

Article

A Link Prediction Algorithm Based on Layer Attention Mechanism for Multiplex Networks

Mingzhou Yang^{1,2,*} and Yongqi He¹¹ School of Information Science and Engineering, Shenyang University of Technology, Shenyang 110870, China; hyq@smail.sut.edu.cn² Shenyang Key Laboratory of Advanced Computing and Application Innovation, Shenyang 110870, China

* Correspondence: yangmz@sut.edu.cn

Abstract

Link prediction is a technique for predicting future or missing relationships between entities based on current network information. Recent studies on link prediction in multiplex networks have improved prediction accuracy by considering inter-layer similarity, but they do not fully exploit the distinct topological information present in each individual network layer. Therefore, this paper proposes a link prediction algorithm based on a layer attention mechanism for multiplex networks, called LATGCN, which simultaneously considers both inter-layer and intra-layer information. Firstly, it uses two-layer GCN to capture the intra-layer embedding representation of a multiplex network. Secondly, it employs the layer attention mechanism to find the layer importance score of each layer of the multiplex network, and utilizes the layer importance scores to weight the intra-layer information according to the difference in importance. Thirdly, we design an embedding representation fusion module to integrate the intra-layer embedding representation and global embedding representation. Finally, experimental results on four real-world social networks and two biological networks show that LATGCN outperforms all compared methods across four evaluation metrics, demonstrating its effectiveness for link prediction.

Keywords: multiplex network; graph convolutional network; graph embedding; link prediction**MSC:** 68T07

Academic Editor: Francisco Chiclana

Received: 30 August 2025

Revised: 1 October 2025

Accepted: 26 November 2025

Published: 27 November 2025

Citation: Yang, M.; He, Y. A Link Prediction Algorithm Based on Layer Attention Mechanism for Multiplex Networks. *Mathematics* **2025**, *13*, 3803. <https://doi.org/10.3390/math13233803>

Copyright: © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Link prediction aims to exploit the underlying topological structure of a network to infer potential or missing relationships between nodes [1]. With the rapid development of information, many social software programs have emerged [2]. In many real-world networks, multiple types of relationships can exist simultaneously between the same pair of nodes. For example, person A and person B may be friends on both WeChat and Weibo, while having no connection on Twitter [3]. Such networks, consisting of multiple single-layer networks, are referred to as multiplex networks, where each layer shares the same set of nodes but differs in edge distribution. There are many applications of link prediction in multiplex networks. In social networks, link prediction can be used to infer unknown links based on existing edges, thereby recommending potential friends [4]. In protein interaction networks, it can predict interactions between proteins [5]. Similarly, in traffic networks, link prediction can help forecast traffic trends [6].

In recent years, the research on link prediction for multiplex networks has received much attention and many methods have been gradually proposed [7]. The research mainly

relies on three frameworks, i.e., a framework based on inter-layer similarity and node similarity, framework based on network representation learning, and hybrid framework based on inter-layer similarity and network representation learning. However, multiplex network link prediction becomes more challenging when considering the complex inter-layer relationships and topologies of multiplex networks. To more accurately predict the relationships in multiplex networks, some methods [8,9] were proposed by considering the complex inter-layer relationships and topologies. However, they failed to fully utilize the rich inter-layer relationships in multiplex networks, which resulted in the differences in the contribution of information from other layers to the target layer and poor model performance. Recent hybrid methods, such as the hybrid framework LISCNE [10], which belongs to network representation learning, aim to extract more comprehensive structural information by incorporating the importance of inter-layer information. LISCNE assigns higher weights to layers more similar to the target layer and lower weights to less similar layers. However, the method still has limitations. It simply obtained the common vector by aggregating the graphs of all layers. When edge distributions differ significantly across layers, this aggregation method struggles to capture such differences.

