Learning from History: Modeling Temporal Knowledge Graphs with Sequential Copy-Generation Networks

Cunchao Zhu\(^1\), Muhao Chen\(^2\), Changjun Fan\(^1\), Guangquan Cheng\(^1\), Yan Zhang\(^3\)

\(^1\) College of Systems Engineering, National University of Defense Technology, China
\(^2\) Viterbi School of Engineering, University of Southern California, USA
\(^3\) École Pour l’Informatique et les Techniques Avancées, France
fanchangjun@nudt.edu.cn

Abstract

Large knowledge graphs often grow to store temporal facts that model the dynamic relations or interactions of entities along the timeline. Since such temporal knowledge graphs often suffer from incompleteness, it is important to develop time-aware representation learning models that help to infer the missing temporal facts. While the temporal facts are typically evolving, it is observed that many facts often show a repeated pattern along the timeline, such as economic crises and diplomatic activities. This observation indicates that a model could potentially learn much from the known facts appeared in history. To this end, we propose a new representation learning model for temporal knowledge graphs, namely CyGNet, based on a novel time-aware copy-generation mechanism. CyGNet is not only able to predict future facts from the whole entity vocabulary, but also capable of identifying facts with repetition and accordingly predicting such future facts with reference to the known facts in the past. We evaluate the proposed method on the knowledge graph completion task using five benchmark datasets. Extensive experiments demonstrate the effectiveness of CyGNet for predicting future facts with repetition as well as de novo fact prediction.

1 Introduction

Knowledge Graphs (KGs) are widely used resources for knowledge representations of real-world facts (or events), inasmuch as it supports countless knowledge-driven tasks in areas such as information retrieval (Liu et al. 2018), natural language understanding (Chen, Chen, and Yu 2019; He et al. 2017), recommender systems (Koren, Bell, and Volinsky 2009) and healthcare (Hao et al. 2020). Traditionally, a KG only possesses facts in a static snapshot, while currently the rapid growing data often exhibit complex temporal dynamics. This calls for new approaches to model such dynamics of facts by assigning the interactions of entities with temporal properties (i.e., known as temporal knowledge graphs, or TKGs.). Representative TKGs include Global Database of Events, Language, and Tone (GDELT) (Leetaru and Schrodt 2013) and Integrated Crisis Early Warning System (ICEWS) (Boschee et al. 2015), which are two well-known event-based data repository that store evolving knowledge about entity interactions across the globe. Figure 1 shows a snippet of ICEWS, which shows several records of diplomatic activities.

Figure 1: A snippet of ICEWS showing several records of diplomatic activities.

Recently, many research efforts have put into representation learning for TKGs. Relevant methods typically encode the temporally evolving facts of entity relations with time-specific embeddings. This provides a versatile and efficient tool to complete future facts of TKGs based on the embedding representations of the past facts. Moreover, it benefits a wide range of downstream applications, e.g. transaction recommendation (Wang, Hu, and Cao 2017), event process induction (Du et al. 2016; Zhang et al. 2020) and social relation prediction (Zhou et al. 2018a). Several temporal knowledge graph embedding (TKGE) methods only focus on calculating latent representations for each snapshot separately, and thus are not capable of capturing the long-term dependency of facts in consecutive time snapshots (Jiang et al. 2016; Dasgupta, Ray, and Talukdar 2018; Wang and Li 2019; Ma, Tresp, and Daxberger 2019; Lacroix, Obozinski, and Usunier 2019). Some other recent attempts encode
the temporally evolving facts of entity relations by incorporating past information from previous snapshots (Goel et al. 2020; Trivedi et al. 2017; Jin et al. 2019). However, the complex evolution of temporal facts, including recurrence and trending patterns, can cause the aforementioned methods to fall short.

In fact, many facts occur repeatedly along the history. For instance, global economic crises happen periodically in about every seven to ten years (Korotayev and Tsirel 2010); the diplomatic activities take place regularly between two countries with established relationships (Fellham 2004); East African animals undergo annual vast migration in every June (Musiega and Kazadi 2004). More specifically, we found that over 80% of all the events throughout the 24 years of ICEWS data (i.e. 1995 to 2019) have already appeared during the previous time period. This phenomenon highlights the importance of leveraging the known facts in predicting the future ones. However, most existing methods do not incorporate the awareness of such evolution patterns in modeling a TKG. On the contrary, to conform with the nature of evolving facts in TKG events, we believe more precise temporal fact inference should make full use of known facts in history. Accordingly, a TKGE model would benefit from learning the temporal recurrence patterns while characterizing the TKGs.

