An Effective Graph Learning based Approach for Temporal Link Prediction: The First Place of WSDM Cup 2022

Qian Zhao
Ant Group
zq317110@alibaba-inc.com

Shuo Yang
Ant Group
kexi.yys@antfin.com

Binbin Hu*
Ant Group
bin.hbb@antfin.com

Zhiqiang Zhang
Ant Group
lingyao.zzq@alipay.com

Yakun Wang
Ant Group
feika.wyk@antgroup.com

Yusong Chen
Ant Group
chenyusong.cys@alibaba-inc.com

Jun Zhou
Ant Group
jun.zhoujun@alibaba-inc.com

Chuan Shi
Beijing University of Posts and Telecommunications
shichuan@bupt.edu.cn

ABSTRACT

Temporal link prediction, as one of the most crucial work in temporal graphs, has attracted lots of attention from the research area. The WSDM Cup 2022 seeks for solutions that predict the existence probabilities of edges within time spans over temporal graph. This paper introduces the solution of AntGraph, which wins the 1st place in the competition. We first analysis the theoretical upper-bound of the performance by removing temporal information, which implies that only structure and attribute information on the graph could achieve great performance. Based on this hypothesis, then we introduce several well-designed features. Finally, experiments conducted on the competition datasets show the superiority of our proposal, which achieved AUC score of 0.666 on dataset A and 0.902 on dataset B, the ablation studies also prove the efficiency of each feature.

KEYWORDS

Link Prediction, Gradient Boosting Decision Trees, Graph Learning, WSDM Cup 2022

1 INTRODUCTION

As graphs are ubiquitously exist in a wide range of real-world applications, many problems can be formulated as specific tasks over graphs. And link prediction [4], as one of the most important task in graph-structured datasets, is widely applied in biology [10], recommendation [3, 14] and finance [12]. Meanwhile, real-world data is usually evolving over time, some following literature [11, 13] consider devising temporal graph learning models to uncover temporal information. However, predicting the links on a temporal graph is more non-trivial. WSDM Cup 2022 calls for solutions that predicting the probability of a link within a period of time. In this paper, we will introduce the solution of AntGraph team, which ranks the first of the competition (achieved AUC score of 0.666 on dataset A and 0.902 on dataset B). And this technical report is organized as following:

• First, we give some statistics on the datasets, do some exploratory analyses and introduce the motivation of the method. According to the data analyses, we surprisingly find that removing the time span information in prediction could also achieve satisfactory performance.
• Subsequently, we introduce the data processing flow, enumerate several feature engineering methods ranging from network embedding to heuristic graph structure.
• Finally, we conduct comprehensive experiments on the competition datasets, which show the effectiveness of our proposal, and exhaustive ablation studies also show the improvement of each kind of feature.

Our source code are publicly available on GitHub\footnote{https://github.com/im0qianqian/WSDM2022TGP-AntGraph}.

2 DATASETS

In this section, we focus on the exploratory of datasets provided by the competition, and an in-depth analysis is presented, followed by the detailed introduction of evaluation metrics.

Table 1: The statistics of two datasets. Note that "# Node" of Dataset B is obtained by the maximum value of node ids, and "Inter." is short for "Intermediate".

|                  | Dataset A | Dataset B |
|------------------|-----------|-----------|
| # Train          | 27,045,268| 8,278,431 |
| # Initial Test   | 8,197     | 3,863     |
| # Inter. Test    | 49,903     | 49,940    |
| # Final Test     | 200,000   | 200,000   |
| # Nodes          | 19,942    | 1,304,045 |
| # Edges          | 27,045,268| 8,278,431 |
| # Node feat.     | 8         | N.A.      |
| # Edge feat.     | N.A.      | 768       |
| # Edge type      | 248       | 14        |

\*Corresponding author.
Table 2: The analysis of the existence of same edges in the initial test set.

