{\text{test}}\|$
ACLED-zero	621	23	20	11	9	14	2,118	931	146
ICEWS21-zero	18,205	253	181	62	130	123	247,764	77,195	1,395
ICEWS22-zero	999	248	181	62	93	155	171,013	47,784	1,956

Table 2: Dataset statistics. Dataset timestamps consist of both training and evaluation timestamps, i.e., $\mathcal{T} = \mathcal{T}_{\text{train}} \cup \mathcal{T}_{\text{eval}}$, $\mathcal{T}_{\text{train}} \cap \mathcal{T}_{\text{eval}} = \emptyset$, $\max(\mathcal{T}_{\text{train}}) < \min(\mathcal{T}_{\text{eval}})$.

4.1 Datasets for Zero-Shot TKGF

As discussed in Sec. 2.1, LM-enhanced TKGF models experience the risk of information leak. To exclude this concern, we construct new benchmark datasets on top of the facts happening after the publication date of T5-11B. We first construct two datasets ICEWS21-zero and ICEWS22-zero based on the Integrated Crisis Early Warning System (ICEWS) (Boschee et al., 2015) KB. ICEWS21-zero contains the facts happening from 2021-01-01 to 2021-08-31, while all the facts in ICEWS22-zero happen from 2022-01-01 to 2022-08-31. Besides, we also construct another dataset ACLED-zero based on a newer KB: The Armed Conflict Location & Event Data Project (ACLED) (Raleigh et al., 2010). Facts in ACLED-zero take place from 2023-08-01 to 2023-08-31. All the facts in all three datasets are based on social-political events described in English. Inspired by (Mirtaheri et al., 2021), our dataset construction process consists of the following steps. (1) For each dataset, we first collect all the facts within the time period of interest from the associated KB and then sort them in the temporal order. (2) Then we split the collected facts into two splits, where the first split contains the facts for model training and the second one has all the facts for evaluation. Any fact from the evaluation split happens later than the maximum timestamp of all the facts from the training split. Since we are studying ZS relations, we exclude the facts in the evaluation split whose entities do not appear in the training split, to avoid the potential impact of unseen entities. (3) We compute the frequencies of all relations in the evaluation split, and set a frequency threshold (40 for ACLED-zero and ICEWS21-zero, 60 for ICEWS22-zero). (4) We take each relation whose frequency is lower than the threshold as a ZS relation, and treat every fact containing it in the evaluation split as ZS evaluation data $\mathcal{G}_{\text{test}}$. We exclude the facts associated with ZS relations from the training split to ensure that models cannot see these relations during training, and take the rest as the training set $\mathcal{G}_{\text{train}}$. The rest of facts in the evaluation split are taken as the regular evaluation data $\mathcal{G}_{\text{valid}}$. We do validation over $\mathcal{G}_{\text{valid}}$ and test over $\mathcal{G}_{\text{test}}$ because we want to study how models perform over ZS relations when they reach the best performance over seen relations. See Table 2 and App. B for dataset statistics.

4.2 Experimental Setup

Training and Evaluation for Zero-Shot TKGF. All TKGF models are trained on $\mathcal{G}_{\text{train}}$. We take the model checkpoint achieving the best validation result on $\mathcal{G}_{\text{valid}}$ as the best model checkpoint, and report their test result on $\mathcal{G}_{\text{test}}$ to study the ZS inference ability. To keep ZS relations "always unseen" during the whole test process, we constrain all models to do LP only based on the training set as several popular TKGF methods, e.g., RE-GCN (Zhu et al., 2021). Some TKGF models, e.g., TiRGN (Li et al., 2022), allow using the ground truth TKG data until the LP query timestamp, including the facts in evaluation sets. This will violate the ZS setting because every unseen relation will occur multiple times in the evaluation data and is no longer ZS after models observe any fact of it. We prevent them from observing evaluation data to maintain the ZS setting. See App. C.5 for detailed explanation. Note that in our work, $\mathcal{G}_{\text{valid}}$ and $\mathcal{G}_{\text{test}}$ share the same time period. This is because we want to make sure that zrLLM can enhance ZS reasoning and simultaneously maintain TKGF models' performance on the facts with seen relations. Improving ZS inference ability at the cost of sacrificing too much performance over seen relations is undesired.

Baselines and Evaluation Metrics. We consider seven recent embedding-based TKGF methods as baselines, i.e., CyGNet (Zhu et al., 2021), TANGO-TuckER/Distmult (Han et al., 2021b), RE-GCN (Li

Datasets	ACLED-zero							ICEWS21-zero							ICEWS22-zero						
	Zero-Shot Relations			Seen Relations			Overall	Zero-Shot Relations			Seen Relations			Overall	Zero-Shot Relations			Seen Relations			Overall
Model	MRR	Hits@1	Hits@10	MRR	Hits@1	Hits@10	MRR	MRR	Hits@1	Hits@10	MRR	Hits@1	Hits@10	MRR	MRR	Hits@1	Hits@10	MRR	Hits@1	Hits@10	MRR
CyGNet	0.487	0.349	**0.791**	**0.751**	0.663	0.903	0.717	0.120	0.046	0.270	0.254	**0.165**	0.432	0.252	0.211	0.098	0.459	**0.315**	0.198	0.540	0.311
CyGNet+	**0.533**	**0.418**	0.753	**0.751**	**0.664**	**0.906**	**0.723**	**0.201**	**0.103**	**0.415**	**0.258**	0.162	**0.447**	**0.257**	**0.286**	**0.167**	**0.542**	**0.315**	**0.200**	**0.545**	**0.314**
TANGO-T	0.052	0.021	0.101	0.774	0.701	0.900	0.681	0.067	0.031	0.132	**0.283**	**0.190**	**0.470**	**0.279**	0.092	0.042	0.187	**0.363**	0.250	0.579	0.352
TANGO-T+	**0.525**	**0.393**	**0.764**	**0.775**	**0.702**	**0.901**	**0.743**	**0.216**	**0.125**	**0.395**	0.280	0.186	0.466	**0.279**	**0.326**	**0.198**	**0.578**	**0.363**	**0.251**	**0.585**	**0.362**
TANGO-D	0.021	0.003	0.049	**0.777**	**0.701**	**0.907**	0.679	0.012	0.005	0.023	0.266	**0.178**	0.439	0.261	0.011	0.002	0.018	**0.350**	0.227	0.569	0.337
TANGO-D+	**0.491**	**0.348**	**0.791**	0.760	0.678	0.901	**0.725**	**0.212**	**0.122**	**0.400**	**0.268**	0.175	**0.453**	**0.267**	**0.311**	**0.186**	**0.574**	**0.350**	**0.239**	**0.570**	**0.348**
RE-GCN	0.441	0.332	0.718	0.730	**0.653**	0.865	0.693	0.200	0.104	0.379	0.277	0.185	**0.456**	0.276	0.280	0.162	**0.616**	0.354	0.243	0.567	0.351
RE-GCN+	**0.529**	**0.393**	**0.784**	**0.731**	0.650	**0.876**	**0.705**	**0.214**	**0.117**	**0.406**	**0.280**	**0.188**	**0.456**	**0.279**	**0.324**	**0.194**	0.595	**0.357**	**0.244**	**0.573**	**0.356**
TiRGN	0.478	0.330	0.745	**0.754**	0.678	**0.886**	0.718	0.189	0.101	0.368	0.275	0.182	0.457	0.273	0.299	0.169	0.570	0.352	0.239	0.575	0.350
TiRGN+	**0.548**	**0.436**	**0.750**	**0.754**	0.679	0.885	**0.727**	**0.221**	**0.130**	**0.410**	**0.279**	**0.185**	**0.464**	**0.278**	**0.333**	**0.203**	**0.602**	**0.353**	**0.240**	**0.577**	**0.352**
RETIA	0.499	0.360	0.795	0.782	0.701	0.924	0.745	» 120 Hours Timeout							0.302	0.166	0.566	0.356	0.245	0.577	0.354
RETIA+	**0.557**	**0.408**	**0.814**	**0.783**	**0.703**	**0.925**	**0.754**								**0.331**	**0.201**	**0.597**	**0.358**	**0.247**	**0.578**	**0.357**
CENET	0.419	0.297	0.593	0.753	0.682	0.869	0.710	0.205	0.101	0.411	0.288	0.196	0.468	0.287	0.270	0.134	0.544	0.379	0.268	0.599	0.375
CENET+	**0.591**	**0.451**	**0.844**	**0.779**	**0.692**	**0.912**	**0.755**	**0.335**	**0.162**	**0.659**	**0.396**	**0.239**	**0.688**	**0.395**	**0.564**	**0.432**	**0.801**	**0.571**	**0.451**	**0.773**	**0.570**

Table 3: LP results. The best results between each baseline and its zrLLM-enhanced version (model name with "+") are marked in bold. TANGO-T and TANGO-D denote TANGO with TuckER (Balazevic et al., 2019) and Distmult (Yang et al., 2015), respectively. RETIA cannot be trained before 120 hours timeout on ICEWS21-zero. Complete results with Hits@3 are presented in App. E.

et al., 2021b), TiRGN (Li et al., 2022), CENET (Xu et al., 2023b) and RETIA (Liu et al., 2023). We couple them with zrLLM and show their improvement in ZS relational learning on TKGs (implementation details in App. C). We employ two evaluation metrics, i.e., mean reciprocal rank (MRR) and Hits@1/3/10. See App. F for detailed definitions. As suggested in (Gastinger et al., 2023), we use the time-aware filtering setting (Han et al., 2021a) for fairer evaluation.

