Temporal Attribute Prediction via Joint Modeling of Multi-Relational Structure Evolution

Sankalp Garg[1*†] , **Navodita Sharma**[2*†] , **Woojeong Jin**[3] and **Xiang Ren**[3]

[1]Indian Institute of Technology Delhi
[2]Indian Institute of Technology Madras
[3]University of Southern California
{sankalp2621998, navoditasharma16}@gmail.com, {woojeong.jin, xiangren}@usc.edu

Abstract

Time series prediction is an important problem in machine learning. Previous methods for time series prediction did not involve additional information. With a lot of dynamic knowledge graphs available, we can use this additional information to predict the time series better. Recently, there has been a focus on the application of deep representation learning on dynamic graphs. These methods predict the structure of the graph by reasoning over the interactions in the graph at previous time steps. In this paper, we propose a new framework to incorporate the information from dynamic knowledge graphs for time series prediction. We show that if the information contained in the graph and the time series data are closely related, then this inter-dependence can be used to predict the time series with improved accuracy. Our framework, DARTNET, learns a static embedding for every node in the graph as well as a dynamic embedding which is dependent on the dynamic attribute value (time-series). Then it captures the information from the neighborhood by taking a relation specific mean and encodes the history information using RNN. We jointly train the model link prediction and attribute prediction. We evaluate our method on five specially curated datasets for this problem and show a consistent improvement in time series prediction results. We release the data and code of model DARTNET for future research[1].

1 Introduction

Many real-world scenarios exist where the time series can effectively be complemented with external knowledge. One such scenario represents information about the trade between countries in the form of a temporally evolving knowledge graph. The information about trade between the countries affects the corresponding currency exchange rate. Using information about trade, we want to better predict the currency exchange rate with high accuracy.

---

*Equal Contribution

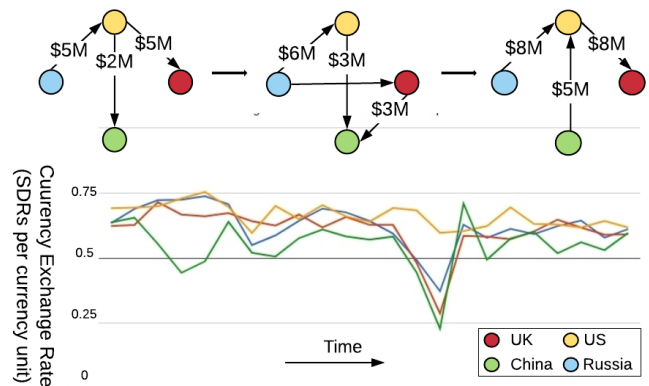†Work done while at USC

[1]https://github.com/INK-USC/DArtNet

Figure 1: Evolving graph structure and attribute values with time in the Dynamic Attributed Graph. The node represents a country and the edges between them represents the value of trade. The graph shows the changing attribute values (currency exchange rate) for each node.

Time Series forecasting deals with the prediction of the data points of the sequence at a future timestamp based on the data available up till the current timestamp. Methods such as Auto-Regressive Integrated Moving Average (ARIMA) model and Kalman filtering [Liu *et al.*, 2011; Lippi *et al.*, 2013] are popular for predicting time series. Representational learning on graph-structured data [Hamilton *et al.*, 2017; Wu *et al.*, 2019; Kipf and Welling, 2016; You *et al.*, 2018] is a widely researched field with considerable focus on temporally-evolving graphs [Trivedi *et al.*, 2018; Pareja *et al.*, 2019; Kumar *et al.*, 2018; Goyal *et al.*, 2018]. The increasing amount of complex data that can be effectively represented using dynamic multi-relational graphs [Neil *et al.*, 2018] has led to this increased focus on dynamic graph modeling. Several methods such as ConvE [Dettmers *et al.*, 2017], RGCN [Schlichtkrull *et al.*, 2018], and DistMult [Yang *et al.*, 2014] have shown admirable results on modeling static, multi-relational graph data for link prediction. There are other approaches that attempt to model dynamic knowledge graphs by incorporating temporal information, and these include Know-Evolve [Trivedi *et al.*, 2017], HyTE [Dasgupta *et al.*, 2018], and TA-DistMult [García-Durán *et al.*, 2018], among others.

The two aforementioned fields of time series prediction and representation learning on graphs have mainly been sep-

arated in the machine learning community. Recently some work has been done in integrating the two fields [Li *et al.*, 2017], which describes a method to incorporate a static, uni-relational graph for traffic flow prediction. However, this method is only limited to static graphs with a single relation. To date, no method has been proposed for integrating temporally evolving graphs and time series prediction. In this paper, we propose a new method for exploiting the information from the dynamic graphs for time series prediction. We propose the use of static learnable embedding to capture the spatial information from knowledge graphs and a dynamic embedding to capture the dynamics of the time series and the evolving graph.