|                | Total | Exist in graph | Exist in graph label = 1 | Exist in graph label = 0 | Not exist label = 1 | Not exist label = 0 |
|----------------|-------|----------------|--------------------------|--------------------------|---------------------|---------------------|
| Dataset A      | 8197  | 7354 (89.72%)  | 3333 (40.66%)            | 4021 (49.05%)            | 183 (2.23%)         | 660 (8.05%)         |
|                |       | 5886 (71.81%)  | 2755 (33.61%)            | 3131 (38.20%)            | 761 (9.28%)         | 1550 (18.91%)       |
| Dataset B      | 3863  | 3195 (82.71%)  | 2123 (54.96%)            | 1072 (27.75%)            | 128 (3.31%)         | 540 (13.98%)        |
|                |       | 2612 (67.62%)  | 1685 (43.62%)            | 927 (24.00%)             | 566 (14.65%)        | 685 (17.73%)        |

Table 3: The performance w.r.t. AUC of our native strategy compared to the baseline model provided by the sponsor on both initial and intermediate (Inter.) test set.

| Method            | Initial test | Inter. test |
|-------------------|--------------|-------------|
| Baseline model    | 0.5110       | 0.5026      |
| Dataset A         |              |             |
| Naive strategy (w.o. edge type) | 0.5428       | 0.5432      |
| Naive strategy (w.i. edge type) | 0.5597       | 0.5687      |
| Dataset B         |              |             |
| Naive strategy (w.o. edge type) | 0.6391       | 0.8655      |
| Native strategy (w.i. edge type) | 0.5867       | 0.8059      |

2.1 A Brief Description

The competition expects participants to adopt a single model (hyperparameters can vary) that works well on two kinds of data simultaneously, and thus correspondingly provides two representative large-scale temporal graph datasets.

- **Dataset A** characterizes a dynamic event graph with entities as nodes and different types of events as edges. Each node may be associated with rich features if available, and except for the edge types, no other information is available for edges.

- **Dataset B** characterizes a user-item graph with users and items as nodes and different types of interactions as edges. Each edge is associated with rich features if available, and no feature information is available for nodes. Noting that the sponsor treats the user-item graph as a bipartite graph. For convenience, we try to convert this graph to an undirected multi-relation graph through shifting item ids. In particular, we perform the above operation by adding the sum of 1 and the maximum value of user ids (denotes as $\text{Offse}_{tu}$) for each original item id, as follows:

$$\text{node}_{id} = \begin{cases} 
\text{node}_{id} & \text{if an user} \\
\text{node}_{id} + \text{Offse}_{tu} & \text{if an item}
\end{cases}$$

Since the competition asks participants to predict whether an edge will exist between two nodes within a given time span, instead of a single timestamp in the graph, a start and an end timestamp is respectively given for each query in test stage. Also, for each dataset, the sponsor provides a train set, an initial test set, an intermediate test set and a final test set, and the labels of intermediate test set and final test set are still not available at present. It is worthwhile to note that only the performance of the model in the final test set determines the ranking of the competition.

In summary, we detailed all necessary statistics of two datasets in Table 1.

2.2 Data Analysis

Generally, an inspiring data analysis could shed some light on the model design, which plays a vital role in various data mining tasks. Based on the originally provided data (i.e., the train set and the initial test set), we perform a series of detailed data analysis as follows:

- **The existence of same edges in the test.** We first analyze whether edges of the initial test set have already existed in the original graph. In this analysis, timestamps are not taken into consideration. As shown in Table 2, we observe that the original graph contains most of edges in the initial test set for both datasets, especially when the edge type is ignored. Surprisingly, we also find that approximately half of edges (i.e., 49.05% without edge type and 33.61% with edge type) existed in the graph keep the same labels for Dataset A, while about three quarters of edges (i.e., 54.96% with edge type and 43.62% without edge type) existed in the graph keep the same labels for Dataset B.

- **Optimal performance without consideration of timestamps.** Secondly, we also explore theoretical upper-bounds on performance without temporal information. We select the data with the same node pair in the initial set, and then calculate the mode or mean value for all labels as the prediction result of these data. Experiments show that the model can still achieve a good performance, as shown in Table 4.