4.3 Comparative Study and Further Analysis

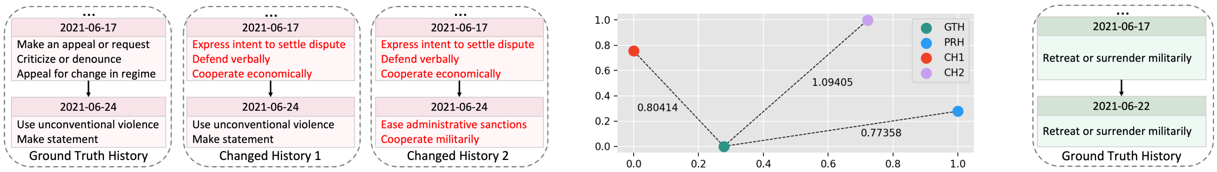
Comparative Study. We report the LP results of all baselines and their zrLLM-enhanced versions in Table 3. We have two findings: (1) zrLLM greatly helps TKGF models in forecasting the facts with unseen ZS relations. (2) In most cases, zrLLM even improves models in predicting the facts with seen relations. The zrLLM-enhanced models whose performance drops over seen relations still achieve better overall performance. These findings prove that embedding-based TKGF models benefit from the semantic information extracted from LLMs, especially when they are dealing with ZS relations.

Ablation Study. We conduct ablation studies from three aspects. (1) First, we directly input the dataset provided relation texts into T5-11B encoder, ignoring the relation explanations generated by GPT-3.5. From Table 4 (-ERD), we observe that in most cases, models' performance drops on the facts with both seen and ZS relations, which proves the usefulness of ERDs. (2) Next, we remove the RHL from all zrLLM-enhanced models. From Table 4 (-RHL), we find that all the considered TKGF models can benefit from RHL, especially CENET. See App. I for more discussion about CENET performance gain. (3) We switch T5-11B to T5-3B to see the impact of LM size on zrLLM. We observe from Table 4 that decreasing the size of T5 harms model performance. This proves that using larger scale LMs can provide semantic information of higher quality, and can be more beneficial to downstream TKGF (whether ZS or not).

Datasets	ACLED-zero MRR			ICEWS21-zero MRR			ICEWS22-zero MRR		
Model	Zero	Seen	Overall	Zero	Seen	Overall	Zero	Seen	Overall
CyGNet+	**0.533**	0.751	**0.723**	**0.201**	**0.258**	**0.257**	**0.286**	**0.315**	**0.314**
- ERD	0.502	0.748	0.716	0.198	0.252	0.251	0.250	0.314	0.311
- RHL	0.503	**0.752**	0.720	0.199	0.256	0.255	0.268	0.297	0.296
T5-3B	0.511	**0.752**	0.721	0.117	0.204	0.202	0.257	**0.315**	0.313
TANGO-T+	0.525	**0.775**	**0.743**	**0.216**	**0.280**	**0.279**	**0.326**	**0.363**	**0.362**
- ERD	0.533	0.772	0.741	0.214	**0.280**	**0.279**	0.320	0.362	0.360
- RHL	0.506	0.755	0.740	0.213	0.277	0.276	0.309	**0.363**	0.361
T5-3B	**0.544**	0.771	0.742	0.206	0.274	0.273	0.323	0.359	0.358
TANGO-D+	**0.491**	**0.760**	**0.725**	**0.212**	**0.268**	**0.267**	**0.311**	**0.350**	**0.348**
- ERD	**0.491**	0.702	0.675	0.205	0.267	0.266	0.285	0.328	0.326
- RHL	0.490	0.725	0.695	0.197	0.224	0.224	0.296	0.324	0.323
T5-3B	0.490	0.701	0.674	0.204	0.223	0.222	0.308	0.284	0.285
RE-GCN+	**0.529**	**0.731**	**0.705**	**0.214**	**0.280**	**0.279**	**0.324**	**0.357**	**0.356**
- ERD	0.489	0.730	0.699	0.211	0.277	0.276	0.294	0.354	0.352
- RHL	0.519	0.726	0.699	0.213	0.277	0.276	0.317	0.350	0.349
T5-3B	0.504	0.721	0.693	0.211	0.259	0.258	0.301	0.354	0.352
TiRGN+	**0.548**	**0.754**	**0.727**	**0.221**	**0.279**	**0.278**	**0.333**	**0.353**	**0.352**
- ERD	0.480	0.747	0.713	0.211	0.275	0.274	0.282	**0.353**	0.350
- RHL	0.515	0.752	0.721	0.215	0.277	0.276	0.320	0.350	0.349
T5-3B	0.498	0.749	0.717	0.208	0.271	0.270	0.325	0.345	0.344
RETIA+	**0.557**	**0.783**	**0.754**	» 120 Hours Timeout			**0.331**	**0.358**	**0.357**
- ERD	0.519	0.777	0.744				0.292	0.354	0.352
- RHL	0.529	0.782	0.749				0.318	0.357	0.355
T5-3B	0.512	0.776	0.742				0.330	0.353	0.352
CENET+	**0.591**	**0.779**	**0.755**	**0.335**	**0.396**	**0.395**	**0.564**	**0.571**	**0.570**
- ERD	0.526	0.737	0.710	0.321	0.374	0.373	0.542	0.570	0.568
- RHL	0.445	0.754	0.714	0.232	0.290	0.289	0.295	0.370	0.367
T5-3B	0.568	0.736	0.714	0.303	0.330	0.329	0.550	0.555	0.554

Table 4: Ablation study (complete results in App. F).

Compare with Previous LM-Enhanced Model. We benchmark two recent LM-enhanced TKGF models PPT (Xu et al., 2023a) and ICL + GPT-NeoX-20B (Lee et al., 2023; Black et al., 2022) (Table 5). PPT finetunes BERT for TKGF. We find that although PPT achieves strong ZS results, it is beaten by several zrLLM-enhanced models. This proves that aligning language space to TKGF is helpful for ZS relational learning and LMs with



(a) (b) (c)

Figure 3: (a) Ground truth and changed relation histories between *United States* and *African Union*. Changed relations are marked in red. Only the histories nearest to 2021-07-03 are shown. (b) t-SNE of encoded GTH, CH1, CH2 (computed with Eq. 4), and predicted history PRH. Numbers beside dashed lines denote point distances (L2 norm). (c) Ground truth relation histories between *United States* and *Afghanistan*.

larger size can be more contributive. ICL shows inferior results. This proves that without finetuning or alignment, LLMs are unable to optimally solve TKGF. zrLLM not only benefits from a large LM but also enables efficient alignment from language to TKG embedding space, which leads to superior performance. See App. G for further discussion.

Datasets	ACLED-zero MRR			ICEWS21-zero MRR			ICEWS22-zero MRR		
Model	Zero	Seen	Overall	Zero	Seen	Overall	Zero	Seen	Overall
PPT	0.532	0.782	0.748	0.212	0.269	0.268	0.323	0.332	0.331
ICL	0.537	0.736	0.709	0.156	0.178	0.177	0.255	0.229	0.230

Table 5: PPT and ICL performance. Implementation details and complete results in App. C.3 and G.

Case Study of RHL We do a case study to show: (1) RHL's HPN is able to capture ground truth relation history (GTH). (2) By capturing temporal relation patterns, RHL helps for better ZS TKGF. We ask zrLLM-enhanced CENET to predict the missing object of the test query $q = (s_q, r_q, ?, t_q) =$ (*United States*, *Reduce or stop military assistance*, ?, 2021-07-03) (answer is o_q = *African Union*) taken from ICEWS21-zero. The GTH of s_q and o_q (Fig. 3a, left) shows a pattern indicating their recent worsening relationship. It can serve as a clue in LP over q because it can be viewed as a "cause" to the query relation r_q which also implies a negative relationship. In other words, the entities with a worsening historical relationship are more likely to be connected with a relation showing their bad relationship currently. Since RHL uses HPN to infer GTH during test, we wish to study whether HPN can achieve reasonable inference to support LP. Based on GTH, we first change all three relations on 2021-06-17 to randomly sampled positive relations seen in the training data and form a changed history 1 (CH1, Fig. 3a, middle). Then we further modify the relations on 2021-06-24 in the same way and form a changed history 2 (CH2, Fig. 3a, right). We use Eq. 4 to encode GTH, CH1, CH2, and visualize them together with the predicted history (PRH) computed with HPN by using t-SNE (van der Maaten and Hinton, 2008) in Fig. 3b. We find that PRH is the closest to GTH and CH1 is closer than CH2 to GTH. The reason why CH2 is much farther from GTH is that CH2 changes more negative relations to positive, greatly changing the semantic meaning stored in GTH. CH1 only introduces changes on 2021-06-17, making it less deviated from GTH. HPN takes the r_q and can keep PRH close to GTH, making zrLLM able to maximally capture the temporal patterns indicated by GTH, while preventing the scalability problem incurred by searching relation histories of all candidate entities. By using RHL, the zrLLM-enhanced CENET can correctly predict o_q, while the model without RHL takes o' = *Afghanistan* as the predicted answer. We present the nearest GTH between s_q and o' in Fig. 3c and find that it indicates a positive relationship which is unlikely to cause r_q right after. During training, RHL learns patterns and matches entity pairs with them (Eq. 8). This enables RHL to exclude the entities that do not fit into the learned patterns from the answer set and make more accurate predictions.