We present the first-ever solution to the problem of time-series prediction with temporal knowledge graphs (TKG). Since, to the best of our knowledge, currently no datasets exist which align with the problem statement, we prepared five suitable datasets through web scraping and evaluate our model. We show that our approach beats the current state-of-the-art methods for time series forecasting on all the five datasets. Our approach also predicts the time series by any number of time steps and does not require a test time graph structure for evaluation.

2 Related Work

We review work using static graphs for time series prediction and work on temporal knowledge graphs.

Time-series Prediction. In addition to the general time-series prediction task, there have been some recent studies on the spatial-temporal forecasting problem. Diffusion Convolutional Recurrent Neural Network (DCRNN) [Li *et al.*, 2017] is a method which incorporates a static, uni-relational graph for time series (traffic flow) forecasting. Traffic flow is modeled as a diffusion process on a directed graph. The method makes use of bidirectional random walks on the graph to capture the spatial dependency and uses an encoder-decoder framework with scheduled sampling for incorporating the temporal dependence. However, this method cannot be extended to temporally evolving graphs as well as multi-relational graphs. Another paper on Relational Time Series Forecasting [Rossi, 2018], also formulates the problem of using dynamic graphs for time series prediction though it is not formulated for multi relational data. Neural relational inference [Kipf *et al.*, 2018] also looks at the inverse problem of predicting dynamics of graph with attribute information.

Temporal Knowledge Graph Reasoning and Link Prediction. There have been several attempts on reasoning on dynamically evolving graphs. HyTE [Dasgupta *et al.*, 2018] is a method for embedding knowledge graphs which views each timestamp in the graph data as a hyperplane. Each head, relation, tail triple at a particular timestamp is projected into the corresponding hyperplane. The translational distance, as defined by the TransE model [Bordes *et al.*, 2013], of the projected embedding vectors, is minimized. TA-Distmult is a temporal-aware version of Distmult. For a quadruple h, r, t, τ, a predicate p is constructed using r, τ which is passed into an GRU. The last hidden state of the GRU is taken
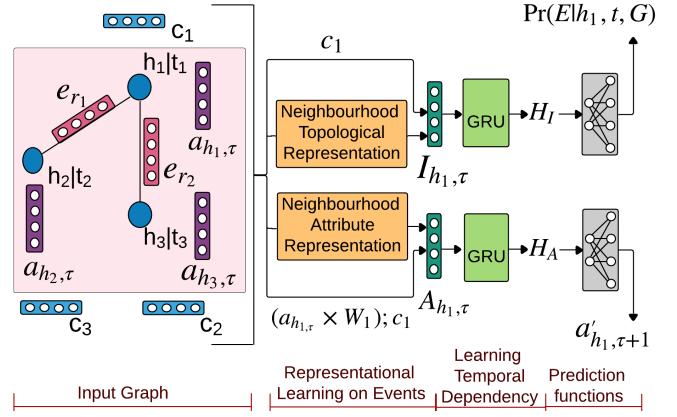

Figure 2: Network for Link Prediction and Attribute Prediction. h_i and t_i denote the ith head and tail respectively. e_{r_i} denotes the embedding for the relation r_i and $a_{h_i,\tau}$ denotes the attribute values associated with h_i at time τ. From neighbouring nodes, topological representation is obtained and using attribute values, attribute representation is obtained which are then fed to recurrent network. The recurrent network's output is used to obtain the predictions.

as the representation of the predicate sequence (e_p). Know-Evolve [Trivedi *et al.*, 2017] models a relationship between two nodes as a multivariate point process. Learned entity embeddings are used to calculate the score for that relation, which is used to modulate the intensity function of the point process. ReNet [Jin *et al.*, 2019] uses the neighborhood aggregation and RNN to capture the spatial and temporal information in the graph.

3 Problem Formulation

A Knowledge Graph is a multi-relational graph that can be represented in the form of triples (h, r, t) where h denotes the head, t denotes the tail, and r is the relation between the nodes h and t. A TKG has a time dimension as well, and the graph can be represented as quadruples in the form (h, r, t, τ) where τ denotes the timestamp at which the relation r exists between the nodes h and t. We now introduce Dynamic Attributed Graphs, formalize our problem statement, and in later sections, present our model for making predictions on Dynamic Attributed Graphs.