2.3 Evaluation Metrics

This competition uses Area Under ROC (AUC) as the evaluation metric for both tasks. Intuitively, the two tasks have different difficulties, and sacrificing one task to do well on the other is not expected. Therefore, the competition further adopt the average of T-scores as the ranking basis for encouraging the model to perform well on different tasks. The formal definition is introduced as...
As mentioned above, the goal of this competition is to predict whether we remove the edge features for Dataset B, since these features are not available for most of edges, i.e., the non-empty ratio is 6.67%.

| Description | Initial test (mode) | Initial test (mean) |
|-------------|---------------------|---------------------|
| Dataset A   | node pair (w.o. edge type) | 0.9040 | 0.9776 |
|             | node pair (w.i. edge type) | 0.9900 | 0.9997 |
| Dataset B   | node pair (w.o. edge type) | 0.8946 | 0.9795 |
|             | node pair (w.i. edge type) | 0.9147 | 0.9875 |

Table 4: Explore the maximum AUC without temporal information.

\[ T_{score} = \frac{AUC - \text{mean}(AUC)}{\text{std}(AUC)} \times 0.1 + 0.5 \]  
\[ \text{AverageOfTscore} = \frac{T_{score_A} + T_{score_B}}{2} \]

where mean(AUC) and std(AUC) represents the mean and standard deviation of AUC of all participants. Clearly, an larger average of T-scores means a better performance.

3 METHODOLOGY

In this section, we introduce our complete solution for large-scale temporal graph link prediction task, which consists of train data construction component, feature engineering component and downstream model training component. In the following, we will zoom into each well designed component.

3.1 Train Data Construction

As mentioned above, the goal of this competition is to predict whether an edge will exist between two nodes within a given time span, whereas each edge in the provided graphs is only associated with a single timestamp. Hence, the inconsistent problem between training and testing severely threatens the generalization of models. In addition, previous data analysis has concluded that this task may not benefit from involving timestamps, therefore, we construct the train data without timestamps as follows:

3.1.1 Negative sampling. For efficient training, we adopt the shuffling based sampling strategy to sample negative instance in batch, rather than the whole node set. Moreover, the timestamps are ignored in our negative sampling process. In particular, our negative sampling process is detailed is follows: i) We denotes edges in the original graphs as the positive instance set, consisting of source nodes, target nodes and relations. ii) We only keep source nodes unchanged, and randomly shuffle target nodes and relations to generate the negative instance set. iii) We combine the above positive and negative instance set, and uniformly sample a certain number of instances to construct the final train set.

3.1.2 Removing redundant features. Firstly, we remove all time related features, including timestamp, start time, end time. Moreover, we remove the edge features for Dataset B, since these features are not available for most of edges, i.e., the non-empty ratio is 6.67%.

3.2 Feature Engineering

3.2.1 LINE embedding. As concluded in the previous data analysis, the first-order relation is of crucially importance in our link prediction task. In order to capture such deep correlation between nodes in an more fine-grained manner, we introduce the LINE embedding, an effective and efficient graph learning framework based on the graph structure to help downstream model capture high-order information:

- Unitary feature w.r.t. individual nodes: i) The degree of this node; ii) The number of different nodes adjacent to this node; iii) The number of different edge types adjacent to this node.
- Binary information w.r.t. node pairs: i) The number of one hop paths between two nodes; ii) The number of two hop paths between two nodes; iii) The number of different edge types between two nodes.
- Ternary information w.r.t. node pairs and edge types: The number of occurrences of this triplet.

3.2.3 Subgraph features. In addition, we also added the following statistical features based on the graph structure to well help downstream model capture high-order information:

- A feature w.r.t. source node, relation, target.
- A feature w.r.t. source node, target.
- A feature w.r.t. relation.
- A feature w.r.t. both source and target.
- A feature w.r.t. both source and target, and their common neighbors.
- A feature w.r.t. all nodes in the graph.