5 Conclusion

We study zero-shot relational learning in TKGF and design an LLM-empowered approach, i.e., zrLLM. zrLLM extracts the semantic information of KG relations from LLMs and introduces it into TKG representation learning. It also uses an RHL module to capture the temporal relation patterns for better reasoning. We couple zrLLM with several embedding-based TKGF models and find that zrLLM provides huge help in forecasting the facts with zero-shot relations, and moreover, it maintains models' performance over seen relations.

6 Limitations

Our limitations can be summarized as follows. First, zrLLM is developed only for enhancing embedding-based TKG forecasting methods. It is not directly applicable to the rule-based methods, e.g., TLogic. Besides, relation history learner inevitably increases model's training and evaluation time since relation patterns are learned with GRUs where recurrent computations are performed along the time axis. More GPU memory is also required for storing relation histories. This hinders the efficiency of zrLLM-enhanced models compared with the original baselines. In the future, we will explore how to generalize our proposed method to rule-based models and try to improve model efficiency. We will also try to experiment zrLLM on more TKG forecasting methods and study whether we can benefit more of them.

References

Ivana Balazevic, Carl Allen, and Timothy M. Hospedales. 2019. Tucker: Tensor factorization for knowledge graph completion. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 5184–5193. Association for Computational Linguistics.

Sid Black, Stella Biderman, Eric Hallahan, Quentin Anthony, Leo Gao, Laurence Golding, Horace He, Connor Leahy, Kyle McDonell, Jason Phang, Michael Pieler, USVSN Sai Prashanth, Shivanshu Purohit, Laria Reynolds, Jonathan Tow, Ben Wang, and Samuel Weinbach. 2022. Gpt-neox-20b: An open-source autoregressive language model. *CoRR*, abs/2204.06745.

Elizabeth Boschee, Jennifer Lautenschlager, Sean O'Brien, Steve Shellman, James Starz, and Michael Ward. 2015. ICEWS Coded Event Data.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Zhongwu Chen, Chengjin Xu, Fenglong Su, Zhen Huang, and Yong Dou. 2023a. Incorporating structured sentences with time-enhanced BERT for fully-inductive temporal relation prediction. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2023, Taipei, Taiwan, July 23-27, 2023*, pages 889–899. ACM.

Zhongwu Chen, Chengjin Xu, Fenglong Su, Zhen Huang, and Yong Dou. 2023b. Meta-learning based knowledge extrapolation for temporal knowledge graph. In *Proceedings of the ACM Web Conference 2023, WWW 2023, Austin, TX, USA, 30 April 2023 - 4 May 2023*, pages 2433–2443. ACM.

Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1724–1734. ACL.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Zifeng Ding, Bailan He, Jingpei Wu, Yunpu Ma, Zhen Han, and Volker Tresp. 2023a. Learning meta-representations of one-shot relations for temporal knowledge graph link prediction. In *International Joint Conference on Neural Networks, IJCNN 2023, Gold Coast, Australia, June 18-23, 2023*, pages 1–10. IEEE.

Zifeng Ding, Jingpei Wu, Bailan He, Yunpu Ma, Zhen Han, and Volker Tresp. 2022. Few-shot inductive learning on temporal knowledge graphs using concept-aware information. In *4th Conference on Automated Knowledge Base Construction*.

Zifeng Ding, Jingpei Wu, Zongyue Li, Yunpu Ma, and Volker Tresp. 2023b. Improving few-shot inductive learning on temporal knowledge graphs using confidence-augmented reinforcement learning. In *Machine Learning and Knowledge Discovery in Databases: Research Track - European Conference, ECML PKDD 2023, Turin, Italy, September 18-22, 2023, Proceedings, Part III*, volume 14171 of *Lecture Notes in Computer Science*, pages 550–566. Springer.

Julia Gastinger, Timo Sztyler, Lokesh Sharma, Anett Schuelke, and Heiner Stuckenschmidt. 2023. Comparing apples and oranges? on the evaluation of methods for temporal knowledge graph forecasting. In *Machine Learning and Knowledge Discovery in Databases: Research Track - European Conference, ECML PKDD 2023, Turin, Italy, September 18-22, 2023, Proceedings, Part III*, volume 14171 of *Lecture Notes in Computer Science*, pages 533–549. Springer.

Zhen Han, Peng Chen, Yunpu Ma, and Volker Tresp. 2021a. Explainable subgraph reasoning for forecasting on temporal knowledge graphs. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Zhen Han, Zifeng Ding, Yunpu Ma, Yujia Gu, and Volker Tresp. 2021b. Learning neural ordinary equations for forecasting future links on temporal knowledge graphs. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 8352–8364. Association for Computational Linguistics.

Zhen Han, Ruotong Liao, Jindong Gu, Yao Zhang, Zifeng Ding, Yujia Gu, Heinz Koeppl, Hinrich Schütze, and Volker Tresp. 2023. ECOLA: Enhancing temporal knowledge embeddings with contextualized language representations. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 5433–5447, Toronto, Canada. Association for Computational Linguistics.

Woojeong Jin, Meng Qu, Xisen Jin, and Xiang Ren. 2020. Recurrent event network: Autoregressive structure inferenceover temporal knowledge graphs. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 6669–6683. Association for Computational Linguistics.

Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Dong-Ho Lee, Kian Ahrabian, Woojeong Jin, Fred Morstatter, and Jay Pujara. 2023. Temporal knowledge graph forecasting without knowledge using in-context learning. *CoRR*, abs/2305.10613.

Yujia Li, Shiliang Sun, and Jing Zhao. 2022. Tirgn: Time-guided recurrent graph network with local-global historical patterns for temporal knowledge graph reasoning. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI 2022, Vienna, Austria, 23-29 July 2022*, pages 2152–2158. ijcai.org.

Zixuan Li, Xiaolong Jin, Saiping Guan, Wei Li, Jiafeng Guo, Yuanzhuo Wang, and Xueqi Cheng. 2021a. Search from history and reason for future: Two-stage reasoning on temporal knowledge graphs. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 4732–4743. Association for Computational Linguistics.

Zixuan Li, Xiaolong Jin, Wei Li, Saiping Guan, Jiafeng Guo, Huawei Shen, Yuanzhuo Wang, and Xueqi Cheng. 2021b. Temporal knowledge graph reasoning based on evolutional representation learning. In *SIGIR '21: The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, Canada, July 11-15, 2021*, pages 408–417. ACM.

Ruotong Liao, Xu Jia, Yunpu Ma, and Volker Tresp. 2023. Gentkg: Generative forecasting on temporal knowledge graph. *CoRR*, abs/2310.07793.

Kangzheng Liu, Feng Zhao, Guandong Xu, Xianzhi Wang, and Hai Jin. 2023. RETIA: relation-entity twin-interact aggregation for temporal knowledge graph extrapolation. In *39th IEEE International Conference on Data Engineering, ICDE 2023, Anaheim, CA, USA, April 3-7, 2023*, pages 1761–1774. IEEE.

Yushan Liu, Yunpu Ma, Marcel Hildebrandt, Mitchell Joblin, and Volker Tresp. 2022. Tlogic: Temporal logical rules for explainable link forecasting on temporal knowledge graphs. In *Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022, Thirty-Fourth Conference on Innovative Applications of Artificial Intelligence, IAAI 2022, The Twelveth Symposium on Educational Advances in Artificial Intelligence, EAAI 2022 Virtual Event, February 22 - March 1, 2022*, pages 4120–4127. AAAI Press.

Ruixin Ma, Biao Mei, Yunlong Ma, Hongyan Zhang, Meihong Liu, and Liang Zhao. 2023. One-shot relational learning for extrapolation reasoning on temporal knowledge graphs. *Data Min. Knowl. Discov.*, 37(4):1591–1608.

Xin Mei, Libin Yang, Xiaoyan Cai, and Zuowei Jiang. 2022. An adaptive logical rule embedding model for inductive reasoning over temporal knowledge graphs. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*, pages 7304–7316. Association for Computational Linguistics.

Mehrnoosh Mirtaheri, Mohammad Rostami, Xiang Ren, Fred Morstatter, and Aram Galstyan. 2021. One-shot learning for temporal knowledge graphs. In *3rd Conference on Automated Knowledge Base Construction, AKBC 2021, Virtual, October 4-8, 2021*.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Z. Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing*

Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada, pages 8024–8035.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick S. H. Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander H. Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 2463–2473. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Clionadh Raleigh, rew Linke, Håvard Hegre, and Joakim Karlsen. 2010. Introducing acled: An armed conflict location and event dataset. *Journal of Peace Research*, 47(5):651–660.

