# The Evaluation of DyHATR Performance for Dynamic Heterogeneous Graphs

**Nasy`an Taufiq Al Ghifari, Gusti Ayu Putri Saptawati*, Masayu Leylia Khodra & Benhard Sitohang**

School of Electrical Engineering and Informatics, Institut Teknologi Bandung,
Jalan Ganesa 10 Bandung 40132, Indonesia
*E-mail: putri@staff.stei.itb.ac.id

**Abstract.** Dynamic heterogeneous graphs can represent real-world networks. Predicting links in these graphs is more complicated than in static graphs. Until now, research interest of link prediction has focused on static heterogeneous graphs or dynamically homogeneous graphs. A link prediction technique combining temporal RNN and hierarchical attention has recently emerged, called DyHATR. This method is claimed to be able to work on dynamic heterogeneous graphs by testing them on four publicly available data sets (Twitter, Math-Overflow, Ecomm, and Alibaba). However, after further analysis, it turned out that the four data sets did not meet the criteria of dynamic heterogeneous graphs. In the present work, we evaluated the performance of DyHATR on dynamic heterogeneous graphs. We conducted experiments with DyHATR based on the Yelp data set represented as a dynamic heterogeneous graph consisting of homogeneous subgraphs. The results show that DyHATR can be applied to identify link prediction on dynamic heterogeneous graphs by simultaneously capturing heterogeneous information and evolutionary patterns, and then considering them to carry out link predicition. Compared to the baseline method, the accuracy achieved by DyHATR is competitive, although the results can still be improved.

**Keywords**: *dynamic; graph mining; heterogeneous; link prediction; performance evaluation.*

## 1 Introduction

A graph can be seen as a tool to model complex real worlds, since it focuses on modeling interaction in the real world. Some examples of real-world interactions modeled in graphs include recommendation systems [1,2], author-article relationships [3,4], artist-film relationships [5], and others. In fact, every relation that exists in the real world can be modeled as a multiple graph. For example, the process of identifying people based on their images, which can be modeled into a homogeneous graph [6], the drug discovery process based on protein targets and disease types, which can be modeled into a heterogeneous graph [7], the influence of the social environment on user preferences can be modeled into a

Received March, 13th, 2023, Revised August 6th, 2023, Accepted for publication August 21st, 2023.
 ISSN: 2337-5787, DOI: 10.5614/itbj.ict.res.appl.2023.17.2.7

dynamic graph [8], and customer preferences for products to be purchased can be modeled into a dynamic heterogeneous graph [9]. In order to gain insight from the various case study examples and types of graphs mentioned previously, the use of graph mining can be used to analyze graph data, such as graph classification, which is useful for determining the category or target label of a graph; node classification, which is used to predict the class of nodes not labeled as property nodes based on other property nodes; link predictions used to find new edges or find repeated edges in a graph so that a graph topology is obtained at the next time interval; and many more applications [10-15].

Link prediction as a part of graph mining has so far been applied to various use cases, such as predicting friendship [16,17], predicting customer purchase intentions [18-20], predicting cures for a disease [7,21], and so on. The link prediction process is different for various types of graphs. For example, link prediction on homogeneous graphs and heterogeneous graphs has different characteristics because in link prediction on homogeneous graphs, the nodes and edges have the same type so that the process steps that occur can be simpler, while the nodes or edges in a heterogeneous graph have different types, so that link prediction on heterogeneous graphs requires a more complex process [7,21-23]. Even though the process steps are very complex, the existing link prediction methods for heterogeneous graphs are focused on static heterogeneous graphs. To develop link prediction on dynamic graphs, research has been conducted using dynamic homogeneous graph types [24]. If you want to carry out link prediction on a dynamic heterogeneous graph, the effort required increases because not only heterogeneous structural information must be processed but also the topological structure of the evolving graph. Recently, Xue, *et al.* [25] proposed a link prediction method based on deep learning. They called this method DyHATR. The idea behind this method is to use a hierarchical attention model to capture heterogeneous information in a snapshot graph and then study the pattern of graph evolution over time using a temporal attention RNN. The snapshot graph is literally an image of the current graph topology.

Of the studies mentioned above, only DyHATR is claimed to be able to perform link prediction on dynamic heterogeneous graphs. However, the research did not prove this claim, because the experiments were carried out on dynamic, partially heterogeneous graph types and dynamic bipartite graphs. To see the performance of DyHATR on dynamic heterogeneous graphs, experiments with dynamic heterogeneous graphs must be conducted. Therefore, the contribution of this research is to evaluate the performance of DyHATR for link prediction on dynamic heterogeneous graphs. The evaluation process was carried out by conducting experiments with the DyHATR method on a data set that represents dynamic heterogeneous graphs, namely the Yelp data set. We evaluated the performance of the DyHATR method on Yelp data by comparing the experimental results with the same baseline as the original DyHATR experiments.

The rest of the paper is organized as follows. Part 2 presents prelimenaries on link prediction and various types of graphs. Part 3 presents related work. Part 4 presents how the DyHATR method works when doing link prediction. Part 5 provides the results of the empirical experiments with the model compared to a state-of-the-art approach using publicly available data sets. Finally, the conclusions and future work are given in Part 6.

## 2 Prelimenaries on Link Prediction and Various Types of Graphs

Link prediction is an important aspect of complex network analysis and includes assessment of potential links and prediction of future connections [26]. If there is an undirected graph $G = (V, E)$, where $V$ is the set of nodes and $E$ is the set of edges provided that relations connecting to itself are not allowed. Then $U$ is a universal set that contains all possible edges. Thus, the set of edges that do not exist is $U - E$. Link prediction is performed to determine which edges will appear in the $U - E$ set in the future.