Problem Definition. A Dynamic Attributed Graph (DAG) is a directed graph, where the edges are multi-relational with time stamp associated with each edge known as an event, and attributes associated with the nodes for that particular time. An event in dynamic graph is represented as a quadruple (h, r, t, τ) where h is the head entity, r is the relation, t is the tail entity and τ is the time stamp of the event. In a dynamic attributed graph, an event (E_τ) is represented as a hextuple $(h, r, t, a_h^\tau, a_t^\tau, \tau)$ where a_h^τ and a_t^τ are the attribute associated with head and tail at time τ. The collection of all the events at a time constitutes a dynamic graph G_τ where $G_\tau = \{E_{\tau_i} = (h_i, r_i, t_i, a_{h_i}^\tau, a_{t_i}^\tau, \tau_i) | \tau_i = \tau, \forall i\}$. The goal of the DAG Prediction problem is to learn a representation of the dynamic graph events and predict the attributes at each node at future timestamps by learning a set of functions to predict the events for the next time step. Link is predicted

jointly, to aid the attribute prediction task. Formally, we want to learn a set of functions $\{F(.)\}$ such that:

$$[G_1, G_2, \ldots, G_\tau] \xrightarrow{\{F(.)\}} [G_{\tau+1}, G_{\tau+2}, \ldots, G_{\tau+T}].$$

We divide the dynamic graph at any time τ in two sets, one consisting of only (h, r, t, τ) and other consisting of only the attribute values. Formally, $G_\tau = G_\tau^I \cup G_\tau^A$, where $G_\tau^I = \{E_{\tau_i}^I = (h_i, r_i, t_i, \tau_i)|\tau_i = \tau, \forall i\}$ and $G_\tau^A = \{a_{h_i}^\tau|\tau_i = \tau, \forall i\}$. We propose to predict G^I and G^A using these set of functions as follows:

$$[G_1, G_2, \ldots, G_\tau] \xrightarrow{\{F(.)\}} [(G_{\tau+1}^I, G_{\tau+1}^A), \ldots, (G_{\tau+T}^I, G_{\tau+T}^A)].$$

We jointly predict the graph structure $G_{\tau'}^I$ and the attribute values $G_{\tau'}^A$ and show that the attribute values are being predicted with greater accuracy than can be done using any existing method.

4 Proposed Framework : DARTNET

We now present our framework for learning the set of functions for predicting the events for the next timestamp, given the history. We name our framework **DARTNET** which stands for **D**ynamic **A**ttributed **Net**work.

We model the changing graph structure and attribute values by learning an entity-specific representation. We define the representation of an entity in the graph as a combination of a static learnable embedding which does not change with time and represent static characteristics of each node, and a dynamic embedding which depends on the attribute value at that time and represent dynamically evolving property of each node. We then aggregate the information using the mean of neighborhood entities. For every entity in the graph, we model history encoding using a Recurrent Neural Network. Finally, we use a fully connected network for attribute prediction and link prediction tasks.

4.1 Representation Learning on Events

The main component of our framework is to learn a representation over the events which will be used for predicting future events. We learn a head-specific representation and then model its history using Recurrent Neural Network (RNN). Let $E_{h,\tau}$ represent the events associated with head h at time τ, i.e., $E_{h,\tau} = \{(h, r_i, t_i, a_h^\tau, a_{t_i}^\tau, \tau_i)|\tau_i = \tau, \forall i\}$. For each entity in the graph, we decompose the information in two parts, static information and dynamic information. Static information does not change over time and represents the inherent information for the entity. Dynamic information changes over time. It represents the information that is affected by all the external variables for the entity. For every entity h in the graph at time τ, we construct an embedding for the entity which consists of two components:

1. Static (Constant) learnable embedding $(\mathbf{c}_h)$ which does not change over time.

2. Dynamic embedding $(\mathbf{d}_{h,\tau} = \mathbf{a}_{h,\tau} \cdot \mathbf{W}_1)$ which changes over time.

where $\mathbf{c}_h \in \mathbb{R}^d$, $a_{h,\tau} \in \mathbb{R}^k$ is the attribute of entity h at time τ and $\mathbf{W}_1 \in \mathbb{R}^{k \times d}$ is learnable parameter. The attribute value

can be a multi-dimensional vector with $k \geq 1$ representing multiple time-series associated with same head. Hence the embedding of entity h becomes $\mathbf{e}_{h,\tau} = (\mathbf{c}_h; \mathbf{d}_{h,\tau})$ where ; is the concatenation operator. For every relation (link) r, we construct a learnable static embedding $\mathbf{e}_r \in \mathbb{R}^d$.

