HiSMatch: Historical Structure Matching based Temporal Knowledge Graph Reasoning

Zixuan Li[1,2,3]*, Zhongni Hou[1,2], Saiping Guan[1,2]†, Xiaolong Jin[1,2], Weihua Peng[3], Long Bai[1,2], Yajuan Lyu[3], Wei Li[3], Jiafeng Guo[1,2], Xueqi Cheng[1,2]

[1]School of Computer Science and Technology, University of Chinese Academy of Sciences;
[2]CAS Key Laboratory of Network Data Science and Technology,
Institute of Computing Technology, Chinese Academy of Sciences; [3]Baidu Inc.

{lizixuan, houzhongni18z, guansaiping, jinxiaolong}@ict.ac.cn
{pengweihua,lvyajuan}@baidu.com

Abstract

A Temporal Knowledge Graph (TKG) is a sequence of KGs with respective timestamps, which adopts quadruples in the form of (*subject*, *relation*, *object*, *timestamp*) to describe dynamic facts. TKG reasoning has facilitated many real-world applications via answering such queries as (*query entity*, *query relation*, *?*, *future timestamp*) about future. This is actually a matching task between a query and candidate entities based on their historical structures, which reflect behavioral trends of the entities at different timestamps. In addition, recent KGs provide background knowledge of all the entities, which is also helpful for the matching. Thus, in this paper, we propose the **Hi**storical **S**tructure **Match**ing (**HiSMatch**) model. It applies two structure encoders to capture the semantic information contained in the historical structures of the query and candidate entities. Besides, it adopts another encoder to integrate the background knowledge into the model. TKG reasoning experiments on six benchmark datasets demonstrate the significant improvement of the proposed HiSMatch model, with up to 5.6% performance improvement in MRR, compared to the state-of-the-art baselines.[1]

1 Introduction

Knowledge Graphs (KGs), which store facts as triples in the form of (*subject*, *relation*, *object*), have been widely applied to many NLP applications, such as question answering (Lan and Jiang, 2020), dialogue generation (He et al., 2017) and recommendation (Wang et al., 2019). However, facts may constantly change over time. Temporal Knowledge Graphs (TKGs) is a kind of KGs that describe such dynamic facts by extending each triple with a timestamp as (*subject*, *relation*, *object*, *timestamp*). Usually, a TKG is represented as a sequence of KG snapshots. The TKG reasoning task is to infer new facts from known ones, which primarily has two settings, interpolation and extrapolation. The former attempts to complete missing facts in history, while the latter aims to predict future facts with historical facts. This paper focuses on the extrapolation setting, which is more challenging and far from being solved (Jin et al., 2020). This task can be seen as answering the query about the future facts (e.g., (*COVID-19*, *Infect*, *?*, *2022-8-1*)) by selecting from all the candidate entities.

The key of answering the queries about future facts is to understand the history thoroughly. All the existing models conduct reasoning based on substructures extracted from the whole history. These substructures can be divided into two types, i.e., query-related history (Jin et al., 2019; Zhu et al., 2021) and candidate-related history (Li et al., 2021b, 2022; Han et al., 2021a; Deng et al., 2020). The former contains the latest historical facts related to the subject and relation in the query, which reflects the behavioral trends of the subject concerning the query relation. The latter contains all the latest historical facts of the candidates without considering the query, which indicates the behavioral trends of all the entities. Both of these two kinds of history are vital to TKG reasoning. Take the query (*COVID-19*, *Infect*, ?, *2022-8-1*) for example, the query-related history contains facts like (*COVID-19*, *Infect*, *, *t*), where *t* is before 2022-8-1. The candidate-related history of a candidate *A*, includes facts reflecting its own behaviors, like (*A*, *, *, *t*) or (*, *, *A*, *t*). In the realistic situation, the occurrence of the fact (*COVID-19*, *Infect*, *A*, *2022-8-1*) is caused by the interactions between these two kinds of history. However, existing models only focus on one kind of history and underestimate the other, which limits their performance on

* This work was done while the first author was doing internship at Baidu Inc.

† Corresponding author.

[1]Our code and data are publicly available for research purpose at https://github.com/Lee-zix/HiSMatch

Findings of the Association for Computational Linguistics: EMNLP 2022, pages 7357 - 7367

TKG reasoning. Overall, it still remains a challenge to model both two kinds of history in a unified framework.

To reduce the computational cost caused by the enormous facts in history, these two kinds of history usually contain one hop facts of the centered entities. Thus, they cannot model the high-order associations among the entities, which is also vital to TKG reasoning.

Motivated by these, we consider both query-related history and candidate-related history under a matching framework and propose the **Hi**storical **S**tructure **Match**ing (**HiSMatch**) model. Specifically, it applies two structure encoders to model the semantic information in the above two kinds of historical structures, respectively. Then, it obtains the matching scores. Both of these two structure encoders contain three components: (1) a structure semantic component to model the structure dependencies among concurrent facts at the same timestamp; (2) a time semantic component to model the time numerical information of the historical facts; (3) a sequential pattern component to mine the behavioral trends from the temporal order information. Additionally, to model the high-order associations among the entities, we consider the most recent KGs as the background knowledge of each query and apply a GCN-based background knowledge encoder to obtain more informative entity representations for the two structure encoders.

Our contributions are summarized as follows:

- We first advocate the importance of modeling both query-related and candidate-related history for TKG reasoning and transform the task into a matching problem between them.
- To solve this problem, we propose HiSMatch to comprehensively capture the information in both historical structures via modeling the structure dependencies among concurrent facts, the time numerical information of historical facts and the temporal order among facts. HiSMatch complementally captures high-order associations among entities by modeling the recent background knowledge.
- Extensive experiments on six commonly used benchmarks demonstrate that HiSMatch achieves significantly better performance (up to 5.6% improvement in MRR) on the TKG reasoning task.

