EXPLAINABLE SUBGRAPH REASONING FOR FORECASTING ON TEMPORAL KNOWLEDGE GRAPHS

Anonymous authors
Paper under double-blind review

ABSTRACT

Interest has been rising lately towards modeling time-evolving knowledge graphs (KGs). Recently, graph representation learning approaches have become the dominant paradigm for link prediction on temporal KGs. However, the embedding-based approaches largely operate in a black-box fashion, lacking the ability to judge the results' reliability. This paper provides a link forecasting framework that reasons over query-dependent subgraphs of temporal KGs and jointly models the graph structures and the temporal context information. Especially, we propose a temporal relational attention mechanism and a human-mimic representation-update scheme to guide the extraction of an enclosing subgraph around the query. The subgraph is then expanded via a temporal neighborhood sampling and pruning. As a result, our approach provides explainable and human-understandable arguments to the forecasting task. We evaluate our model on four benchmark temporal knowledge graphs for the link forecasting task. While being more explainable, our model also obtains a relative improvement of up to 17.7 % on MRR compared to the previous best KG forecasting methods. We also conduct a survey with 53 respondents, and the results show that the reasoning arguments extracted by the machine for knowledge forecasting are aligned with human understanding.

1 INTRODUCTION

Reasoning, a process of inferring new knowledge from available facts, has long been considered to be an essential subject in artificial intelligence (AI). Recently, the KG-augmented reasoning process has been studied in (Das et al., 2017; Ren et al., 2020), where knowledge graphs store factual information in form of triples (s, p, o), e.g. (*California, locatedIn, USA*). In particular, s (subject) and o (object) are expressed as nodes in knowledge graphs and p (predicate) as an edge type. Most knowledge graph models assume that the underlying graph is static. However, in the real world, facts and knowledge change with time, which can be treated as time-dependent multi-relational data. To accommodate time-evolving multi-relational data, temporal KGs have been introduced (Boschee et al., 2015), where temporal events are represented as a quadruple by extending the static triplet with timestamps describing when these events occurred, i.e. (Barack Obama, inaugurated, as president of the US, 2009/01/20). In this work, we focus on forecasting on temporal KGs to infer future events based on past events. Forecasting on temporal KGs can improve a plethora of AI applications such as decision support in various domains, e.g., personalized health care and finance. These use cases often require the predictions made by the learning models to be interpretable, such that users can understand and rely on the predictions. However, current machine learning approaches (Trivedi et al., 2017; Jin et al., 2019) for temporal KG forecasting operate in a black-box fashion, where they design an embedding-based score function to estimate the correctness of a quadruple. These models can not clearly show which events contribute to the prediction and lack explainability to the forecasting. Thus, purely data-driven 'black box' methods do not give any information about the predictions' reliability, making them less suitable for real-world applications.

Explainable approaches for reasoning on graphs can generally be categorized into post-hoc interpretable and integrated transparent methods (Došilović et al., 2018). Post-hoc interpretable approaches (Ying et al., 2019) aim to interpret the results of a black-box model, while integrated transparent approaches (Das et al., 2017; Qiu et al., 2019; Wang et al., 2019) have an explainable internal mechanism. In particular, most integrated transparent (Lin et al., 2018; Hildebrandt et al., 2020) models employ path-based methods, e.g., reinforcement learning, to derive an explicit reasoning path and demonstrate a transparent reasoning process. The path-based methods focus on finding the answer to a query within a single reasoning chain. However, many complicated queries require multiple supporting reasoning chains rather than just one reasoning path. Recent work (Xu et al., 2019; Teru et al., 2019) has shown that reasoning over local subgraph structures substantially boosts performance while keeping interpretability. However, these explainable models cannot only be applied to temporal graph-structured data because they do not take time information into account. The time span between events often shows important regularities on their relative importance for the forecasting on temporal multi-relational graphs. For example, events in the distant past may be less informative and predictive to the current graph structure and thus to forecasting. Therefore, this work aims to design a transparent forecasting mechanism on temporal KGs that can generate informative explanations of the predictions and enables humans to check their reliability.

In this paper, we propose an e**x**plainabl**e** **r**easoning framework for forecasting on **t**emporal knowl**e**dge graphs, xERTE, which employs a sequential reasoning process over local subgraphs of temporal KGs. For example, to forecast a future link, xERTE starts from the query subject and then samples its temporal neighbors to construct a dynamic subgraph around it. Furthermore, xERTE prunes and samples the subgraph to further expand it step-by-step. After several rounds of expansion and pruning, the query object is predicted from nodes in the final subgraph. Thus, xERTE can naturally visualize the inference process and provide a concise and compact graphical explanation to the forecasting based on the extracted subgraph. To guide the model to extend the subgraph in the direction of the query's interest, we propose a two-folds temporal relational graph attention mechanism. First, inspired by Vaswani et al. (2017), we apply an attention mechanism to learn a vector representation of the target entity as a weighted sum of embeddings of its past neighbors across different timestamps. Second, we use a shared embedding layer containing static and time-dependent entity embeddings, allowing our model to capture both global structural information and temporal aspects about the graph. Besides, to mimic human's reasoning behavior, we develop a novel representation update mechanism. When humans perform a reasoning process, their impressions of observed entities will update as new arguments have been found. Thus, it is necessary to ensure that all existing entities in a subgraph can receive messages from newly added edges. To this end, the proposed representation update mechanism enables every entity to receive messages from its farthest neighbors in the subgraph by performing message passing along each edge only once.

The major contributions of this work are as follows. **(1)** We develope the first explainable model based on temporal relational attention mechanisms for forecasting on temporal KGs. **(2)** Unlike most black-box embedding-based models, xERTE visualizes the reasoning process and provides an interpretable inference graph to emphasize important arguments and influences of past events to the forecasting. Thus, humans can inspect the arguments' reliability to prevent inappropriate predictions using misleading data. **(3)** To mimic human's reasoning behavior, we design a reverse representation update mechanism, where newly found entities will affect the hidden representations of explored entities. **(4)** The dynamical pruning procedure enables our model to perform reasoning on large-scale temporal knowledge graphs with millions of edges. **(5)** To evaluate our approach, we apply our model for forecasting future links between observed nodes on four large-scale temporal knowledge graphs, showing that our method outperforms all state-of-the-art baselines. **(6)** We conduct a survey with 53 respondents to evaluate whether the model is aligned with human understanding.

2 PRELIMINARIES

Let $\mathcal{E}$ and $\mathcal{P}$ represent a finite set of entities, and predicates, respectively. A temporal knowledge graph is a collection of timestamped facts written as quadruples. A quadruple $q = (e_s, p, e_o, t)$ describes a temporal fact at a timestamp t, where $p \in \mathcal{P}$ represents a timestamped and labeled edge between a subject entity $e_s \in \mathcal{E}$ and an object entity $e_o \in \mathcal{E}$. The temporal knowledge graph forecasting task aims to predict unknown links at future timestamps based on observed events.

Definition 1 *(Temporal KG forecasting). Let $\mathcal{F}$ represents the set of all ground-truth quadruples, and $(e_{s,q}, p_q, e_{o,q}, t_q) \in \mathcal{F}$ denotes the target quadruple. Given a query $(e_{s,q}, p_q, ?, t_q)$ derived from the target quadruple and a set of observed facts $\mathcal{O} = \{(e_i, p_k, e_j, t_l) \in \mathcal{F} | t_l < t_q\}$, the temporal KG forecasting task is to predict the missing object entity $e_{o,q}$. Specifically, we consider all entities*

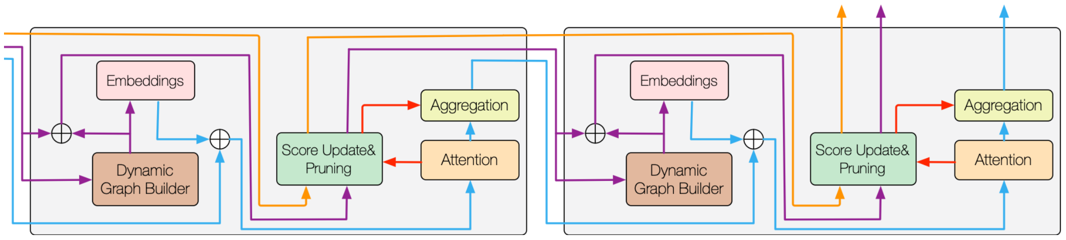


Figure 1: Model Architecture. Arrows represent as follows: → plausibility score, → attention score, → node sampling and pruning, → node representations. (details in main text.)

in the set $\mathcal{E}$ as candidates and rank object entities $e_o \in \mathcal{E}$ by their likelihood to form a true triple together with the given subject-predicate-pair at timestamp t_q[1].

Inference Graph: For a given query $q = (e_q, p_q, ?, t_q)$, we build an inference graph $\mathcal{G}_{inf}$ to visualize the reasoning process. Unlike temporal KGs, each entity in $\mathcal{G}_{inf}$ is associated with a timestamp, and thus, each node is an entity-timestamp pair. The inference graph is a directed acyclic graph in which a link points from a node with an earlier timestamp to a node with a later timestamp. We initialize $\mathcal{G}_{inf}$ with a node $v_q = (e_q, t_q)$ consisting of the query subject and the query time. Then we expand $\mathcal{G}_{inf}$ by adding neighbors of e_q in the temporal KG. For example, suppose that (e_q, p_k, e_j, t') is a ground truth quadruple where $t' < t_q$, we add the node (e_j, t') into $\mathcal{G}_{inf}$ and link it with v_q. We provide an example in Figure 5 in the appendix to illustrate the inference graph.

Definition 2 *(Node in Inference Graph and its Temporal Neighborhood). Let $\mathcal{E}$ represents all entities, $\mathcal{F}$ denotes all ground-truth quadruples, and t represents a timestamp. A node in an inference graph $\mathcal{G}_{inf}$ is defined as an entity-timestamp pair $v = (e_i, t), e_i \in \mathcal{E}$. We define the set of prior neighbors of v as $\mathcal{N}_{v=(e_i,t)} = \{(e_j, t')|(e_i, p_k, e_j, t') \in \mathcal{F} \wedge (t' < t)\}$. For simplicity, we denote prior neighbors as $\mathcal{N}_v$. Similarly, we define the set of posterior neighbors of v as $\overline{\mathcal{N}}_{v=(e_i,t)} = \{(e_j, t')|(e_i, p_k, e_j, t') \in \mathcal{F} \wedge (t' > t)\}$. We denote them as $\overline{\mathcal{N}}_v$ for short.*

The definition of *prior neighbors* and *posterior neighbors* are introduced for clarifying the message passing mechanism in Section 4.3. We use $\mathcal{P}_{uv}$ to represent the set of observed predicates connecting node u and node v. Besides, we define the set of edges between v and its prior neighbors as $\mathcal{Q}_v$, where $q \in \mathcal{Q}_v$ is a *prior edge* of v.