To this end, we propose a new representation learning method for TKGs based on a novel time-aware copy mechanism. The proposed CyGNet (Temporal Copy-Generation Network) is not only able to predict future facts from the whole entity vocabulary, but also capable of identifying facts with recurrence, and accordingly selecting such facts based on the historical vocabulary of entities that form facts only appeared in the past. This behaves similarly to the copy mechanism in the abstractive summarization task (Gu et al. 2016) in natural language generation (NLG), which allows a language generator to choose to copy subsequences from the source text, so as to help generate summaries that preserve salient information in the source text. Inspired by this mechanism, when predicting a future fact of quadruple \((s_i, p_j, ?, t_T)\), we can treat the known facts \(\{(s_i, p_j, o_k, t_0), (s_i, p_j, o_k, t_0), \ldots, (s_i, p_j, o_k, T_{t-1})\}\) appeared in the previous snapshots as the source text in abstractive summarization, and predict the future facts based solely on the known facts from the historical vocabulary \(\{o_k, ..., o_k\}\). As shown in Figure 2, CyGNet consists of two modes of inference, namely Copy mode and Generation mode. We illustrate CyGNet’s two inference modes with an example prediction of the 2018 NBA championship. Accordingly, all 18 NBA champion teams before 2018 are collected as the historical entity vocabulary, and the total 30 NBA teams are considered as the whole entity vocabulary.

To complete the query \((\text{NBA.champion},?,2018)\), CyGNet utilizes the Copy mode to predict the entity probabilities from the known entity vocabulary, and adopts the Generation mode to infer the entity probabilities from the whole entity vocabulary. Then, both probability distributions are combined as the final prediction.

We conduct extensive experiments on five benchmark TKG datasets, which demonstrate that CyGNet can effectively model TKG data via the combination of the Copy mode and the Generation mode in both learning and inference. Accordingly, CyGNet achieves more precise future fact predictions in most datasets than state-of-the-art (SOTA) methods that do not take special consideration of temporal facts with recurrence patterns.

This work presents the following contributions:

1. We investigate the underlying phenomena of temporal facts with recurrence, and propose to make reference to known facts in history when learning to infer future facts in TKGs;
2. We propose a novel TKGE model CyGNet via the time-aware copy-generation mechanism, which combines two modes of inference to make predictions based on either the historical vocabulary or the whole entity vocabulary, hence being more consistent to the aforementioned evolution pattern of TKG facts;
3. We conduct extensive experiments on five public TKG benchmarks datasets, and demonstrate its effectiveness in future fact (link) prediction.

## 2 Related Work

We discuss three lines of relevant research. Each has a large body of work, of which we can only provide a highly selected summary.

### Static KG Embeddings

A large number of methods are developed to model static KGs without temporally dynamic facts, which have been summarize in recent surveys (Wang et al. 2017; Ji et al. 2020; Dai et al. 2020). A class of these methods are the translational models (Bordes et al. 2013; Wang et al. 2014; Ji et al. 2015), which models a relation between two entity vectors as a translation vector. Another class of models are the semantic matching models that measures plausibility of facts using a triangular norm (Yang et al. 2015; Trouillon et al. 2016; Sun et al. 2018). Some other models are based on deep neural network approaches using feed-forward or convolutional layers on top of the embeddings (Schlichtkrull et al. 2018; Dettmers et al. 2018; Schlichtkrull et al. 2018). However, these methods do not capture temporally dynamic facts.

### Temporal KG Embeddings

More recent attempts have been made to model the evolving facts in TKGs. TTransE (Jiang et al. 2016) is an extension of TransE (Bordes et al. 2013) by embedding temporal information into the score function. HyTE (Dasgupta, Ray, and Talukdar 2018) replaces the unit normal vector of the hyperplane projection in TransH (Wang et al. 2014) with a time-specific normal vector. Know-Evolve (Trivedi et al. 2017) learns non-linearly evolving entity representations over time which models the occurrence of a fact as a temporal point process. DE-SimplE (Goel et al. 2020) leverages diachronic embeddings to represent entities at different timestamps and employs the same score function as SimpleE (Kazemi and Poole 2018) to score the plausibility of a quadruple. Based on the Tucker decomposition (Balazevic, Allen, and Hospedales 2019), ConT (Ma, Tresp, and Daxberger 2019) learns a new core tensor for each timestamp. However, these models do not provide a
mechanism to capture the long-term dependency of facts in consecutive time snapshots.