To capture the neighbourhood information of entity h at time τ from the dynamic graph, we propose two spatial embeddings: attribute embedding $(\mathbf{A}_{h,\tau})$ and interaction embedding $(\mathbf{I}_{h,\tau})$. $\mathbf{A}_{h,\tau}$ captures the spatio-attribute information from the neighbourhood of the entity h and $\mathbf{I}_{h,\tau}$ captures the spatio-interaction information from the neighbourhood of the entity h. Mathematically, we can define the spatial embeddings as:

$$\mathbf{A}_{h,\tau} = (\mathbf{e}_{h,\tau}; \frac{1}{|E_{h,\tau}|}\Sigma_{(h,r_j,t_j,a_h^\tau,a_{t_j}^\tau,\tau) \in E_{h,\tau}}(\mathbf{e}_{t_j,\tau}; \mathbf{e}_{r_j}) \cdot \mathbf{W}_2),$$
(1)

$$\mathbf{I}_{h,\tau} = (\mathbf{c}_h; \frac{1}{|E_{h,\tau}|}\Sigma_{(h,r_j,t_j,a_h^\tau,a_{t_j}^\tau,\tau) \in E_{h,\tau}}(\mathbf{c}_{t_j}; \mathbf{e}_{r_j}) \cdot \mathbf{W}_3),$$
(2)

where $|E_{h,\tau}|$ is the cardinality of set $E_{h,\tau}$ and $\mathbf{W}_2 \in \mathbb{R}^{3d \times d}$ and $\mathbf{W}_3 \in \mathbb{R}^{2d \times d}$ are learnable parameters.

4.2 Learning Temporal Dependency

The embeddings $\mathbf{I}_{h,\tau}$, $\mathbf{A}_{h,\tau}$ capture the spatial information for entity h at time τ. For predicting the information at future time, we need to capture the temporal dependence of the information. To keep track of the interactions and the attribute evolution over time, we model the history using Gated Recurrent Unit [Cho *et al.*, 2014], an RNN. For the head s, we define the encoded attribute history at time τ as the sequence $[A_{h,1}, A_{h,2}, \ldots, A_{h,\tau-1}]$ and the encoded interaction history at time τ as the sequence $[I_{h,1}, I_{h,2}, \ldots, I_{h,\tau-1}]$. These sequence provide the full information about the evolution of the head h till time τ. To represent in sequences, we model the encoded attribute and encoded interaction history for head h as follows:

$$\mathbf{H}_A(h, \tau) = GRU^1(\mathbf{A}_{h,\tau}, \mathbf{H}_A(h, \tau - 1)), \quad (3)$$

$$\mathbf{H}_I(h, r, \tau) = GRU^2((\mathbf{I}_{h,\tau}; \mathbf{c}_h; \mathbf{e}_{r_j}), \mathbf{H}_I(h, r, \tau - 1)), \quad (4)$$

where the vector $\mathbf{H}_A(h, \tau)$ captures the spatio-temporal information for the attribute evolution i.e. captures how the attribute value of the entity evolves over time with respect to the evolving graph structure, while the vector $\mathbf{H}_I(h, r, \tau)$ captures the spatio-temporal information of how the relation r is associated with the entity h over time. We show the DARTNET, its input and output in Figure 2.

4.3 Prediction Functions

The main aim of the model is to be able to predict the future attribute values as well as the interaction events. To get the complete information of the event at next time step, we perform the prediction in two steps: (1) prediction of the attribute values for the whole graph and (2) prediction of the interaction events for the graph. We know $G_{\tau+1} = \{E_{\tau_i} = (h_i, r_i, t_i, a_{h_i}^\tau, a_{t_i}^\tau, \tau_i)|\tau_i = \tau + 1, \forall i\}$. To predict $G_{\tau+1}$, we

Dataset	# Train	# Valid	# Test	# Nodes	# Rel	# Granularity
AGT	463,188	57,898	57,900	58	178	Monthly
CAC(small)	2070	388	508	90	1	Yearly
CAC(large)	116,933	167,047	334,096	20,000	1	Yearly
MTG	270,362	39,654	74,730	44	90	Monthly
AGG	3,879,878	554,268	1,108,538	6,635	246	Monthly

Table 1: Statistics of the five datasets used in the experiments.

divide it into two sets $G^I_{\tau+1}$ and $G^A_{\tau+1}$. The attribute values of $G^A_{\tau+1}$ are predicted directly and modelled as follows:

$$a'_{h,\tau+1} = f_A(\mathbf{H}_A(h,\tau), \mathbf{c}_h). \tag{5}$$

The attribute value for the entity is a function of the spatio-attribute history for the entity and the static information about the entity. Attribute prediction requires graph structures, so we also predict graph structures. The probability of $G^I_{\tau+1}$ is modeled as:

$$Pr(G^I_{\tau+1}|G_1, \ldots, G_\tau) = \prod_{E^I_k \in G^I_{\tau+1}} Pr(E^I_k|G_1, \ldots, G_\tau).$$

At $\tau + 1$, we can write this probability as

$$Pr(E^I_k|G_1, \ldots, G_\tau) = Pr(h_k, r_k, t_k|G_1, \ldots, G_\tau)$$
$$= Pr(t_k|h_k, r_k, G_1, \ldots, G_\tau) \cdot Pr(h_k, r_k|G_1, \ldots, G_\tau).$$