3 Related Work

Representation learning is an expressive and popular paradigm underlying many KG models. The embedding-based approaches for knowledge graphs can generally be categorized into bilinear models (Nickel et al., 2011; Balažević et al., 2019), translational models (Bordes et al., 2013; Sun et al., 2019), and deep-learning models (Dettmers et al., 2017; Schlichtkrull et al., 2018). However, the above methods cannot use rich temporal dynamics available on temporal knowledge graphs. To this end, several studies have been conducted for temporal knowledge graph forecasting (Leblay & Chekol, 2018; García-Durán et al., 2018; Ma et al., 2018; Dasgupta et al., 2018; Trivedi et al., 2017; Jin et al., 2019; Goel et al., 2019; Lacroix et al., 2020). Nevertheless, the methods are largely black box, lacking the ability to interpret their prediction results. Recently, several explainable reasoning methods for knowledge graphs have been proposed (Das et al., 2017; Xu et al., 2019; Hildebrandt et al., 2020; Teru et al., 2019). However, these explainable methods can only deal with static KGs, while our model is designed for interpretable forecasting on temporal KGs.

4 Our Model

4.1 Subgraph Reasoning Process

The key contribution of this paper is a novel explainable subgraph reasoning model for temporal knowledge graph forecasting. Specifically, the model conducts the reasoning process on a dynamically expanded inference graph $\mathcal{G}_{inf}$ extracted from the temporal KG. This inference graph gives an interpretable graphical explanation about the final prediction. Given a query $q = (e_q, p_q, ?, t_q)$, we initialize the inference graph as the query entity e_q and define the tuple of (e_q, t_q) as a node in the inference graph. The inference graph expands by sampling prior neighbors that are linked with e_q before t_q. In general, the expansion will grow rapidly and cover almost all nodes after a few steps. To prevent the inference graph from exploding, we reduce the number of nodes by pruning the nodes that are less relevant to the query. Hence, we propose a query-dependent temporal relational attention mechanism in Section 4.3 to identify the nodes' importance in the inference graph related to the query q. Next, by expanding the inference graph, we sample the prior neighbors of the remaining nodes after pruning. As this process iterations, the inference graph allocates more and more information from increasingly larger neighborhoods of the temporal KG. Besides, we add self-loop edges to allow xERTE to stop at a node if it believes reaching the ground-truth. After running L expansion steps, the model selects the entity with the highest plausibility score in $\mathcal{G}_{inf}$ as the prediction of the missing entity in the query, where the inference graph serves as a graphical explaination.

4.2 Neighborhood Sampling

To reduce the scope and complexity, we sample a subset of prior edges $\hat{\mathcal{Q}}_v \in \mathcal{Q}_v$ at each inference step if a node $v = (e_i, t)$ has been involved in a large number of events that occurred before t. We denote the prior neighbors and posterior neighbors of node v after the sampling as $\hat{N}_v$ and $\overline{\hat{N}}_v$, respectively. The sampling can be uniform if there is no bias, or the sampling can be temporally biased using a non-uniform distribution. For instance, we may want to sample more edges closer to the current time point as the events that took place long ago may have less impact on the current structure and, thus, the inference. Specifically, we propose three different sampling strategies: (1) **Uniform sampling**. Each prior edge $q_v \in \mathcal{Q}_v$ of node v has the same probability of being selected: $\mathbb{P}(q_v) = 1/|\mathcal{Q}_v|$. (2) **Time-aware exponentially weighted sampling**. We temporally bias the neighborhood sampling using an exponential distribution and assign the probability $\mathbb{P}(q_v = (e_i, p_k, e_j, t')) = \exp(t' - t)/\sum_{(e_i, p_l, e_m, t'') \in \mathcal{Q}_v} \exp(t'' - t)$ to each prior neighbor, which negatively correlates with the time difference between node v and its prior neighbor (e_j, t'). (3) **Time-aware linearly weighted sampling**. We use a linear function to bias the sampling. Compared to the second strategy, the quadruples occurred in early stages have a higher probability of being sampled. We have empirically found that the second strategy is most beneficial to our framework and provide a detailed ablation study in Section 5.2.

4.3 Temporal Relational Attention Mechanism

Here, we propose a temporal relational graph attention (TRGA) mechanism for identifying the relevant arguments in the inference graph related to a given query q. The TRGA mechanism computes a query-dependent contribution score for each edge. Besides, similar to GraphSAGE (Hamilton et al., 2017) and GAT (Veličković et al., 2017), the TRGA mechanism performs a local representation aggregation. To avoid misusing future information, we only allow message passing from prior neighbors to posterior neighbors. Specifically, for each node v in the inference graph, the aggregation function fuses the representation of node v and its prior neighbors $\mathcal{N}_v$ to output a time-aware representation for v. Since entities may play different roles depending on the predicate they are associated with, we also incorporate the predicate embeddings in the attention mechanism. If the node v has a large number of prior neighbors, we sample a subset $\hat{\mathcal{N}}_v \subset \mathcal{N}_v$ using different sampling schemes described in Section 4.2. Instead of treating all prior neighbors with equal importance, we take the query information into account and assign varying importance levels to each prior neighbor

[1]Throughout this work, we add reciprocal relations for every quadruple, i.e. we add (e_o, p^{-1}, e_s, t) for every (e_s, p, e_o, t). Hence, the restriction to predict object entities does not lead to a loss of generality.

u by calculating a query-dependent attention score as

$$e^l_{vu}(q, p_k) = \mathbf{W}^l_{sub}(\mathbf{h}^{l-1}_v || \mathbf{p}_k || \mathbf{h}^{l-1}_{e_q} || \mathbf{p}_q) \mathbf{W}^l_{obj}(\mathbf{h}^{l-1}_u || \mathbf{p}_k || \mathbf{h}^{l-1}_{e_q} || \mathbf{p}_q), \tag{1}$$

where $e^l_{vu}(q, p_k)$ is the attention score of the edge (v, p_k, u) regarding the query $q = (e_q, p_q, ?, t_q)$, p_k corresponds to the predicate between node u and node v, $\mathbf{p}_k$ and $\mathbf{p}_q$ are predicate embeddings. $\mathbf{h}^l_v$ denotes the hidden representation of v at the l^{th} inference step, $\mathbf{W}^l_{sub}$ and $\mathbf{W}^l_{obj}$ are two weight matrices for capturing the interactions between query features and subject/object features, $||$ denotes the concatenation operation. Then, we compute the normalized attention score $\alpha^l_{vu}(q, p_k)$ using the *softmax function*, which is written as

$$\alpha^l_{vu}(q, p_k) = \frac{\exp(e^l_{vu}(q, p_k))}{\sum_{w \in \hat{\mathcal{N}}_v} \sum_{p_z \in \mathcal{P}_{vw}} e^l_{vw}(q, p_z)}. \tag{2}$$

Once obtained, we aggregate the representation of prior neighbors and weight them using the normalized attention scores:

$$\widetilde{\mathbf{h}}^l_v(q) = \sum_{u \in \hat{\mathcal{N}}_v} \sum_{o_z \in \mathcal{P}_{vu}} \alpha^l_{vu}(q, p_k) \mathbf{h}^{l-1}_u(q), \tag{3}$$

where we initialize the hidden representations in the case $l = 0$ using the entity embeddings. Similar to (Hamilton et al., 2017), we combine the current hidden representation $\mathbf{h}^{l-1}_v(q)$ of node v with the aggregated neighborhood vector $\widetilde{\mathbf{h}}^l_v(q)$ and feed them into a fully connected layer with a nonlinear activation function, as shown below

$$\mathbf{h}^l_v(q) = \sigma(\mathbf{W}^l_h f(\mathbf{h}^{l-1}_v, \widetilde{\mathbf{h}}^l_v(q))) \tag{4}$$

$$f(\mathbf{x}, \mathbf{y}) = \gamma \mathbf{x} + (1 - \gamma)\mathbf{y}, \tag{5}$$

where $\mathbf{h}^l_v(q)$ denotes the final representation output at the l^{th} inference step, and γ is a hyperparameter.

4.4 Reverse Representation Update Mechanism

When humans perform a reasoning process, the impression of observed objects might change as new arguments have been found. For example, we want to predict the development trend of company A. We knew that A has the largest market portion, which gives us a high expectation about A. However, a new argument shows that conglomerate B enters this market as a strong competitor. Although the new argument is not directly related to A, it indicates that there will be a high competition between A and B, which lowers our expectations about A. To mimic human reasoning behavior, we should ensure that all existing nodes in an inference graph $\mathcal{G}_{inf}$ can receive messages from a new node when the new node is added to the inference graph. However, since $\mathcal{G}_{inf}$ expands once at each inference step, it might include l-hop neighbors of the query subject at the l^{th} step. The vanilla solution is to iterate the message passing l times at the l^{th} inference step, which means that we need to run the message passing $(1 + L) \cdot L/2$ times in total for L inference steps. To avoid the quadratic increase of message passing iterations, we propose a novel reverse representation update mechanism for subgraph reasoning. Recall that, for each node, we use its prior neighbors to update its representation. And at each inference step, we expand $\mathcal{G}_{inf}$ by adding prior neighbors of existing nodes. For example, assuming that we are at the fourth inference step, for a node that has been added at the second step, we only need to aggregate messages from nodes added at the third and fourth steps. Hence, we can update the representations of nodes in reverse order as they have been added in $\mathcal{G}_{inf}$. Specifically, at the l^{th} inference step, we first update the representations of nodes added at the $(l-1)^{th}$ inference step, then the nodes added at $(l-2)$, and so forth until $l = 0$, as shown in Algorithm 1 in the appendix. In this way, we compute messages along each edge in $\mathcal{G}_{inf}$ only once and ensure that every node can receive messages from its farthest prior neighbors.

4.5 Time-aware Entity Embeddings

In temporal knowledge graphs, graph structures are no longer static as entities and their links evolve over time. Entities naturally have some features that change over time and some features that remain fixed. Thus, we represent the latent feature embeddings of an entity $e_i \in \mathcal{E}$ at a time t with a static low-dimensional vector and a functional representation of time to encode both long-term stationary properties and time-dependent behavior, which is defined as $\mathbf{e}_i = [\bar{\mathbf{e}}_i, \mathbf{\Phi}(t)]^T \in \mathbb{R}^{d_S + d_T}$. Here, $\bar{\mathbf{e}}_i \in R^{d_S}$ represents the static embeddings that captures time-invariant features and global dependencies over the temporal KG. $\mathbf{\Phi}(t) \in R^{d_T}$ denotes a functional time encoding that captures temporal dependencies between entities (Xu et al., 2020). d_S and d_T represent the dimensionality of static embeddings and time embeddings, which can be tuned according to the temporal fraction of the given dataset. Thus, our model is able to predict unseen events based on both the stationary and time-evolving structural features.

4.6 Score Function and Learning

After L reasoning steps, we get a final inference graph. We rank all entities in the inference graph according to their plausibility scores and choose the entity with the highest score as our prediction. Specifically, we compute the plausibility score $a^l_{e_i}$ of an entity e_i regarding a query q as follows:

$$a^l_v = \sum_{u \in \widetilde{\mathcal{N}}_v} \sum_{p_z \in \mathcal{P}_{vu}} \alpha^l_{vu}(q, p_z) a^{l-1}_u, \qquad a^l_{e_i} = g(a^l_v | v(e) = e_i), \tag{6}$$

where $v(e)$ represents the entity included in node v, a^l_v denotes the plausibility score of node v at the l^{th} inference step, and $g(\cdot)$ represents a score aggregation function. As stated in Definition 2, each node in inference graph is an entity-timestamp pair. To assign each entity a unique score, we aggregate the plausibility scores of nodes where their entities are the same. We tried two score aggregation functions $g(\cdot)$, which are summation and mean. We have conducted an ablation study and found that the summation aggregation performs better. Detailed results are shown in Section 5.2. We prune the inference graph at each inference step by remaining the nodes with M largest node scores, where M is a prefixed parameter. To reduce the time complexity, we recalculate attention scores and plausibility scores on the pruned inference graph. Thus, the gradients of removed nodes do not need to be considered in backpropagation, significantly speeding up the training procedure.