To accurately extract the embedding representation for each layer of the network and determine inter-layer similarity, we propose a link prediction algorithm based on layer attention mechanism for multiplex networks, called LATGCN, which simultaneously addresses the above limitations and enhances the accuracy of link prediction. Firstly, an embedding representation extraction module is proposed to efficiently generate low-dimensional node embeddings, which are capable of effectively capturing the topological structure information of each layer in the multiplex network. Secondly, to improve the accuracy of link prediction, a layer attention mechanism module is designed, enabling the model to more effectively exploit inter-layer relationships and enhance the comprehensiveness of the embedding representations. Thirdly, we design an embedding representation fusion module which integrates the intra-layer embedding representations and global embedding representation to obtain comprehensive and high-quality multiplex network topology information. Finally, the comprehensive experiments are conducted on six publicly available realistic network datasets. Our method is compared to representative and state-of-the-art methods on four metrics. The experimental results confirm the superiority of LATGCN.

The rest of the paper is organized as follows. Section 2 briefly describes related work. Section 3 first describes the multiplex network link prediction problem and then describes our proposed model in detail. Experimental results and analysis are shown in Section 4. Section 5 summarizes the paper and outlines our future work.

2. Related Work

In this section, we review existing work related to link prediction, including methods for single-layer networks as well as approaches specifically designed for multiplex networks.

2.1. Link Prediction Methods for Single-Layer Networks

Many researchers have tackled the problem of link prediction in single-layer networks by considering the similarity between nodes. Common Neighbors (CN) [11] was the classical similarity metric which considered the number of common neighbors of two nodes. It considered that the more common neighbors there were, the higher the similarity between two nodes. Since then, many new metrics have been proposed inspired by CN, such as Salton [12], Sorenson [13], and HPI [14]. Adamic-Adar (AA) [15] differentiated the contributions of different neighbors by assigning weights. Based on AA, Zhou

et al. [16] proposed Resource Allocation, which assumes that common neighbors with lower degree contribute more to node similarity. Lü et al. [17] further proposed Local Path, which considered local path information and was based on the idea that two nodes were more likely to form a link if multiple shorter paths existed between them. Aziz [18] integrated information on common neighbors, node degrees, and the distribution of edges connected to the target node. However, the above methods only considered the local information of the network and neglected the global topology. To address this limitation, subsequent researchers proposed Katz [19], which considered the global path information. The Katz metric measured the contribution of all possible paths between two nodes, assigning higher weights to shorter paths. However, its computational complexity was high, making it unsuitable for large-scale networks.

Some other researchers have approached link prediction in single-layer networks by considering node representation learning. DeepWalk [20], also known as a graph embedding algorithm, was inspired by Word2vec [21] in natural language processing. DeepWalk treated random walk sequences in a network as sentences and nodes as words, thereby learning node representations in a continuous vector space analogous to word embeddings. Subsequently, Grover et al. [22] extended DeepWalk by introducing biased random walk strategies that incorporated both depth-first and breadth-first sampling mechanisms, leading to the development of Node2vec. Methods such as GraRep [23] and SDNE [24], which preserve higher-order neighborhood information, were then proposed. The convolution operator in Convolutional Neural Network [25] inspired the development of Graph Neural Network [26]. Researchers extended the convolutional operator to network representation learning and proposed Graph Convolutional Network (GCN) [27]. Graph Generative Adversarial Network [28] combined generative and discriminative models with network representations to learn low-dimensional vector representations of nodes. To address the challenge of learning embeddings for only some nodes in large-scale networks, SepNE [29] was proposed, which independently learned the representations of different subsets of nodes during network separation. Yao et al. [30] proposed the MCAS, which introduced multi-scale subgraphs as input graphs. It obtained complementary network structures from different perspectives and used contrastive learning to achieve information balance among multi-scale subgraphs. Su et al. [31] proposed the VGNN DP, a new approach to solve the Gaussian prior problem. This method combined variational graph auto-encoder with diffusion models to advance the learning of more pliable prior distributions.