Some other methods are designed to model graph sequences, which can be applied to capture long-term dependency of TKG facts. TA-DistMult (García-Durán, Dumancic, and Niepert 2018) utilizes a recurrent neural network to learn time-aware representations of relations and uses standard scoring functions from DistMult (Yang et al. 2015). GCRN (Seo et al. 2018) combines GCN (Kipf and Welling 2017) for graph-structured data and RNN to identify simultaneously meaningful spatial structures and dynamic patterns. DyREP (Trivedi et al. 2018) divides the dynamic graph network into two separate processes of global and local topological evolution, and proposes a two-time scale deep temporal point process to jointly model the two processes. Know-Evolve, DyREP and GCRN have also been combined with MLP decoders to predict future facts, as presented by Jin et al. (2019). The previous SOTA method in this line of research, i.e. RE-NET (Jin et al. 2019) models event (fact) sequences jointly with an RNN-based event encoder and an RGCN-based (Schlichtkrull et al. 2018) snapshot graph encoder.

Copy Mechanism The copy mechanism has been previously applied to NLG tasks, particularly for abstractive summarization. Vinyals, Fortunato, and Jaity (2015) proposed the Pointer Networks in which a pointer mechanism is used to select the output sequence directly from the input sequence. However, the Pointer Network cannot make prediction using lexemes that are external to the input sequence. COPYNET (Gu et al. 2016) solves this problem in a hybrid end-to-end model by combining the pointer network (or termed as copy mechanism in its context) with a generation mechanism that yields lexemes that do not appear in the input. SeqCopyNet (Zhou et al. 2018b) improves the copy mechanism to not only copy single lexemes, but also copies subsequences from the input text. Our work is inspired to use the copy mechanism for the characteristics of TKGs, which to the best of our knowledge, is the first work that incorporates the copy mechanism into modeling TKGs. This is consistent to the nature that temporal knowledge may contain recurrence patterns along the trending timeline.

3 Method

In this section, we introduce the proposed model, named CyGNet, for fact/link prediction in TKGs. We start with the notations, and then introduce the model architecture as well as its training and inference procedures in detail.

3.1 Notations

TKGs incorporate temporal information in traditional KGs. In a TKG, each of the fact captures the relation (or predicate) $p \in R$ for subject and object entity $s \in E$ and $o \in E$ at time step $t \in T$, where $E$, $R$ denote the corresponding vocabularies of entities and relations respectively, and $T$ is the set of timestamps. Boldfaced $s$, $p$, $o$, $t$ represent the embedding vectors of subject entity $h$, predicate $p$, object...
entity \( o \) and time step \( t \) in a temporal fact. \( \mathcal{G}_t \) is a snapshot of the TKG at a time \( t \), and \( q = (s, p, o, t) \) denotes a quadruple (fact) in \( \mathcal{G}_t \). A TKG is built upon a set of fact quadruples ordered ascendingly based on their timestamps, i.e., \( \mathcal{G} = \{ \mathcal{G}_1, \mathcal{G}_2, \ldots, \mathcal{G}_T \} \), where the same quadruple is removed for redundancy. For each subject entity and predicate pair at time step \( t_k \), we define a delimited subset of \( E \) that is specific to \( (s, p, t_k) \) (namely a historical vocabulary for \( (s, p, t_k) \)) as \( \mathbf{H}^{(s,p)}_{t_k} \), which contains all the entities that have served as object entities in facts with the subject entity \( s \) and the predicate \( p \) along the known snapshots \( \mathcal{G}_{(t_1, t_{k-1})} = \{ \mathcal{G}_1, \mathcal{G}_2, \ldots, \mathcal{G}_{k-1} \} \) before \( t_k \), where the historical vocabulary \( \mathbf{H}^{(s,p)}_{t_k} \) is an \( N \)-dimensional multi-hot indicator vector and \( N \) is the cardinality of \( \mathcal{E} \), the value of entities in the historical vocabulary are marked 1 while others are 0. Prediction of a missing temporal fact aims to infer the missing object entity given \( (s, p, ?, t) \) (or the subject entity given \( (?, p, o, t) \), or the predicate with subject entity and object entity given \( (s, ?, o, t) \)). Without loss of generality, we describe our model as predicting the missing object entity in a temporal fact, although the model can be easily extended to predicting other elements including the subject entity and the predicate.