We split quadruples of a temporal KG into *train*, *validation*, and *test* sets by timestamps, ensuring (timestamps of training set)<(timestamps of validation set)<(timestamps of test set). We use the binary cross-entropy as the loss function, which is defined as $\mathcal{L} = (-1/N) \sum_{m=1}^{N} 1/|\mathcal{E}^{inf}_m| \left(\mathbf{y}^T_m \log(\mathbf{a}_m) + (\mathbf{1} - \mathbf{y}_m)^T \log(\mathbf{1} - \mathbf{a}_m)\right)$, where N is the number of training samples, $\mathbf{a}_m$ denotes the vector containing plausibility scores of all entities in the final inference graph regarding a training sample q_m, and $\mathbf{y}_m$ represents the vector of binary labels indicating whether entities in the inference graph are genuine for q_m. We use *Adam* optimizer (Kingma & Ba, 2014) to learn model parameters.

5 Experiments

5.1 Datasets and Baselines

Integrated Crisis Early Warning System (ICEWS) (Boschee et al., 2015) and YAGO (Mahdisoltani et al., 2013) have established themselves in the research community as benchmark datasets of temporal KGs. The ICEWS dataset contains information about political events with time annotations, e.g., (Barack Obama, visit, Malaysia, 2014-02-19). We evaluate our model on three subsets of the ICEWS dataset: ICEWS14, ICEWS18, and ICEWS05-15 contain event facts in 2014, 2018, and the facts between 2005 to 2015, respectively. The YAGO dataset is a temporal knowledge base that fuses information from Wikipedias with the English WordNet dataset (Miller, 1995). Following the experimental settings of HyTE (Dasgupta et al., 2018), we use a subset and only deal with year level granularity by dropping the month and date information. We compare our approach and baseline methods by performing the link prediction task on the ICEWS14, ICEWS18, ICEWS0515, and YAGO datasets. The statistics of the datasets are provided in Appendix C.

We compare xERTE with benchmark temporal KG and static KG reasoning models. From the temporal KG reasoning models, we compare our model with several state-of-the-art methods, including TTransE (Leblay & Chekol, 2018), TA-DistMult/TA-TransE (García-Durán et al., 2018), DE-SimplE(Goel et al., 2019), TNTComplEx (Lacroix et al., 2020), and RE-Net (Jin et al., 2019). From the static KG reasoning models, we choose TransE (Bordes et al., 2013), DistMult (Yang et al., 2014), and ComplEx (Trouillon et al., 2016) where we compress temporal knowledge graphs into a static, cumulative graph by ignoring the time information. Implementation details of the baselines and our model please see Appendix E.

5.2 Experimental Results and Ablation Study

Datasets	**ICEWS14 - filtered**				**ICEWS05-15 - filtered**				**ICEWS18 - filtered**				**YAGO - filtered**			
Model	MRR	HITS@1	HITS@3	HITS@10	MRR	Hits@1	Hits@3	Hits@10	MRR	Hits@1	Hits@3	Hits@10	MRR	Hits@1	Hits@3	Hits@10
TransE	22.48	13.36	25.63	41.23	22.55	13.05	25.61	42.05	12.24	5.84	12.81	25.10	11.69	10.37	11.96	13.83
DistMult	27.67	18.16	31.15	46.96	28.73	19.33	32.19	47.54	10.17	4.52	10.33	21.25	11.98	10.20	12.31	14.93
ComplEx	30.84	21.51	34.48	49.58	31.69	21.44	35.74	52.04	21.01	11.87	23.47	39.87	12.07	10.42	12.36	14.82
TTransE	13.12	2.95	16.75	34.29	15.45	4.85	19.32	37.73	8.45	1.87	8.98	22.46	5.67	1.40	9.02	11.19
TA-DistMult	27.35	17.28	31.26	47.53	26.58	16.72	27.31	44.88	16.81	8.43	18.26	35.63	11.62	10.58	11.90	13.91
TA-TransE	18.32	0.00	30.25	50.02	20.58	1.86	32.98	53.22	13.66	0.01	18.31	37.97	6.75	2.24	11.12	12.33
DE-SimplE	32.67	24.43	35.69	49.11	35.02	25.91	38.99	52.75	19.30	11.53	21.86	34.80	11.73	10.70	12.10	13.51
TNTComplEx	32.12	23.35	36.03	49.13	27.54	19.52	30.80	42.86	21.23	13.28	24.02	36.91	53.29	51.18	54.03	56.63
RE-Net	38.28	28.68	41.34	54.52	42.97	31.26	46.85	63.47	28.81	19.05	32.44	47.51	54.87	47.51	57.84	65.81
xERTE	**42.61**	**34.09**	**47.87**	**59.76**	**49.81**	**40.48**	**55.71**	**67.14**	**30.19**	**21.97**	**34.76**	**46.95**	**61.37**	**57.72**	**64.80**	**66.45**
	±0.07	±0.09	±0.04	±0.11	±0.06	±0.08	±0.03	±0.05	±0.02	±0.04	±0.04	±0.03	±0.14	±0.19	±0.03	±0.09

Table 1: Link forecasting results on four datasets. Compared metrics are filtered MRR (%) and Hits@1/3/10 (%). The best results among all models are in bold.

Comparison results Table 1 summarizes the filtered results of the link prediction task on the ICEWS and YAGO datasets. A detailed evaluation protocol is provided in Appendix D. Overall, xERTE significantly outperforms all other baseline models on these datasets in MRR, Hits@1/3/10, while being more interpretable. Compared to the strongest baseline RE-Net, xERTE obtains a relative improvement of 12% and 19% in MRR and Hits@1, which are averaged on all four datasets. Especially, xERTE achieves more gains in Hits@1 than in Hits@10. It confirms the assumption that subgraph reasoning helps xERTE make a sharp prediction by exploiting local structures. To assess the importance of each component, we conduct the several ablation studies and show their results in the following.

Representation update analysis We train a model without updating the representations of nodes found in previous inference steps to find out how the reverse representation update contributes to our model. Since the reverse representation update ensures that each node can receive messages from every node in the inference graph, we expect this mechanism could to help nodes mine available information. This update mechanism should be especially important for nodes that only have been involved in a small number of events. Since the historical information of such nodes is quite limited, it is very challenging to forecast their future behavior. In Figure 2a and 2b we show the metrics of Hits@1 and Hits@10 as a function of the number of entities in the final inference graph. It can be observed the model with the reverse update mechanism performs better in general. In particular, this update mechanism significantly improves the performance if the query subject only has a small number of (multi-hop) neighbors in the final subgraph, which meets our expectation.

Time-aware representation analysis To verify the importance of the time components, we evaluate the performance of a model without time encoding. It can be observed in Figure 2c, removing the time-dependent part from entity representations sacrifices the model's performance significantly. Recall that each node in the inference graph $\mathcal{G}_{inf}$ is associated with a timestamp, the same entity might appear several times in $\mathcal{G}_{inf}$ with different timestamps. Besides, to get a unified score for each entity, we aggregate the plausibility scores of nodes where their entities are the same. Figure 2d shows that the summation aggregator brings a considerable gain in each metric on ICEWS14.

Sampling analysis We run experiments with different sampling strategies proposed in Section 4.2. To assess the necessity of the time-aware weighted sampling, we propose a deterministic version of time-aware weighted sampling, where we chronologically sort the prior edges of node v in terms of their timestamps and select the last N edges to build the subset $\hat{\mathcal{Q}}_v$. The experimental results are provided in Table 2. We find that different sampling strategies have a considerable influence on model's performance. Sampling strategies that bias towards recent quadruples perform better. Specifically, the exponentially time-weighted strategy performs better than the linear time-weighted strategy and the deterministic last-N-quadruples strategy.

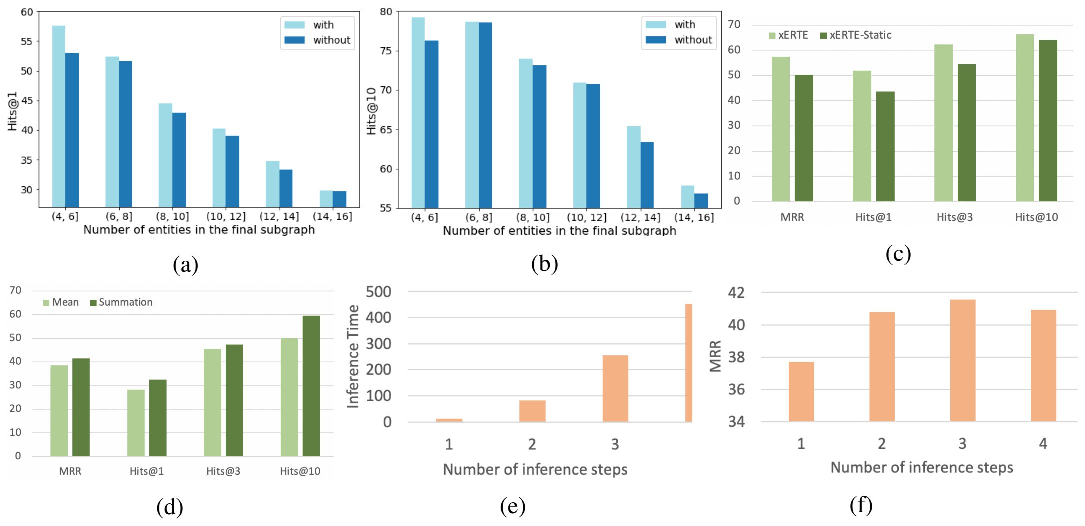


Figure 2: Ablation Study. (a)-(b) We compare the model with/without the reverse representation update in terms of Hits@1(%) and Hits@10(%) on ICEWS14, respectively. (c) Temporal embeddings analysis on YAGO. We refer the model only with static embeddings as xERTE-Static. (d) Plausibility score aggregation function analysis on ICEWS14. (e) Inference time (seconds) on the test set of ICEWS14 regarding different inference step settings $L \in \{1, 2, 3, 4\}$. (f) MRR(%) on ICEWS14 regarding different inference step settings L.

Time cost analysis Training time is a deal-breaker for time-aware models. We report the training time of xERTE and baseline models in Figure 4 at the beginning of the appendix. We see that the model has an on-par training speed with the strongest baseline RE-Net. The time cost of xERTE is affected not only by the scale of a dataset but also by the number of inference steps L. Thus, we run experiments of training time and predictive power regarding different settings of L and show the results in Figures 2e and 2f. We can see that the model achieves the best performance with $L = 3$ while the training time increases as L goes up. To make the computation more efficient, we develop a series of segment operations for subgraph reasoning. Please see Appendix H for more details.