2.2. Link Prediction Methods for Multiplex Networks

In terms of link prediction in multiplex networks, Gallo et al. [32] proposed MultiSAGE, a generalization of GraphSAGE [33] that enables embeddings in multiplex networks and effectively integrates transitions between intra-layer and inter-layer edges. Nasiri et al. [34] proposed a local random walk based on an extended version of the pure random walk, called multiplex local random walk, to address the link prediction problem in multiplex networks. Bai et al. [35] regarded link prediction in multiplex networks as a multi-attribute decision-making problem, where potential links in the target layer were considered alternatives and the layers as attributes. The similarity of potential links in each layer was treated as an attribute value. The alternatives were ranked using the TOPSIS method, and the attributes were weighted by inter-layer correlation, achieving good prediction performance. Najari et al. [8] introduced a link prediction method based on inter-layer similarity for multiplex networks and proposed an inter-layer similarity metric that fused both inter-layer and intra-layer node features, thereby improving prediction accuracy. Shan et al. [36] proposed a supervised link prediction method in multiplex networks, which formulated link prediction as a binary classification problem. LISCNE [10] exploited both

common and local features of multiplex networks while leveraging inter-layer similarity. Gao et al. [9] proposed a new embedding method called ILAR to capture and fully utilize multiplex networks with multiple relationships and complex layer correlations. ILAR employed two convolutional modules to capture sufficient complementarities and correlations in multiplex networks. Wang et al. [37] introduced the LPGRI method, which enhanced link prediction in multiplex networks by integrating intra-layer and inter-layer information, where the contribution of inter-layer information was quantified through a global relevance index.

Despite the remarkable effectiveness of previous link prediction methods for multiplex networks, their shortcomings still deserve our attention. In general, they do not sufficiently exploit the inter-layer relationships. As a result of the above elucidation, we are motivated to adopt the following method to better understand and explore the topology and potential links in multiplex networks. In comparison with existing models, the proposed LATGCN is capable of comprehensively exploiting both intra-layer and inter-layer information within multiplex networks.

3. Methodology

In this section, we first describe link prediction problem for multiplex networks and then provide a detailed overview of our model architecture, including explanations of its individual modules.

3.1. Problem Description of Link Prediction for Multiplex Networks

A multiplex network which consists of N nodes and k single-layer networks is defined as $MG = (G^1, G^2, \dots, G^k)$, where $0 < k \leq K$, and K denotes the total number of layers in the multiplex network. $G^k = (V, \xi^k)$ denotes the k th layer network, where $V = (v^1, v^2, \dots, v^N)$ and $E = (\xi^1, \xi^2, \dots, \xi^k)$ represent the set of N nodes and the set of edges in all layers respectively. The topology of this layer can be represented by an adjacency matrix A^k . The embedding representation of node $v_i \in V$ in each relation is defined as $\mathbf{h}_i \in \mathbb{R}^{K \times d_1}$ ($d_1 \ll N$). The global embedding representation of the node in all relations is defined as $\mathbf{g}_i \in \mathbb{R}^{d_2}$ ($d_2 \ll N$). The edge probability function is defined as $p(v_i^k, v_j^k) \in [0, 1]$. The LATGCN model we propose is specifically designed to predict intra-layer links within a multiplex network. In particular, it leverages information from other layers to predict the links within the target layer G^k .

3.2. The Proposed Model

The proposed LATGCN aims to capture the features of target nodes of a multiplex network carrying information from each layer. The architecture of the LATGCN model is shown in Figure 1, which consists of three major components, i.e., embedding representation extraction module, layer attention mechanism module, and embedding fusion module. Firstly, the embedding representation extraction module is designed to embed the topology information of the multiplex network into the node features. Secondly, the generated feature representations are fed into the layer attention mechanism module to obtain the layer importance scores. Thirdly, the extracted features and layer importance scores are processed by the embedding fusion module to obtain the final embedding representations. Fourthly, the decoder is used to generate the edge vectors and then the activation function is used to get the probabilities of the existence of edges. Finally, a loss function is used to supervise the entire training process of the model. In the following, we provide a detailed description of the LATGCN model.