In this work, we consider the case that probability of (h, r) is independent of the past graphs $\{G_1, \ldots, G_\tau\}$, and model it using uniform distribution, leading to

$$Pr(G^I_{\tau+1}|G_1, \ldots, G_\tau) \propto \prod_{E^I_k \in G^I_{\tau+1}} Pr(t_k|h_k, r_k, G_1, \ldots, G_\tau).$$

For predicting the interaction at future timestamp, we model the probability of the tail as follows:

$$Pr(t|h, r, G_1, \ldots, G_\tau) = f_I(\mathbf{H}_I(h, r, \tau), \mathbf{c}_h, \mathbf{e}_r). \tag{6}$$

The functions f_A and f_I can be any function. In our experiments we use the functions as a single-layered feed-forward network.

4.4 Parameter Learning

We use multi-task learning [Ruder, 2017; Caruana, 1997] loss for optimizing the parameters. We minimize the attribute prediction loss and graph prediction loss jointly. The total loss $L = L_I + \lambda L_A$, where L_I is interaction loss, L_A is attribute loss and λ is a hyperparameter deciding the weight of both the tasks. For the attribute loss, we use the mean-squared error $L_A = \frac{1}{N} \sum_{i=1}^{N} (\mathbf{a}'_{h_i,\tau_i} - \mathbf{a}_{h_i,\tau_i})^2$, where a'_{h_i,τ_i} is the predicted attribute and a_{h_i,τ_i} is the ground truth attribute. For the interaction loss, we use the standard multi-class cross entropy loss, $L_I = \sum_{i=1}^{N} \sum_{c=1}^{M} y_j \log(Pr(t = c|h_i, r_i))$, where the M is the number classes i.e. number of entities in our case.

4.5 Forecasting Over Time

At each inference time, DARTNET predicts future interactions and attributes based on the previous observations, i.e.,

$[(G^I_i, G^A_i)]^\tau_{i=0}$. To predict interactions and attributes at time $\tau + \Delta\tau$, DARTNET adopts multi-step inference and predicts in a sequential manner. At each time step, we compute the probability $Pr(t|h, r, G_1, \ldots, G_\tau)$ to predict $G^A_{\tau+1}$. We rank the tails predicted and choose the top-k tails as the predicted values. We use the predicted tails as $G^A_{\tau+1}$ for further inference. Also, we predict attributes, which yields $G^I_{\tau+1}$. Now we have graph structure and attributes at time $\tau + 1$. We repeat this process until we get $(G^I_{\tau+\Delta\tau-1}, G^A_{\tau+\Delta\tau-1})$. Then we can predict interactions and attributes at time $\tau+\Delta\tau$ based on $[(G^I_i, G^A_i)]^{\tau+\Delta\tau-1}_{i=0}$.

5 Experiments

In this section, we evaluate our proposed method DARTNET on a temporal attribute prediction task on five datasets. The attribute prediction task predicts future attributes for each node.

We evaluate our proposed method on two tasks: (1) predicting future attributes associated with each node on five datasets; (2) studying variations and parameter sensitivity of our proposed method. We will summarize the datasets, evaluation metrics, and baseline methods in the following sections.

5.1 Datasets

Due to the unavailability of datasets satisfying our problem statement, we curated appropriate datasets by scraping the web. We created and tested our approach on the datasets described below. Statistics of datasets are described in Table 1.

Attributed Trade graph (ATG). This dynamic graph represents the net export from one country (node) to another, where each edge belongs to an order of trade segment (in a million dollars). The month-averaged currency exchange rate of the corresponding country in SDRs per currency unit is the time series attribute value.

Co-authorship-Citation dataset (CAC). Each edge in the graph represents the collaboration between the authors (node) of the research paper. The number of citations per year for an author is the corresponding time series attribute for the node.

Multi-attributed Trade graph (MTG). This is a subset of ATG, with a multi attributed time series representing monthly Net Export Price Index and the value of International Reserves assets in millions of US dollars.

Attributed GDELT graph (AGG). Global Database of Events, Language, and Tone (GDELT) represents a different type of event in a month between entities like political leaders, organizations, and countries, etc. Here only country nodes are associated with a time-series attribute, which is taken as the Currency Exchange Rate.