Haohai Sun, Jialun Zhong, Yunpu Ma, Zhen Han, and Kun He. 2021. TimeTraveler: Reinforcement learning for temporal knowledge graph forecasting. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8306–8319, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurélien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. Llama: Open and efficient foundation language models. *CoRR*, abs/2302.13971.

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(86):2579–2605.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Ruijie Wang, Zheng Li, Dachun Sun, Shengzhong Liu, Jinning Li, Bing Yin, and Tarek F. Abdelzaher. 2022. Learning to sample and aggregate: Few-shot reasoning over temporal knowledge graphs. In *NeurIPS*.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, Ed H. Chi, Tatsunori Hashimoto, Oriol Vinyals, Percy Liang, Jeff Dean, and William Fedus. 2022. Emergent abilities of large language models. *Trans. Mach. Learn. Res.*, 2022.

Wenjie Xu, Ben Liu, Miao Peng, Xu Jia, and Min Peng. 2023a. Pre-trained language model with prompts for temporal knowledge graph completion. In *Findings of the Association for Computational Linguistics: ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 7790–7803. Association for Computational Linguistics.

Yi Xu, Junjie Ou, Hui Xu, and Luoyi Fu. 2023b. Temporal knowledge graph reasoning with historical contrastive learning. In *Thirty-Seventh AAAI Conference on Artificial Intelligence, AAAI 2023, Thirty-Fifth Conference on Innovative Applications of Artificial Intelligence, IAAI 2023, Thirteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2023, Washington, DC, USA, February 7-14, 2023*, pages 4765–4773. AAAI Press.

Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2015. Embedding entities and relations for learning and inference in knowledge bases. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Shuai Zhang, Yi Tay, Lina Yao, and Qi Liu. 2019. Quaternion knowledge graph embeddings. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 2731–2741.

Cunchao Zhu, Muhao Chen, Changjun Fan, Guangquan Cheng, and Yan Zhang. 2021. Learning from history: Modeling temporal knowledge graphs with sequential copy-generation networks. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 4732–4740. AAAI Press.

A Algorithm

We provide algorithms to show the whole process of using zrLLM to enhance TKGF models. First, zrLLM generates LLM-based relation representations by using GPT-3.5 and T5-11B (Algorithm 1). Then we train zrLLM jointly with TKGF baseline models (Algorithm 2). The trained models are then used for evaluation (Algorithm 3).

B Further Details of Zero-Shot Datasets

For each dataset, we provide the distribution of all zero-shot relations' frequncies in Fig. 4. We take the relations with lowest frequencies as zero-shot relations when we construct datasets, following

Algorithm 1: Generate LLM-based Relation Representations

Input: Relations $\mathcal{R}$, relation text of all relations provided by the TKG dataset $\text{TEXT}_{\mathcal{R}}$
for batch = *1: B* **do**
  Take a batch of n relations from $\mathcal{R}$
  Pick out their relation texts from $\text{TEXT}_{\mathcal{R}}$
  Write prompt with the relation texts `// Fig. 2`
  Input the prompt into GPT-3.5
  Extract the ERDs from the output of GPT-3.5
  Input the ERDs into T5-11B's encoder
  Store the output of T5-11B's encoder
return T5-encoded text representation $\bar{\mathbf{H}}_r$ for every $r \in \mathcal{R}$

Algorithm 2: Model Training with zrLLM

Input: Entities $\mathcal{E}$, relations $\mathcal{R}$, timestamps $\mathcal{T}$, T5-encoded text representations $\{\bar{\mathbf{H}}_r\}$ for $\mathcal{R}$, training set $\mathcal{G}_{\text{train}}$
Align $\{\bar{\mathbf{H}}_r\}$ to TKG embedding space and get $\{\bar{\mathbf{h}}_r\}$ `// Eq. 1, 2`
for epoch = *1: V* **do**
  for batch = *1: B* **do**
    Take a batch of training facts $\{(s, r, o, t)\} \in \mathcal{G}_{\text{train}}$
    Find the relation history of s and o before t for each (s, r, o, t)
    Encode relation history until $t-1$ `// Eq. 4`
    Compute the predicted history with HPN `// Eq. 5`
    Compute history-related MSE loss $\mathcal{L}_{\text{hist}}$ `// Eq. 6`
    Compute the representation of the r-related temporal relation pattern `// Eq. 7`
    Compute the RHL-based score `// Eq. 8`
    Input $\{\bar{\mathbf{h}}_r\}$ into TKGF baseline and compute LP score
    Compute total score for the training batch `// Eq. 9`
    Compute TKGF model loss $\mathcal{L}_{\text{TKGF}}$ `// Eq. 10`
    Compute RHL-based loss $\mathcal{L}_{\text{RHL}}$
    Compute total loss $\mathcal{L}_{\text{total}}$ `// Eq. 12`
    Update model parameters using gradient of $\bigtriangledown \mathcal{L}_{\text{total}}$
return trained zrLLM-enhanced TKGF model

Algorithm 3: Model Evaluation with zrLLM

Input: Entities $\mathcal{E}$, relations $\mathcal{R}$, timestamps $\mathcal{T}$, LLM-based relation representations $\{\bar{\mathbf{h}}_r\}$ for $\mathcal{R}$, training set $\mathcal{G}_{\text{train}}$, validation set $\mathcal{G}_{\text{valid}}$, test set $\mathcal{G}_{\text{test}}$
if *evaluation set is* $\mathcal{G}_{valid}$ **then**
  $\mathcal{G}_{\text{eval}} = \mathcal{G}_{\text{valid}}$
else
  $\mathcal{G}_{\text{eval}} = \mathcal{G}_{\text{test}}$
for batch = *1: B* **do**
  Take a batch of evaluation facts $\{(s_q, r_q, o_q, t_q)\} \in \mathcal{G}_{\text{eval}}$
  Derive LP queries $\{(s_q, r_q, ?, t_q)\}$
  Input $\{r_q\}$ into HPN and compute the predicted history `// Eq. 5`
  Compute the representation of the r_q-related temporal relation pattern for each LP query `// Eq. 7`
  Compute the RHL-based score of each candidate entity $e \in \mathcal{E}$ for each LP query `// Eq. 8`
  Input $\{\bar{\mathbf{h}}_r\}$ into TKGF baseline and compute LP score of each candidate entity $e \in \mathcal{E}$ for each LP query
  Compute total score of each candidate entity $e \in \mathcal{E}$ for each LP query in the batch `// Eq. 9`
  Rank candidate entities $\mathcal{E}$ with their total scores in the descending order
  Compute and record the rank of the ground truth missing entity o_q for each LP query
Compute MRR and Hits@1/3/10
return MRR and Hits@1/3/10

previous few-shot relational TKG learning frameworks, e.g., OAT (Mirtaheri et al., 2021) and MOST (Ding et al., 2023a). The proportion of zero-shot relations for each dataset is high. 14 out of 23; 123 out of 253; 155 out of 248 relations in ACLED-zero; ICEWS21-zero; ICEWS22-zero are zero-shot relations. This ensures the diversity of relation types in test sets.

C Implementation Details

All experiments are implemented with PyTorch (Paszke et al., 2019) on a server equipped with an AMD EPYC 7513 32-Core Processor and a single NVIDIA A40 with 48GB memory. All the experimental results are the average of three runs with different random seeds.

C.1 Baseline Implementation Details

The details of each TKGF baseline is as follows.

- **CyGNet.** We use the official code of CyGNet[4]. We search hyperparameters of baseline CyGNet following Table 6. The best hyperparameters are marked as bold. For each dataset, we do 4 trials to try different hyperparameter settings. We run 5 epochs for each trail and take the one with the best validation result as the best hyperparameter setting.

Dataset **Hyperparameter**	**ACLED-zero** CyGNet	**ICEWS21-zero** CyGNet	**ICEWS22-zero** CyGNet
Embedding Size	{100, **200**}	{100, **200**}	{100, **200**}
Alpha (Eq. 9 in (Zhu et al., 2021))	{**0.2**, 0.5}	{**0.2**, 0.5}	{**0.2**, 0.5}

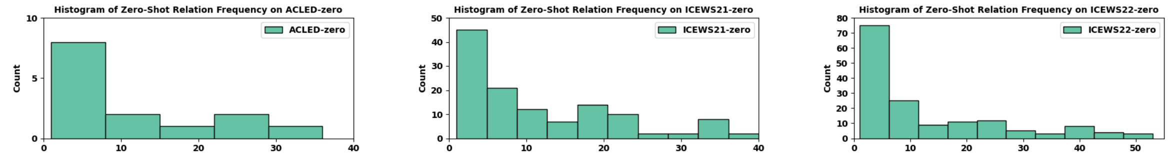
Table 6: CyGNet hyperparameter searching strategy.