2 Related Work

TKG Reasoning under the interpolation setting focuses on completing the missing facts at past timestamps (Liao et al., 2021; Goel et al., 2020; Wu et al., 2020; Han et al., 2020a; Jiang et al., 2016; Dasgupta et al., 2018; Garcia-Duran et al., 2018; Xu et al., 2021). For example, TTransE (Leblay and Chekol, 2018) extends the idea of TransE (Bordes et al., 2013) by adding the temporal order constraints among facts. Also, HyTE (Dasgupta et al., 2018) projects the entities and relations to time-related hyperplanes to generate time-aware representations. TNTComplEx (Lacroix et al., 2020) performs 4th-order tensor factorization to get the time-aware representations of entities. However, they cannot obtain the representations of the unseen timestamps and are not suitable for the extrapolation setting.

TKG Reasoning under the extrapolation setting aims to predict facts at future timestamps. According to the historical structure the models focus on, the existing models can be categorized into two groups: query-based and candidate-based models.

Query-based models focus on modeling the query-related history. For example, RE-NET (Jin et al., 2020) models the query-related subgraph sequence. GHNN (Han et al., 2020c) introduces the temporal point process to model the precise time information and takes the 1-hop subgraphs of the query entity into consideration. CyGNet (Zhu et al., 2021) captures repetitive patterns by modeling repetitive facts with the same subject and relation to the query. xERTE (Han et al., 2020b) learns a dynamic pruning procedure to find the query-related subgraphs. CluSTeR (Li et al., 2021a) and TITer (Sun et al., 2021) both adopt reinforcement learning to discover query-related paths in history.

Candidate-based models encode the latest historical facts of all the candidate entities without considering the query, and query are considered only in the decoding phase. RE-GCN and its extension CEN (Li et al., 2021b, 2022) designs an evolutional model to get the representations of all the candidates by modeling history at a few latest timestamps. TANGO (Han et al., 2021a) utilizes neural ordinary differential equations to model the structure information for each candidate entity. Glean (Deng et al., 2020) introduces unstructured textual information to enrich the candidate-related history.

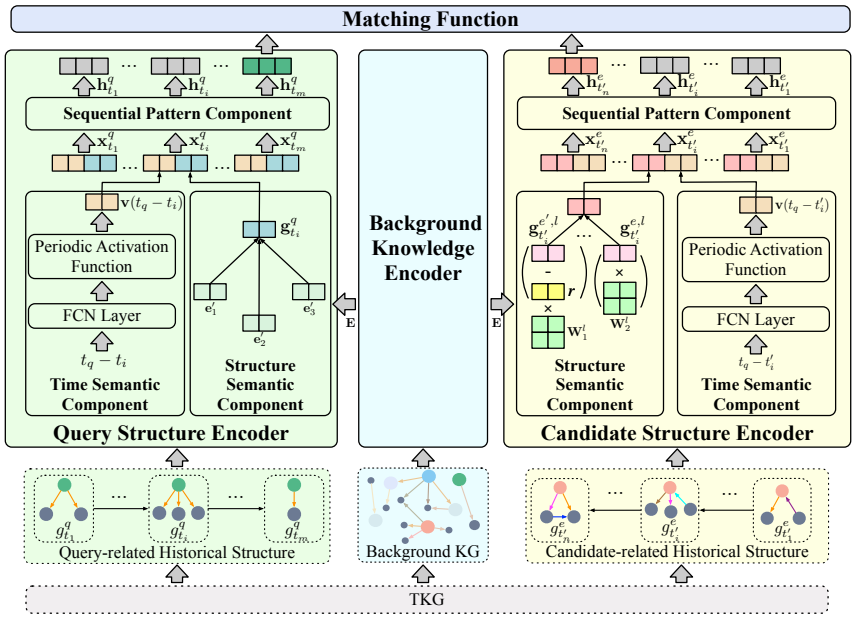

Figure 1: An illustrative diagram of the proposed HiSMatch model.

Above all, none of the existing models focus on both two kinds of history in a unified framework. HiSMatch considers these two kinds of history under the matching framework and takes the advantages of both kinds of models.

3 Problem Formulation

A TKG $G = \{G_0, ..., G_t, ..., G_T\}$ is a sequence of KGs, each of which contains facts occurred at timestamp t, i.e., $G_t = \{\mathcal{E}, \mathcal{R}, \mathcal{F}_t\}$, where $\mathcal{E}$ is the set of entities, $\mathcal{R}$ is the set of relations and $\mathcal{F}_t$ is the set of facts that occurred at t. Each fact is a quadruple (e_s, r, e_o, t), where $e_s, e_o \in \mathcal{E}$ and $r \in \mathcal{R}$. For each fact in TKG, we add the inverse quadruple (e_o, r^{-1}, e_s, t) into TKG, correspondingly. The TKG reasoning task aims to predict the missing object via answering a query $q = (e_q, r_q, ?, t_q)$ with the historical KGs given. Note that, when predicting the missing subject of a query $q = \{?, r_q, e_q, t_q\}$, we can convert the query into $q = \{e_q, r_q^{-1}, ?, t_q\}$.

4 The HiSMatch model

HiSMatch aims to captures the semantic similarity contained in the query-related and candidate-related historical structures. For each query time, it first embeds the background knowledge into the initial entity representations. With the initial representations as input, it maps the semantic information in these two historical structures into the vectorized representations of structures. Based on the structure representations, matching scores are calculated.

Thus, as shown in Figure 1, HiSMatch consists of four parts: the query structure encoder, the candidate structure encoder, the background knowledge encoder, and the matching function. First, two kinds of historical structures and a background knowledge graph are derived from the TKG. Then, the background knowledge encoder gets the representations of the entities with the background knowledge graph as input (Section 4.3). With the learned representations as input, two structure encoders use three components to integrate three kinds of semantic information into the representations of query-related structure and candidate-related structure, respectively (Section 4.1 and 4.2). Finally, the matching function calculates the scores between the query and candidates based on the representations of their historical structures (Section 4.4).