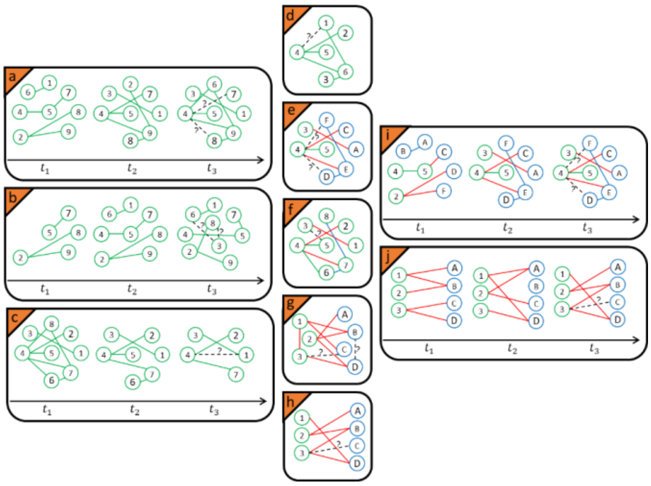

**Figure 1** Illustration of various types of graphs.

The way to implement link prediction for each type of graph varies depending on the type of graph being processed. Graph types can be categorized based on several criteria, namely based on changes in topology and based on the types of nodes and edges. Based on topology changes, graphs can be categorized into two types [27], namely:

1. Static graphs – a type of graph that has a fixed topology (Figure 1(d)).
2. Dynamic graphs – a type of graph that changes its topology during a certain time interval (Figure 1(a)).
   Dynamic graphs can be further categorized based on the topological changes that occur:
   a) Fully dynamic graphs – a type of graph that has additions and deletions at the nodes and/or edges (Figure 1(a)).
   b) Partial dynamic graphs – a type of graph where nodes and/or edges are added or deleted (Figures 1(b) and 1(c)). Thus, after a certain time interval there are only two possible topologies, namely a dense graph or a sparse graph.

Based on the types of nodes and edges, graphs can be categorized into two types [6], namely:

1. Homogeneous graphs – a type of graph that has a single node type and a single edge type (Figure 1(d)).
2. Heterogeneous graphs – a type of graph that has several types of nodes and several types of edges (Figure 1(e)).
   Heterogeneous graphs can be further categorized based on the variation in the types of nodes connected to the edges:
   a) Fully heterogeneous graphs – a type of graph that has several types of nodes and several types of edges (Figure 1(e)).
   b) Partial heterogeneous graphs – a type of graph that has several types of nodes or several types of edges (Figures 1(f) and 1(g)).
   c) Bipartite graphs – a type of graph that has two types of nodes and one or several types of edges, where all types of edges only connect a node in node set $V_i$ to a node in node set $V_j$. There is no edge that connects nodes in the set of nodes $V_i$ of a homogeneous subgraph (Figure 1(h)).

Based on the explanation above, even more complex types of graphs can be distinguished, such as:

1. Dynamic homogeneous graphs – a type of graph that has a single node type and a single edge type, which changes its topology during a certain time interval (Figure 1(a)) [27].
2. Static heterogeneous graphs – a type of graph that has several types of nodes and several types of edges, but the topology of the graph remains the same (Figure 1(e)) [6].
3. Dynamic heterogeneous graphs – a type of graph that has several types of nodes and several types of edges that experience topological changes over a certain time interval (Figure 1(i)) [28,29].
4. Dynamic heterogeneous bipartite graphs – a type of graph that has two types of nodes and one or more types of edges, where all types of edges only

connect a node in the set of nodes $V_i$ to a node in the set of nodes $V_j$. There is no edge that connects nodes in the set of nodes $V_i$ of a homogeneous subgraph and the topology of the graph changes over a certain time interval (Figure 1(j)) [30].

With the various types of graphs described above, it turns out that the type of graph affects the link prediction process. For simple graph types such as static homogeneous graphs, the way to do link prediction on them is simpler than for more complex graph types such as dynamic heterogeneous graphs. This is because the structural and semantic information possessed by dynamic heterogeneous graphs is richer than in the case of static homogeneous graphs.

## 3 Related Work

Link prediction has been classified into four types by Daud [31], namely similarity, probability, algorithmic, and hybrid approaches. Among the similarity type, there are three categories, namely global indices, local indices, and quasi-local indices. Among the probabilistic type, there are four categories, namely probability tensor factorization models, probability latent variables, Markov models, and link label modeling. There are three categories for the algorithmic type, namely metaheuristic, matrix factorization, and machine learning. The hybrid type are approaches that combine two or more other approaches.

Xue's [25] modeling of graphs by encoding the network structure into non-linear space and representing network nodes as low-dimensional features, commonly referred to as network embedding [32,33], includes link prediction using machine learning. So far, there have been several studies that made link predictions for dynamically heterogeneous graphs based on network embedding. MetaDynaMix [34] integrates metapath-based topology features and latent representations to study heterogeneity and temporal evolution. Change2vec [35] focuses on measuring changes within snapshots rather than learning all the structural information from each snapshot, and also uses a metapath-based model to capture heterogeneous information. Both of the above methods focus on short-term evolutionary information between adjacent snapshots of the dynamic network and thus are insufficient to capture long-term evolutionary patterns. Recently, Sajadmanesh, *et al.* [36] used a recurrent neural network model to study long-term evolution patterns of dynamic networks over a metapath-based model and proposed a non-parametric generalized linear model, NP-GLM, to predict continuous-time relationships. Yin, *et al.* [37] proposed the DHNE method, which studies historical and current heterogeneous information and models evolutionary patterns by constructing a comprehensive historical-current network based on sequential snapshots. Then, DHNE performs metapath-based random walks and dynamic heterogeneous skip-gram models to capture node representations. Kong, *et al.* [38] introduced a dynamic heterogeneous information network embedding method called HA-LSTM. It uses graph convolutional networks (GCN) to study heterogeneous information networks and uses attention models and long-short time memory (LSTM) to capture information that evolves over time.

The application of link prediction for dynamic and heterogeneous graph types is important because many real-world networks are dynamic, that is, their network structure develops over time and is also heterogeneous, where the network structure is filled with nodes and edges that have many types. Thus, the current need for dynamic heterogeneous graph analysis is growing. For example, a customer-product network is usually a heterogeneous graph because it has two types of nodes that represent each different entity. Further, the relationships that are formed can also vary, namely customer-customer relationships, customer-product relations, and product-product relations. The nodes and edges in the customer-product network also evolve, indicating dynamic customer activity.