### 3.2 Model Components

As shown in Figure 3, our model combines two modes of inference, namely **Copy mode** and **Generation mode**, where the former seeks to select entities from a specific historical vocabulary that forms repeated facts in history while the latter predicts entities from the whole entity vocabulary. When predicting a quadruple \((s_1, p_3, ?, T + 1)\) (as shown in Figure 3), the Copy mode should infer the probability of entities in the historical vocabulary \(\{s_3, s_4, s_5, \ldots, s_m\}\) which have served as the object entities in facts of the subject entity \(s_1\) and the predicate \(p_3\) along the known snapshots \(\mathcal{G}_{(t_1, t_T)}\). On the other hand, the Generation mode estimates the probability of every entity in the whole entity vocabulary to answer a query. Then CyGNet combines the probabilistic predictions from both Copy mode and Generation mode to output the final prediction.

We first process the training set to obtain the historical vocabulary for each subject entity and predicate combination \((s, p, t)\) on every \( t \) in training snapshots, i.e. \(\{\mathbf{h}^{(s,p)}_{t_1}, \mathbf{h}^{(s,p)}_{t_2}, \ldots, \mathbf{h}^{(s,p)}_{t_T}\}\), where \(\mathbf{h}^{(s,p)}_{t_k}\) is an \(N\)-dimensional multi-hot indicator vector that includes all object entities served as object entities in facts with the subject \(s\) and the predicate \(p\) in snapshot \(\mathcal{G}_{t_k}\). As shown in Figure 4, we train the model sequentially on each snapshot, similar to the idea of recursion, trained by incrementally maintain the historical vocabulary for all the previous snapshots. When we evaluate CyGNet’s performances in the validation set and test set, the maximum historical vocabulary from the whole training set will be used.

Specifically, for each query quadruple \((s, p, ?, t_k)\) on time \(t_k\), the training process extends the historical vocabulary that specific to \((s, p, t_k)\) from that of the previous snapshot, as formalized below:

\[
\mathbf{H}^{(s,p)}_{t_k} = \mathbf{h}^{(s,p)}_{t_1} + \mathbf{h}^{(s,p)}_{t_2} + \ldots + \mathbf{h}^{(s,p)}_{t_{k-1}},
\]

where \(\mathbf{H}^{(s,p)}_{t_k}\) is an \(N\)-dimensional multi-hot indicator vector where 1 is marked for all entities in the current historical vocabulary. We now introduce the two inference modes in the following.

**Copy Mode** The Copy mode is designed to identify facts with recurrence, and accordingly predict the future facts by copying from known facts in history.

If query \((s, p, ?, t_k)\) has the historical vocabulary \(\mathbf{H}^{(s,p)}_{t_k}\) specific to the subject entity \(s\) and the predicate \(p\) at time step \(t_k\), CyGNet will increase the probability estimated for the object entities that are selected in the historical vocabulary. In detail, the Copy mode first generates an index vector \(v_q\) with an MLP:

\[
t_k = t_{k-1} + t_u,
\]

\[
v_q = \tanh(W_c[s, p, t_k] + b_c),
\]

4735
where $\mathbf{W}_g \in \mathbb{R}^{3d \times N}$ and $\mathbf{b}_g \in \mathbb{R}^N$ are trainable parameters. $t_0$ is a unit step of time and $t_3 = t_0$. The index vector $\mathbf{v}_q$ is an $N$-dimensional vector, where $N$ is the cardinality of the whole entity vocabulary $\mathcal{E}$.