Sampling Strategies	MRR	HITS@1	HITS@3	HITS@10
Uniform	36.26	27.66	41.39	53.96
Time-aware exponentially weighted	41.56	32.49	47.27	59.63
Time-aware linearly weighted	38.21	29.25	43.77	56.07
Last-N-edges	39.84	31.31	45.04	57.40

Table 2: Comparison between model variants with different sampling methods on ICEWS14 : MRR (%) and Hits@1/3/10 (%).

5.3 Graphical Explanation and Human Evaluation

The extracted inference graph provides a graphical explanation for the forecasting prediction. To demonstrate which arguments are important for the reasoning process, we assign each edge in the inference graph a contribution score. Specifically, the contribution score is defined as $c_{uv}(q, p_k) = e_{uv}(q, p_k)\alpha_v$ (see Equation 1 and 6 for definitions of e_{uv} and α_v), where node u is a prior neighbor of node v in terms of the predicate p_k. Thus, the users can trace back the important arguments that the prediction results mainly depend on. We study a query chosen from the test set, where we predict whom will Catherine Ashton visit on Nov. 9, 2014 and show the final inference graph in Figure 3. The node size indicates the value of the plausibility score; the darkness of the edge color indicates the contribution score of the edge. In this case, the model's prediction is Oman. We can see that *(Catherine Ashton, express intent to meet or negotiate, Oman, 2014-11-04)* is the most important event to support this answer. We provide more visualization results Appendix F.

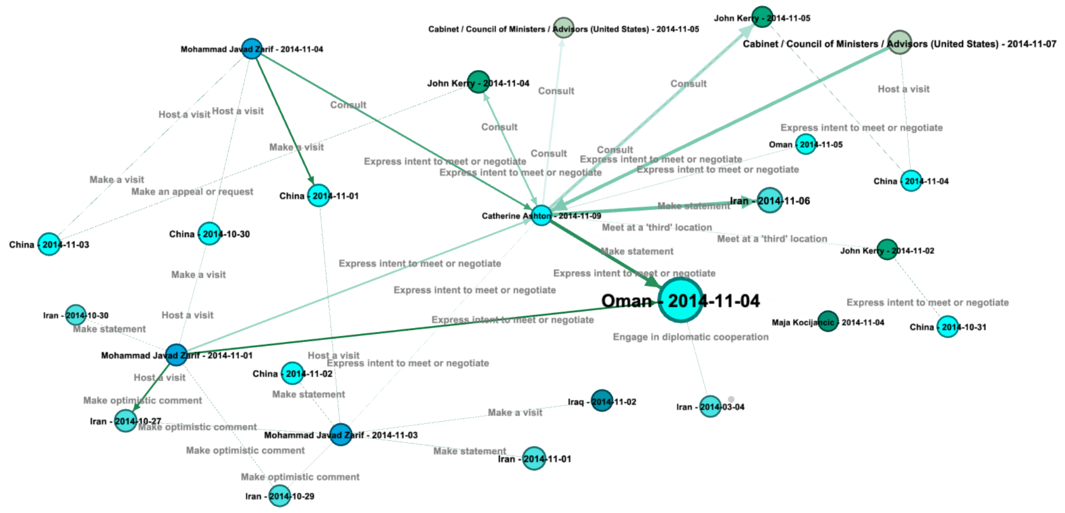

Mohammad Javad Zarif - 2014-11-04
Cabinet / Council of Ministers / Advisors (United States) - 2014-11-05
John Kerry - 2014-11-05
Cabinet / Council of Ministers / Advisors (United States) - 2014-11-07
John Kerry - 2014-11-04
Catherine Ashton - 2014-11-09
Oman - 2014-11-04
Oman - 2014-11-05
China - 2014-11-04
Iran - 2014-11-06
China - 2014-11-01
China - 2014-10-30
China - 2014-11-03
John Kerry - 2014-11-02
Maja Kocijancic - 2014-11-04
China - 2014-10-31
Iran - 2014-10-30
Mohammad Javad Zarif - 2014-11-01
China - 2014-11-02
Iraq - 2014-11-02
Iran - 2014-03-04
Iran - 2014-10-27
Mohammad Javad Zarif - 2014-11-03
Iran - 2014-11-01
Iran - 2014-10-29
Consult
Host a visit
Make a visit
Make an appeal or request
Express intent to meet or negotiate
Make statement
Meet at a 'third' location
Engage in diplomatic cooperation
Make optimistic comment

Figure 3: The inference graph for the query (Catherine Ashton, Make a visit, ?, 2014/11/09) from ICEWS14. The cyan node with the entity *Catherine Ashton* and the timestamp 2014/11/09 represents the given query subject and the query timestamp. The biggest cyan nodes represents the object predicted by xERTE. The node size indicates the value of the node score. Also, the edges' color indicates the contribution score of the edge, where darkness increases, as the contribution score goes up. The entity at an arrow's tail, the predicate on the arrow, the entity and the timestamp at the arrow's head build a genuine quadruple.

To assess whether the arguments are informative for users in an objective setting, we conducted a survey where respondents judge the relevance of the extracted arguments to the prediction. More concretely, we set up an online quiz consisting of 7 rounds. Each round is centered around a query sampled from the test set of ICEWS14/ICEWS0515. Along with the query statement and the ground-truth answer, we present the human respondents with two arguments in the inference graph with high contribution scores and two arguments with low contribution scores in a randomized order. Specifically, each argument is based on a chronological reasoning path that connects the query subject with an object candidate. For example, given a query (police, arrest, ?, 2014-12-28), an extracted argument is that police made statements to lawyers on 2014-12-08, then lawyers were criticized by citizens on 2014-12-10. In each round, we set up three questions to ask the participants to choose the most relevant argument, the most irrelevant argument, and sort the arguments according to their relevance. Then we rank the arguments according to their contribution scores computed by our model and check whether the relevance order classified by the respondents matches that estimated by our models. We surveyed 53 participants, and the average accuracy of all questions is 70.5%. Moreover, based on a majority vote, 18 out of 21 questions were answered correctly, indicating that the extracted inference graphs are informative, and the model is aligned with human intuition. The complete survey and a detailed evaluation are reported in Appendix I.

6 Conclusion and Future Work

We proposed an explainable reasoning approach for forecasting future links on large-scale temporal knowledge graphs. The model extracts a query-dependent subgraph from temporal KGs by selecting neighbors of nodes recursively and uses a novel temporal relational attention mechanism to guide the extraction. The subgraph can be seen as a graphical explanation of the prediction, making our model more interpretable than other baseline methods. We proposed a reverse representation update mechanism that provides a considerable gain to the nodes with a small amount of links in history. A survey shows that the arguments included in the subgraph are informative for humans.

REFERENCES

Ivana Balažević, Carl Allen, and Timothy M Hospedales. Tucker: Tensor factorization for knowledge graph completion. *arXiv preprint arXiv:1901.09590*, 2019.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. In *Advances in neural information processing systems*, pp. 2787–2795, 2013.

Elizabeth Boschee, Jennifer Lautenschlager, Sean O'Brien, Steve Shellman, James Starz, and Michael Ward. Icews coded event data. *Harvard Dataverse*, 12, 2015.

Rajarshi Das, Shehzaad Dhuliawala, Manzil Zaheer, Luke Vilnis, Ishan Durugkar, Akshay Krishnamurthy, Alex Smola, and Andrew McCallum. Go for a walk and arrive at the answer: Reasoning over paths in knowledge bases using reinforcement learning. *arXiv preprint arXiv:1711.05851*, 2017.

Shib Sankar Dasgupta, Swayambhu Nath Ray, and Partha Talukdar. Hyte: Hyperplane-based temporally aware knowledge graph embedding. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 2001–2011, 2018.

Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. Convolutional 2d knowledge graph embeddings. *arXiv preprint arXiv:1707.01476*, 2017.

Filip Karlo Došilović, Mario Brčić, and Nikica Hlupić. Explainable artificial intelligence: A survey. In *2018 41st International convention on information and communication technology, electronics and microelectronics (MIPRO)*, pp. 0210–0215. IEEE, 2018.

Alberto García-Durán, Sebastijan Dumančić, and Mathias Niepert. Learning sequence encoders for temporal knowledge graph completion. *arXiv preprint arXiv:1809.03202*, 2018.

Rishab Goel, Seyed Mehran Kazemi, Marcus Brubaker, and Pascal Poupart. Diachronic embedding for temporal knowledge graph completion. *arXiv preprint arXiv:1907.03143*, 2019.

Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *Advances in neural information processing systems*, pp. 1024–1034, 2017.

Marcel Hildebrandt, Jorge Andres Quintero Serna, Yunpu Ma, Martin Ringsquandl, Mitchell Joblin, and Volker Tresp. Reasoning on knowledge graphs with debate dynamics. *arXiv preprint arXiv:2001.00461*, 2020.

Woojeong Jin, Changlin Zhang, Pedro Szekely, and Xiang Ren. Recurrent event network for reasoning over temporal knowledge graphs. *arXiv preprint arXiv:1904.05530*, 2019.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Timothée Lacroix, Guillaume Obozinski, and Nicolas Usunier. Tensor decompositions for temporal knowledge base completion. *arXiv preprint arXiv:2004.04926*, 2020.

Julien Leblay and Melisachew Wudage Chekol. Deriving validity time in knowledge graph. In *Companion Proceedings of the The Web Conference 2018*, pp. 1771–1776, 2018.

Xi Victoria Lin, Richard Socher, and Caiming Xiong. Multi-hop knowledge graph reasoning with reward shaping. *arXiv preprint arXiv:1808.10568*, 2018.

Yunpu Ma, Volker Tresp, and Erik A Daxberger. Embedding models for episodic knowledge graphs. *Journal of Web Semantics*, pp. 100490, 2018.

Farzaneh Mahdisoltani, Joanna Biega, and Fabian M Suchanek. Yago3: A knowledge base from multilingual wikipedias. 2013.

George A Miller. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11): 39–41, 1995.

Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. A three-way model for collective learning on multi-relational data. In *Icml*, volume 11, pp. 809–816, 2011.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, pp. 8024–8035, 2019.

Lin Qiu, Yunxuan Xiao, Yanru Qu, Hao Zhou, Lei Li, Weinan Zhang, and Yong Yu. Dynamically fused graph network for multi-hop reasoning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 6140–6150, 2019.

Hongyu Ren, Weihua Hu, and Jure Leskovec. Query2box: Reasoning over knowledge graphs in vector space using box embeddings. *arXiv preprint arXiv:2002.05969*, 2020.

Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. Modeling relational data with graph convolutional networks. In *European Semantic Web Conference*, pp. 593–607. Springer, 2018.

Yizhou Sun, Jiawei Han, Xifeng Yan, Philip S Yu, and Tianyi Wu. Pathsim: Meta path-based top-k similarity search in heterogeneous information networks. *Proceedings of the VLDB Endowment*, 4(11):992–1003, 2011.

Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. Rotate: Knowledge graph embedding by relational rotation in complex space. *arXiv preprint arXiv:1902.10197*, 2019.

Komal K Teru, Etienne Denis, and William L Hamilton. Inductive relation prediction by subgraph reasoning. *arXiv*, pp. arXiv–1911, 2019.

Rakshit Trivedi, Hanjun Dai, Yichen Wang, and Le Song. Know-evolve: Deep temporal reasoning for dynamic knowledge graphs. *arXiv preprint arXiv:1705.05742*, 2017.

Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. Complex embeddings for simple link prediction. International Conference on Machine Learning (ICML), 2016.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pp. 5998–6008, 2017.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017.

Xiang Wang, Dingxian Wang, Canran Xu, Xiangnan He, Yixin Cao, and Tat-Seng Chua. Explainable reasoning over knowledge graphs for recommendation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 5329–5336, 2019.

Da Xu, Chuanwei Ruan, Evren Korpeoglu, Sushant Kumar, and Kannan Achan. Inductive representation learning on temporal graphs. *arXiv preprint arXiv:2002.07962*, 2020.

Xiaoran Xu, Wei Feng, Yunsheng Jiang, Xiaohui Xie, Zhiqing Sun, and Zhi-Hong Deng. Dynamically pruned message passing networks for large-scale knowledge graph reasoning. *arXiv preprint arXiv:1909.11334*, 2019.

Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. Embedding entities and relations for learning and inference in knowledge bases. *arXiv preprint arXiv:1412.6575*, 2014.

Zhitao Ying, Dylan Bourgeois, Jiaxuan You, Marinka Zitnik, and Jure Leskovec. Gnnexplainer: Generating explanations for graph neural networks. In *Advances in neural information processing systems*, pp. 9244–9255, 2019.

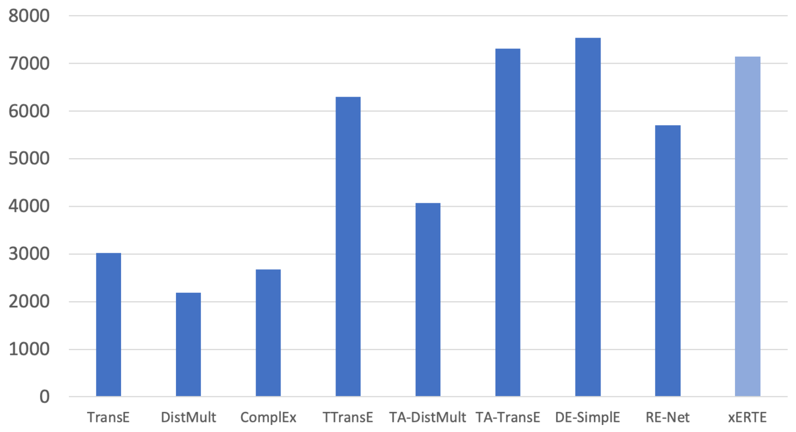


Figure 4: Training time (seconds) of baselines and xERTE on ICEWS14. We measure the training time until models are overfitting. The training time of xERTE is in light blue and corresponds to the standard hyperparameter settings in Section E.

Algorithm 1 Reverse Node Representations Update at the l^{th} inference step

```
Require: Inference graph $\mathcal{G}_{inf}$, nodes in the inference graph $\mathcal{V}$, nodes that have been added into $\mathcal{G}_{inf}$
    at the $l^{th}$ inference step $\mathcal{V}^l$, sampled prior neighbors $\hat{\mathcal{N}}_v$, hidden representation at the $(l\text{-}1)^{th}$ step
    $\mathbf{h}_v^{l-1}$, entity embeddings $\mathbf{e}_i$, weight matrices $\mathbf{W}_{sub}$, $\mathbf{W}_{obj}$, and $\mathbf{W}_h$, query $q = (e_q, p_q, ?, t_q)$.
Ensure: Hidden representations $\mathbf{h}_v^l$ at the $l^{th}$ inference step, $\forall v \in \mathcal{V}$.
 1: for $t = l-1, ..., 0$ do
 2:   for $v \in \mathcal{V}^t$ do
 3:     for $u \in \hat{\mathcal{N}}_v$ do
 4:       $e_{vu}^l(q, p_k) = \mathbf{W}_{sub}^l(\mathbf{h}_v^{l-1}||\mathbf{p}_k||\mathbf{h}_{e_q}^{l-1}||\mathbf{p}_q)\mathbf{W}_{obj}^l(\mathbf{h}_u^{l-1}||\mathbf{p}_k||\mathbf{h}_{e_q}^{l-1}||\mathbf{p}_q)$,
 5:       $\alpha_{vu}^l(q, p_k) = \frac{\exp(e_{vu}^l(q,p_k))}{\sum_{w\in\hat{\mathcal{N}}_v}\sum_{p_z\in\mathcal{P}_{vw}} e_{vw}^l(q,p_z)}$
 6:     end for
 7:     $\widetilde{\mathbf{h}}_v^l(q) = \sum_{u\in\hat{\mathcal{N}}_v}\sum_{o_z\in\mathcal{P}_{vu}} \alpha_{vu}^l(q,p_k)\mathbf{h}_u^{l-1}(q)$,
 8:     $\mathbf{h}_v^l(q) = \sigma(\mathbf{W}_h^l f(\mathbf{h}_v^{l-1}, \widetilde{\mathbf{h}}_v^l(q)))$
 9:   end for
10: end for
11: Return $\mathbf{h}_v, \forall v \in \mathcal{V}$.
```

A Related Work

A.1 Knowledge Graph Models

Representation learning is an expressive and popular paradigm underlying many KG models. The key idea is to embed entities and relations into a low-dimensional vector space. The embedding-based approaches for knowledge graphs can generally be categorized into bilinear models (Nickel et al., 2011; Balažević et al., 2019), translational models (Bordes et al., 2013; Sun et al., 2019), and deep-learning models (Dettmers et al., 2017; Schlichtkrull et al., 2018). However, the above methods lack the ability to use rich temporal dynamics available on temporal knowledge graphs. To this end, several studies have been conducted for link prediction on temporal knowledge graphs (Leblay & Chekol, 2018; García-Durán et al., 2018; Ma et al., 2018; Dasgupta et al., 2018; Trivedi et al., 2017; Jin et al., 2019; Goel et al., 2019; Lacroix et al., 2020). Ma et al. (2018) developed extensions of static knowledge graph models by adding a timestamp embedding to their score functions. Besides, García-Durán et al. (2018) suggested a straight forward extension of some existing static knowledge graph models that utilize a recurrent neural network (RNN) to encode predicates with temporal tokens derived by decomposing given timestamps. Also, HyTE (Dasgupta et al., 2018) embeds time information in the entity-relation space by arranging a temporal hyperplane

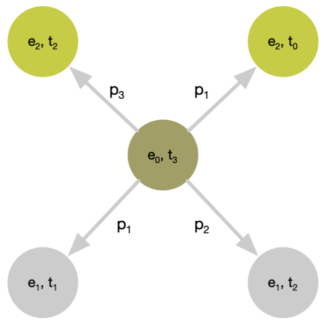


Figure 5: The inference graph for the query $(e_0, p_1, ?, t_3)$. The entity at an arrow's tail, the predicate on the arrow, the entity and the timestamp at the arrow's head build a true quadruple. Specifically, the true quadruple in this graph are as follows: $\{(e_0, p_1, e_1, t_1)$, (e_0, p_2, e_1, t_2), (e_0, p_3, e_2, t_2), $(e_0, p_1, e_2, t_0)\}$. Note that t_3 is larger than t_0, t_1, t_2.

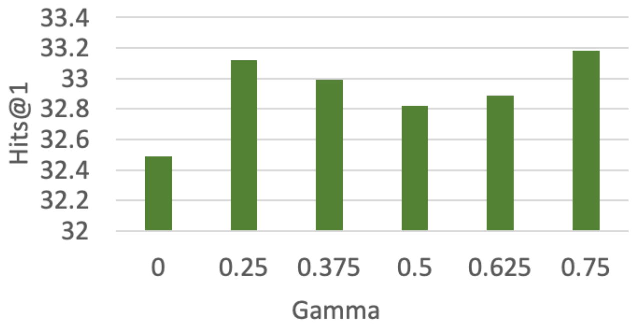


Figure 6: Model's performance on ICEWS14 regarding different values of γ.

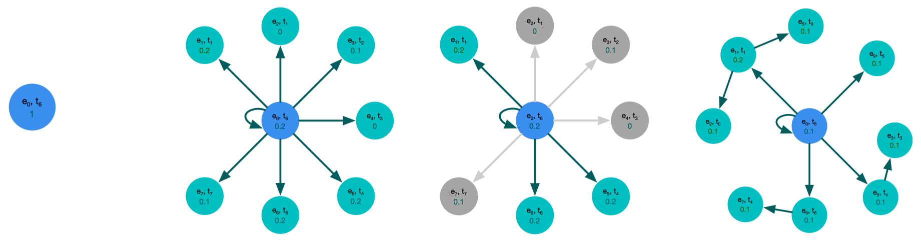
to each timestamp. However, these models cannot generalize to unseen timestamps because they only learn embeddings for observed timestamps. In contrast, Know-Evolve (Trivedi et al., 2017) learns evolving entity representations using point processes, being able to infer embeddings at a future timestamp. Nevertheless, the methods are largely black box, lacking the ability to interpret their prediction results while our main focus is to employ an integrated transparency mechanism for archiving human-understandable results.

A.2 Explainable reasoning on Knowledge Graphs

Recently, several explainable reasoning methods for knowledge graphs have been proposed (Das et al., 2017; Xu et al., 2019; Hildebrandt et al., 2020) . Das et al. (2017) proposed a reinforcement learning based path searching approach to display the query subject and predicate to the agents and let them perform a policy guided walk to the correct object entity. The reasoning paths produced by the agents can explain the prediction results to some degree. Also, Hildebrandt et al. (2020) framed the link prediction task as a debate game between two reinforcement learning agents that extract arguments from knowledge graphs and allow users to understand the decision the agents. Besides, and more related to our work, Xu et al. (2019) models a sequential reasoning process by dynamically constructing an input-dependent sub-graph. The difference here is that these explainable methods can only deal with static KGs, while our model is designed for interpretable forecasting on temporal KGs.

B Workflow

We show the workflow of the subgraph reasoning process in Figure 7. The model conducts the reasoning process on a dynamically expanding inference graph $\mathcal{G}_{inf}$ extracted from the temporal KG. This inference graph gives an interpretable graphical explanation about the final prediction. Given



(a) Initialization. (b) Expansion. (c) Pruning. (d) New Expansion.

Figure 7: Inference step by step illustration. Node prediction scores are attached to the nodes. Gray nodes are removed by pruning procedure.

a query $q = (e_q, p_q, ?, t_q)$, we initialize the inference graph with the query entity e_q and define the tuple of (e_q, t_q) as a node in the inference graph (Figure 7a). The inference graph expands by sample neighbors that have been linked with e_q before t_q, as shown in Figure 7b. The expansion would go rapidly that it covers almost all nodes after a few steps. To prevent the inference graph from exploding, we constrain the number of nodes by pruning the nodes that are less related to the query (Figure 7c) . Here, we propose a query-dependent temporal relational attention mechanism in Section 4.3 to identify the nodes' importance in the inference graph for query q and aggregate information from nodes' local neighbors. Next, we sample the past neighbors of the remaining nodes in the inference graph to expand it further, as shown in Figure 7d. As this process iterates, the inference graph incrementally gains more and more information from increasingly larger neighborhoods of the temporal KG. After running L expansion steps, the model predicts the missing entity in the query based on the entities in the inference graph, where the inference graph serves as a graphical explaination for the prediction.