4.1 Query Structure Encoder

The query-related historical structure should reflect the behavioral trends of the query. Motivated by this, for a query $q = (e_q, r_q, ?, t_q)$, the query-related historical structure consists of the latest historical facts with the same subject e_q and relation r_q. These facts co-occuring at the same timestamp t form a subgraph g_t^q centered on e_q. Then, we obtain a subgraph sequence $\{g_{t_1}^q, ..., g_{t_i}^q, ..., g_{t_m}^q\}$, where $t_1 < ... < t_i < ... < t_m < t_q$ and m is the maximum length of the sequence.

Three kinds of information are vital in the above historical structure, namely, the structure semantic

information of each subgraph, the time numerical information of each subgraph, and the temporal order information across subgraphs. To model these three kinds of information, we design three components as follows:

Structure Semantic Component. The structure semantic information captures the associations among the query entity and other entities through the query relation and implies possible answer entities. Since all the concurrent facts, which having the same subject and relation to the query, form a one-hop homogeneous graph, we simply perform mean pooling over all the neighbor entities in each subgraph $g^q_{t_i}$ to get the structure semantic representation $\mathbf{g}^q_{t_i}$ of the subgraph,

$$\mathbf{g}^q_{t_i} = \frac{1}{|N^q_{t_i}|} \sum_{e \in N^q_{t_i}} \mathbf{e}, \tag{1}$$

where $N^q_{t_i}$ is the set of the neighborhood of the query entity e_q in $g^q_{t_i}$ and $\mathbf{e}$ is the representation of each entity e calculated by the background knowledge encoder (see Section 4.3).

Time Semantic Component. Previous works (Jin et al., 2019, 2020) only consider the temporal order of the facts but ignore their time numerical information. A much earlier fact and a recent one contribute equally when they have the same order in the subgraph sequence. Actually, the recent fact is more important. Motivated by this, we model the time numerical information by encoding the time interval $d = t_q - t_i$, into the time representation $\mathbf{v}(d)$. However, giving each time interval a learnable time representation always meets the time sparsity problem (i.e., the time interval used in the test phase may not exist in the training phase). Thus, we model any time interval by rescaling a learnable time unit $\mathbf{w}_t$ with a time bias $\mathbf{b}_t$,

$$\mathbf{v}(d) = cos(d\mathbf{w}_t + \mathbf{b}_t). \tag{2}$$

Since some facts occur periodically, such as elections, we additionally apply the periodic activation function, i.e., cosine function, on $\mathbf{v}(d)$.

Sequential Pattern Component. Furthermore, the temporal order information in the subgraph sequence implies sequential patterns of the query entity. To integrate the sequential patterns into the representation of the query, we use Gated Recurrent Unit (GRU) to model the subgraph sequence. First, for every timestamp t_i $(i = 1, 2, ..., m)$, we concatenate structure semantic representation and the time semantic representation from the above two components as the input of GRU,

$$\mathbf{x}^q_{t_i} = [\mathbf{g}^q_{t_i}; \mathbf{v}(d)]. \tag{3}$$

Then these representations $\{\mathbf{x}^q_{t_1}, ..., \mathbf{x}^q_{t_i}, ..., \mathbf{x}^q_{t_m}\}$ are fed into GRU recursively,

$$\mathbf{h}^q_{t_i} = GRU(\mathbf{h}^q_{t_{i-1}}, \mathbf{x}^q_{t_i}), \tag{4}$$

where $i \in \{1, 2, ..., m\}$ and $\mathbf{h}^q_{t_0}$ is the randomly initialized hidden representations for GRU. The final representation of query $(e_q, r_q, ?, t_q)$ is the ouput of GRU at the final step, i.e., $\mathbf{h}^q_{t_q} = \mathbf{h}^q_{t_m}$.

4.2 Candidate Structure Encoder

The candidate-related historical structure reflects the behavioral trends of each candidate entity. For each candidate entity e, we use its 1-hop subgraphs at latest historical timestamps to form a subgraph sequence $\{g^e_{t'_1}, ..., g^e_{t'_i}, ..., g^e_{t'_n}\}$. n is the maximum length of the sequence. Actually, this structure is similar to the query-related historical structure, the difference is that each subgraph in the sequence is multi-relational. Therefore, we use an encoder similar to the query structure encoder, except the calculation of the structure semantic representation of each subgraph. More specifically, we adopt the CompGCN (Vashishth et al., 2019) instead of the mean pooling operation, to capture the semantic information of different relations[2]. The representation of the candidate entity e at timestamp t_i, is calculated by a CompGCN with ω_1 layers. Thus, the representation of the $(l+1)$-th layer is

$$\begin{aligned}\mathbf{h}^{e,l+1}_{t'_i} = & f\Big(\frac{1}{c_e} \sum_{e' \in N^e_{t'_i}} \mathbf{W}^l_1(\mathbf{h}^{e',l}_{t'_i} - \mathbf{r}) \\ & + \mathbf{W}^l_2 \mathbf{h}^{e,l}_{t'_i}\Big),\end{aligned} \tag{5}$$

where $\mathbf{r}$ is the representation of relation r; $\mathbf{h}^{e',l}_{t'_i}$ denotes the l-th layer representation of entity e' at t'_i timestamp; $\mathbf{W}^l_1$, $\mathbf{W}^l_2$ are the weight matrices of the l-th layer; c_e is a normalization constant, which equals to the in-degree of entity e. Note that the input representations of all entities are also

[2]Note that, the CompGCN layers can be replaced by other relation-aware GCNs. We further analyze them in Section5.4

calculated by the background knowledge encoder, which will be introduced in Section 4.3.