	Method	ATG (10^{-3})	CAC(small) (10^{-2})	CAC(large) (10^{-4})	MTG (10^{-4})	AGG (10^{-4})
w/o graph	Historic Average	1.636	4.540	9.810	14.930	600.000
	VAR[2][Hamilton, 1994]	3.961	6.423	10.330	9.490	300.000
	ARIMA	1.463	4.245	9.102	2.860	51.240
	Seq2Seq model [Sutskever et al., 2014]	1.323	4.554	8.080	2.975	28.000
with graph	ConvE+GRU (1 layer) [3][Dettmers et al., 2017]	0.763	3.899	8.220	7.240	202.580
	ConvE+GRU (2 layers)	0.728	4.321	8.440	9.460	206.640
	HyTE+GRU (1 layer) [4][Dasgupta et al., 2018]	4.041	40.234	8.089	37.170	7.430
	HyTE+GRU (2 layers)	1.531	40.885	8.230	17.410	2.070
	TA-Distmult+GRU (1 layer) [García-Durán et al., 2018]	0.847	3.584	9.456	16.880	3.250
	TA-Distmult+GRU (2 layer)	0.796	3.432	9.034	9.770	7.030
	RENet (mean)+GRU (1 layer) [Jin et al., 2019]	0.793	4.073	9.022	5.020	203.320
	RENet (mean)+GRU (2 layers)	0.857	3.865	8.856	4.348	200.220
	RENet (RGCN)+GRU (1 layer)	0.620	3.718	8.998	5.170	203.120
	RENet (RGCN)+GRU (2 layers)	0.550	3.984	8.201	12.700	201.560
	DARTNET	**0.115**	**3.423**	**7.054**	**0.496**	**0.848**

Table 2: Performance comparison on Attribute Prediction on various datasets. DARTNET performs best on all the datasets. Relational baselines generally perform better the non-relational baselines. [Smaller is better]

5.2 Evaluation Metrics

The aim is to predict attribute values at each node at future timestamps. For this purpose, Mean Squared Error (MSE) loss is used. The lower MSE indicates better performance.

5.3 Baseline Methods

We show that the results produced by our model outperform those of the existing time series forecasting models. We compare against two kinds of attribute prediction methods.

Time series prediction without TKG. These methods do not take into account the graph data and make predictions using just the time-series history available. We compare our model to Historic Average (HA), Vector AutoRegressive (VAR) model [Hamilton, 1994], Autoregressive Integrated Moving Average (ARIMA) model, and GRU based Seq2Seq [Sutskever et al., 2014]. HA makes predictions based on the weighted average of the previous time series values, ARIMA uses lagged observations for prediction and VAR predicts multiple time series simultaneously by capturing linear interdependencies among multiple time series.

Time series prediction with TKG. Node embeddings are learned using graph representational learning methods (both static and temporal) like RE-NET, HyTE, TA-DistMult, and ConvE. For each node, the attribute value at a particular timestamp is concatenated with the corresponding node embedding, and this data is passed into an GRU network for making predictions.

Hyperparameter settings. All models are implemented in PyTorch using Adam Optimizer for training. The best hyperparameters are chosen using the validation dataset. Typically increasing value of λ gives better results, and the best results on each dataset are reported.

5.4 Main Results

The results for the attribute prediction on different datasets are reported in Table 2. We see that our method DARTNET outperforms every other baseline for the attribute prediction by a large margin. From the results it is clear, that the neural network based models outperform the other baselines on these complicated datasets proving their long term modeling capacity. We observe that the relational methods using graph information, generally outperform the non-relational methods on attribute prediction. Large increase in performance is observed for more complicated datasets like MTG and AGG. This suggests that it is the right direction for research to use relational methods for attribute prediction. DART-NET outperforms other relational methods, which does not jointly train embeddings for attribute prediction and link prediction. This suggests that joint training of embeddings for attribute prediction and link prediction improves the performance on attribute prediction rather than training embeddings separately and then using it for attribute prediction.

5.5 Performance Analysis

To study the effects of changes in model parameter sharing and hyperparameter sensitivity on prediction, we perform several ablation studies for DARTNET on four datasets as AGG does not have attribute values over all nodes.

1. **Decoupling of Attribute prediction and Interaction prediction tasks.** We decouple the shared parameters c_h and $\mathbf{W}_1$ for both tasks and observe the performance. More formally, we use a different embedding for both tasks, i.e $\mathbf{c}_h^I, \mathbf{W}_1^I$ and $\mathbf{c}_h^A, \mathbf{W}_1^A$ as the parameters for link prediction and attribute prediction task respectively.

2. **Sharing history.** We study the effect of using the same history embedding for both link prediction and attribute prediction. This will help us study if similar history information is required for both the tasks. Here the H_I

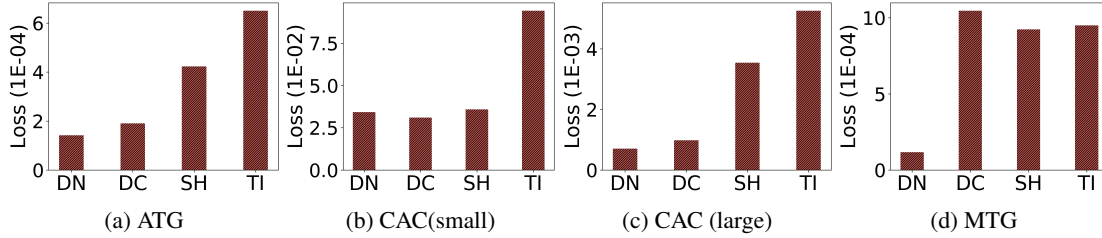(a) ATG   (b) CAC(small)   (c) CAC (large)   (d) MTG