## 4 Dynamic Heterogeneous Network Embedding Method

The dynamic heterogeneous network embedding method called DyHATR, was proposed by Xue, *et al.* [25] for predicting edges that exist in dynamic heterogeneous graphs using two unique models: (1) a hierarchical attention model to study snapshots of static heterogeneous graphs; (2) a temporal attentive RNN model that is in charge of capturing the pattern of graph evolution. Figure 2 shows the architecture of DyHATR.

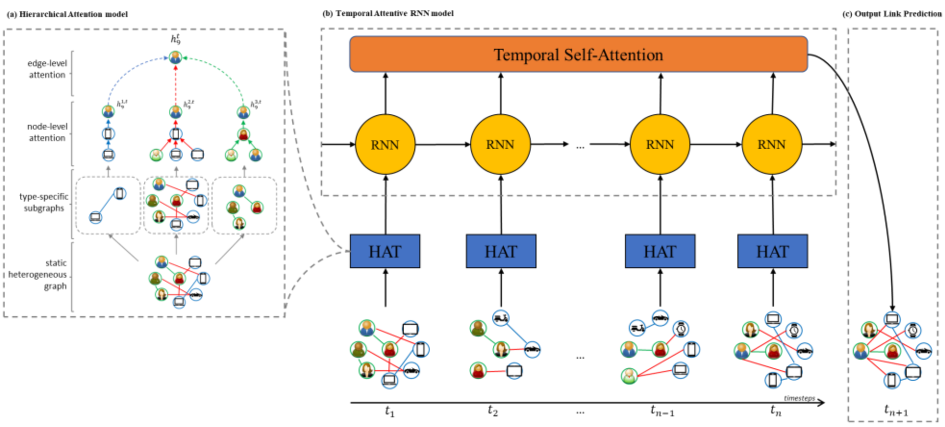


**Figure 2** Overall workflow of the DyHATR method [25].

The hierarchical attention model shown in Figure 2(a) processes each snapshot of a static heterogeneous graph. Firstly, we need to divide this heterogeneous graph into several subgraphs based on edge type. These subgraphs further represent the importance weight of each node's neighborhood, which is part of node-level attention.

In edge-level attention, some information is integrated from subgraphs that have different edge types so that the importance weights of various edge types can be studied and generate novel embeddings. First, edge-specific embedding is combined with a non-linear transformation function that is mapped to the same feature space. After capturing heterogeneity information from a static graph snapshot using the hierarchical attention model, a temporary evolution pattern is modeled (see Figure 2(b)), usually indicated by the appearance and disappearance of nodes and edges over time. Since recurrent neural network (RNN) has shown promising performance for dynamic network embedding methods [39,40], Xue, *et al.* [25] adopted it to model sequential information and learning temporal evolutionary patterns. At this stage, Xue et al. [25] provides two temporal learner options, namely long short-term memory (LSTM) and gated-recurrent-unit (GRU), to be experimented with in combination with DyHATR.

The output of the RNN model obtained is a concatenate of state vectors, after which the most recent state is selected as the final embedding of a node. Xue, *et al.* [25] argue that this method could result in information loss and is unable to record the most important embedding features. To capture important feature vectors in their model, Xue, *et al.* [25] applies a temporal-level attention model to the output of the RNN model, shown in Figure 2(b). By developing a temporal-level self-attention model, evolutionary patterns can be better captured across dynamic graphs. The addition of the temporal-level self-attention model after the RNN model anticipates the dynamic linkages in the last snapshot.

**Table 1** Statistical data sets used by DyHATR [25].

| Data sets | Nodes | Edges | Node Types | Edge Types | Snapshots |
|---|---|---|---|---|---|
| Twitter | 100,000 | 63,410 | 1 | 3 | 7 |
| Math-Overflow | 24,818 | 506,550 | 1 | 3 | 11 |
| EComm | 37,724 | 91,033 | 2 | 4 | 11 |
| Alibaba | 16,620 | 93,956 | 2 | 3 | 11 |

Table 1 presents a summary of the four original data sets tested using DyHATR. In the Twitter and Math-Overflow data sets, there is only one node type, so the term heterogeneous only refers to the edges. Thus, the Twitter and Math-Overflow data sets are partially heterogeneous graph types as show in Figure 1(f). The EComm and Alibaba data sets have several types of nodes and several types of edges so they deserve to be called heterogeneous graphs. These two data sets also have eleven snapshots, which means that they experience topology changes over certain time intervals. Thus, the Ecomm and Alibaba data sets are dynamic heterogeneous graphs. However, the four types of edges in the EComm data set have several meanings, namely: add-to-favorite, add-to-chart, buy, and click. These four edge types only connect the user entity to the item entity. There is no edge that connects the user entity to the user or connects the item entity to the item, commonly referred to as a homogeneous relationship. Because the four types of edges that exist in the EComm data set only connect user entities to item entities, even though the relationship types vary, the type of graph that is suitable for this condition is a dynamic heterogeneous bipartite graph, like the one shown in Figure 1(j). The same is true for Alibaba's data set.

The DyHATR method uses a hierarchical attention model, which includes node-level attention and edge-level attention to capture static snapshot heterogeneity information. This hierarchical attention model outperformed the metapath2vec model for the Twitter and EComm data sets with two different integration methods. As shown in Figure 3 below, the hierarchical attention model was effective in learning heterogeneous information from each snapshot.