To minimize the probability of some entities that do not form known facts with $s$ and $p$ in history (i.e., those being uninterested to the Copy mode), we first make modifications to $\mathbf{H}_{t_k}^{(s,p)}$. $\mathbf{H}_{t_k}^{(s,p)}$ changes the index value for an uninterested entity in $\mathbf{H}_{t_k}^{(s,p)}$ to a small negative number. Therefore, CyGNet can delimit the candidate space by adding the index vector $\mathbf{v}_q$ and the changed multi-hot indicator vector $\mathbf{H}_{t_k}^{(s,p)}$, minimize the probability of the uninterested entities, and then estimate the probability of the object entities in the historical vocabulary with a softmax function:

$$
c_q = \mathbf{v}_q + \mathbf{H}_{t_k}^{(s,p)},
$$

$$
p(c) = \text{softmax}(c_q),
$$

where $c_q$ is an $N$-dimensional index vector, such that the values corresponding to uninterested entities in $c_q$ are assigned close to zero. $p(c)$ is a vector whose size equals to that of the whole entity vocabulary, and is to represent the prediction probabilities on the historical vocabulary. Eventually, the largest dimension of $p(c)$ indicates the object entity to be copied from the historical vocabulary. The merit of the Copy mode is that it is able to learn to predict from a much delimited candidate space than the entire entity vocabulary. However, facts can also be newly appearing in an upcoming snapshot. And we therefore need a Generation mode to predict such facts.

**Generation Mode** Given the same aforementioned query $(s, p, ?, t_k)$, the Generation mode is responsible for predicting facts by selecting the object entity from the whole entity vocabulary $\mathcal{E}$. The prediction made by the generation mode regards the predicted fact as an entirely new fact without any references to history. Similar to the Copy mode, the Generation mode also generates an index vector $\mathbf{g}_q$ whose size equals to the size of the candidate space $\mathcal{E}$, and is normalized with a softmax function to make predictions:

$$
g_q = \mathbf{W}_g|s, p, t_k| + \mathbf{b}_g,
$$

$$
p(g) = \text{softmax}(g_q),
$$

where $\mathbf{W}_g \in \mathbb{R}^{3d \times N}$ and $\mathbf{b}_g \in \mathbb{R}^N$ are trainable parameters. Similar to $p(c)$ in the Copy mode, $p(g)$ represents the predicted probability on the whole entity vocabulary. The maximum value in $p(g)$ indicates the object entity we predict in the whole entity vocabulary by the Generation mode. The Generation mode complements the Copy mode with the ability for de novo fact prediction.

### 3.3 Learning Objective

Predicting the (object) entity when given a query $(s, p, ?, t)$ can be viewed as a multi-class classification task, where each class corresponds to one object entity. The learning objective is to minimize the following cross-entropy loss $\mathcal{L}$ on all facts of the TKG snapshots that exist during training:

$$
\mathcal{L} = -\sum_{t \in T} \sum_{i \in \mathcal{E}} \sum_{k=1}^{K} o_{it} \ln p(y_{ik}|s, p, t)
$$

where $o_{it}$ is the $i$-th ground truth object entity in the snapshot $G_t$, $p(y_{ik}|s, p, t)$ is the combined probability value of the $k$-th object entity in the snapshot $G_t$ when the $i$-th ground truth object entity is $o_i$.

### 3.4 Inference

Without loss of generality, we describe the inference process as predicting the missing object in a temporal fact, although this process can be easily extended to predicting the subject and the relation as well. To make prediction regarding a query $(s, p, ?, t_k)$, both Copy and Generation modes give a predicted object entity with the highest probability within their candidate spaces, whereof the Copy mode may make predictions from a much smaller candidate space than the entire entity vocabulary. To ensure that the sum of the probability equals 1 for all entities in $\mathcal{E}$, a coefficient $\alpha$ is incorporated to adjust the weight between the Copy mode and the Generation mode. CyGNet combines the probabilistic predictions from both the Copy mode and the Generation mode by adding the probability of each entity given by these two modes. The final prediction $o_i$ will be the entity that receives the highest combined probability, as defined below:

$$
p(o|s, p, t) = \alpha \ast p(c) + (1 - \alpha) \ast p(g),
$$

$$
o_i = \text{argmax}_{o \in \mathcal{E}} p(o|s, p, t),
$$

where $\alpha \in [0, 1]$, $p(o|s, p, t)$ is the whole entity vocabulary size vector which contains the probability of all entities.

### 4 Experiments

In this section, we demonstrate the effectiveness of CyGNet with five public TKG datasets. We first explain experimental settings in detail, including details about baselines and datasets. After that, we discuss the experimental results. We also conduct an ablation study to evaluate the importance of different components of CyGNet$^1$.