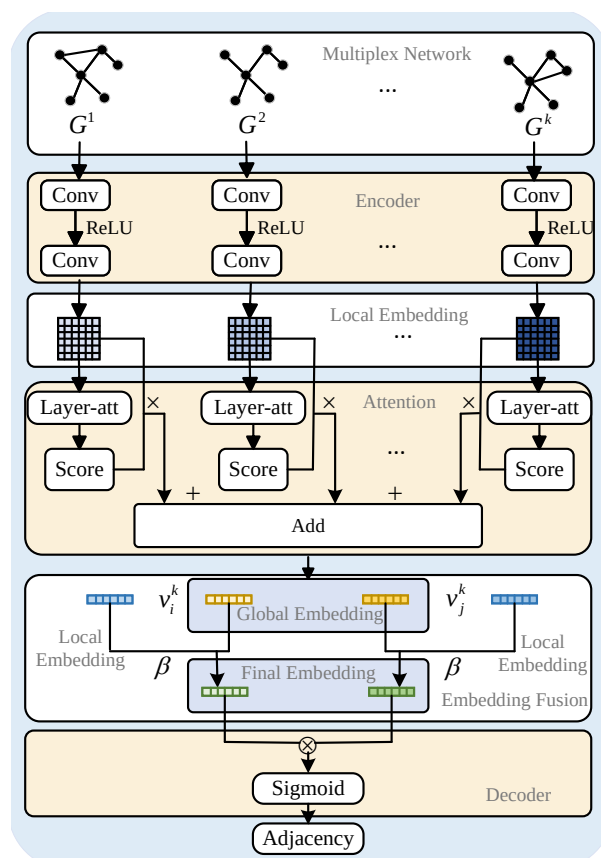


Figure 1. Overall framework of the proposed LATGCN model. (1) Input: a multiplex network; (2) embedding representation extraction module: learns node embeddings in each layer through two-layer GCN; (3) layer attention mechanism module: assigns higher weights to layers with greater impact on neighboring nodes; (4) embedding representation fusion module: fuses local and global embedding representations.

3.2.1. Embedding Representation Extraction Module

The embedding representation extraction module aims to efficiently extract the node features of the relevant nodes of a multiplex network. As shown in Figure 2, LATGCN uses the output of Node2vec as the input to a two-layer GCN and obtains the embedding representations of the relevant nodes.

A multiplex network is defined as $MG = (G^1, G^2, \dots, G^k)$, where $G^k = (V, \xi^k)$ denotes the k th layer ($0 < k \leq K$). We represent the topology of each layer using an adjacency matrix $\mathbf{A}^k$, and use the adjacency matrices along with the initial features of the multiplex network as inputs to the GCN. Then, the task of embedding representation extraction for the k th-layer network is defined as:

$$\mathbf{H}_k^{l+1} = \sigma \left(\tilde{\mathbf{D}}^{k-\frac{1}{2}} \tilde{\mathbf{A}}^k \tilde{\mathbf{D}}^{k-\frac{1}{2}} \mathbf{H}_k^{(l)} \mathbf{W}_k^{(l)} \right), \quad (1)$$

where $\tilde{\mathbf{A}}^k = \mathbf{A}^k + \mathbf{I}_N^k$ is the self-looped adjacency matrix of the k th-layer network G^k . $\mathbf{I}_N$ denotes the identity matrix, and $\tilde{\mathbf{D}}_{ii}^k = \sum_j \tilde{\mathbf{A}}_{ij}^k$ denotes the diagonal matrix. $\mathbf{W}_k^{(l)}$ represents the trainable weight matrix at layer l for the k th-layer network. $\sigma(\cdot)$ represents the activation function which uses ReLU($\cdot$) in this paper. The input feature matrix at layer l is denoted as $\mathbf{H}_k^{(l)} = R^{N \times d_0}$, where N is the number of nodes and d_0 is the dimension of the initial embeddings. For the initial input $\mathbf{H}_k^{(0)}$, we use the output of Node2vec.