C Dataset Statistics

Dataset	N_{train}	N_{valid}	N_{test}	N_{ent}	N_{rel}	$N_{timestamp}$	Time granularity
ICEWS14	63510	13610	13610	7128	230	365	day
ICEWS18	373018	45995	49545	23033	256	304	day
ICEWS0515	323829	69600	67900	10488	251	4017	day
YAGO	51205	10973	10973	10038	10	188	year

Table 3: Dataset Statistics

Dataset	$\|\mathcal{E}_{tr}\|$	$\|\mathcal{E}_{tr}\|/\|\mathcal{E}\|$	$\|\mathcal{E}_{tr+val}\|$	$\|\mathcal{E}\|$
ICEWS14	6180	86.7	6710	7128
ICEWS18	28866	92.8	29924	31110
ICEWS0515	8853	9792	84.4	10488
YAGO	7904	78.7	9008	10038

Table 4: Unseen nodes (new emerging entities) in validation set and test set. $|\mathcal{E}_{tr}|$ denotes the number of entities in the training set, $|\mathcal{E}_{tr+val}|$ represents the number of entities in the training set and validation set, $|\mathcal{E}|$ denotes the number of entities in the whole dataset.

We provide the statistics of datasets in Table 3. Since we split each dataset into subsets by timestamps, ensuring (timestamps of training set) $<$ (timestamps of validation set) $<$ (timestamps of test set), a considerable amount of entities in test sets are new. We report the number of entities in each subset in Table 4.

Besides, since we focus local neighbors, it is necessary to answer the question of ”what is the accuracy of the prediction if we randomly choose an entity from the one-hop or two-hop neighbors?”.

It is equivalent to the ratio of the number of ground truth entities to the number of prior neighbors. We show the results regarding the one-hop neighborhood in Table 5 and the results regarding the two-hop neighborhood in Table 6.

Data set	$\lvert\mathcal{N}_{s(p,t_q)}\rvert$	$\lvert\mathcal{N}_{s(t<t_q)}\rvert$	*Ratio*
ICEWS14	3.69	28.29	0.13
ICEWS18	1.77	32.31	0.05
ICEWS0515	5.91	35.70	0.16

Table 5: Events with the same object in the latest N events of a query (s, p, o, t_q). $|\mathcal{N}_{s(p,t_q)}|$ denotes the average number of **one-hop** prior neighbors. $|\mathcal{N}_{s(t<t_q)}|$ denotes the averaged number of ground truth entities of the query. *Ratio* means the ratio of $|\mathcal{N}_{s(p,t_q)}|$ to $|\mathcal{N}_{s(t<t_q)}|$.

Data set	$\lvert\mathcal{N}_{s(p,t_q)}\rvert$	$\lvert\mathcal{N}_{s(t<t_q)}\rvert$	*Ratio*
ICEWS14	30.95	904.53	0.034
ICEWS18	20.68	1122.13	0.018
ICEWS0515	60.72	1343.17	0.045

Table 6: Events with the same object in the latest N events of a query (s, p, o, t_q). $|\mathcal{N}_{s(p,t_q)}|$ denotes the average number of **two-hop** prior neighbors. $|\mathcal{N}_{s(t<t_q)}|$ denotes the averaged number of ground truth entities of the query. *Ratio* means the ratio of $|\mathcal{N}_{s(p,t_q)}|$ to $|\mathcal{N}_{s(t<t_q)}|$.

D Evaluation protocol

For each quadruple $q = (e_s, p, e_o, t)$ in the test set $\mathcal{G}_{test}$, we create two queries: $(e_s, p, ?, t)$ and $(e_o, p^{-1}, ?, t)$, where p^{-1} denotes the reciprocal relation of p. For each query, the model ranks all entities $\mathcal{E}^L_{inf}$ in the final inference graph according to their scores. If the ground truth entity does not appear in the final subgraph, we set its rank as $|\mathcal{E}|$ (the number of entities in the dataset). Let ψ_{e_s} and ψ_{e_o} represent the rank for e_s and e_o of the two queries respectively, we evaluate our models using standard metrics across the link prediction literature: *mean reciprocal rank (MRR)*: $\frac{1}{2\cdot|\mathcal{G}_{test}|}\sum_{q\in\mathcal{G}_{test}}(\frac{1}{\psi_{e_s}}+\frac{1}{\psi_{e_o}})$ and *Hits*@$k(k \in \{1, 3, 10\})$: the percentage of times that the true entity candidate appears in the top k of ranked candidates.

In this paper, we consider two different filtering settings. The first one is following the ranking technique described in Bordes et al. (2013), where we remove from the list of corrupted **triplets** all the **triplets** that appear either in the training, validation, or test set. We name it *static filtering*. Trivedi et al. (2017) and Jin et al. (2019) use this filtering setting for reporting their results on temporal KG forecasting. However, assume this filtering setting is not appropriate for evaluating the link prediction on temporal KGs. For example, there is a test quadruple (Barack Obama, visit, India, Jan. 25, 2015), and we perform the object prediction (Barack Obama, visit, ?, Jan. 25, 2015). Besides, we have observed the quadruple (Barack Obama, visit, Germany, Jan. 18, 2013) in the training set. According to the static filtering, (Barack Obama, visit, Germany, Jan. 25, 2015) will be considered as a genuine quadruple and filtered accordingly since the triplet (Barack Obama, visit, Germany) appears in the training set in the quadruple (Barack Obama, visit, Germany, Jan. 18, 2015). However, the triplet (Barack Obama, visit, Germany) is only temporally valid on Jan. 18, 2013 but not on Jan. 25, 2015. Therefore, we propose another filtering scheme, which is more appropriate for the link forecasting task on temporal KGs. We name it *time-aware filtering*. In this case, we only filter out the **quadruples** that are genuine at the timestamp when the link forecasting is conducted . In other words, if the quadruple (Barack Obama, visit, Germany, Jan. 25, 2015) does not appear at the query time Jan. 25, 2015, this quadruple is considered as corrupted and filtered out. In Table 1, we report the time-aware filtered results of baselines and our model. Additionally, we provide the raw results in Table 7.

Datasets	ICEWS14 - raw				ICEWS05-15 - raw				ICEWS18 - raw				YAGO - raw			
Model	MRR	HITS@1	HITS@3	HITS@10	MRR	Hits@1	Hits@3	Hits@10	MRR	Hits@1	Hits@3	Hits@10	MRR	Hits@1	Hits@3	Hits@10
TransE	21.66	12.85	24.31	40.14	22.28	12.76	25.29	41.93	12.08	5.72	12.59	24.90	11.14	9.61	11.65	13.54
DistMult	27.24	17.69	30.75	46.80	28.29	18.73	31.86	47.37	9.89	4.28	10.04	20.94	11.91	10.11	12.27	14.89
ComplEx	30.40	21.01	34.12	49.33	31.22	20.81	35.22	51.89	20.42	11.22	22.89	39.47	11.49	9.85	12.03	14.28
TTransE	13.09	2.93	16.70	34.25	15.41	4.82	19.21	37.70	8.44	1.85	8.95	22.38	5.67	1.40	9.02	11.19
TA-DistMult	25.19	15.60	29.09	44.44	23.29	14.16	25.56	41.40	14.79	6.97	15.91	31.25	11.44	10.13	11.88	13.83
TA-TransE	17.41	0.00	28.39	48.04	19.07	1.78	30.87	50.17	12.24	0.01	17.07	36.80	6.71	2.12	10.94	12.19
RE-Net	35.33	26.06	39.27	53.13	40.52	29.45	45.84	62.36	27.87	18.12	31.60	46.94	52.71	46.45	56.81	65.02
xERTE	**41.56** ±0.06	**32.49** ±0.08	**47.27** ±0.02	**59.63** ±0.14	**47.71** ±0.06	**38.18** ±0.05	**55.01** ±0.07	**65.51** ±0.09	**28.83** ±0.01	**20.14** ±0.03	**33.66** ±0.05	**47.08** ±0.04	**57.37** ±0.15	**51.84** ±0.27	**62.36** ±0.01	**66.39** ±0.12

Table 7: Link forecasting results on four datasets. Compared metrics are raw MRR (%) and raw Hits@1/3/10 (%). The best results among all models are in bold.

E Implementation

We implemented our model and all baselines in PyTorch (Paszke et al., 2019). Besides, we use the datasets augmented with reciprocal relations to train all baseline models. We tune hyperparameters of our models using a grid search. We set the learning rate to be 0.0002, the batch size to be 128, the inference step L to be 3, the static embedding dimensions d_S to be 256, and time embedding dimensions d_T to be 256. Besides, we sample 30 edges from the prior neighbors for each node in the inference graph and prune 10 edges regarding its attention value. We implement TTransE, TA-TransE/TA-DistMult, and RE-Net based on the code[†] provided in (Jin et al., 2019). We use the released code [‡] to implement DE-SimplE and TNTComplEx. We use the binary cross-entropy loss to train these baselines and optimize hyperparameters by early stopping according to MRR on the validation set.

F Visualization

We provide an additional inference visualization example in Figure 8.

G Reverse representation update mechanism for subgraph reasoning

In this section, we explain an additional reason why we have to update node representations along edges selected in previous inference steps. We show our intuition by a simple query in Figure 9 with two inference steps. We do not apply the pruning procedure. First, we check the equations without updating node representations along previously selected edges.

$$\begin{aligned}\textbf{First inference step:}\quad \mathbf{h}_0^1 &= f(\mathbf{h}_0^0, \mathbf{h}_1^0, \mathbf{h}_2^0, \mathbf{h}_3^0)\\ \mathbf{h}_1^1 &= \mathbf{h}_1^0\\ \mathbf{h}_2^1 &= \mathbf{h}_2^0\\ \mathbf{h}_3^1 &= \mathbf{h}_3^0\end{aligned}$$

$$\begin{aligned}\textbf{Second inference step:}\quad \mathbf{h}_0^2 &= f(\mathbf{h}_0^1, \mathbf{h}_1^1, \mathbf{h}_2^1, \mathbf{h}_3^1)\\ &= f(\mathbf{h}_0^1, \mathbf{h}_1^0, \mathbf{h}_2^0, \mathbf{h}_3^0)\\ \mathbf{h}_1^2 &= f(\mathbf{h}_1^1, \mathbf{h}_4^0, \mathbf{h}_5^0)\\ &= f(\mathbf{h}_1^0, \mathbf{h}_4^0, \mathbf{h}_5^0)\\ \mathbf{h}_3^2 &= f(\mathbf{h}_3^1, \mathbf{h}_6^0)\\ &= f(\mathbf{h}_3^0, \mathbf{h}_6^0)\\ \mathbf{h}_2^2 &= f(\mathbf{h}_2^1, \mathbf{h}_7^0, \mathbf{h}_8^0)\\ &= f(\mathbf{h}_2^0, \mathbf{h}_7^0, \mathbf{h}_8^0)\end{aligned}$$

[†]https://github.com/INK-USC/RE-Net

[‡]https://github.com/BorealisAI/de-simple, https://github.com/facebookresearch/tkbc.