#### 4.1 Experimental Setup

**Datasets** We evaluate CyGNet on the task of link prediction using five benchmark datasets, namely ICEWS18, ICEWS14, GDELT, WIKI and YAGO specifically. ICEWS records political facts (events) with timestamps, e.g., (Donald Trump, visit, France, 2018-04-10), and there are two benchmark datasets extracted from two time periods in this knowledge base i.e., ICEWS18 ([Boschee et al. 2015]; from 1/1/2018 to 10/31/2018) and ICEWS14 ([Trivedi et al. 2017]; from 1/1/2014 to 12/31/2014). GDELT ([Leetaru and Schrodt 2013]) is a catalog of human societal-scale behavior and beliefs extracted from news media, and the experimental dataset is collected from 1/1/2018 to 1/31/2018 with a time granularity of 15 mins. The WIKI and YAGO datasets

$^1$The released source code and documentation are available at https://github.com/CunchaoZ/CyGNet.
and Talukdar 2018), TA-DistMult (García-Durán, Duman et al. 2018), TTransE (Jiang et al. 2016), HyTE (Dasgupta, Ray, and Talukdar 2018) and RotatE (Sun et al. 2018). Temporal methods include TransE (Bordes et al. 2013), DistMult (Yang et al. 2015), ComplEX (Trouillon et al. 2016), R-GCN (Schlichtkrull et al. 2018), ConvE (Dettmers et al. 2018) and baseline methods generally perform better than TTransE and HyTE. We believe this is due to that TTransE and HyTE only learn representations independently for each snapshot, while lack capturing the long-term dependency. Table 2 also shows that CyGNet significantly outperforms other baselines on ICEWS18, ICEWS14 and GDELT. For instance, CyGNet

### Evaluation Protocols

Following prior work (Jin et al., 2019), we split each dataset except for ICEWS14 into training, validation and testing set into 80%/10%/10% splits in the chronological order, whereas ICEWS14 is not provided with a validation set. We report Mean Reciprocal Ranks (MRR) and Hits@1/3/10 (the proportion of correct test cases that are ranked within top 1/3/10). We also enforce the filtered evaluation constraint that has been widely adopted in prior work (Jin et al. 2019; Dasgupta, Ray, and Talukdar 2018).

### Model Configurations

The values of hyper-parameters are determined according to the MRR performance on each validation set. Specifically, since ICEWS14 does not come with a validation set, we directly carry forward the same hyper-parameter settings from ICEWS18 to ICEWS14. The coefficient α is tuned from 0.1 to 0.9 with a step of 0.1. It is set to 0.8 for ICEWS18 and ICEWS14, 0.7 for GDELT, and 0.5 for YAGO and WIKI. The model parameters are first initialized with Xavier initialization (Glorot and Bengio 2010), and are optimized using an AMSGrad optimizer with a learning rate of 0.001. The batch size is set to 1024. The training epoch is limited to 30, which is enough for convergence in most cases. The embedding dimension is set to 200 to be consistent with the setting by Jin et al. (2019). The baseline results are also adopted from Jin et al. (2019).

### Results

Tables 2 and 3 report the link prediction results by CyGNet and baseline methods on five TKG datasets. On the first three datasets, static KGE methods fell far behind RE-NET or CyGNet due to their inability to capture temporal dynamics. It can also be observed that all static KGE methods generally perform better than TTransE and HyTE. We believe this is due to that TTransE and HyTE only learn representations independently for each snapshot, while lack capturing the long-term dependency. Table 2 also shows that CyGNet significantly outperforms other baselines on ICEWS18, ICEWS14 and GDELT. For instance, CyGNet

### Table 2: Statistics of the datasets.

<table>
<thead>
<tr>
<th>Method</th>
<th>ICEWS18</th>
<th>ICEWS14</th>
<th>GDELT</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>#Data</td>
<td>#Entities</td>
<td>#Relation</td>
</tr>
<tr>
<td>TransE</td>
<td>23,033</td>
<td>256</td>
<td>373,018</td>
</tr>
<tr>
<td>DistMult</td>
<td>12,498</td>
<td>260</td>
<td>323,895</td>
</tr>
<tr>
<td>ComplEx</td>
<td>7,691</td>
<td>240</td>
<td>1,734,399</td>
</tr>
<tr>
<td>R-GCN</td>
<td>12,554</td>
<td>24</td>
<td>539,286</td>
</tr>
<tr>
<td>ConvE</td>
<td>10,623</td>
<td>10</td>
<td>161,540</td>
</tr>
</tbody>
</table>