Then, the structure semantic representation of the subgraph $g^e_{t'_i}$, i.e., $\mathbf{g}^e_{t'_i}$, equals to the representation of the centered candidate e from the last layer of CompGCN, i.e., $\mathbf{g}^e_{t'_i} = \mathbf{h}^{e,\omega_1}_{t'_i}$.

Similar to the query structure encoder, we use another GRU to model the subgraph sequence. The input of GRU at timestamp t'_i is

$$\mathbf{x}^e_{t'_i} = [\mathbf{g}^e_{t'_i}; \mathbf{v}(d)], \tag{6}$$

where $d = t_q - t'_i$ and $\mathbf{v}(d)$ is the time interval representation calculated by a shared time semantic component introduced in Section 4.1. Finally, the output of GRU at the last step t'_n is used as the representation of candidate entity e at t_q, i.e., $\mathbf{h}^e_{t_q} = \mathbf{h}^e_{t'_n}$.

4.3 Background Knowledge Encoder

The above two historical structures are local information centered on the query entity or the candidate entity, which focus on describing the behavioral trends of the entities. However, these two kinds of structures may miss some important entities that have high-order associations with the query entity or the candidate entity in the whole TKG. Since the recent history is more important, for query timestamp t_q, we gather the latest k KGs into a cumulative graph $\mathcal{G}_{t_q}$, called background knowledge graph. Formally, $\mathcal{G}_{t_q} = \{\mathcal{E}, \mathcal{R}, \hat{\mathcal{F}}_{t_q} = \{(e_s, r, e_o) | (e_s, r, e_o, t) \in \mathcal{F}_t, t_q - k \leq t < t_q\}\}$, where $\hat{\mathcal{F}}_{t_q}$ is a set of facts. We adopt another CompGCN with ω_2 layers to model it since it is also a multi-relational graph. The representations of all entities are calculated as follows,

$$\mathbf{E} = CompGCN(\mathbf{E}', \mathbf{R}, \mathcal{G}_{t_q}), \tag{7}$$

where $\mathbf{E}'$ is the randomly initialized entity representation matrix and $\mathbf{E}$ is used as the input entity representation matrix of the aforementioned two structure semantic components. $\mathbf{R}$ is the relation representation matrix, which is shared with the structure semantic component. For the entities that have no facts in the background knowledge graph, an self-loop operation is conduct to get its representation. Note that, the background knowledge graph changes along the query time and $\mathbf{E}$ is different for different t_q.

4.4 Matching Function

With the representation $\mathbf{h}^q_{t_q}$ of the query and the representation $\mathbf{h}^e_{t_q}$ of each candidate entity e at timestamp t_q as input, the matching function calculates the score of the quadruple (e_q, r_q, e, t_q). As previous work (Vashishth et al., 2019; Li et al., 2021b) shows the convolutional score functions get good performance on reasoning tasks, ConvTransE (Shang et al., 2019) is chosen as the matching fucntion, which contains 1D convolution and fully-connected layers, denoted by $ConvTransE(\cdot)$. To describe the behavioral information of the query entity in the query representation, an sum-up operation is performed between $\mathbf{h}^q_{t_q}$ and $\mathbf{h}^{e_q}_{t_q}$. Then, the score for eacht candidate entity e is calculated as follows:

$$\begin{aligned}\phi(e_q, r_q, e, t_q) = \\ \sigma(\mathbf{h}^e_{t_q} ConvTransE(\mathbf{h}^q_{t_q} + \mathbf{h}^{e_q}_{t_q}, \mathbf{r_q})),\end{aligned} \tag{8}$$

where $\sigma(\cdot)$ is the sigmoid function.

4.5 Training Details

The training objective is to minimize the cross-entropy loss:

$$L(\Theta) = \sum_{t=0}^{T} \sum_{(e_s, r, e_o, t) \in \mathcal{F}_t} \sum_{e \in \mathcal{E}} y^e_t \log \phi(e_s, r, e, t), \tag{9}$$

where T is the number of timestamps in the training set; $y^e_t = 1$ if e equals to e_o, otherwise 0; $\phi(e_s, r, e, t)$ is the matching score between the query $(e_s, r, ?, t)$ and the candidate entity e; Θ are all the model parameters.

5 Experiments

We compare HiSMatch with a number of baselines on six datasets to validate its effectiveness. In addition, we conduct ablation study to analyze the importance of its different parts. We also evaluated the effects of different kinds of GCN layers in the candidate structure encoder and the background knowledge encoder. Besides, we study the maximum time interval that HiSMatch models.

5.1 Experimental setup

5.1.1 Datasets

To evaluate the effectiveness of HiSMatch, we use the following six benchmark TKGs: ICEWS14 (Li et al., 2021b), ICEWS14* (Han et al., 2020b), ICEWS18 (Jin et al., 2020), ICEWS05-15 (Li et al., 2021b), GDELT (Jin et al., 2020) and WIKI (Leblay and Chekol, 2018). The first four datasets with the time granularity of 24 hours

Datasets	$\lvert\mathcal{V}\rvert$	$\lvert\mathcal{R}\rvert$	$\lvert\mathcal{E}_{train}\rvert$	$\lvert\mathcal{E}_{valid}\rvert$	$\lvert\mathcal{E}_{test}\rvert$	Time granularity	Snapshot numbers
ICEWS14	7,128	230	74,845	8,514	7,371	24 hours	365
ICEWS14*	7,128	230	63,685	13,823	13,222	24 hours	365
ICEWS18	23,033	256	373,018	45,995	49,545	24 hours	365
ICEWS05-15	10,094	251	368,868	46,302	46,159	24 hours	4017
GDELT	7,691	240	1,734,399	238,765	305,241	15 mins	2975
WIKI	12,554	24	539,286	67,538	63,110	1 year	232

Table 1: Statistics of the datasets ($|\mathcal{E}_{train}|$, $|\mathcal{E}_{valid}|$, $|\mathcal{E}_{test}|$ are the sizes of training, validation, and test sets.).

were extracted from the large-scale event-based database, Integrated Crisis Early Warning System. The ICEWS14, ICEWS14* and ICEWS18 datasets contain events in 2014 and 2018, respectively, and the ICEWS05-15 dataset contains events occurred from 2005 to 2015. GDELT is extracted from the global database of events, language, and tone (Leetaru and Schrodt, 2013), which has a fine-grained time granularity of 15 minutes. WIKI is a TKG with the largest time granularity of one year. The statistics of the datasets are listed in Table 1.