4 EXPERIMENTS

4.1 Overall Performance

Performance in the leaderbord. We present the results of top five teams from the leaderboard in Table 6. We observation that our solution achieve the best performance in Dataset A and competitive performance in Dataset B. As the best performance achieved w.r.t. the final ranking metric further indicating that our solution works well on both kinds of data simultaneously.
We summarize the comparison results in Table 5, and we have
work embedding methods, i.e.,

In this section, we perform a series of ablation studies to
analyses
4.2 Ablation studies

the impact of kinds of features proposed in Section 3.2, incl uding
LINE embedding, node crossing features and subgraph featur es.

improvement of 7.22% on dataset A and 53.83% on dataset B, respec-
tively.

provement of 7.22% on dataset A and 53.83% on dataset B, respe c-
using raw feature, our final submitted model achieve a relati ve im-

<sparser than Dataset A, and thus subgraph structure are hard ly
in Dataset A. An intuitive explanation is that Dataset B is mu ch
subgraph features in Dataset B, while opposite trend is obse rved
Line features show a greater improvement than models involv ing
in most cases with the incorporation of all features. ii) Inv olving
achieve better performance, and the best performance is yie lded

baseline
DeepWalk [5]
TransE [1]
RotatE [7]
ComplEx [9]
LINE [8]
catboost (raw input data)
+ LINE embedding
+ Subgraph features
+ LINE embedding + Subgraph features
+ LINE embedding + Node crossing features
+ All (submitted version)

1 AntGraph (Ours)
2 nothing here
3 NodeInGraph
4 We can [mask]!
5 IDEAS Lab UT

Table 5: Overall experimental results of different methods in two datasets.

| Model                        | Dataset A | Dataset B |
|------------------------------|-----------|-----------|
|                             | Initial AUC | Inter. AUC | Initial AUC | Inter. AUC |
| baseline                     | 0.5110     | 0.5026    | 0.5100      | 0.5026     |
| DeepWalk [5]                 | 0.5352     | 0.5707    | 0.5246      | 0.4985     |
| TransE [1]                   | 0.5182     | 0.5614    | 0.6389      | 0.8903     |
| RotatE [7]                   | 0.5315     | 0.5736    | 0.6323      | 0.8981     |
| ComplEx [9]                  | 0.5514     | 0.5821    | 0.6359      | 0.9014     |
| LINE [8]                     | 0.6072     | 0.6320    | 0.6425      | 0.8905     |
| catboost (raw input data)    | 0.6045     | 0.6222    | 0.5545      | 0.5869     |
| + LINE embedding             | 0.6377 (+5.49%) | 0.6540 (+5.11%) | 0.6399 (+15.40%) | 0.9013 (+53.57%) |
| + Subgraph features          | 0.6611 (+9.36%) | 0.6673 (+7.25%) | 0.5861 (+5.70%) | 0.7561 (+28.83%) |
| + LINE embedding + Subgraph features | 0.6619 (+9.50%) | 0.6659 (+7.02%) | 0.6368 (+14.84%) | 0.8978 (+52.97%) |
| + LINE embedding + Node crossing features | 0.6573 (+8.73%) | 0.6673 (+7.25%) | 0.6504 (+17.29%) | 0.9001 (+53.37%) |
| + All (submitted version)    | 0.6657 (+10.12%) | 0.6671 (+7.22%) | 0.6459 (+16.48%) | 0.9028 (+53.83%) |

Table 6: Top five results in the final leaderboard.

| Rank | Team name     | Dataset A Final AUC | Dataset B Final AUC | Average of T/100 |
|------|---------------|---------------------|---------------------|-----------------|
| 1    | AntGraph(Ours) | 0.666001            | 0.630737            | 1531            |
| 2    | nothing here  | 0.662482            | 0.906923            | 0.628942        |
| 3    | NodeInGraph   | 0.627821            | 0.865567            | 0.585137        |
| 4    | We can [mask]!| 0.603621            | 0.898232            | 0.572372        |
| 5    | IDEAS Lab UT  | 0.605264            | 0.873949            | 0.566849        |

Compare to baselines. We compare our methods with other 6
methods, including the official baseline\(^2\) and several classic net-
work embedding methods, i.e., DeepWalk [5], TransE [1], RotatE [7],
LINE [8], and ComplEx [9]. The experimental results in Table 5
shows that our solution outperforms all baselines by a considera-
able margin.