### Table 2: Results (in percentage) on three datasets. The best results are boldfaced, and the second best ones are underlined.
achieves the improvements of 10.80% (MRR), 12.10% (Hits@1), 11.29% (Hits@3), and 7.19% (Hits@10) over GDELT. We later found that GDELT has 62.3% of the groups where the objects have repeated for at least once in history, whereas the recurrence rate of subject entities is around 23.4%. The imbalanced recurrence rate has hindered the combined learning of CyGNet’s two components, hence caused a deteriorated effect to its overall prediction performance. While CyGNet performs much better on the other datasets with more balanced recurrence rates of subject and object entities, how to tackle this shortcoming of CyGNet with a more robust meta-learning framework is a meaningful direction for further studies.

We also observe in Table 3 that CyGNet does not always perform the best on WIKI and YAGO, especially on WIKI. To give further analyses, we separate the results for predicting subject entities and those for object entities (Table 4). We can observe that CyGNet performs much better in predicting object entities on WIKI. This is in fact due to the unbalanced proportions of subjects and objects in participating recurring facts on WIKI. More specifically, if we group the facts by subjects and relations, we observe as many as 62.3% of the groups where the objects have repeated for at least once in history, whereas the recurrence rate of subject entities (in facts grouped by relations and objects) is around 23.4%. The imbalanced recurrence rate has hindered the combined learning of CyGNet’s two components, hence caused a deteriorated effect to its overall prediction performance. While CyGNet performs much better on the other datasets with more balanced recurrence rates of subject and object entities, how to tackle this shortcoming of CyGNet with a more robust meta-learning framework is a meaningful direction for further studies.

To give further analyses, we separate the results for predicting subject entities and those for object entities (Table 4). We can observe that CyGNet performs much better in predicting object entities on WIKI. This is in fact due to the unbalanced proportions of subjects and objects in participating recurring facts on WIKI. More specifically, if we group the facts by subjects and relations, we observe as many as 62.3% of the groups where the objects have repeated for at least once in history, whereas the recurrence rate of subject entities (in facts grouped by relations and objects) is around 23.4%. The imbalanced recurrence rate has hindered the combined learning of CyGNet’s two components, hence caused a deteriorated effect to its overall prediction performance. While CyGNet performs much better on the other datasets with more balanced recurrence rates of subject and object entities, how to tackle this shortcoming of CyGNet with a more robust meta-learning framework is a meaningful direction for further studies.

We believe this is because the original Generation mode in CyGNet is noticeably better than the Copy mode and the Generation mode are both important. Removing the Copy mode can lead to a drop of 12.11% in MRR, as well as drastic drops of other metrics, which shows that learning to predict future facts by referring to the known facts in the past can be helpful. On the other hand, the removal of the Generation mode leads to a drop of 4.09% in MRR, which counts for the loss of the model’s ability for de novo fact prediction. These results further explain that the promising performance by CyGNet is due to both the capability of learning from history, and identifying and predicting new facts from scratch.

CyGNet-Generation-new is another variant of CyGNet. The difference between CyGNet-Generation-new and the complete CyGNet is that the former utilizes the Generation mode to predict new facts in the whole entity vocabulary modulo the historical vocabulary. The performance of CyGNet is noticeably better than CyGNet-Generation-new. We believe this is because the original Generation mode in CyGNet can also strengthen the prediction in cases where a future fact is repeated. This merit is however discarded in the modified Generation mode of CyGNet-Generation-new.