5.1.2 Evaluation Metrics

We employ widely used $Hits@N$ and Mean Reciprocal Rank (MRR) to evaluate the performance of the models. $Hits@N$ measures the proportion of correct entities whose scores rank less than or equal to N. In this paper, $N \in \{1, 3, 10\}$, i.e., the results in terms of $Hits@1$, $Hits@3$, and $Hits@10$ are reported. MRR measures the average of these reciprocal ranks and is the most typical metric for TKG reasoning. Previous work (Han et al., 2020b, 2021a; Li et al., 2021a,b) points out that the traditional filtered setting is flawed as it ignores the time of the fact and filters all facts with the same entity and relation before ranking. Actually, only the facts occurring at the same time should be filtered. Thus, we calculate the results under the more reasonable time-aware filtered setting following Sun et al. (2021); Han et al. (2021a), which only filters out the quadruples occurring at the query time.

5.1.3 Baselines

The HiSMatch model is compared with three kinds of models: KG reasoning models, interpolation TKG reasoning models and extrapolation TKG reasoning models. For the KG reasoning models, DistMult (Yang et al., 2015), ComplEx (Trouillon et al., 2016), ConvE (Dettmers et al., 2018), ConvTransE (Shang et al., 2019), RotatE (Sun et al., 2018) are compared. For the TKG reasoning models, HiSMatch is compared to the interpolation TKG reasoning models, including TTransE (Leblay and Chekol, 2018), TA-DistMult (Garcia-Duran et al., 2018), DE-SimplE (Goel et al., 2020) and TNTComplEx (Lacroix et al., 2020). Besides, as for the extrapolation TKG reasoning models, we choose the newest seven baselines including TANGO-DistMult (Han et al., 2021b), TANGO-Tucker (Han et al., 2021b), xERTE (Han et al., 2020b), TITer (Sun et al., 2021), CyGNet (Zhu et al., 2021), RE-NET (Jin et al., 2020) and RE-GCN (Li et al., 2021b).

5.1.4 Implementation Details

The dimensions of the entities and relations are set to 128, and the dimension of the time semantic representation is set to 32 for all the datasets. For the structure semantic encoders, the optimal lengths of historical structures of query m and candidate entities n are equal in this paper. For ICEWS14, ICEWS18, ICEWS05-15 and GDELT, they are set to 5; while 6 for ICEWS14* and 1 for WIKI; the number of layers ω_1 of the CompGCN is set to 1 for GDELT and 2 for the other datasets; the GRU layers is set to 1 for all the datasets and the output dimension of the GRU unit is set to 128. For the background knowledge encoder, the latest KG number k is experimentally set to 4, 1, 2, 1, 2, 2 for ICEWS14, ICEWS18, GDELT, ICEWS14*, ICEWS05-15, and WIKI, respectively; we set the dropout rate for each layer to 0.2 and the layer of CompGCN in the background knowledge encoder, ω_2, to 2, for all the datasets. For the matching function, the number of kernels is set to 50, the kernel size is set to 2×3, and the dropout rate is set to 0.2, for all the datasets. Adam (Kingma and Ba, 2014) is adopted for parameter learning with the learning rate 0.001. All the experiments are carried out on 32G Tesla V100.

5.2 Experimental Results

The experimental results of HiSMatch and all the baselines on TKG reasoning are presented in Tables 2 and 3. It can be seen that HiSMatch consis-

Model	ICE14				ICE05-15				GDELT			
	MRR	H@1	H@3	H@10	MRR	H@1	H@3	H@10	MRR	H@1	H@3	H@10
DistMult	15.44	10.91	17.24	23.92	17.95	13.12	20.71	29.32	8.68	5.58	9.96	17.13
ComplEx	32.54	23.43	36.13	50.73	32.63	24.01	37.50	52.81	16.96	11.25	19.52	32.35
ConvE	35.09	25.23	39.38	54.68	33.81	24.78	39.00	54.95	16.55	11.02	18.88	31.60
ConvTransE	33.80	25.40	38.54	53.99	33.03	24.15	38.07	54.32	16.20	10.85	18.38	30.86
RotatE	21.31	10.26	24.35	44.75	24.71	13.22	29.04	48.16	13.45	6.95	14.09	25.99
TTransE	13.72	2.98	17.70	35.74	15.57	4.80	19.24	38.29	5.50	0.47	4.94	15.25
TA-DistMult	25.80	16.94	29.74	42.99	24.31	14.58	27.92	44.21	12.00	5.76	12.94	23.54
DE-SimplE	33.36	24.85	37.15	49.82	35.02	25.91	38.99	52.75	19.70	12.22	21.39	33.70
TNTComplEx	34.05	25.08	38.50	50.92	27.54	9.52	30.80	42.86	19.53	12.41	20.75	33.42
TANGO-DistMult	-	-	-	-	40.71	31.23	45.33	58.95	-	-	-	-
TANGO-Tucker	-	-	-	-	42.86	32.72	48.14	62.34	-	-	-	-
xERTE	40.02	32.06	44.63	56.17	46.62	37.84	52.31	63.92	18.09	12.30	20.06	30.34
TITer	40.87	32.28	45.45	57.10	47.69	37.95	52.92	65.81	15.46	10.98	15.61	24.31
CyGNet	35.05	25.73	39.01	53.55	36.81	26.61	41.63	56.22	18.48	11.52	19.57	31.98
RE-NET	36.93	26.83	39.51	54.78	43.32	33.43	47.77	63.06	19.62	12.42	21.00	34.01
RE-GCN	40.39	30.66	44.96	59.21	48.03	37.33	53.85	68.27	19.64	12.42	20.90	33.69
HiSMatch	**46.42**	**35.91**	**51.63**	**66.84**	**52.85**	**42.01**	**59.05**	**73.28**	**22.01**	**14.45**	**23.80**	**36.61**

Table 2: Performance (in percentage) on ICEWS14, ICEWS05-15 and GDELT. We average the results of HiSMatch over five runs and the best results are in bold.