Figure 3: Variation of Attribute Loss for attribute prediction on ATG, CAC(small), CAC(large), MTG for different variants of DARTNET [Legend - DN: DARTNET, DC: Decoupled, SH: Sharing History, TI: Time Independent]. Our method outperforms other methods. We see that our method performs substantially better than decoupled variant, same history setting and the time-independent setting. This suggests that joint modelling of link prediction and attribute prediction is helpful for attribute prediction. It also shows separate history encoding vectors are required for prediction and temporal information is essential for link prediction. [Smaller is better]
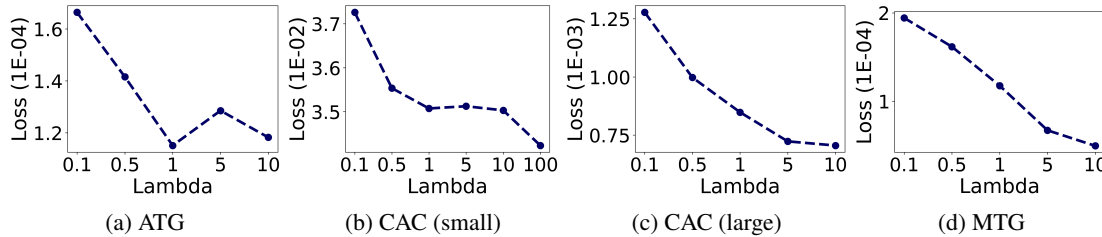

(a) ATG   (b) CAC (small)   (c) CAC (large)   (d) MTG

Figure 4: Sensitivity analysis of Attribute Loss with λ on ATG, CAC(small), CAC(large), MTG. We see a general trend that the Attribute Loss value decreases with increase in λ. This is also the expected trend. [Smaller is better]

does not explicitly get the related information so that we can share the weights. Hence the new equations become:

$$\mathbf{H}_I(h, r, \tau) = \mathbf{H}_A(h, \tau),$$

where the parameters of both the RNNs are shared.

3. **Study of Time-Dependent Information.** We evaluate the performance of our model in the absence of any temporal information. Hence we do not encode any history for any task and directly predict the tails and the attribute values at a future timestamp. Hence equations are:

$$\mathbf{a}'_{h,\tau+1} = f_A(\mathbf{c}_h),$$
$$P(t|h, r, G_1, \ldots, G_\tau) = f_I(\mathbf{c}_h, \mathbf{l}_r).$$

Analysis on Variants of DARTNET. Figure 3 shows the variation of Attribute Loss with different variants of DART-NET proposed in Section 5.5. From Figure 3, we observe that our model outperforms the decoupled variant by a large margin for attribute prediction. This confirms the hypothesis that joint training of attribute prediction and link prediction performs better than training separately. We also see that sharing history for attribute prediction and link prediction deteriorates the results, which indicates that the history encoding information required for link prediction and attribute prediction is quite different from each other. Lastly, the time-independent variant of our framework performs poorly. This clearly indicates that the temporal evolution information is essential for proper inference.

Sensitivity analysis of hyperparameter λ. We perform the sensitivity analysis of parameter λ, which specifies the weight given to both the tasks. We show the variation of MSE loss for

attribute prediction task with λ. Figures 4 shows the variation of Attribute Loss with increasing λ. In Figure 4, we observe that Attribute value decreases with increasing λ. As expected, as increasing lambda favors the optimization of attribute loss while decreasing lambda favors the link prediction.

6 Conclusion and Future Work

In this paper, we propose to jointly model the attribute prediction and link prediction on a temporally evolving graph. We propose a novel framework DARTNET, which uses two recurrent neural networks to encode the history for the graph. The framework shares the parameter for the two tasks and jointly trains the two tasks using multi-task learning. Through various experiments, we show that our framework is able to achieve better performance on attribute prediction than the previous methods indicating that external knowledge is useful for time series prediction. Interesting future work includes the link prediction on graph level rather than on subject and relation level in a memory-optimized way.

Acknowledgements

References

[Bordes *et al.*, 2013] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 2787–2795. Curran Associates, Inc., 2013.

[Caruana, 1997] Rich Caruana. Multitask learning. *Machine Learning*, 28(1):41–75, 1997.

[Cho *et al.*, 2014] KyungHyun Cho, Bart van Merrienboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder-decoder approaches. *CoRR*, abs/1409.1259, 2014.