- **TANGO-TuckER/Distmult.** We use the official code of TANGO[5]. We search hyperparameters of baseline TANGO-TuckER/Distmult following Table 7. The best hyperparameters are marked as bold. For each dataset, we do 6 (TANGO-TuckER) and 9 (TANGO-Distmult) trials to try different hyperparameter settings. We run 10 epochs for each trail and take the one with the best validation result as the best hyperparameter setting.

Dataset	**ACLED-zero**		**ICEWS21-zero**		**ICEWS22-zero**	
Hyperparameter	TuckER	Distmult	TuckER	Distmult	TuckER	Distmult
Embedding Size	{100, **200**}	{100, 200, **300**}	{**100**, 200}	{**100**, 200, 300}	{100, **200**}	{100, **200**, 300}
History Length	{**4**, 6, 10}	{**4**, 6, 10}	{**4**, 6, 10}	{**4**, 6, 10}	{**4**, 6, 10}	{**4**, 6, 10}

Table 7: TANGO hyperparameter searching strategy.

[4]https://github.com/CunchaoZ/CyGNet
[5]https://github.com/TemporalKGTeam/TANGO



(a) ACLED-zero. (b) ICEWS21-zero. (c) ICEWS22-zero.

Figure 4: Zero-shot Relation frequency on all zero-shot TKGF datasets. Horizontal axis denotes the appearance times, i.e., frequency. Vertical axis denotes the number of relations.

- **RE-GCN.** We use the official code of RE-GCN[6]. We search hyperparameters of baseline RE-GCN following Table 8. The best hyperparameters are marked as bold. For each dataset, we do 4 trials to try different hyperparameter settings. We run 10 epochs for each trail and take the one with the best validation result as the best hyperparameter setting.

Dataset Hyperparameter	ACLED-zero RE-GCN	ICEWS21-zero RE-GCN	ICEWS22-zero RE-GCN
Embedding Size	{100, **200**}	{**100**, 200}	{100, **200**}
History Length	{**3**, 9}	{3, **9**}	{3, **9**}

Table 8: RE-GCN hyperparameter searching strategy.

- **TiRGN.** We use the official code of TiRGN[7]. We search hyperparameters of baseline TiRGN following Table 9. The best hyperparameters are marked as bold. For each dataset, we do 12 trials to try different hyperparameter settings. We run 10 epochs for each trail and take the one with the best validation result as the best hyperparameter setting.

Dataset Hyperparameter	ACLED-zero TiRGN	ICEWS21-zero TiRGN	ICEWS22-zero TiRGN
Embedding Size	{100, **200**}	{**100**, 200}	{100, **200**}
History Length	{**3**, 9}	{3, **9**}	{3, **9**}
Alpha (Eq. 11 in (Li et al., 2022))	{**0.3**, 0.5, 0.7}	{**0.3**, 0.5, 0.7}	{**0.3**, 0.5, 0.7}

Table 9: TiRGN hyperparameter searching strategy.

- **RETIA.** We use the official code of RETIA[8]. We search hyperparameters of baseline RETIA following Table 10. The best hyperparameters are marked as bold. For each dataset, we do 4 trials to try different hyperparameter settings. We run 10 epochs for each trail and take the one with the best validation result as the best hyperparameter setting.

Dataset Hyperparameter	ACLED-zero RETIA	ICEWS21-zero RETIA	ICEWS22-zero RETIA
Embedding Size	{100, **200**}	{**100**, 200}	{100, **200**}
History Length	{**3**, 9}	{3, **9**}	{3, **9**}

Table 10: RETIA hyperparameter searching strategy.

- **CENET.** We use the official code of CENET[9]. We search hyperparameters of baseline CENET following Table 11. The best hyperparameters are marked as bold. For each dataset, we do 4 trials to try different hyperparameter settings. We run 5 epochs for each trail and take the one with the best validation result as the best hyperparameter setting.

Dataset Hyperparameter	ACLED-zero CENET	ICEWS21-zero CENET	ICEWS22-zero CENET
Embedding Size	{**100**, 200}	{100, **200**}	{100, **200**}
Mask Strategy	{**soft**, hard}	{**soft**, hard}	{**soft**, hard}

Table 11: CENET hyperparameter searching strategy.

The hyperparameters not discussed above follow the settings reported in the original papers.

C.2 zrLLM Implementation Details

We fix the hyperparameters searched from the baselines and additionally search zrLLM-specifc hyperparameters for zrLLM-enhanced models. The hyperparameter searching strategy and the best hyperparameter settings regarding the zrLLM-enahnced baselines are reported in Table 12. Note that γ can be either a learnable parameter or a fixed scalar. When γ is not fixed, γ Value means the initialized parameter value during training. For each zrLLM-enhanced model, in each dataset, we do 36 trials to try different hyperparameter settings. We run 7 epochs for each trail and take the one with the best validation result as the best hyperparameter setting.

[6]https://github.com/Lee-zix/RE-GCN
[7]https://github.com/Liyyy2122/TiRGN
[8]https://github.com/CGCL-codes/RETIA
[9]https://github.com/xyjigsaw/CENET

Dataset	**ACLED-zero**				**ICEWS21-zero**				**ICEWS22-zero**			
Model	α	γ Type	γ Value	η	α	γ Type	γ Value	η	α	γ Type	γ Value	η
CyGNet+	{1, **0.1**}	{**Fixed**, Unfixed}	{**1**, 0.01, 0.001}	{1.2, **1**}	{1, **0.1**}	{**Fixed**, Unfixed}	{1, 0.01, **0.001**}	{1.2, **1**}	{1, **0.1**}	{**Fixed**, Unfixed}	{1, 0.01, **0.001**}	{1.2, **1**}
TANGO-T+	{1, **0.1**}	{Fixed, **Unfixed**}	{**1**, 0.01, 0.001}	{**1.2**, 1}	{1, **0.1**}	{Fixed, **Unfixed**}	{**1**, 0.01, 0.001}	{**1.2**, 1}	{1, **0.1**}	{**Fixed**, Unfixed}	{**1**, 0.01, 0.001}	{**1.2**, 1}
TANGO-D+	{1, **0.1**}	{**Fixed**, Unfixed}	{**1**, 0.01, 0.001}	{**1.2**, 1}	{1, **0.1**}	{Fixed, **Unfixed**}	{**1**, 0.01, 0.001}	{**1.2**, 1}	{1, **0.1**}	{Fixed, **Unfixed**}	{**1**, 0.01, 0.001}	{**1.2**, 1}
RE-GCN+	{1, **0.1**}	{**Fixed**, Unfixed}	{1, 0.01, **0.001**}	{1.2, **1**}	{1, **0.1**}	{**Fixed**, Unfixed}	{1, **0.01**, 0.001}	{1.2, **1**}	{1, **0.1**}	{**Fixed**, Unfixed}	{1, **0.01**, 0.001}	{1.2, **1**}
TiRGN+	{1, **0.1**}	{Fixed, **Unfixed**}	{1, 0.01, **0.001**}	{1.2, **1**}	{1, **0.1**}	{**Fixed**, Unfixed}	{1, **0.01**, 0.001}	{1.2, **1**}	{1, **0.1**}	{**Fixed**, Unfixed}	{1, **0.01**, 0.001}	{1.2, **1**}
RETIA+	{1, **0.1**}	{Fixed, **Unfixed**}	{1, **0.01**, 0.001}	{**2**, 1}	-	-	-	-	{1, **0.1**}	{**Fixed**, Unfixed}	{1, **0.01**, 0.001}	{**2**, 1}
CENET+	{1, **0.1**}	{Fixed, **Unfixed**}	{**1**, 0.01, 0.001}	{1.2, **1**}	{1, **0.1**}	{Fixed, **Unfixed**}	{**1**, 0.01, 0.001}	{1.2, **1**}	{1, **0.1**}	{Fixed, **Unfixed**}	{1, **0.01**, 0.001}	{1.2, **1**}

Table 12: zrLLM hyperparameter searching strategy. The best settings are marked as bold.

Dataset	**ACLED-zero**		**ICEWS21-zero**		**ICEWS22-zero**	
Model	Training Time (h)	GPU Memory (MB)	Training Time (h)	GPU Memory (MB)	Training Time (h)	GPU Memory (MB)
CyGNet+	0.03	2,216	17.87	7,470	4.80	9,574
TANGO-T+	0.05	2,716	8.64	34,186	2.82	20,120
TANGO-D+	0.11	3,064	10.88	34,034	0.70	19,250
RE-GCN+	0.06	1,587	14.70	26,420	3.85	19,168
TiRGN+	0.10	2,654	11.67	36,780	2.40	15,976
RETIA+	0.13	4,274	-	-	9.33	26,328
CENET+	0.03	1,429	48.94	6,750	12.54	5,639
PPT	0.47	7,654	84.68	9,078	59.35	7,678

Table 13: Computational resources required by zrLLM-enhanced models and PPT.