Next, the overall processing equation of the multiplex network for the embedding representation extraction module is defined as:

$$\mathbf{H}_1^{l+1}, \mathbf{H}_2^{l+1}, \dots, \mathbf{H}_k^{l+1} = f(\mathbf{A}^1, \mathbf{H}_1^l; \mathbf{A}^2, \mathbf{H}_2^l; \dots; \mathbf{A}^k, \mathbf{H}_k^l), \quad (2)$$

where $\mathbf{H}_k^{l+1}$ ($0 < k \leq K$) is the output of the k th-layer network after applying GCN. $\mathbf{H}_k^l$ represents the set of node embedding vectors learned by Node2vec on the k th-layer. $\mathbf{A}^k$ is the adjacency matrix of the k th-layer multiplex network.

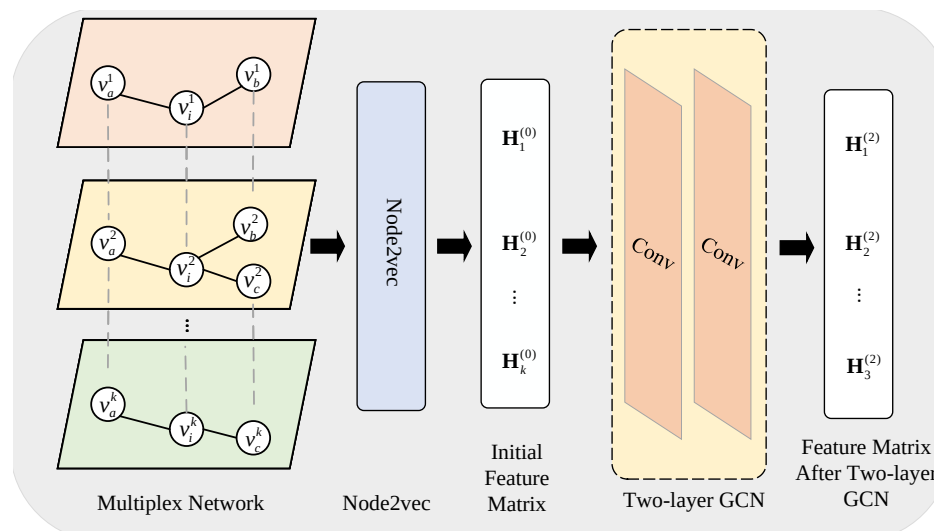


Figure 2. Embedding extraction process in the LATGCN model. (1) Node2vec is used to extract the initial feature matrix for the multiplex network; (2) the initial feature matrix is then processed by a two-layer GCN.

Inspired by GAT [38], this paper employs the layer attention mechanism to evaluate the contribution of nodes from other layers to the nodes in the target layer, after obtaining the node embedding representations by using the GCN model. This is achieved by analyzing the relationships among their neighbors across layers in the network. The mechanism assigns importance scores to the nodes in each layer, which indicates the importance of the nodes in other layers with respect to the nodes in the target layer, and accordingly adjusts the embedding matrix of each layer to ensure that information from important layers has a more significant impact on the global embedding representation.

3.2.2. Layer Attention Mechanism Module

In this paper, to capture the differences in information across layers relative to the target layer, LATGCN considers the influence of neighboring nodes on the target node. Specifically, it aggregates the effects of first-order neighbors, normalizes the per-layer influence, and assigns higher weights to layers where the neighboring nodes have greater impact. The schematic of the layer attention mechanism is shown in Figure 3.