<table>
<thead>
<tr>
<th>Method</th>
<th>WIKI</th>
<th>YAGO</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MRR</td>
<td>Hits@3</td>
</tr>
<tr>
<td>TransE</td>
<td>46.68</td>
<td>49.71</td>
</tr>
<tr>
<td>DistMult</td>
<td>46.12</td>
<td>49.81</td>
</tr>
<tr>
<td>ComplEx</td>
<td>47.84</td>
<td>50.08</td>
</tr>
<tr>
<td>R-GCN</td>
<td>37.57</td>
<td>39.66</td>
</tr>
<tr>
<td>ConvE</td>
<td>47.57</td>
<td>50.10</td>
</tr>
<tr>
<td>RotatE</td>
<td>50.67</td>
<td>50.71</td>
</tr>
<tr>
<td>Hybrid</td>
<td>45.02</td>
<td>45.12</td>
</tr>
<tr>
<td>TTransE</td>
<td>31.74</td>
<td>36.25</td>
</tr>
<tr>
<td>TA-DistMult</td>
<td>48.09</td>
<td>49.51</td>
</tr>
<tr>
<td>DyRep+MLP</td>
<td>11.60</td>
<td>12.74</td>
</tr>
<tr>
<td>R-GCRN+MLP</td>
<td>47.71</td>
<td>48.14</td>
</tr>
<tr>
<td>RE-NET</td>
<td>51.97</td>
<td>52.07</td>
</tr>
<tr>
<td>CyGNet</td>
<td>45.50</td>
<td>50.79</td>
</tr>
</tbody>
</table>

Table 3: Results (in percentage) on WIKI and YAGO. The best results are boldfaced, and the second best are underlined. Hits@1 was not reported by Jin et al. (2019) on these datasets, for which CyGNet achieved 41.73% on WIKI and 60.04% on YAGO.

Table 4: Results (in percentage) for predicting subject or object entities separately on WIKI.

<table>
<thead>
<tr>
<th>Method</th>
<th>Predicting subjects</th>
<th>Predicting objects</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MRR</td>
<td>Hits@3</td>
</tr>
<tr>
<td>RE-NET</td>
<td>50.44</td>
<td>50.80</td>
</tr>
<tr>
<td>CyGNet</td>
<td>37.80</td>
<td>42.20</td>
</tr>
</tbody>
</table>

Table 5: Results (in percentage) by different variants of our model on ICEWS18.

### 4.3 Ablation Study

To help understand the contribution of different model components of CyGNet, we present an ablation study. To do so, we create variants of CyGNet by adjusting the use of its model components, and compare the performance on the ICEWS18 dataset. Besides, more analyses are given in Appendix A (Zhu et al. 2021) due to space limitation.

From the results in Tables 5, we observe that the Copy mode and the Generation mode are both important. Removing the Copy mode can lead to a drop of 12.11% in MRR, as well as drastic drops of other metrics, which shows that learning to predict future facts by referring to the known facts in the past can be helpful. On the other hand, the removal of the Generation mode leads to a drop of 4.09% in MRR, which counts for the loss of the model’s ability for de novo fact prediction. These results further explain that the promising performance by CyGNet is due to both the capability of learning from history, and identifying and predicting new facts from scratch.

CyGNet-Generation-new is another variant of CyGNet. The difference between CyGNet-Generation-new and the complete CyGNet is that the former utilizes the Generation mode to predict new facts in the whole entity vocabulary modulo the historical vocabulary. The performance of CyGNet is noticeably better than CyGNet-Generation-new. We believe this is because the original Generation mode in CyGNet can also strengthen the prediction in cases where a future fact is repeated. This merit is however discarded in the modified Generation mode of CyGNet-Generation-new.

### 5 Conclusion

Characterizing and inferring temporal knowledge is a challenging problem. In this paper, we leverage the copy mechanism to tackle this problem, based on the hypothesis that a future fact can be predicted from the facts in history. The proposed CyGNet is not only able to predict facts from the whole open world, but is also capable of identifying facts with recurrence and accordingly selecting such future facts based on the known facts appeared in the past. The presented results demonstrate CyGNet’s promising performance for predicting future facts in TKGs. For future work, we plan to improve the sequential copy mechanism by identifying globally salient entities and events (Fan et al. 2019) in TKGs. Grounding dated documents to TKGs with the obtained embedding representations is also a meaningful direction. Besides, we are interested in leveraging the proposed techniques to help understand dynamic event processes in natural language text (Chen et al. 2020; Zhang et al. 2020).
Acknowledgments

We appreciate the anonymous reviewers for their insightful comments. This work is partially supported by NSFC-71701205, NSFC-62073333. Muhao Chen’s work is supported by Air Force Research Laboratory under agreement number FA8750-20-2-10002, and by the DARPA MCS program under Contract No. N660011924033.

References


Garcia-Durán, A.; Dumancic, S.; and Niepert, M. 2018. Learning Sequence Encoders for Temporal Knowledge Graph Completion. In EMNLP.


Citeseer.