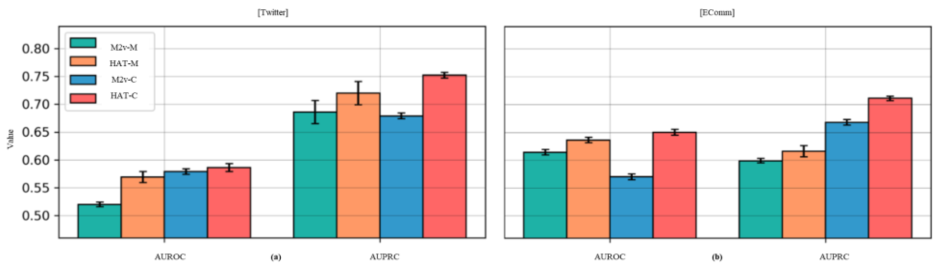


**Figure 3** Comparison results between hierarchical attention (HAT) and meta-path2vec (m2v) models [25].

Another contribution of the DyHATR method is the attentional RNN temporal model. As previously explained, the actual link prediction process for dynamic heterogeneous graphs can only run up to the RNN model, but the addition of this temporal self-attention model provides an improvement of the performance of the DyHATR method, as shown in Figure 4. This shows the superiority of the temporal attentive RNN model in dynamic information modeling. Although the hierarchical attention model and the addition of the temporal self-attention model improve the performance of the DyHATR method, it was not proven that the DyHATR method works well in link prediction on dynamic heterogeneous graphs because the experiments were only carried out on dynamic partially heterogeneous graphs and dynamic bipartite graphs. Therefore, it was necessary to retest the DyHATR method using a data set that represents a dynamic heterogeneous graph.

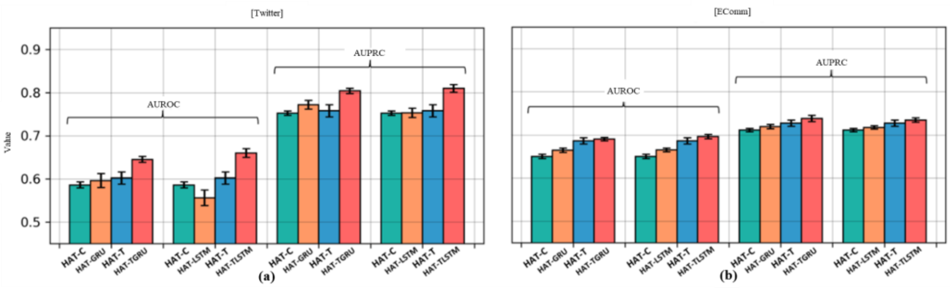


**Figure 4** Results of comparison of different components in DyHATR [25].

## 5 DyHATR Experiments for Dynamic Heterogeneous Graphs

Based on the previous explanation, we conducted an experiment with the Yelp data set as a dynamic heterogeneous graph. To evaluate the performance of DyHATR on this data set, we also considered DHNE and NP-GLM in this experiment. Moreover, the experiment focused on answering the two following research questions:

1. RQ1: Can DyHATR proposed by Xue, *et al.* [25] do link prediction on dynamic heterogeneous graphs?
2. RQ2: Can DyHATR proposed by Xue, *et al.* [25] outperform state-of-the-art methods in link prediction on dynamic heterogeneous graphs?

Before analyzing the link prediction results, the first scenario in this experiment was to find the right parameter values to produce the best accuracy in link prediction. After the best parameter values were found, the results of the link prediction were analyzed by comparing the results with a predetermined baseline. The results compared were the accuracy and the execution time of each method.

### 5.1 Data Sets

To answer RQ1 and RQ2, we used only one publicly available data set, namely the Yelp[1] data set. This data set contains review records from many local business users in America, such as reservations, mechanics, and restaurants or cafes, from 2004 to 2021. The reason for using Yelp data in this study was because Yelp data provides time information for each transaction activity and provides information on the user-to-user relationship and item category similarity. Thus, the Yelp data

[1] https://www.yelp.com/dataset

is considered suitable for a relationship scheme in the form of a dynamic heterogeneous graph.

Users and businesses can be represented as nodes in the graph. Friendships between users, category similarities between businesses, and user reviews of items can be represented as edges in the graph. This condition is in accordance with the type of heterogeneous graph that has homogeneous subgraphs. In addition, the existence of information on the time when the relationship occurs qualifies a graph as dynamic, so that the Yelp data available on the Yelp.com website meets the criteria of a dynamic heterogeneous graph.

Table 2 presents statistics from the Yelp data that were used in the experiment. Six different graphs were used, where each graph is a record of transactions for a week or a month. Each graph had a different number of nodes and edges because it corresponds to the reality of transactions that occurred. One snapshot is one day. Although some of them look like the data sets in Table 1, we can confirm that these types of graphs are different.

**Table 2** Statistics of Yelp data sample.

| Graph | \|V\| | \|E\| | Node Types | Edge Types | Snapshots |
|---|---|---|---|---|---|
| W1 | 4,664 | 6,998 | 2 | 3 | 7 |
| M2 | 18,071 | 25,977 | 2 | 3 | 28 |
| M3 | 24,968 | 37,318 | 2 | 3 | 31 |
| M4 | 21,723 | 28,362 | 2 | 3 | 30 |
| M5 | 27,275 | 39,460 | 2 | 3 | 31 |
| M6 | 24,009 | 33,880 | 2 | 3 | 30 |

## 5.2 Experiment Environment

The task of link prediction on a dynamic heterogeneous graph is to study heterogeneous and dynamic information simultaneously in the previous $G_1, \dots, G_t$ in order to predict new links as well as repeated links in snapshots of $G_{t+1}$. Each predicted edge weight is later classified as connected or not connected. Then it is compared with the ground truth graph to produce AUROC and AUPRC scores as evaluation metrics. DyHATR and both baselines were run five times and the average was recorded.

The DyHATR model and both baselines were tested using Ubuntu Server 20.04.2 LTS with an Intel Core i5-7200 CPU @2.50 GHz, 4 GB RAM and 15 GB hard disk. The DyHATR code was implemented in Tensorflow 1.14 and Python 3.6. In the DyHATR model, we re-evaluated the parameters that affect the performance of the model by conducting an experiment to find out how many heads are suitable for the HAT and TAT models so that they can capture node feature embedding. The final embedding dimension was also searched for the best value. Several optimizers, such as Stochastic Gradient Derivation and Adam optimizer, were tried for use in the DyHATR model to update and optimize parameters. In the link prediction section, we will use the metric evaluation function from the scikit-learn library.

### 5.3 Experiment Baseline

The models explained below were used as benchmarks in the performance comparison.