[Dasgupta *et al.*, 2018] 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*, pages 2001–2011, Brussels, Belgium, October-November 2018. Association for Computational Linguistics.

[Dettmers *et al.*, 2017] Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. Convolutional 2d knowledge graph embeddings. *CoRR*, abs/1707.01476, 2017.

[García-Durán *et al.*, 2018] Alberto García-Durán, Sebastian Dumancic, and Mathias Niepert. Learning sequence encoders for temporal knowledge graph completion. *CoRR*, abs/1809.03202, 2018.

[Goyal *et al.*, 2018] Palash Goyal, Nitin Kamra, Xinran He, and Yan Liu. Dyngem: Deep embedding method for dynamic graphs. *CoRR*, abs/1805.11273, 2018.

[Hamilton *et al.*, 2017] William L. Hamilton, Rex Ying, and Jure Leskovec. Representation learning on graphs: Methods and applications. *CoRR*, abs/1709.05584, 2017.

[Hamilton, 1994] James Douglas Hamilton. *Time series analysis*. Princeton Univ. Press, Princeton, NJ, 1994.

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

[Kipf and Welling, 2016] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *CoRR*, abs/1609.02907, 2016.

[Kipf *et al.*, 2018] Thomas N. Kipf, Ethan Fetaya, Kuan-Chieh Wang, Max Welling, and Richard S. Zemel. Neural relational inference for interacting systems. In *ICML*, 2018.

[Kumar *et al.*, 2018] Srijan Kumar, Xikun Zhang, and Jure Leskovec. Learning dynamic embeddings from temporal interactions. *CoRR*, abs/1812.02289, 2018.

[Li *et al.*, 2017] Yaguang Li, Rose Yu, Cyrus Shahabi, and Yan Liu. Graph convolutional recurrent neural network: Data-driven traffic forecasting. *CoRR*, abs/1707.01926, 2017.

[Lippi *et al.*, 2013] Marco Lippi, Matteo Bertini, and Paolo Frasconi. Short-term traffic flow forecasting: An experimental comparison of time-series analysis and supervised learning. *IEEE Transactions on Intelligent Transportation Systems*, 14:871–882, 2013.

[Liu *et al.*, 2011] Wei Liu, Yu Zheng, Sanjay Chawla, Jing Yuan, and Xie Xing. Discovering spatio-temporal causal interactions in traffic data streams. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '11, pages 1010–1018, New York, NY, USA, 2011. ACM.

[Neil *et al.*, 2018] Daniel Neil, Joss Briody, Alix Lacoste, Aaron Sim, Paidi Creed, and Amir Saffari. Interpretable graph convolutional neural networks for inference on noisy knowledge graphs. *CoRR*, abs/1812.00279, 2018.

[Pareja *et al.*, 2019] Aldo Pareja, Giacomo Domeniconi, Jie Chen, Tengfei Ma, Toyotaro Suzumura, Hiroki Kanezashi, Tim Kaler, and Charles E. Leisersen. Evolvegcn: Evolving graph convolutional networks for dynamic graphs. *CoRR*, abs/1902.10191, 2019.

[Rossi, 2018] Ryan A. Rossi. Relational time series forecasting. *The Knowledge Engineering Review*, 33:e1, 2018.

[Ruder, 2017] Sebastian Ruder. An overview of multi-task learning in deep neural networks. *CoRR*, abs/1706.05098, 2017.

[Schlichtkrull *et al.*, 2018] Michael Schlichtkrull, Thomas N. Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. Modeling relational data with graph convolutional networks. *Lecture Notes in Computer Science*, page 593–607, 2018.

[Sutskever *et al.*, 2014] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. Sequence to sequence learning with neural networks. *CoRR*, abs/1409.3215, 2014.

[Trivedi *et al.*, 2017] Rakshit Trivedi, Hanjun Dai, Yichen Wang, and Le Song. Know-evolve: Deep reasoning in temporal knowledge graphs. *CoRR*, abs/1705.05742, 2017.

[Trivedi *et al.*, 2018] Rakshit Trivedi, Mehrdad Farajtabar, Prasenjeet Biswal, and Hongyuan Zha. Representation learning over dynamic graphs. *CoRR*, abs/1803.04051, 2018.

[Wu *et al.*, 2019] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and Philip S. Yu. A comprehensive survey on graph neural networks. *CoRR*, abs/1901.00596, 2019.

[Yang *et al.*, 2014] Bishan Yang, Wen tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. Embedding entities and relations for learning and inference in knowledge bases. *CoRR*, abs/1412.6575, 2014.

[You *et al.*, 2018] Jiaxuan You, Rex Ying, Xiang Ren, William L. Hamilton, and Jure Leskovec. Graphrnn: A deep generative model for graphs. *CoRR*, abs/1802.08773, 2018.