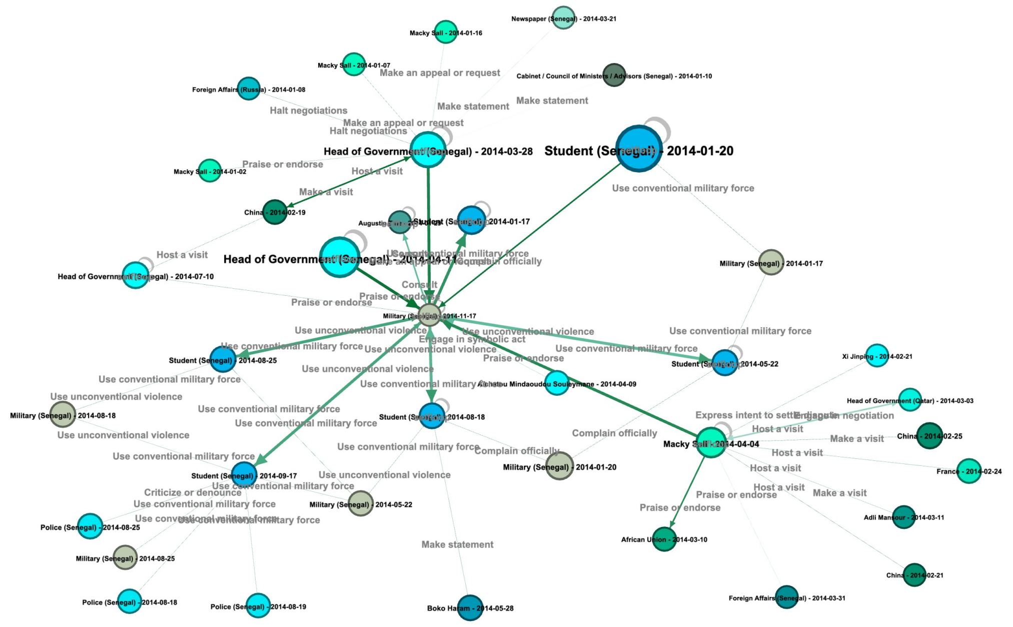


Figure 8: The inference graph for the query (?, praise or endorse, Senegalese military, 2014/11/17) from ICEWS14. The ground truth subject is the head of Senegalese government. The node size indicates the value of the node score. Also, the edges' color indicates the contribution score of the edge, where darkness increases, as the contribution score goes up. The entity at an arrow's tail, the predicate on the arrow, the entity and the timestamp at the arrow's head build a genuine quadruple.

Note that $\mathbf{h}_0^2$ is updated with $\mathbf{h}_1^0, \mathbf{h}_2^0, \mathbf{h}_3^0$ and has nothing to do with $\mathbf{h}_4^0, \mathbf{h}_5^0, \mathbf{h}_6^0, \mathbf{h}_7^0, \mathbf{h}_8^0$, i.e., two-hop neighbors. In comparison, if we have full access to the graph all the time, we are able to aggregate information of two-hop neighbors in the second layer.

If we update the node representations along previously selected edges, the update in second layer changes to:

$$\begin{aligned}
\textbf{Second inference step part a: } & \mathbf{h}_4^2 = \mathbf{h}_4^0 \\
& \mathbf{h}_5^2 = \mathbf{h}_5^0 \\
& \mathbf{h}_6^2 = \mathbf{h}_6^0 \\
& \mathbf{h}_7^2 = \mathbf{h}_7^0 \\
& \mathbf{h}_8^2 = \mathbf{h}_8^0 \\
\textbf{Second inference step part b: } & \mathbf{h}_1^2 = f(\mathbf{h}_1^1, \mathbf{h}_4^2, \mathbf{h}_5^2) \\
& \mathbf{h}_2^2 = f(\mathbf{h}_2^1, \mathbf{h}_7^2, \mathbf{h}_8^2) \\
& \mathbf{h}_3^2 = f(\mathbf{h}_3^1, \mathbf{h}_6^2)
\end{aligned}$$

$$\textbf{Second inference step part c: } \quad \mathbf{h}_0^2 = f(\mathbf{h}_0^1, \mathbf{h}_1^2, \mathbf{h}_2^2, \mathbf{h}_3^2)$$

Thus, the node 1 3 receive messages from node 4 8, i.e. $\mathbf{h}_1^2 = f(\mathbf{h}_1^1, \mathbf{h}_4^2, \mathbf{h}_5^2)$. And they can pass the information to the query subject (node 0), i.e., $\mathbf{h}_0^2 = f(\mathbf{h}_0^1, \mathbf{h}_1^2, \mathbf{h}_2^2, \mathbf{h}_3^2)$.

H SEGMENT OPERATIONS

The degree of entities in temporal KGs, i.e., ICEWS, varies from thousands to a single digit. Thus, the size of inference graphs of each query is also different. To optimize the batch training, we define

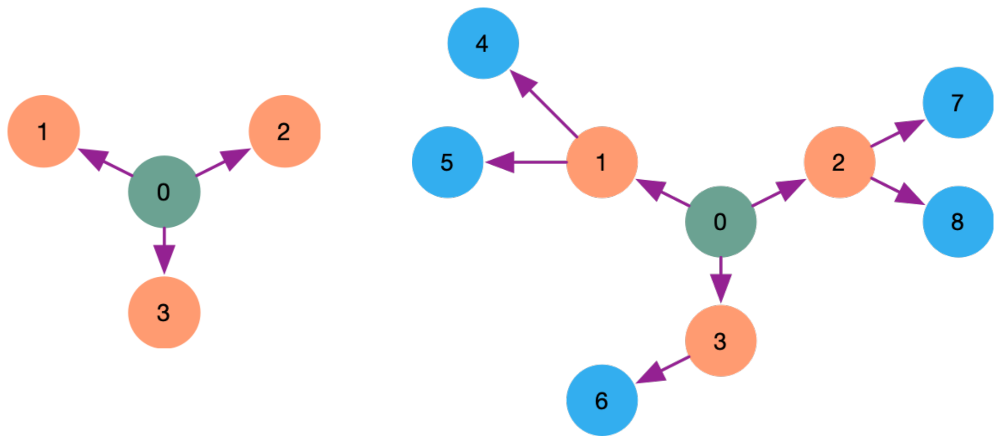


Figure 9: A simple example with two inference steps for illustrating reverse node representation update schema. The graph is initialized with the green node. In the first step (the left figure), orange nodes are selected; and in the second step (the right figure), blue nodes are selected.

an array to record all nodes in inference graphs for a batch of queries. Each node is represented by (inference graph index, entity index, timestamp, node index). The node index is the unique index to distinguish the same node in different inference graphs.

Note that the inference graphs of two queries may overlap, which means they have the same node in their inference graphs. But the query-dependent node representations would be distinct in different inference graphs. To avoid mixing information across different queries, we need to make sure that tensor operations can be applied separately to nodes in different inference graphs. Instead of iterating through each inference graph, we develop a series of segment operations based on matrix multiplication. The segment operations significantly improve time efficiency and reduce the time cost. We report the improvement of time efficiency on ICEWS14 in Table 8. Additionally, we list two examples of segment operations in the following.

Segment Sum Given a vector $\mathbf{x} \in \mathbb{R}^d$ and another vector $\mathbf{s} \in \mathbb{R}^d$ that indicates the segment index of each element in $\mathbf{x}$, the segment sum operator returns the summation for each segment. For example, we have $\mathbf{x} = [3, 1, 5]^T$ and $\mathbf{s} = [0, 0, 1]^T$, which means the first two element of $\mathbf{x}$ belong to the 0^{th} segment and the last elements belongs to the first segment. The segment sum operator returns $[4, 5]^T$ as the output. It is realized by creating a sparse matrix $\mathbf{Y} \in \mathbb{R}^{n \times d}$, where n denotes the number of segments. We set 1 in positions $\{(\mathbf{s}[i], i), \quad \forall i \in \{0, ..., d\}\}$ of $\mathbf{Y}$ and pad other positions with zeros. Finally, we multiply $\mathbf{Y}$ with $\mathbf{x}$ to get the sum of each segment.

Segment Softmax The standard softmax function $\sigma : \mathbb{R}^K \to \mathbb{R}^K$ is defined as:

$$\sigma(\mathbf{z})_i = \frac{\exp{(z_i)}}{\sum_{j=1}^K \exp(z_j)}$$

The segment softmax function has two inputs: $\mathbf{z} \in \mathbb{R}^K$ contains elements to normalize and $\mathbf{s} \in \mathbb{R}^K$ denotes the segment index of each element. It is then defined as:

$$\sigma(\mathbf{z})_i = \frac{\exp{(z_i)}}{\sum_{j \in \{k | s_k = s_i, \forall k \in \{0,...,K\}\}} \exp{(z_j)}}$$

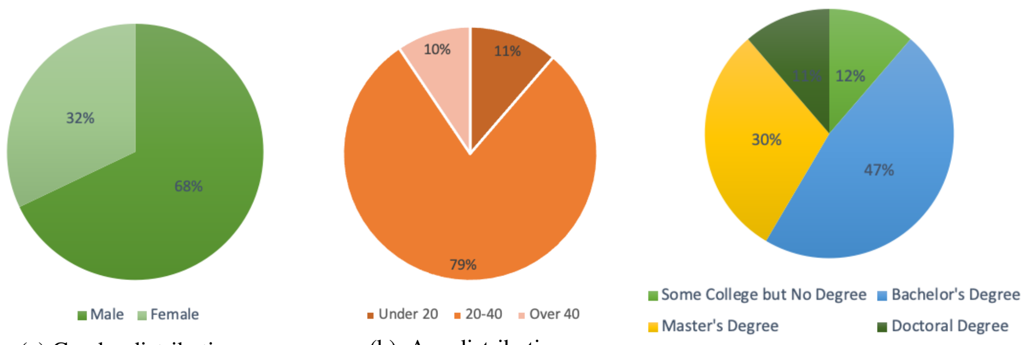
, where s_i denotes the segment that z_i is in.

The segment softmax function can be calculated by the following steps:

1. We apply the exponential function to each element of $\mathbf{z}$ and then apply the segment sum operator to get a denominator vector $\mathbf{d}$. We need broadcast $\mathbf{d}$ such that it aligns with $\mathbf{z}$, which means $\mathbf{d}[i]$ is the summation of segment $\mathbf{s}[i]$.
2. We apply element-wise division between $\mathbf{d}$ and $\mathbf{z}$.

Computations	Time Cost using Iterator	Time Cost using Segment Operation
Aggregation of Node Score	11.75s	0.004s
Aggregation of Entity Score	3.62s	0.026s
Softmax	1.56s	0.017s
Node Score Normalizaion	0.000738s	0.000047s

Table 8: Reduction of Time Cost for a batch on ICEWS14



(a) Gender distribution. (b) Age distribution. (c) Education level.

Figure 10: Information about respondent population.

I Survey

In this section, we provide the online survey (see Section 5.3 of the main body) and the evaluation statistics based on 53 respondents. To avoid biasing the respondents, we did not inform them about the type of project that we have been working on. Further, all questions are permuted at random.

We set up the quiz consisting of 7 rounds. In each round, we sample a query from the test set of ICEWS14/ICEWS0515. Along with the query statement and the ground-truth object, we present the users two arguments extracted from the inference graph with high contribution scores and two arguments with low contribution scores in randomized order. The respondents are supposed to judge the relevance of the statements to the query in two levels, namely relevant or less relevant. There are three questions in each round that ask the participants to give the most relevant statement, the most irrelevant statement, and rank the statements according to their relevance. The answer to the first question is classified as correct if a participant gives one of the two relevant statements as the most relevant statement. Similarly, the answer to the second question is classified as correct if the participant gives one of the two removed statements as the most irrelevant statement. For the relevance ranking task, the answer is right if the participant ranks the two relevant statements higher than the removed statements.