The input to the layer attention mechanism is defined as $\mathbf{h}^k = \{\mathbf{h}_1^k, \mathbf{h}_2^k, \dots, \mathbf{h}_N^k\}$, where $\mathbf{h}_i^k \in \mathbb{R}^F$ is a set of node features and N is the number of nodes. F is the number of each node's features. To obtain sufficient expressive power, we need to transform the input features into higher level features, which requires at least one learnable linear transformation. For this purpose, a shared linear transformation $\mathbf{W}^k \in \mathbb{R}^{F' \times F}$ parameterized by a trainable weight matrix is applied to each node to perform self-attention, i.e., a kind of shared attention mechanism $a: \mathbb{R}^{F'} \times \mathbb{R}^F \rightarrow \mathbb{R}$. Thus, the importance of the feature of node v_j to node v_i is defined as:

$$e_{ij}^k = a(\mathbf{W}^k \mathbf{h}_i^k, \mathbf{W}^k \mathbf{h}_j^k), \quad (3)$$

where e_{ij}^k is the attention coefficient of node v_j to node v_i .

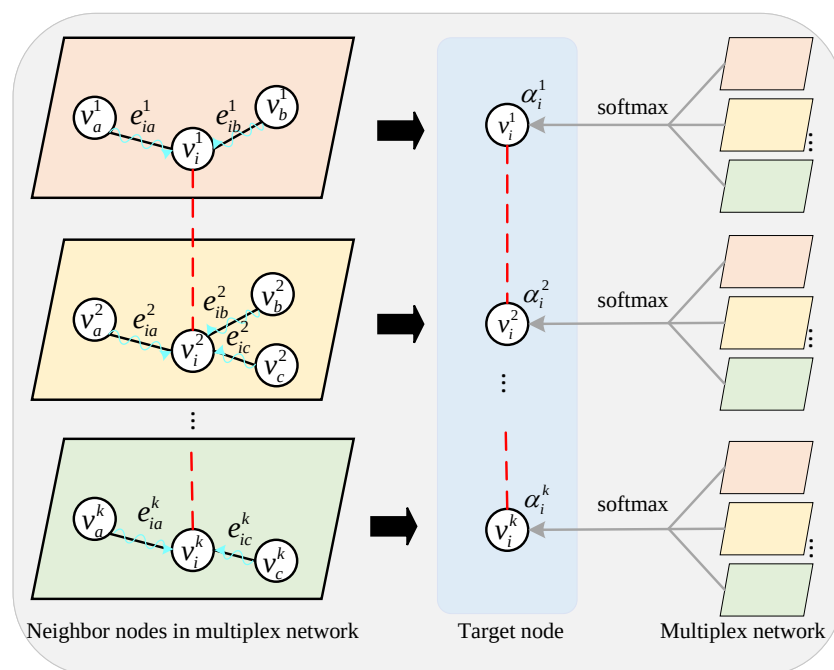


Figure 3. Layer attention mechanism in LATGCN. (1) It aggregates the effects of first-order neighbors; (2) it assigns higher weights to layers where the neighboring nodes have greater impact.

In this paper, the layer attention mechanism is a single-layer feed-forward neural network parameterized by the weight vector $\mathbf{a}^k \in \mathbb{R}^{2F'}$, and activated by LeakyReLU. After full expansion, the importance of the feature of node v_j to node v_i can be expressed as:

$$e_{ij}^k = \text{LeakyReLU}\left((\mathbf{a}^k)^\top (\mathbf{W}^k \mathbf{h}_i^k \parallel \mathbf{h}_j^k)\right), \quad (4)$$

where $\top$ denotes the transposition and $\parallel$ denotes the concatenation operation.

In order to make the layer importance scores more easily comparable across layers, we normalize the choice of node v_j for each layer of the multiplex network. The layer importance score is defined as:

$$\alpha_i^k = \frac{\sum_{j \in N_i} \exp(e_{ij}^k)}{\sum_{p=1}^P \sum_{m \in N_i} \exp(e_{im}^p)}, \quad (5)$$

where α_i^k is the layer importance score of node v_i at the k th-layer ($0 < k \leq K, 0 < \alpha_i^k < 1$) and N_i is the first order neighbor of the node v_i .