C.3 Implementation Details of PPT and ICL

We use the official code of PPT[10] and ICL[11]. For PPT, we use the default hyperparameter setting used for ICEWS14 when we implement it on all our new datasets. Since PPT only explores object entity prediction in its original implementation, we add the subject entity prediction part and report the overall result. We achieve subject prediction by first deriving the inverse relation texts for each relation in each TKG dataset, e.g., use *Inversed Reduce or stop military assistance* to represent the inverse relation of the relation *Reduce or stop military assistance*, and then turning each subject prediction query $(?, r_q, o_q, t_q)$ to an object prediction query $(o_q, r_q^{-1}, ?, t_q)$, where r_q^{-1} stands for the inverse relation of r_q. For ICL, we use the lexical-based prompt because we are dealing with zero-shot relations where text information is important. We also employ the unidirectional entity-focused history, which achieves best results on ICEWS14 as reported in ICL's original paper. We use the default history length of 20 for all datasets.

C.4 Computational Resource Usage

We report the computational resources for all zrLLM-enhanced models and PPT in Table 13. Training time denotes the period of time a model requires to reach its best validation performance. PPT requires extremely long time for sampling and thus has high time consumption. Note that zrLLM loads T5 to generate LM-based relation representations. This process takes a substantial amount of GPU memoery. However, in our work, we store the output of T5's encoder as saved parameters and use them in downstream zero-shot TKGF with any zrLLM-enhanced model. This prevents from high memory demand during model training and evaluation. We use Fig. 5 to illustrate the direct comparison among zrLLM-enhanced models and PPT regarding their required computational resources during training.

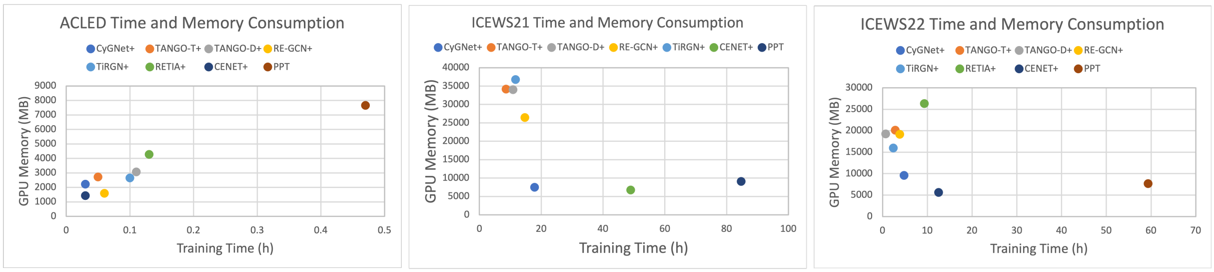
ICL loads GPT-NeoX-20B that requires huge memory consumption. We use two NVIDIA A40 for all its experiments. Since ICL does not require training, we only report its validation and test time here. For ACLED-zero, GPU memory usage is 90,846 MB. Validation time is 0.63 h and test time is 0.12 h. For ICEWS21-zero, GPU memory usage is 90,868 MB. Validation time is 35.48 h and test time is 0.82 h. For ICEWS22-zero, GPU memory usage is 91,458 MB. Validation time is 22.98 h and test time is 1.15 h.

C.5 Zero-Shot Evaluation Setting Explanation

To keep zero-shot relations "always unseen" during the whole evaluation process, we constrain all models to do LP only based on the training set. Among all TKGF models, TANGO, RE-GCN, TiRGN and RETIA use recurrent neural structures to model historical TKG information from a short sequence of timestamps prior to the prediction timestamp. We constrain them to only use the latest training data, i.e., from $t_{\text{train\_max}} - k$ to $t_{\text{train\_max}}$, to encode historical information during evaluation. k is the considered history length and $t_{\text{train\_max}} = \max(\mathcal{T}_{\text{train}})$

[10]https://github.com/JaySaligia/PPT

[11]https://github.com/usc-isi-i2/isi-tkg-icl



(a) ACLED-zero. (b) ICEWS21-zero. (c) ICEWS22-zero.

Figure 5: Computational resources required during training of zrLLM-enhanced models and PPT.

is the maximum timestamp in the training data. For CyGNet and CENET, they have originally met our restriction of not observing any ground truth evaluation data during evaluation, and thus can be directly implemented in our zero-shot setting. Another point worth noting is that RHL requires ground truth relation history. We restrict zrLLM to only capture the relation history across the whole training time period to prevent from exposing zero-shot relations during evaluation.

D Evaluation Metrics Details

We employ two evaluation metrics, i.e., mean reciprocal rank (MRR) and Hits@1/3/10. For every LP query q, we compute the rank θ_q of the ground truth missing entity. We define MRR as: $\frac{1}{|\mathcal{G}_{\text{test}}|}\sum_q \frac{1}{\theta_q}$ (the definition is similar for $\mathcal{G}_{\text{valid}}$). Hits@1/3/10 denote the proportions of the predicted links where ground truth missing entities are ranked as top 1, top3, top10, respectively. As explored and suggested in (Gastinger et al., 2023), we also use the time-aware filtering setting proposed in (Han et al., 2021a) for fairer evaluation.

E Complete Comparative Study Results

We report the complete results of comparative study in Table 14 and 15.

F Complete Ablation Study Results

We report the complete ablation study results in Table 16.

G Complete Results of Previous LM-Enhanced TKGF Model

We report the complete results of previous LM-enhanced TKGF models in Table 14 and 15.

H Further Discussion about RHL

In RHL, temporal relation patterns are captured by only using LLM-based relation representations. Since for all relations (whether zero-shot or not), their LLM-based representations contain semantic information extracted from the same LLM, the learned HPN can do reasonable relation history prediction even with an input of unseen zero-shot relation. If we learn hidden representations for each relation based on graph contexts (as most TKGF models do), zero-shot relations cannot be easily processed by HPN anymore. In this case, zero-shot relations will not have a meaningful representation without any observed associated fact, and therefore, HPN cannot detect its meaning and will fail to find reasonable relation history.

I CENET Performance gain with RHL

We find that RHL can greatly increase CENET's TKGF performance, whether on zero-shot or not. We think it is because RHL provides temporal relation patterns that helps CENET to better predict the entities that are highly-dependent on the historical TKG information. Assume we have an LP query $(s_q, r_q, ?, t_q)$, CENET computes for each entity a query-related historical dependency feature and a non-historical dependency feature. If the ground truth missing entity o_q is a historical entity (s_q, r_q, o_q appear frequently in the facts of previous TKG data), its historical dependency feature will be dominant in its entity representation. Meanwhile, CENET uses a binary classifier to recognize whether the missing object entity of each query exists in the set of historical entities. This process helps to greatly decrease the influence of non-historical entities during inference, making the prediction easier (because the non-historical entities are to great extent ignored when model makes decision and the number of potential candidate be-

Datasets	ICEWS21-zero									ICEWS22-zero									
	Zero-Shot Relations				Seen Relations				Overall	Zero-Shot Relations				Seen Relations				Overall
Model	MRR	Hits@1	Hits@3	Hits@10	MRR	Hits@1	Hits@3	Hits@10	MRR	MRR	Hits@1	Hits@3	Hits@10	MRR	Hits@1	Hits@3	Hits@10	MRR
CyGNet	0.120	0.046	0.130	0.270	0.254	**0.165**	0.293	0.432	0.252	0.211	0.098	0.240	0.459	**0.315**	0.198	**0.373**	0.540	0.311
CyGNet+	**0.201**	**0.103**	**0.226**	**0.415**	**0.258**	0.162	**0.294**	**0.447**	**0.257**	**0.286**	**0.167**	**0.324**	**0.542**	**0.315**	**0.200**	0.364	**0.545**	**0.314**
TANGO-T	0.067	0.031	0.069	0.132	**0.283**	**0.190**	**0.319**	**0.470**	**0.279**	0.092	0.042	0.100	0.187	**0.363**	0.250	0.407	0.579	0.352
TANGO-T+	**0.216**	**0.125**	**0.245**	**0.395**	0.280	0.186	0.313	0.466	**0.279**	**0.326**	**0.198**	**0.388**	**0.578**	**0.363**	**0.251**	**0.409**	**0.585**	**0.362**
TANGO-D	0.012	0.005	0.011	0.023	0.266	**0.178**	0.298	0.439	0.261	0.011	0.002	0.007	0.018	**0.350**	0.227	**0.394**	0.569	0.337
TANGO-D+	**0.212**	**0.122**	**0.237**	**0.400**	**0.268**	0.175	**0.303**	**0.453**	**0.267**	**0.311**	**0.186**	**0.374**	**0.574**	**0.350**	**0.239**	0.393	**0.570**	**0.348**
RE-GCN	0.200	0.104	0.231	0.379	0.277	0.185	0.309	**0.456**	0.276	0.280	0.162	0.321	**0.616**	0.354	0.243	**0.398**	0.567	0.351
RE-GCN+	**0.214**	**0.117**	**0.246**	**0.406**	**0.280**	**0.188**	**0.314**	**0.456**	**0.279**	**0.324**	**0.194**	**0.376**	0.595	**0.357**	**0.244**	**0.398**	**0.573**	**0.356**
TiRGN	0.189	0.101	0.209	0.368	0.275	0.182	0.308	0.457	0.273	0.299	0.169	0.358	0.570	0.352	0.239	0.399	0.575	0.350
TiRGN+	**0.221**	**0.130**	**0.246**	**0.410**	**0.279**	**0.185**	**0.323**	**0.464**	**0.278**	**0.333**	**0.203**	**0.383**	**0.602**	**0.353**	**0.240**	**0.400**	**0.577**	**0.352**
RETIA	» 120 Hours Timeout									0.302	0.166	0.349	0.566	0.356	0.245	0.401	0.577	0.354
RETIA+										**0.331**	**0.201**	**0.384**	**0.597**	**0.358**	**0.247**	**0.402**	**0.578**	**0.357**
CENET	0.205	0.101	0.232	0.411	0.288	0.196	0.318	0.468	0.287	0.270	0.134	0.318	0.544	0.379	0.268	0.423	0.599	0.375
CENET+	**0.335**	**0.162**	**0.455**	**0.659**	**0.396**	**0.239**	**0.502**	**0.688**	**0.395**	**0.564**	**0.432**	**0.649**	**0.801**	**0.571**	**0.451**	**0.651**	**0.773**	**0.571**
PPT	0.212	0.120	0.240	0.403	0.269	0.172	0.304	0.462	0.268	0.323	0.191	0.376	0.598	0.332	0.219	0.377	0.556	0.331
ICL	0.156	0.096	0.180	0.300	0.178	0.120	0.206	0.308	0.177	0.255	0.162	0.303	0.460	0.229	0.158	0.264	0.393	0.230