1. NP-GLM [36] – In addition to learning long-term evolutionary patterns of dynamic networks on top of metapath-based models, NP-GLM uses a recurrent neural network model to predict continuous-time relationships.
2. DHNE [37] – Using both historical and current heterogeneous information, DHNE develops comprehensive historical-current networks based on consecutive snapshots and models evolutionary patterns. Then, DHNE uses dynamic heterogeneous skip-gram modeling and metapath-based random walks to capture representations of nodes.

We compared DyHATR with two approaches, NP-GLM and DHNE. The reason was because these two methods represent dynamic heterogeneous network embedding methods and include the DyHATR baseline. In addition, both of them also provided the source code in their papers. To ensure a fair comparison, we fixed the final embedding size for each baseline at 16. All baselines' hyper-parameters were properly optimized.

### 5.4 Experimental Results

#### 5.4.1 Parameters Sensitivity

With this experiment we wanted to find out the parameter values that can produce the highest AUROC/AUPRC score and the fastest execution time so that the best configuration is obtained. For this reason, a number of runs were carried out by changing several parameter values to observe their effect on the results and the execution time of the link prediction task performed. The observed parameters included:

1. Optimizer – The DyHATR method provides several types of optimizers, namely Adam, Stochastic Gradient Descent (SGD), AdaDelta (Adade), RMSprop (RMSP), and Momentum.
2. Dimensions of the final embedding output – The final embedding size for all baselines was set to the same value for making a fair comparison.
3. Number of heads in hierarchical attention (HAT) – The hierarchical attention model has a multi-head mechanism so that the number of heads could be adjusted.

4. Number of heads in temporal attentive (TAT) – The same as before, there was no specific range for the determination of the number of heads. However, we varied one parameter to check the sensitivity while the other parameters were kept fixed.
5. Temporal learner – The DyHATR method provides two types of temporal learner, namely LSTM and GRU.

Five parameter configurations were observed, and the graph used in this experiment was a W1 graph. These parameters needed to be configured in order to get the right constant value information to support the best results. The results of this functional test were useful for selecting constant parameter values for further testing, namely of link prediction. Some samples of the experimental results can be seen in Table 3.

**Table 3** Sample experimental parameter sensitivity results.

| Optimizer | Dimensions | HAT | TAT | Temporal Learner | epoch | AUROC | AUPRC | Time (h.m.s) |
|---|---|---|---|---|---|---|---|---|
| Adam | 16 | 4 | 4 | LSTM | 10 | **0,7655** | 0,6869 | 0.49.15 |
| SGD | 16 | 4 | 4 | LSTM | 10 | 0,5656 | 0,4298 | 0.58.11 |
| Adade | 16 | 4 | 4 | LSTM | 10 | 0,4898 | 0,4140 | 0.58.56 |
| RMSP | 16 | 4 | 4 | LSTM | 10 | 0,7338 | 0,5758 | 0.59.10 |
| Momentum | 16 | 4 | 4 | LSTM | 10 | Error | Error | - |
| Adam | 16 | 4 | 4 | GRU | 10 | 0,7708 | 0,6670 | 1.11.47 |
| SGD | 16 | 4 | 4 | GRU | 10 | 0,5787 | 0,4331 | 0.48.0 |
| Adade | 16 | 4 | 4 | GRU | 10 | 0,5003 | 0,4678 | 0.47.50 |
| RMSP | 16 | 4 | 4 | GRU | 10 | **0,8152** | 0,7819 | 0.49.36 |
| Momentum | 16 | 4 | 4 | GRU | 10 | Error | Error | - |
| RMSP | 16 | 4 | 4 | GRU | 1 | 0,8232 | 0,8173 | 0.56.21 |
| RMSP | 16 | 4 | 4 | GRU | 4 | **0,8407** | 0,8279 | 0.56.25 |
| RMSP | 16 | 4 | 4 | GRU | 7 | 0,7846 | 0,7871 | 0.57.2 |
| RMSP | 8 | 4 | 4 | GRU | 4 | 0,7674 | 0,7858 | 1.08.20 |
| RMSP | 32 | 4 | 4 | GRU | 4 | 0,7365 | 0,7586 | 1.11.13 |
| RMSP | 64 | 4 | 4 | GRU | 4 | 0,8382 | 0,8263 | 1.14.50 |
| RMSP | 128 | 4 | 4 | GRU | 4 | 0,7124 | 0,7149 | 1.23.47 |
| RMSP | 16 | 2 | 4 | GRU | 4 | 0,7941 | 0,8046 | 0.52.25 |
| RMSP | 16 | 3 | 4 | GRU | 4 | 0,8383 | 0,8285 | 0.54.14 |
| RMSP | 16 | 8 | 4 | GRU | 4 | 0,8226 | 0,7991 | 1.03.34 |
| RMSP | 16 | 4 | 2 | GRU | 4 | 0,8678 | 0,8343 | 0.54.37 |
| RMSP | 16 | 4 | 3 | GRU | 4 | Error | Error | - |
| RMSP | 16 | 4 | 8 | GRU | 4 | 0,8091 | 0,7978 | 0.58.4 |
| **RMSP** | **16** | **3** | **2** | **GRU** | **4** | **0,8700** | **0,8442** | **0.53.15** |
| RMSP | 16 | 3 | 8 | GRU | 4 | 0,8172 | 0,8103 | 0.53.32 |
| RMSP | 16 | 2 | 8 | GRU | 4 | 0,7761 | 0,7826 | 1.04.54 |
| RMSP | 16 | 8 | 2 | GRU | 4 | 0,7458 | 0,7843 | 1.35.50 |

Table 3 shows some samples of the observed parameter configurations. In the observed changes in the type of optimizer, it can be concluded that the use of the RMSprop optimizer resulted in the highest AUROC/AUPRC scores compared to the other optimizers. Then, from the two temporal learners provided, the highest AUROC value was achieved when using the GRU. With respect to the dimension of the final embedding output, when the dimension was equal to 16, DyHATR achieved the highest AUROC value in this experiment. Then the optimal numbers of multi-heads for HAT and TAT were 3 and 2 for graph W1 from the Yelp data set, respectively.