After calculating the layer importance score for each layer of node v_i , LATGCN weights and fuses the layer importance scores with the node embedding representations, i.e., local embedding representations of the corresponding layers to obtain the global embedding representation $\mathbf{g}_i$.

In order to more accurately capture the complex relationship in the multiplex network, the global embedding is defined as:

$$\mathbf{g}_i = \alpha_i^1 \mathbf{h}_i^1 + \alpha_i^2 \mathbf{h}_i^2 + \dots + \alpha_i^K \mathbf{h}_i^K, \quad (6)$$

where $\mathbf{g}_i$ is the global embedding representation of node v_i . $\mathbf{h}_i = \{\mathbf{h}_i^1, \mathbf{h}_i^2, \dots, \mathbf{h}_i^K\}$ ($\mathbf{h}_i \in \mathbb{R}^F$) is the embedding representation of node v_i in each layer and K is the number of layers of the multiplex network.

3.2.3. Embedding Representation Fusion Module

In multiplex networks, node relationships within the target layer and node relationships in other layers are considered. The local embedding representation focuses on the node relationships and structural features of the target layer, while the global embedding representation integrates the information in the multiplex network. Therefore, the final embedding representation must be computed based on the local embedding representation and the global embedding representation. Then, an embedding representation fusion module is proposed, which introduces a learnable parameter to weight the local and the global embedding representation, thereby producing the final node embedding in the target layer. The parameter β is automatically adjusted during the training process to optimally balance the contributions of local information and global information. The final embedding representation is defined as:

$$\mathbf{H}_i^k = \beta \mathbf{h}_i^k + (1 - \beta) \mathbf{g}_i, \quad (7)$$

where $\mathbf{h}_i^k$ denotes the local embedding representation of node v_i at the target layer k ($0 < k \leq K$). $\mathbf{H}_i^k$ is the final embedding representation of the multiplex network at the target layer k for node v_i . The parameter β ($0 < \beta < 1$) is trainable and is initialized using a uniform distribution.

Finally, the obtained final embedding representations of the target layer nodes v_i and v_j are used as inputs to the decoder. We perform a Hadamard accumulation operation on the final embedding representations $\mathbf{H}_i^k$ and $\mathbf{H}_j^k$ to obtain the edge vector $\mathbf{H}_{ij}^k$, as shown in Equation (8). The sigmoid activation function is then used to obtain the probability p that the link exists, as shown in Equation (9).

$$\mathbf{H}_{ij}^k = \mathbf{H}_i^k \odot \mathbf{H}_j^k. \quad (8)$$

$$p = \text{Sigmoid}(\mathbf{H}_{ij}^k) = \frac{1}{1 + \exp(-\mathbf{H}_{ij}^k)}. \quad (9)$$

3.2.4. Model Optimization

Our goal is to predict the adjacency matrix of the target layer k based on the existing adjacency matrix of the multiplex network. We set the edge labels present in the target layer to 1 and the edge labels absent to 0. Supervised learning is carried out and the BCEWithLogitLoss is used to measure the difference between the true value and the predicted value. The loss function and the activation function are respectively shown in Equations (10) and (11).

$$L(s, y) = -[y \log(\sigma(s)) + (1 - y) \log(1 - \sigma(s))], \quad (10)$$

$$\sigma(s) = \frac{1}{1 + e^{-s}}, \quad (11)$$

where s is the logit score predicted by the model and y is the true label (0 or 1). $\sigma(\cdot)$ is the activation function that converts the logit to a probability value.