Model	ICE14*				ICE18				WIKI			
	MRR	H@1	H@3	H@10	MRR	H@1	H@3	H@10	MRR	H@1	H@3	H@10
DistMult	16.16	11.42	17.94	25.30	11.51	7.03	12.87	20.86	10.89	8.92	10.97	16.82
ComplEx	21.28	14.49	23.11	35.20	22.94	15.19	27.05	42.11	24.47	19.69	27.28	34.83
ConvE	34.50	24.83	38.56	53.88	24.51	16.23	29.25	44.51	14.52	11.44	16.36	22.36
ConvTransE	33.47	25.15	38.15	53.30	22.11	13.94	26.44	42.28	10.60	8.67	11.94	16.93
RotatE	20.88	10.26	23.90	44.03	12.78	4.01	14.89	31.91	46.10	41.89	49.65	51.98
TTransE	13.43	3.11	17.32	34.55	8.31	1.92	8.56	21.89	29.27	21.67	34.43	42.39
TA-DistMult	26.47	17.09	30.22	45.41	16.75	8.61	18.41	33.59	44.53	39.92	48.73	51.71
DE-SimplE	32.67	24.43	35.69	49.11	19.30	11.53	21.86	34.80	45.43	42.60	47.71	49.55
TNTComplEx	32.12	23.35	36.03	49.13	21.23	13.28	24.02	36.91	45.03	40.04	49.31	52.03
TANGO-DistMult	24.70	16.36	27.26	41.35	26.65	17.92	30.08	44.09	51.15	49.66	52.16	53.35
TANGO-Tucker	26.25	17.30	29.07	44.18	28.68	19.35	32.17	47.04	50.43	48.52	51.47	53.58
xERTE	40.79	32.70	45.67	57.30	29.98	22.05	33.46	44.83	71.14	68.05	76.11	79.01
TITer	41.73	32.74	46.46	58.44	29.98	22.05	33.46	44.83	75.50	72.96	77.49	79.02
CyGNet	32.73	23.69	36.31	50.67	24.93	15.90	28.28	42.61	33.89	29.06	36.10	41.86
RE-NET	38.28	28.68	41.43	54.52	28.81	19.05	32.44	47.51	49.66	46.88	51.19	53.48
RE-GCN	41.78	31.58	46.65	61.51	30.58	21.01	34.34	48.75	77.55	73.75	80.38	83.68
HiSMatch	**45.82**	**35.84**	**50.79**	**65.08**	**33.99**	**23.91**	**37.90**	**53.94**	**78.07**	**73.89**	**81.32**	**84.65**

Table 3: Performance (in percentage) on ICEWS14*, ICESW18 and WIKI.

tently outperforms all the baselines on all the six TKGs, which indicates its effectiveness and superiority. Especially on ICEWS14 and ICEWS05-15, HiSMatch achieves the most significant improvements of 5.6% and 4.8% in MRR, respectively. In more detail, we have the following observations: (1) HiSMatch outperforms all the KG reasoning models because it can capture both the time information for each fact and sequential patterns in TKGs; (2) HiSMatch performs much better than those interpolation models because they cannot learn representations for unseen timestamps; (3) More importantly, HiSMatch gets better results than all the extrapolation baselines, which proves the superiority of modeling both two kinds of history, i.e., query-related history and candidate-related history; (4) It can be seen that the baselines (e.g., TITer) focusing on query-related history are usually strong on precision and get good results on $Hits@1$ while the baselines focus on the candidate-related history (e.g., RE-GCN) are more capable on recall and get good results on $Hits@10$. In a word, by transforming the TKG reasoning task into a matching task, HiSMatch utilizes both two kinds of history more comprehensively. Moreover, HiSMatch captures more high-order associations via the background knowledge graph. Therefore, it gets the best performances in all the metrics.

By conducting experiments on six datasets with different time granularities, we found that the time granularity partly determines what is vital to the TKG reasoning task. Take the two most typical datasets for example, (1) GDELT has the most fine-grained time granularity (15 minutes) and the results of all the baselines are similarly poor, com-

pared with those of the other datasets. There are more timestamps in history when the time granularity gets more fine-grained, which requires the model to capture history at more timestamps. Under the matching framework, HiSMatch can capture longer historical information than candidate-based models and more comprehensive history than query-based models. Thus, it gets better results (2.3% in MRR); (2) Contrary to GDELT, WIKI has the largest time granularity (1 year). In this situation, the behavioral trends implied in history at fewer timestamps are vital for the reasoning. Moreover, there are more structural dependencies in each KG due to the large time granularity. Thus, RE-GCN focuses on modeling the global structure at the latest a few timestamps and gets strong performance on this data. Still, HiSMatch outperforms it by modeling the two kinds of substructures and the background knowledge.

5.3 Ablation Study

To further analyze how each part of HiSMatch contributes to the final results, we report the MRR results of the HiSMatch variants on the validation sets on three typical datasets, namely, ICEWS14, ICEWS18 and WIKI, in Table 4.