In addition, we also observed the execution time of DyHATR with different configurations of parameters. Regarding the epoch value, when the value was equal to 4, DyHATR achieved the highest AUROC score of 0.87 and also had the fastest execution time, at 53 minutes 15 seconds. This time can be considered fast because it is below the average of the overall successful execution times. Thus, the answer to RQ1 is that the DyHATR method is able to make link prediction on dynamic heterogeneous graphs. However, to assess whether the results obtained by DyHATR are good or not, in the next experiment we compared the results with state-of-the-art methods in link prediction on dynamic heterogeneous graph types.

### 5.4.2 Task of Link Prediction

With this experiment, we wanted to answer RQ2. Before starting, it was necessary to conduct data preprocessing first. The data preprocessing technique carried out was adjusted to the input format for each method. The graphs used in this experiment were M2, M3, M4, M5, and M6. These graphs represent transaction data from different months. During the experiment, each method processed one graph five times to validate the link prediction results. The results recorded in Table 4 are the averages of the five trials carried out. In addition, the parameter configuration used in DyHATR was the best configuration from the experimental parameter sensitivity results. Meanwhile, the parameter configuration for all baselines was specifically optimized for optimal results.

**Table 4** Experimental results for the link prediction task on dynamic heterogeneous graphs.

| Methods | Graphs | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | M2 | | M3 | | M4 | | M5 | | M6 | |
| | AUROC | AUPRC | AUROC | AUPRC | AUROC | AUPRC | AUROC | AUPRC | AUROC | AUPRC |
| NP-GLM | 0,6282 | **0,6250** | 0,5568 | 0,5209 | **0,6343** | **0,6309** | 0,5246 | 0,5112 | 0,5653 | 0,5721 |
| DHNE | 0,6037 | 0,5201 | 0,5424 | 0,5061 | 0,5152 | 0,5544 | 0,5608 | **0,5723** | **0,6336** | 0,6031 |
| DyHATR-TGRU | **0,6304** | 0,6043 | **0,5946** | **0,5384** | 0,5802 | 0,5277 | 0,5870 | 0,5473 | 0,5707 | 0,5217 |
| DyHATR-TLSTM | 0,5746 | 0,5693 | 0,5152 | 0,5092 | 0,6216 | 0,5910 | **0,6253** | 0,5069 | 0,6212 | **0,6415** |

Note: values in bold indicate the highest AUROC/AUPRC scores achieved by the method for each graph data.

The experimental results for the link prediction task are summarized in Table 4. Out of the five graphs tested, DyHATR achieved the best performance for three of them, judging from the AUROC score. When viewed from the AUPRC score, DyHATR achieved the best performance for two graphs out of five graphs. The highest AUROC and AUPRC achieved by DyHATR on graph M3 were 0.5946 and 0.5384, respectively, were significantly higher than the second highest score achieved by NP-GLM (0.5568 for AUROC and 0.5209 for AUPRC). For graphs M2 and M5, the highest AUROC scores achieved by DyHATR were 0.6304 and 0.6253, respectively, but the highest AUPRC scores were achieved by baseline (0.625 by NP-GLM and 0.5723 by DHNE). Then for graph M6, DyHATR only managed to get the first highest AUPRC score with 0.6415 and got the second highest AUROC score after DHNE. The last was graph M4, where DyHATR-TLSTM was below the highest scorer, NP-GLM, which was able to get AUROC and AUPRC scores of 0.6343 and 0.6309, respectively. Note that DyHATR did not always get the best results for every tested graph but the percentage of DyHATR to get the highest score was higher than baseline. This shows that DyHATR is competitive in link prediction on dynamic heterogeneous graphs.

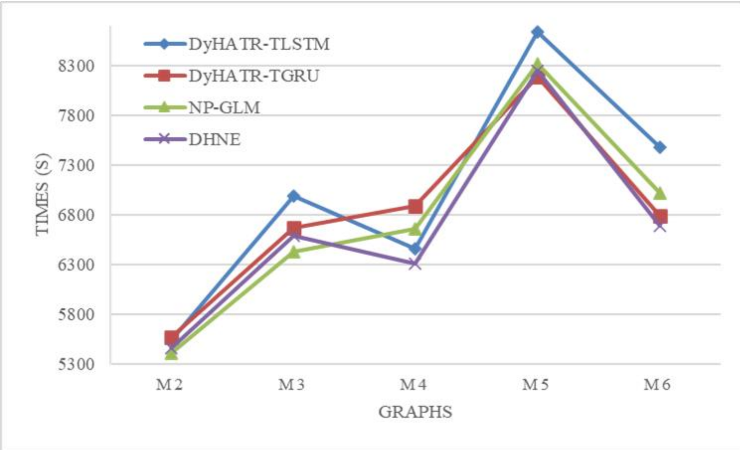


**Figure 5** Average execution time for link prediction.

In addition, we also recorded the time it took for each method to process a graph. Each method processed one graph five times, so the execution time shown in Figure 5 is the average time of the five experiments performed. For graph M3, the average execution times of NP-GLM and DHNE were 6427 s and 6594 s, respectively. For this graph, DyHATR was slightly slower than the baseline method (6675 s for DyHATR-TGRU and 6986 s for DyHATR-TLSTM). However, DyHATR was faster than baseline several times, for example for graphs M4 and M6, respectively. DyHATR-TLSTM and DyHATR-TGRU were slightly faster than NP-GLM. However, the differences were not very significant and most of the DyHATRs were indeed slightly slower than the baseline method due to the large parameter scale in the attention model as well as the RNN model.

## 6 Conclusion

To meet the criteria of a dynamic heterogeneous graph, an existing link prediction method is required to be able to study both evolutionary and heterogeneous information simultaneously. DyHATR is one of the newest network embedding methods and is claimed to be able to perform link prediction on dynamic heterogeneous graphs by capturing heterogeneous information and evolutionary patterns simultaneously, but this was not proven in experiments. Our research showed that the DyHATR method can indeed be applied to perform link prediction on dynamic heterogeneous graphs. Although the AUROC and AUPRC scores were somewhat lower than those of other methods, the execution times of DyHATR were faster. To conclude, DyHATR is a promising method for link prediction on dynamic heterogeneous graphs. We suggest that future work for DyHATR should focus on modifying the structure of the RNN model in order to improve its performance.