Table 14: Complete LP results on ICEWS21-zero and ICEWS22-zero. We also report PPT and ICL's performance.

Datasets	ACLED-zero								
	Zero-Shot Relations				Seen Relations				Overall
Model	MRR	Hits@1	Hits@3	Hits@10	MRR	Hits@1	Hits@3	Hits@10	MRR
CyGNet	0.487	0.349	0.565	**0.791**	**0.751**	0.663	**0.827**	0.903	0.717
CyGNet+	**0.533**	**0.418**	**0.592**	0.753	**0.751**	**0.664**	0.821	**0.906**	**0.723**
TANGO-T	0.052	0.021	0.049	0.101	0.774	0.701	0.826	0.900	0.681
TANGO-T+	**0.525**	**0.393**	**0.606**	**0.746**	**0.775**	**0.702**	**0.827**	**0.901**	**0.743**
TANGO-D	0.021	0.003	0.017	0.049	**0.777**	**0.701**	**0.833**	**0.907**	0.679
TANGO-D+	**0.491**	**0.348**	**0.560**	**0.791**	0.760	0.678	0.818	0.901	**0.725**
RE-GCN	0.441	0.332	0.466	0.718	0.730	**0.653**	0.783	0.865	0.693
RE-GCN+	**0.529**	**0.393**	**0.612**	**0.784**	**0.731**	0.650	**0.789**	**0.876**	**0.705**
TiRGN	0.478	0.330	0.572	0.745	**0.754**	0.678	0.806	**0.886**	0.718
TiRGN+	**0.548**	**0.436**	**0.607**	**0.750**	**0.754**	**0.679**	**0.807**	0.885	**0.727**
RETIA	0.499	0.360	0.586	0.795	0.782	0.701	**0.844**	0.924	0.745
RETIA+	**0.557**	**0.408**	**0.676**	**0.814**	**0.783**	**0.703**	0.842	**0.925**	**0.754**
CENET	0.419	0.297	0.522	0.593	0.753	0.682	0.808	0.869	0.710
CENET+	**0.591**	**0.451**	**0.687**	**0.844**	**0.779**	**0.692**	**0.849**	**0.912**	**0.755**
PPT	0.532	0.388	0.651	0.787	0.782	0.693	0.842	0.942	0.748
ICL	0.537	0.452	0.620	0.661	0.736	0.668	0.794	0.853	0.709

Table 15: Complete LP results on ACLED-zero. We also report PPT and ICL's performance.

comes smaller). Historical entities have abundant relation histories associated with the query subject, and therefore, the temporal relation patterns captured by RHL are highly informative. When zrLLM computes RHL-based score, model can greatly benefit from the relation patterns and better learn the historical dependency features of historical entities. As a result, RHL enables CENET to achieve huge performance gain.

J Related Work Details

Traditional TKG Forecasting Methods. As discussed in Sec. 1, traditional TKGF methods are trained to forecast the facts containing the KG relations (and entities) seen in the training data, regardless of the case where zero-shot relations (or entities) appear as new knowledge arrives[12]. These methods can be categorized into two types: embedding-based and rule-based. Embedding-based methods learn hidden representations of KG relations and entities (some also learn time representations), and perform link forecasting by inputting learned representations into a score function for computing scores of fact quadruples. Most existing embedding-based methods, e.g., (Jin et al., 2020; Han et al., 2021b; Li et al., 2021b, 2022; Liu et al., 2023), learn evolutional entity and relation representations by jointly employing graph neural networks (Kipf and Welling, 2017) and recurrent neural structures, e.g., GRU (Cho et al., 2014). Historical TKG information are recurrently encoded by the models to produce the temporal sequence-aware evolutional representations for future prediction. Some other approaches (Han et al., 2021a; Sun et al., 2021; Li et al., 2021a) start from each LP query[13] and traverse the temporal history in a TKG to search for the prediction answer. Apart from them, CyGNet (Zhu et al., 2021) achieves forecasting purely based on the appearance of historical facts. Another recent work CENET (Xu et al., 2023b) trains contrastive representations of LP queries to identify highly correlated entities in either historical or non-historical facts. Compared with the rapid advancement in developing embedding-based TKGF methods, rule-based TKGF has still not been extensively explored. One popular rule-based TKGF method is TLogic (Liu et al., 2022). It extracts temporal logic rules from TKGs and uses a symbolic reasoning module for LP. Based on it, ALRE-IR (Mei et al., 2022) proposes an adaptive logical rule embedding model

[12] Some works of traditional TKGF methods, e.g., TANGO (Han et al., 2021b), have discussions about models' ability to reason over the facts regarding unseen entities. Note that this is not their main focus but an additional demonstration to show their models' inductive power, i.e., these models are not designed for inductive learning on TKGs.

[13] A TKG LP query is denoted as $(s, r, ?, t)$ (object prediction query) or $(?, r, o, t)$ (subject prediction query).