Impact of the Query Structure Encoder. To demonstrate how the query structure encoder contributes to the final results of HiSMatch, we remove the query structure encoder and use the representation of the query entity from the candidate structure encoder as the representation of the query. The results are denoted as *-query* in Table 4. It can be seen that *-query* performs consistently worse than HiSMatch on all the datasets. It is because that query-related historical structure can model the query more accurately by modeling the repetitive facts focused on the query relation.

Impact of the Candidate Structure Encoder. The results denoted as *-candidate* in Table 4 demonstrate the performance of HiSMatch without modeling the candidate-related history. More specifically, we directly add a fully connection layer after the query structure encoder to get the scores of all entities following (Jin et al., 2020). It can be observed that ignoring the candidate-related historical structure has a great impact on the results. Candidate-related history contains rich information that describes the behavioral trends about all candidate entities, which is helpful to select the correct answer. Especially on WIKI, the dataset with the largest time granularity as mentioned in Section 5.2, entities have more associations among each other at each timestamp and thus contain rich behaviors.

Model	ICE14	ICE18	WIKI
HiSMatch	**47.89**	**35.18**	**79.91**
-query	44.86 (-3.0)	33.16 (-2.0)	77.36 (-2.6)
-cantidate	40.04 (-7.9)	29.12 (-6.1)	63.26 (-16.7)
-background	44.26 (-3.6)	32.87 (-2.3)	73.54 (-6.4)
-time	45.01 (-2.9)	33.25 (-1.9)	78.60 (-1.3)

Table 4: MRR results (in percentage) by different variants of HiSMatch on three datasets.

Impact of the Background Knowledge Encoder. *-background* in Table 4 denotes a variant of HiSMatch that uses the learned representations of entities without the background knowledge encoder. Note that, in the training phase, the randomly initialized representations of entities will be learned and updated. In the test phase, the model uses the learned representations of entities as the input. It can be observed that the performance of *-background* is worse than HiSMatch on all the datasets, especially on the WIKI, which has a time granularity of one year. There are more high order associations in the background knowledge graph. Thus, the background knowledge is more important for WIKI than other datesets.

Impact of the Time Semantic Component. To demonstrate how the time semantic component contributes to the final results, we remove the time semantic component and only use the outputs of structure semantic component as the inputs of the sequential pattern component. The results are denoted as *-time* in Table 4. It can be seen that the time semantic component is useful on all the datasets. It is because the time semantic component describes the time numerical information so that it can help HisMatch to distinguish different time intervals between the history and the query.

Model	ICE14	ICE18	WIKI
HiSMatch (CompGCN)	**47.89**	**35.18**	**79.91**
HiSMatch (CompGCN-mult)	46.12	34.45	73.45
HiSMatch (RGCN)	47.03	35.08	74.83
HiSMatch (KBAT)	47.53	34.78	77.12

Table 5: Performance (in percentage) of HisMatch with different kinds of GCNs.

5.4 Comparative Study on Different GCNs

To further study the impact of different kinds of GCNs in the candidate structure

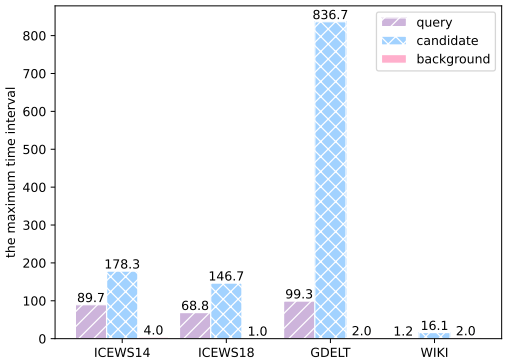

Figure 2: Statistic of maximum time intervals in history on four datasets.

encoder and the background knowledge encoder, we replace CompGCN in these two encoders with CompGCN-mult (Vashishth et al., 2019), RGCN (Schlichtkrull et al., 2018) and KBAT (Nathani et al., 2019). The MRR results on the validation sets of ICE14, ICE18, and WIKI are reported in Table 5. It can be seen that HiSMatch (CompGCN) gets the best performance. For ICE14 and ICE18, the two datasets with the time granularities of one day, the structure dependencies are relatively simple. Thus, different GCNs get similar performances. While for WIKI, the dateset with the time granularities of one year, there are more structural dependencies in the candidate-related history and the background knowledge graph. Therefore, the performance gap caused by the capabilities of GCNs becomes more significant.

5.5 Study on Maximum Time Interval

To explore the maximum time intervals between the query time and the history that HiSMatch models, we conduct statistics on the maximum time interval of historical facts in the query-related historical structure ($\Delta t = t_q - t_1$) and candidate-related historical structure ($\Delta t' = t_q - t'_1$) under the optimal parameters (We report the average maximum time interval on the validation sets). We also report the time interval of background knowledge graphs (k) for comparison. As shown in Figure 2, Δt and $\Delta t'$ are both much larger than k. The results demonstrate that the two historical structures can modeling the long-term behaviors of the query and candidates. The background knowledge graph focuses on model high-order associations among all the facts at the latest a few timestamps, which models the global structure dependencies in a much shorter time interval. It can be seen that Δt on GDELT is more than 800 and the value is only around 16 on WIKI, which verifies the discussion in Section 5.2.

6 Conclusion

In this paper, we considered both two kinds of history, namely, the query-related history and the candidate-related history in TKG reasoning and transformed the task into a matching problem between them for the first time. We further proposed the HiSMatch model, which applies two structure encoders to calculate the representations of historical structures of the queries and candidates, respectively. Each encoder contains a structure semantic component to model the concurrent structure among entities, a time semantic component to model the time numerical information of facts, and a sequence pattern component to capture the temporal orders. Besides, HiSMatch integrates the background knowledge into the representations of entities. Experimental results on six benchmark datasets demonstrate the superiority of HiSMatch.