## References

[1] Yang, K. & Toni, L., *Graph-Based Recommendation System*, 2018 IEEE Global Conference on Signal and Information Processing, GlobalSIP 2018 - Proceedings, pp. 798-802, Feb. 2019. DOI: 10.1109/GLOBALSIP.2018.8646359.

[2] Salamat, A., Luo, X. & Jafari, A., *HeteroGraphRec: A Heterogeneous Graph-Based Neural Networks for Social Recommendations*, Knowl Based Syst, **217**, 106817, Apr. 2021. DOI: 10.1016/j.knosys.2021.106817.

[3] Sun, Y., Barber, R., Gupta, M., Aggarwal, C. C. & Han, J., *Co-Author Relationship Prediction in Heterogeneous Bibliographic Networks*, in Proceedings – 2011 International Conference on Advances in Social Networks Analysis and Mining, ASONAM 2011, pp. 12-128, 2011. DOI: 10.1109/ASONAM.2011.112.

[4] Amin, M.I. & Murase, K., *Link Prediction in Scientists Collaboration with Author Name and Affiliation*, in Proceedings - 2016 Joint 8th International Conference on Soft Computing and Intelligent Systems and 2016 17th International Symposium on Advanced Intelligent Systems, SCIS-ISIS 2016, Institute of Electrical and Electronics Engineers Inc., pp. 233-238, Dec. 2016. DOI: 10.1109/SCIS-ISIS.2016.0058.

[5] Ketkar, N.S., Holder, L.B. & Cook, D.J., *Mining in the Proximity of Subgraphs*, 2006.

[6] Feng, Y., *Homogeneous and Heterogeneous Relational Graph for Visible-infrared Person Re-identification*, ArXiv, Sep. 2021. DOI: 10.48550/arxiv.2109.08811.

[7] Shahreza, M.L., Ghadiri, N., Mousavi, S.R., Varshosaz, J. & Green, J.R., *Heter-LP: A Heterogeneous Label Propagation Algorithm and Its*

*Application in Drug Repositioning*, J Biomed Inform, **68**, pp. 167-183, Apr. 2017. DOI: 10.1016/j.jbi.2017.03.006.
[8] Xiong, F., Wang, X., Pan, S., Yang, H., Wang, H. & Zhang, C., *Social Recommendation with Evolutionary Opinion Dynamics*, IEEE Trans Syst Man Cybern Syst, **50**(10), pp. 3804–3816, Oct. 2020. DOI: 10.1109/TSMC.2018.2854000.
[9] Rafailidis, D. & Nanopoulos, A., *Modeling the Dynamics of User Preferences in Coupled Tensor Factorization*, in RecSys 2014- Proceedings of the 8th ACM Conference on Recommender Systems, Association for Computing Machinery, Inc, pp. 321-324, Oct. 2014. DOI: 10.1145/2645710.2645758.
[10] Yang, J., Mcauley, J. & Leskovec, J., *Community Detection in Networks with Node Attributes*, in IEEE 13th International Conference on Data Mining, IEEE, pp. 1151-1156, 2013.
[11] Bilgic, M., Namata, G.M. & Getoor, L., *Combining Collective Classification and Link Prediction*, in in Workshop on Mining Graphs and Complex Structures at the IEEE International Conference on Data Mining, 2007.
[12] Rossi, R.G., Lopes, A.D.A. & Rezende, S.O., *Optimization and Label Propagation in Bipartite Heterogeneous Networks to Improve Transductive Classification of Texts*, Inf Process Manag, **52**(2), pp. 217-257, Mar. 2016. DOI: 10.1016/j.ipm.2015.07.004.
[13] Himmelstein, D.S. & Baranzini, S.E., *Heterogeneous Network Edge Prediction: A Data Integration Approach to Prioritize Disease-Associated Genes*, PLoS Comput Biol, **11**(7), e1004259, Jul. 2015. DOI: 10.1371/journal.pcbi.1004259.
[14] Wang, Z., Liang, J., Li, R. & Qian, Y., *An Approach to Cold-Start Link Prediction: Establishing Connections between Non-Topological and Topological Information*, IEEE Trans Knowl Data Eng., **28**(11), pp. 2857-2870, 2016. DOI: 10.1109/TKDE.2016.2597823.
[15] Sawhney, K., Prasetio, M.C. & Paul, S., *Community Detection Using Graph Structure and Semantic Understanding of Text*, 2017.
[16] Ning, H., Dhelim, S. & Aung, N., *PersoNet: Friend Recommendation System Based on Big-Five Personality Traits and Hybrid Filtering*, IEEE Trans Comput Soc Syst, **6**(3), pp. 394-402, Jun. 2019. DOI: 10.1109/TCSS.2019.2903857.
[17] Berkani, L., *A Semantic and Social-based Collaborative Recommendation of Friends in Social Networks*, Softw Pract Exp, **50**(8), pp. 1498-1519, Aug. 2020. DOI: 10.1002/spe.2828.
[18] Sharma, P.K., Rathore, S. & Park, J.H., *Multilevel Learning Based Modeling for Link Prediction and Users' Consumption Preference in Online Social Networks*, Future Generation Computer Systems, **93**, pp. 952–961, Apr. 2019. DOI: 10.1016/j.future.2017.08.031.

[19] Natarajan, S., Vairavasundaram, S., Natarajan, S. & Gandomi, A.H., *Resolving Data Sparsity and Cold Start Problem in Collaborative Filtering Recommender System Using Linked Open Data*, Expert Syst Appl, **149**, 113248, Jul. 2020. DOI: 10.1016/J.ESWA.2020.113248.

[20] Liu, G., *An Ecommerce Recommendation Algorithm Based on Link Prediction*, Alexandria Engineering Journal, **61**(1), pp. 905-910, Jan. 2022. DOI: 10.1016/J.AEJ.2021.04.081.