I.1 Population

We provide the information about gender, age, education level of the respondents in Figure 10.

I.2 AI Quiz

You will participate in a quiz consisting of eight rounds. Each round is centered around an international event. Along with the event, we also show you four reasons that argue why the given event happened. While some arguments may be informative and explain the occurrence of this event, others may irrelevant to this event. Your task is to find the most relevant argument and most irrelevant argument, and then sort all four arguments according to their relevance. Don't worry if you feel that you cannot make an informed decision: Guessing is part of this game!

Additional Remarks: Please don't look for external information (e.g., Google, Wikipedia) or talk to other respondents about the quiz. But you are allowed to use a dictionary if you need vocabulary clarifications.

Example

Given an event, please rank the followed arguments according to their relevance to the given event. Especially, please select the most relevant reason, the most irrelevant reason, and rank the relevance from high to low.

Event: French government made an optimistic comment about China on 2014-11-24.

A. First, on 2014-11-20, South Africa engaged in diplomatic cooperation with Morocco. Later, on 2014-11-21, a representative of the Morocco government met a representative of the French government.

B. First, on 2014-11-18, the Chinese government engaged in negotiation with the Iranian government. Later, on 2014-11-21, a representative of the French government met a representative of the Chinese government.

C. On 2014-11-23, the French hosted a visit by Abdel Fattah Al-Sisi.

D. A representative of the French government met a representative of the Chinese government on 2014-11-21.

Correct answer

Most relevant: D Most irrelevant: A Relevance ranking: D B C A

Tasks

1. Event: On 2014-12-17, the UN Security Council accused South Sudan.

A. South Africa engaged in diplomatic cooperation with South Sudan on 2014-12-11.

B. First, on 2014-11-17, Uhuru Muigai Kenyatta accused UN Security Council. Later, on 2014-11-26, the UN Security Council provided military protection to South Sudan.

C. On 2014-12-16, UN Security Council threatened South Sudan with sanctions.

D. South Sudan hosted the visit of John Kerry on 2014-12-16.

Most relevant: Most irrelevant: Relevance ranking:

2. Event: Indonesia police arrested and retained an Indonesia citizen at 2014-12-28.

A. The Indonesia police claimed that an attorney denounced the citizen on 2014-12-10.

B. Zaini Abdullah endorsed the Indonesia citizen on 2014-12-25.

C. The Indonesia police made an optimistic comment on the citizen on 2014-12-14.

D. The Indonesia police investigated the citizen on 2014-12-08.

Most relevant: Most irrelevant: Relevance ranking:

3. Event: A citizen from Greece protested violently against the police of Greece on 2014-11-17.

A. The Greek head of government accused the political party "Coalition of the Radical Left" on 2014-05-25.

B. Greek police refused to surrender to the Greek head of government on 2014-10-15.

C. Greek citizens gathered support on behalf of John Kerry on 2014-11-17.

D. Greek police arrested and detained another Greek police officer on 2014-11-04.

Most relevant: Most irrelevant: Relevance ranking:

4. Event: Raúl Castro signed a formal agreement with Barack Obama on 2014-12-17.

A. First, on 2009-01-28, Dmitry A. Medvedev made statements to Barack Obama. Later, on 2009-01-30, Raúl Castro negotiated with Dmitry A. Medvedev.

B. Raúl Castro visited Angola on 2009-07-22.

C. Raúl Castro hosted a visit of Evo Morales on 2011-09-19.

D. First, on 2008-11-05, Evo Morales hosted a visit of Barack Obama. Later, on 2011-09-19, Raúl Castro appeal for de-escalation of military engagement to Evo Morales.

Most relevant: Most irrelevant: Relevance ranking:

5. Event: The head of the government of Ukraine considered to make a policy option with Angela Merkel on 2015-07-10.

A. First, on 2014-07-04, the armed rebel in Ukraine used unconventional violence to the military of Ukraine. Later, on 2014-07-10, the head of government of Ukraine made statements to the armed rebel in Ukraine.

B. The head of the government of Ukraine expressed intent to meet with Angela Merkel on 2014-10-30.

C. First, on 2014-07-04, the armed rebel in Ukraine used unconventional violence to the military of Ukraine. Later, on 2014-07-19, the head of government of Ukraine made statements to the armed rebel in Ukraine.

D. The head of the government of Ukraine consulted with Angela Merkel on 2015-06-06.

Most relevant: Most irrelevant: Relevance ranking:

6. Event: On 2014-08-09, Ukraine police arrested a member of the Ukraine military.

A. First, on 2014-07-23, a member of Ukraine parliament consulted the head of the Ukraine government. Later, on 2014-07-24, the head of government made a statement to the Ukraine police.

B. First, on 2014-06-25, the military of Ukraine used violence to an armed rebel that occurred in Ukraine. Later, on 2014-07-10, the armed rebel used violence to the Ukraine police.

C. First, on 2005-02-20, the military of Ukraine made a statement to the head of the government of Ukraine. Later, on 2005-07-18, the head of government of Ukraine appealed for a change in leadership of the Ukraine police.

D. On 2014-07-31, the head of the Ukraine government praised the Ukraine police.

Most relevant: Most irrelevant: Relevance ranking:

7. Event: The Office of Business Affairs of Bahrain negotiated with the Labor and Employment Ministry of Bahrain on 2015-07-16.

A. First, on 2014-07-27, the undersecretary of Bahrain made statements to the Labor and Employment Ministry of Bahrain. Later, on 2015-01-21, an officer of Business Affairs of Bahrain signed a formal agreement with the undersecretary of Bahrain.

B. On 2012-01-21, the office of Business Affairs of Bahrain expressed intent to provide policy support to the employees in Bahrain.

C. First, on 2006-11-01, the employees in Bahrain made statements with the special Rapporteurs of the United Nation. Later, on 2011-05-11, the office of Business Affairs of Bahrain reduced relations with the employees in Bahrain.

D. A representative of the Labor and Employment Ministry of Bahrain consulted with a representative of the Office of Business Affairs of Bahrain on 2014-01-31.

Most relevant: Most irrelevant: Relevance ranking:

I.3 Ground Truth Answers

Question 1:

Most relevant: B/C   Most irrelevant: A/D

Relevance ranking: BCAD/BCDA/CBAD/CBDA

Question 2:

Most relevant: A/D   Most irrelevant: B/C

Relevance ranking: ADBC/ADCB/DABC/DACB

Question 3:

Most relevant: B/D   Most irrelevant: A/C

Relevance ranking: BDAC/BDCA/DBAC/DBCA

Question 4:

Most relevant: A/D   Most irrelevant: B/C

Relevance ranking: ADBC/ADCB/DABC/DACB

Question 5:

Most relevant: B/D   Most irrelevant: A/C

Relevance ranking: BDAC/BDCA/DBAC/DBCA

Question 6:

Most relevant: B/C.   Most irrelevant: A/D

Relevance ranking: BCAD/BCDA/CBAD/CBDA

Question 7:

Most relevant: A/D   Most irrelevant: B/C

Relevance ranking: ADBC/ADCB/DABC/DACB

I.4 EVALUATION

The evaluation results of 53 respondents are shown in Figure 11.

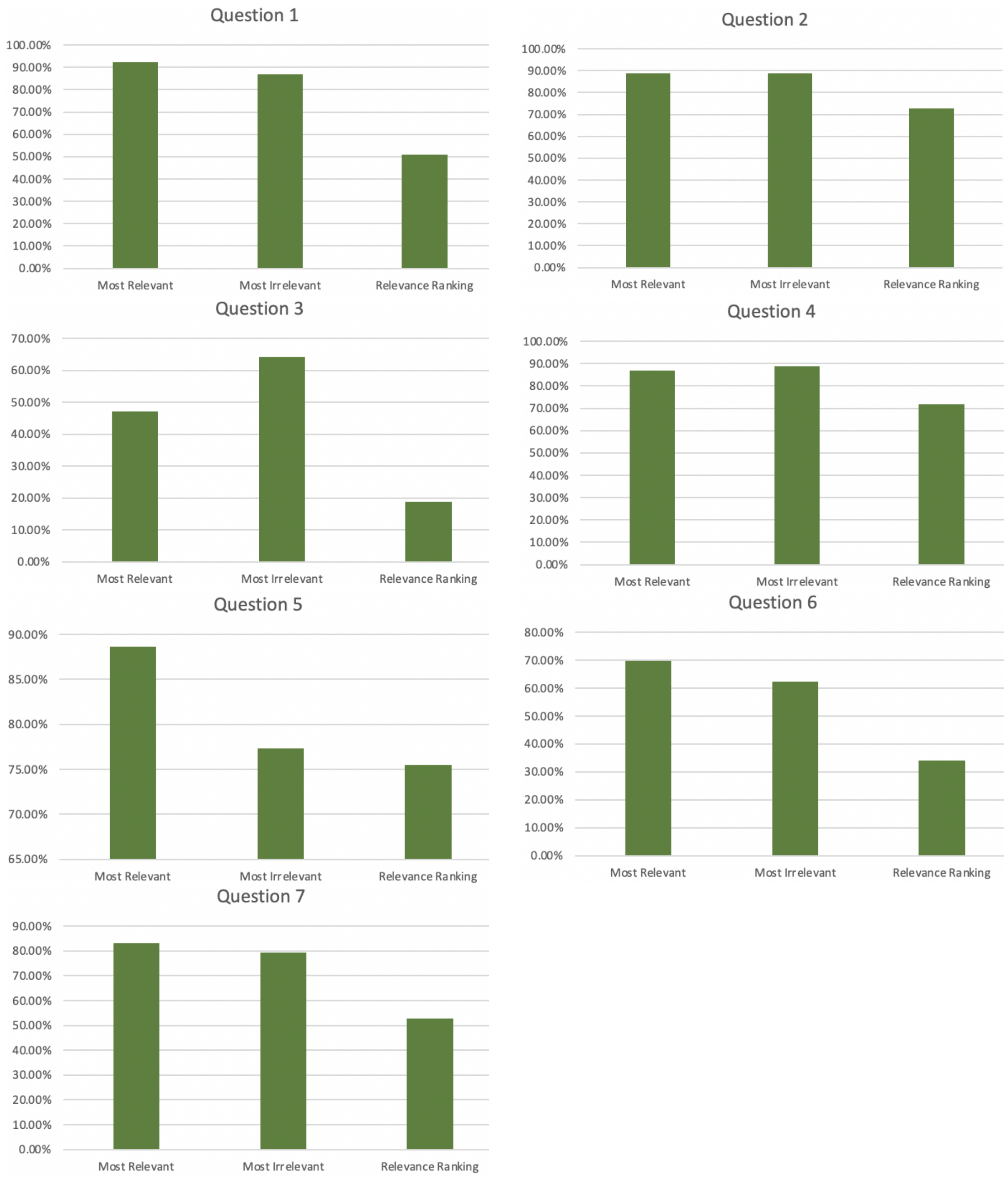


Figure 11: The accuracy of the survey questions.