7 Limitations

The limitations of this work can be concluded into two points: (1) HiSMatch uses a heuristic history finding strategy to get two kinds of history, which may lose some critical facts. Although it uses the background knowledge encoder to consider more historical facts, a learnable history finding strategy is more helpful. (2) HiSMatch is an initial attempt to apply the matching framework to solve the TKG reasoning task using two separate encoders for each kind of history, which fails to model the interactions between the two kinds of history explicitly. Designing a cross-encoder to match history more comprehensively is a good direction for future studies.

Acknowledgments

The work is supported partly by the National Natural Science Foundation of China under grants U1911401, 62002341 and 61772501, the GFKJ Innovation Program, Beijing Academy of Artificial Intelligence under grant BAAI2019ZD0306, and the Lenovo-CAS Joint Lab Youth Scientist Project.

References

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. *Advances in neural information processing systems*, 26.

Shib Sankar Dasgupta, Swayambhu Nath Ray, and Partha Talukdar. 2018. Hyte: Hyperplane-based temporally aware knowledge graph embedding. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pages 2001–2011.

Songgaojun Deng, Huzefa Rangwala, and Yue Ning. 2020. Dynamic knowledge graph based multi-event forecasting. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1585–1595.

Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. 2018. Convolutional 2d knowledge graph embeddings. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Alberto Garcia-Duran, Sebastijan Dumančić, and Mathias Niepert. 2018. Learning sequence encoders for temporal knowledge graph completion. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4816–4821.

Rishab Goel, Seyed Mehran Kazemi, Marcus Brubaker, and Pascal Poupart. 2020. Diachronic embedding for temporal knowledge graph completion. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 3988–3995.

Zhen Han, Peng Chen, Yunpu Ma, and Volker Tresp. 2020a. Dyernie: Dynamic evolution of riemannian manifold embeddings for temporal knowledge graph completion. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7301–7316.

Zhen Han, Peng Chen, Yunpu Ma, and Volker Tresp. 2020b. Explainable subgraph reasoning for forecasting on temporal knowledge graphs. In *International Conference on Learning Representations*.

Zhen Han, Zifeng Ding, Yunpu Ma, Yujia Gu, and Volker Tresp. 2021a. 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*, pages 8352–8364.

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*, pages 8352–8364, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Zhen Han, Yunpu Ma, Yuyi Wang, Stephan Günnemann, and Volker Tresp. 2020c. Graph hawkes neural network for forecasting on temporal knowledge graphs. *arXiv preprint arXiv:2003.13432*.

He He, Anusha Balakrishnan, Mihail Eric, and Percy Liang. 2017. Learning symmetric collaborative dialogue agents with dynamic knowledge graph embeddings. *arXiv preprint arXiv:1704.07130*.

Tingsong Jiang, Tianyu Liu, Tao Ge, Lei Sha, Sujian Li, Baobao Chang, and Zhifang Sui. 2016. Encoding temporal information for time-aware link prediction. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2350–2354.

Woojeong Jin, Meng Qu, Xisen Jin, and Xiang Ren. 2019. Recurrent event network: Autoregressive structure inference over temporal knowledge graphs. *arXiv preprint arXiv:1904.05530*.

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)*, pages 6669–6683.

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

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

Yunshi Lan and Jing Jiang. 2020. Query graph generation for answering multi-hop complex questions from knowledge bases. Association for Computational Linguistics.

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

Kalev Leetaru and Philip A Schrodt. 2013. Gdelt: Global data on events, location, and tone, 1979–2012. In *ISA annual convention*, volume 2, pages 1–49. Citeseer.

Zixuan Li, Saiping Guan, Xiaolong Jin, Weihua Peng, Yajuan Lyu, Yong Zhu, Long Bai, Wei Li, Jiafeng Guo, and Xueqi Cheng. 2022. Complex evolutional pattern learning for temporal knowledge graph reasoning. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 290–296, Dublin, Ireland. Association for Computational Linguistics.

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. *arXiv preprint arXiv:2106.00327*.

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 *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 408–417.

Siyuan Liao, Shangsong Liang, Zaiqiao Meng, and Qiang Zhang. 2021. Learning dynamic embeddings for temporal knowledge graphs. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, pages 535–543.

Deepak Nathani, Jatin Chauhan, Charu Sharma, and Manohar Kaul. 2019. Learning attention-based embeddings for relation prediction in knowledge graphs. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4710–4723.

Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *European semantic web conference*, pages 593–607. Springer.

Chao Shang, Yun Tang, Jing Huang, Jinbo Bi, Xiaodong He, and Bowen Zhou. 2019. End-to-end structure-aware convolutional networks for knowledge base completion. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 3060–3067.

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.

Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. 2018. Rotate: Knowledge graph embedding by relational rotation in complex space. In *International Conference on Learning Representations*.

Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex embeddings for simple link prediction. In *International conference on machine learning*, pages 2071–2080. PMLR.

Shikhar Vashishth, Soumya Sanyal, Vikram Nitin, and Partha Talukdar. 2019. Composition-based multi-relational graph convolutional networks. In *International Conference on Learning Representations*.

Xiang Wang, Xiangnan He, Yixin Cao, Meng Liu, and Tat-Seng Chua. 2019. Kgat: Knowledge graph attention network for recommendation. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 950–958.

Jiapeng Wu, Meng Cao, Jackie Chi Kit Cheung, and William L Hamilton. 2020. Temp: Temporal message passing for temporal knowledge graph completion. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5730–5746.

Yonghui Xu, Shengjie Sun, Huiguo Zhang, Changan Yi, Yuan Miao, Dong Yang, Xiaonan Meng, Yi Hu, Ke Wang, Huaqing Min, et al. 2021. Time-aware graph embedding: A temporal smoothness and task-oriented approach. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 16(3):1–23.

Bishan Yang, Scott Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2015. Embedding entities and relations for learning and inference in knowledge bases. In *Proceedings of the International Conference on Learning Representations (ICLR) 2015*.

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 *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 4732–4740.