Overall, both of observations verify the effectiveness of our
proposal.

4.2 Ablation studies

In this section, we perform a series of ablation studies to analys es
the impact of kinds of features proposed in Section 3.2, including
LINE embedding, node crossing features and subgraph features.
We summarize the comparison results in Table 5, and we have fol lowing observations: i) All extracted feature help base model
achieve better performance, and the best performance is yielded
in most cases with the incorporation of all features. ii) Involv ing
LINE features show a greater improvement than models involving
subgraph features in Dataset B, while opposite trend is observed
in Dataset A. An intuitive explanation is that Dataset B is much
sparser than Dataset A, and thus subgraph structure are hardly
exploited in Dataset B. Overall, compared to the base model only
using raw feature, our final submitted model achieve a relative im-
provement of 7.22% on dataset A and 53.83% on dataset B, respec-
tively.

\(^2\)https://github.com/dglai/WSDM2022-Challenge

5 CONCLUSION

This paper describes our solution for WSDM 2022 Challenge - Tem-
poral Link Prediction. For this task, we design a novel negative
sampling strategy, and combined with data analysis to delete re-
dundant information. We introduce the LINE embedding to pro-
vide local and global features of graph. At the same time, we de-
sign node crossing features and subgraph features. In the end, our
team AntGraph was ranked the 1st place on the final leaderboard.

REFERENCES

[1] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Ok-
sana Yakhnenko. 2013. Translating embeddings for modeling multi-relational
data. In NIPS.
[2] Binbin Hu, Yuan Fang, and Chuan Shi. 2019. Adversarial learning on heteroge-
neous information networks. In SIGKDD: 120–129.
[3] Binbin Hu, Chuan Shi, Wayne Xin Zhao, and Philip S Yu. 2018. Leveraging meta-
path based context for top-n recommendation with a neural co-attention model.
In SIGKDD: 1531–1540.
[4] Ajay Kumar, Shashank Sheshar Singh, Kuldip Singh, and Bhaskar Biwas. 2020.
Link prediction techniques, applications, and performance: A survey. Physical A:
Statistical Mechanics and its Applications 553 (2020), 124289.
[5] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. Deepwalk: Online learn-
ing of social representations. In SIGKDD: 701–710.
[6] Liudmila Prokhorenkova, Gleb Gusev, Aleksandr Vorobev, Anna Veronika Doro-
gush, and Andrey Gulin. 2018. CatBoost: unbiased boosting with categorical
features. In NIPS.
[7] Zhiqing Sun, Zhi-Hong Deng, Jian-Jun Nie, and Jian Tang. 2019. RotatE: Knowl-
edge Graph Embedding by Relational Rotation in Complex Space. In ICLR.
[8] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei.
2015. LINE: Large-scale Information Network Embedding. In WWW.
[9] Theo Trouillon, Johannes Welbl, Sebastian Riedel, Eric Gaussier, and Guillaume
Bouchard. 2016. Complex embeddings for simple link prediction. In ICML: 2071–
2080.
[10] Turki Turki and Zhi Wei. 2017. A link prediction approach to cancer drug sensi-
tivity prediction. BMC systems biology 11, 5 (2017), 1–14.
[11] Xiaohong Wang, Ding Lyu, Mengzian Li, Yang Xia, Qi Yang, et al. 2021. APAN:
Asynchronous Propagation Attention Network for Real-time Temporal Graph
Embedding. In SIGMOD: 2628–2638.
[12] Shuo Yang, Binbin Hu, Zhiquang Zhang, et al. 2021. Inductive Link Prediction
with Interactive Structure Learning onAttributed Graph. In ECML-PKDD: 383–
398.
[13] Shuo Yang, Zhiquang Zhang, Jun Zhou, et al. 2020. Financial Risk Analysis for
SMEs with Graph-based Supply Chain Mining. In WWW: 4661–4667.
[14] Rex Ying, Ruining He, Kaifeng Chen, Pong Ekombehatchai, William I. Hamilton,
and Jure Leskovec. 2018. Graph convolutional neural networks for web-scale rec-
ommender systems. In SIGKDD: 974–983.