Datasets	ACLED-zero							ICEWS21-zero							ICEWS22-zero						
	Zero-Shot Relations			Seen Relations			Overall	Zero-Shot Relations			Seen Relations			Overall	Zero-Shot Relations			Seen Relations			Overall
Model	MRR	Hits@1	Hits@10	MRR	Hits@1	Hits@10	MRR	MRR	Hits@1	Hits@10	MRR	Hits@1	Hits@10	MRR	MRR	Hits@1	Hits@10	MRR	Hits@1	Hits@10	MRR
CyGNet+	**0.533**	**0.418**	**0.753**	0.751	**0.664**	**0.906**	**0.723**	**0.201**	**0.103**	**0.415**	**0.258**	**0.162**	**0.447**	**0.257**	**0.286**	**0.167**	**0.542**	**0.315**	0.200	0.545	**0.314**
- ERD	0.502	0.386	0.743	0.748	0.660	0.902	0.716	0.198	0.102	0.379	0.252	0.161	0.429	0.251	0.250	0.136	0.503	0.314	0.198	**0.546**	0.311
- RHL	0.503	0.356	0.751	**0.752**	0.663	0.901	0.720	0.199	0.100	0.398	0.256	0.159	0.445	0.255	0.268	0.144	0.536	0.297	0.181	0.531	0.296
T5-3B	0.511	0.414	0.684	**0.752**	0.663	0.905	0.721	0.117	0.068	0.186	0.204	0.127	0.348	0.202	0.257	0.135	0.521	**0.315**	**0.201**	0.540	0.313
TANGO-T+	0.525	0.393	0.764	**0.775**	0.702	**0.901**	**0.743**	**0.216**	**0.125**	0.395	**0.280**	0.186	0.466	**0.279**	**0.326**	**0.198**	**0.578**	**0.363**	**0.251**	**0.585**	**0.362**
- ERD	0.533	0.408	**0.770**	0.772	0.692	0.898	0.741	0.214	0.122	0.389	**0.280**	**0.187**	0.465	**0.279**	0.320	0.193	0.576	0.362	0.250	0.584	0.360
- RHL	0.506	0.374	0.749	0.755	**0.704**	**0.901**	0.740	0.213	0.118	**0.407**	0.277	0.181	**0.469**	0.276	0.309	0.190	0.574	**0.363**	0.250	0.584	0.361
T5-3B	**0.544**	**0.425**	0.769	0.771	0.697	0.896	0.742	0.206	0.119	0.375	0.274	0.182	0.454	0.273	0.323	0.193	0.576	0.359	0.246	0.579	0.358
TANGO-D+	**0.491**	0.348	**0.791**	**0.760**	**0.678**	**0.901**	**0.725**	**0.212**	**0.122**	**0.400**	**0.268**	**0.175**	**0.453**	**0.267**	**0.311**	**0.186**	0.574	**0.350**	**0.239**	**0.570**	**0.348**
- ERD	**0.491**	**0.350**	0.771	0.702	0.578	0.898	0.675	0.205	0.111	0.398	0.267	0.174	0.449	0.266	0.285	0.159	0.541	0.328	0.213	0.550	0.326
- RHL	0.490	0.344	0.772	0.725	0.628	0.890	0.695	0.197	0.107	0.390	0.224	0.132	0.412	0.224	0.296	0.175	0.552	0.324	0.212	0.547	0.323
T5-3B	0.490	0.341	0.786	0.701	0.576	0.897	0.674	0.204	0.109	0.393	0.223	0.131	0.408	0.222	0.308	0.177	**0.582**	0.284	0.173	0.510	0.285
RE-GCN+	**0.529**	0.393	**0.784**	**0.731**	**0.650**	**0.876**	**0.705**	**0.214**	0.117	**0.406**	**0.280**	**0.188**	**0.456**	**0.279**	**0.324**	**0.194**	**0.595**	**0.357**	**0.244**	**0.573**	**0.356**
- ERD	0.489	0.375	0.724	0.730	**0.650**	0.865	0.699	0.211	0.119	0.397	0.277	0.185	0.454	0.276	0.294	0.168	0.560	0.354	0.242	0.571	0.352
- RHL	0.519	**0.396**	0.757	0.726	0.646	0.836	0.699	0.213	0.119	0.405	0.277	0.185	0.455	0.276	0.317	0.184	0.589	0.350	0.241	0.562	0.349
T5-3B	0.504	0.361	0.767	0.721	0.638	0.864	0.693	0.211	**0.121**	0.384	0.259	0.171	0.427	0.258	0.301	0.174	0.577	0.354	0.243	0.570	0.352
TiRGN+	**0.548**	**0.436**	0.750	**0.754**	**0.679**	0.885	**0.727**	**0.221**	**0.130**	**0.410**	**0.279**	**0.185**	**0.463**	**0.278**	**0.333**	**0.203**	**0.602**	**0.353**	**0.240**	**0.577**	**0.352**
- ERD	0.480	0.387	0.673	0.747	0.669	0.882	0.713	0.211	0.120	0.387	0.275	0.181	0.460	0.274	0.282	0.157	0.544	**0.353**	**0.240**	0.576	0.350
- RHL	0.515	0.400	**0.753**	0.752	0.675	**0.887**	0.721	0.215	0.124	0.391	0.277	0.183	0.461	0.276	0.320	0.190	0.593	0.350	0.239	0.569	0.349
T5-3B	0.498	0.389	0.722	0.749	0.675	0.879	0.717	0.208	0.118	0.392	0.271	0.180	0.448	0.270	0.325	0.189	0.594	0.345	0.233	0.565	0.344
RETIA+	**0.557**	**0.408**	**0.814**	**0.783**	**0.703**	**0.925**	**0.754**								**0.331**	**0.201**	**0.597**	**0.358**	**0.247**	0.578	**0.357**
- ERD	0.519	0.391	0.765	0.777	0.692	0.917	0.744				» 120 Hours Timeout				0.292	0.163	0.562	0.354	0.242	0.576	0.352
- RHL	0.529	0.368	0.796	0.782	0.701	0.923	0.749								0.318	0.191	0.583	0.357	0.244	**0.580**	0.355
T5-3B	0.512	0.385	0.766	0.776	0.690	0.917	0.742								0.330	0.200	0.595	0.353	0.242	0.573	0.352
CENET+	**0.591**	**0.451**	**0.844**	**0.779**	**0.692**	**0.912**	**0.755**	**0.335**	**0.162**	0.659	**0.396**	**0.239**	**0.688**	**0.395**	**0.564**	**0.432**	**0.801**	**0.571**	**0.451**	0.773	**0.570**
- ERD	0.526	0.373	0.785	0.737	0.653	0.870	0.710	0.321	0.156	**0.665**	0.374	0.216	0.683	0.373	0.542	0.388	0.799	0.570	0.448	**0.774**	0.568
- RHL	0.445	0.367	0.565	0.754	0.685	0.862	0.714	0.232	0.128	0.446	0.290	0.202	0.469	0.289	0.295	0.168	0.560	0.370	0.262	0.588	0.367
T5-3B	0.568	0.426	0.819	0.736	0.646	0.900	0.714	0.303	0.158	0.568	0.330	0.203	0.712	0.329	0.550	0.413	0.798	0.555	0.431	0.765	0.554

Table 16: Complete results of ablation studies.

to encode temporal logical rules into rule representations. This makes ALRE-IR both a rule-based and an embedding-based method. Experiments in TLogic and ALRE-IR have proven that rule-based TKGF methods have strong ability in reasoning over zero-shot unseen entities connected by the seen relations, however, they are not able to handle unseen relations since the learned rules are strongly bounded by the observed relations. In our work, we implement zrLLM on embedding-based TKGF models because (1) embedding-based methods are much more popular; (2) zrLLM utilizes LLM to generate relation representations, which is more compatible with embedding-based methods.

Inductive Learning on TKGs. Inductive learning on TKGs has gained increasing interest. It refers to developing models that can handle the relations and entities unseen in the training data. TKG inductive learning methods can be categorized into two types. The first type of works focuses on reasoning over unseen entities (Ding et al., 2022; Wang et al., 2022; Ding et al., 2023b; Chen et al., 2023a), while the second type of methods aims to deal with the unseen relations (Mirtaheri et al., 2021; Ding et al., 2023a; Ma et al., 2023). Most of inductive learning methods are based on few-shot learning (FSL) (e.g., FILT (Ding et al., 2022), MetaTKGR (Zhang et al., 2019), FITCARL (Ding et al., 2023b), OAT (Mirtaheri et al., 2021), MOST (Ding et al., 2023a) and OSLT (Ma et al., 2023)). They first compute inductive representations of newly-emerged entities or relations based on K-associated facts (K is a small number, e.g., 1 or 3) observed during inference, and then use them to predict the facts regarding few-shot elements. One limitation of these works is that the inductive representations cannot be learned without the K-shot examples, making them hard to solve the zero-shot problems. Different from FSL methods, SST-BERT (Chen et al., 2023a) pre-trains a time-enhanced BERT (Devlin et al., 2019) for TKG reasoning. It achieves inductive learning over unseen entities but has not shown its ability in reasoning zero-shot relations. Another recent work MTKGE (Chen et al., 2023b) is able to concurrently deal with both unseen entities and relations. However, it requires a support graph containing a substantial number of data examples related to the unseen entities and relations, which is far from the zero-shot problem that we focus on.

TKG Reasoning with Language Models. Recently, more and more works have introduced LMs into TKG reasoning. SST-BERT (Chen et al., 2023a) generates a small-scale pre-training corpus based on the training TKGs and pre-trains an LM for encoding TKG facts. The encoded facts are then fed into a scoring module for LP. ECOLA (Han et al., 2023) aligns facts with additional fact-related texts and proposes a joint training framework that enhances TKG reasoning with BERT-encoded language representations. PPT (Xu et al., 2023a) converts TKGF into the pre-trained LM masked token prediction task and finetunes a BERT for TKGF. It directly input TKG facts into the LM for answer prediction. Apart from them, one recent work (Lee et al., 2023) explores the possibility of using in-context learning (ICL) (Brown et al., 2020) with LLMs to make predictions about future facts without fintuning. Another recent work GenTKG (Liao et al., 2023) finetunes an LLM, i.e., Llama2-7B (Touvron et al., 2023), and let the LLM directly generate the LP answer in TKGF. It mines temporal logical rules and uses them to retrieve historical facts for prompt generation.

Although the above-mentioned works have shown success of LMs in TKG reasoning, they have limitations: (1) None of these works has studied whether LMs can be used to better reason the zero-shot relations. (2) By only using ICL, LLMs are beaten by traditional TKG reasoning methods in performance (Lee et al., 2023). The performance can be greatly improved by finetuning LLMs (as in GenTKG (Liao et al., 2023)), but finetuning LLMs requires huge computational resources. (3) Since LMs, e.g., BERT and Llama2, are pre-trained with a huge corpus originating from diverse information sources, it is inevitable that they have already seen the world knowledge before they are used to solve TKG reasoning tasks. Most popular TKGF benchmarks are extracted from the TKGs constructed before 2020, e.g., ICEWS14, ICEWS18 and ICEWS05-15 (Jin et al., 2020). The facts inside are based on the world knowledge before 2019, which means LMs might have encountered them in their training corpus, posing a threat of information leak to the LM-driven TKG reasoning models. To this end, we (1) draw attention to studying the impact of LMs on zero-shot relational learning in TKGs; (2) make a compromise between performance and computational efficiency by not fintuning LMs or LLMs but adapting the LLM-provided semantic information to non-LM-based TKGF methods; (3) construct new benchmarks where the facts are all happening from 2021 to 2023, which avoids the possibility of information leak when we utilize T5-11B that was released in 2020.