[21] Hwang, T. & Kuang, R., *A Heterogeneous Label Propagation Algorithm for Disease Gene Discovery*, in 10th SIAM International Conference on Data Mining, SDM, pp. 583-594, 2010. DOI: 10.1137/1.9781611972801.51.

[22] Chen, X., Liu, M.X. & Yan, G.Y., *Drug-Target Interaction Prediction by Random Walk on the Heterogeneous Network*, Mol Biosyst, **8**(7), pp. 1970-1978, Jul. 2012. DOI: 10.1039/c2mb00002d.

[23] Yan, X.Y., Zhang, S.W. & Zhang, S.Y., *Prediction of Drug-Target Interaction by Label Propagation with Mutual Interaction Information Derived from Heterogeneous Network*, Mol Biosyst, **12**(2), pp. 520-531, Jan. 2016. DOI: 10.1039/c5mb00615e.

[24] Xu, X., *Distributed Temporal Link Prediction Algorithm Based on Label Propagation*, Future Generation Computer Systems, **93**, pp. 627-636, Apr. 2019. DOI: 10.1016/j.future.2018.10.056.

[25] Xue, H., Yang, L., Jiang, W., Wei, Y., Hu, Y. & Lin, Y., *Modeling Dynamic Heterogeneous Network for Link Prediction Using Hierarchical Attention with Temporal RNN*, Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), **12457** LNAI, pp. 282-298, 2021. DOI: 10.1007/978-3-030-67658-2_17/COVER.

[26] Martínez, V., Berzal, F. & Cubero, J.C., *A Survey of Link Prediction in Complex Networks*, ACM Comput Surv, **49**(4), pp. 1-33, Dec. 2016. DOI: 10.1145/3012704.

[27] Harary, F. & Gupta, G., *Dynamic Graph Models*, Math Comput Model, **25**(7), pp. 79-87, Apr. 1997. DOI: 10.1016/S0895-7177(97)00050-2.

[28] Maleki, N., Padmanabhan, B. & Dutta, K., *Representing Social Networks as Dynamic Heterogeneous Graphs*, in IEEE International Conference on Data Mining Workshops, ICDMW, IEEE Computer Society, pp. 714-723, 2022. DOI: 10.1109/ICDMW58026.2022.00098.

[29] Shi, C., Wang, X. & Yu, P.S., *Dynamic Heterogeneous Graph Representation*, In: Heterogeneous Graph Representation Learning and Applications. Artificial Intelligence: Foundations, Theory, and Algorithms., pp. 107-143, 2022. DOI: 10.1007/978-981-16-6166-2_5.

[30] Xing, Z., Song, R., Teng, Y. & Xu, H., *Dynhen: A Heterogeneous Network Model for Dynamic Bipartite Graph Representation Learning*,

Neurocomputing, **508**, pp. 47-57, Oct. 2022. DOI: 10.1016/J.NEUCOM.2022.08.050.

[31] Daud, N.N., Ab Hamid, S.H., Saadoon, M., Sahran, F. & Anuar, N. B., *Applications of Link Prediction in Social Networks: A Review*, Journal of Network and Computer Applications, **166**, Academic Press, 102716, Sep. 15, 2020. DOI: 10.1016/j.jnca.2020.102716.

[32] Nguyen, G.H., Lee, J.B., Rossi, R.A., Ahmed, N.K., Koh, E. & Kim, S., *Continuous-Time Dynamic Network Embeddings*, The Web Conference 2018 - Companion of the World Wide Web Conference, WWW 2018, pp. 969-976, Apr. 2018. DOI: 10.1145/3184558.3191526.

[33] Shi, C., Hu, B., Zhao, W.X. & Yu, P.S., *Heterogeneous Information Network Embedding for Recommendation*, IEEE Trans Knowl Data Eng, **31**(2), pp. 357-370, Feb. 2019. DOI: 10.1109/TKDE.2018.2833443.

[34] Fard, A.M., Bagheri, E. & Wang, K., *Relationship Prediction in Dynamic Heterogeneous Information Network*s, in Azzopardi, L., Stein, B., Fuhr, N., Mayr, P., Hauff, C., Hiemstra, D. (eds) *Advances in Information Retrieval*, ECIR 2019, Lecture Notes in Computer Science, **11437**, Springer, Cham., 2019. DOI: 10.1007/978-3-030-15712-8_2

[35] Bian, R., Dobbie, G., Koh, Y.S. & Divoli, A., *Network Embedding and Change Modeling in Dynamic Heterogeneous Networks*, SIGIR 2019 - Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 861-864, Jul. 2019. DOI: 10.1145/3331184.3331273.

[36] Sajadmanesh, S., Bazargani, S., Zhang, J. & Rabiee, H.R., *Continuous-Time Relationship Prediction in Dynamic Heterogeneous Information Networks*, ACM Trans Knowl Discov Data, **13**(4), 44, 2019. DOI: 10.1145/3333028.

[37] Yin, Y., Ji, L.X., Zhang, J.-P. & Pei, Y.-L., *DHNE: Network Representation Learning Method for Dynamic Heterogeneous Networks*, IEEE Access, **7**, pp. 134782-134792, 2019. DOI: 10.1109/ACCESS.2019.2942221.

[38] Kong, C., Li, H., Zhang, L., Zhu, H. & Liu, T., *Link Prediction on Dynamic Heterogeneous Information Networks*, Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), **11917** LNCS, pp. 339-350, Nov. 2019. DOI: 10.1007/978-3-030-34980-6_36.

[39] Sankar, A., Wu, Y., Gou, L., Zhang, W. & Yang, H., *Dynamic Graph Representation Learning via Self-Attention Networks*, in Workshop on Representation Learning on Graphs and Manifolds in ICLR, 2019.

[40] Goyal, P., Chhetri, S. R. & Canedo, A., *Dyngraph2vec: Capturing Network Dynamics Using Dynamic Graph Representation Learning*, Knowl Based Syst, **187**, 104816, Jan. 2020. DOI: 10.1016/J.KNOSYS.2019.06.024.