3.3. Complexity Analysis

The inputs to the proposed model LATGCN are the multiplex network, the initial features, the maximum number of iterations, and the patience for early stopping. The model consists of two layers of GCN, a layer attention mechanism, and an embedding fusion module. In the final stage, a decoder followed by an activation function is applied to estimate the probability of link existence. Its output is the predicted adjacency matrix of the target layer.

For ease of analysis, we clarify the meaning of the symbols here. We define N , E_{max} , F_0 , F_1 , F_2 , F_3 , K , M , and J as the number of nodes of the multiplex network, the maximum value of the number of edges in each layer of the multiplex network, the initial feature dimensionality, the first layer GCN output dimensionality, the second layer GCN output dimensionality, the layer attention output dimensionality, the number of layers of the multiplex network, the number of predicted edges, and the number of epochs for training, respectively. To compute the complexity of LATGCN, we focus on the local embedding representation extraction module, the layer attention mechanism module, the embedding representation fusion module, the decoder part, and the loss function part.

After a comprehensive analysis of our proposed LATGCN, the local embedding representation extraction module uses two-layer GCN and the time complexity of it is $O(K(E_{max}F_0 + NF_1 + E_{max}F_1 + NF_2))$. The time complexity of the layer attention mechanism module is $O(K(E_{max}F_2 + E_{max}F_3))$. The time complexity of the embedding representation fusion module is $O(KNF_2)$. The time complexity of the decoder part is $O(MF_2)$. The time complexity of the loss function part is $O(M)$. As F_0 , F_1 , F_2 , and F_3 increase, both runtime and memory requirements grow. When F_0 and F_1 are relatively large, the two-layer GCN has a greater impact on runtime, whereas when F_2 and F_3 is large, the layer attention mechanism exerts a stronger influence on runtime. Since F_0 , F_1 , F_2 and F_3 are closer to each other, the overall time complexity can be simplified as $O(J(K(E_{max}F + NF) + MF))$, where $F = \max\{F_1, F_2, F_3\}$. Obviously, $M < E_{max}$. For the sparser layer, $M < E_{max} \leq N$; for the denser layer, $E_{max} > N$. Thus, for sparser layers, the time complexity per iteration is $O(KFN)$, while for denser layers, it is $O(KE_{max}F)$. As the number of layers K in a multiplex network increases, the time complexity grows, thereby increasing computational costs.

4. Experiment

In this section, we conduct a series of experiments to evaluate the performance of our proposed LATGCN model. Firstly, the datasets, the experimental settings, the evaluation metrics, and the baseline methods are described. Secondly, a parameter sensitivity analysis of LATGCN will be performed. Finally, the proposed LATGCN is compared with several state-of-the-art methods and the experimental results are analyzed.

4.1. Datasets

In this paper, we conducted experiments on six widely used public network datasets, which include CS-Aarhus [39], CKM-Physicians-Innovation [40], Twitter-Foursquare [41,42], ff-tw-yt [43], Rattus [44], and Sacchpomb [44]. These datasets are described below and the statistics of these datasets are shown in Table 1.

CS-Aarhus (CS): This multiplex network consists of 61 nodes and five types of edges. The nodes are staff members of the Department of Computer Science at Aarhus University. These types of edges are Facebook, Casual, Work, Collaboration, and Lunch.

CKM-Physicians-Innovation (CKM): This multiplex network consists of 246 nodes and three types of edges. The nodes are physicians from four different towns in the United States, and the edges capture their interactions related to the adoption of new drugs. These types of edges are seeking advice, discussing cases, and building friendships.

Twitter-Foursquare (TF): This multiplex network consists of 1564 nodes and two types of edges, corresponding to user interactions on Twitter and Foursquare.

ff-tw-yt (FF): This multiplex network consists of 6407 nodes and three types of edges, representing user interactions on FriendFeed, Twitter, and YouTube.

Rattus (RA): This multiplex network consists of 2640 nodes and six types of edges, indicating the genetic interactions in Rattus Norvegicus. These types of edges are Physical association, Direct interaction, Colocalization, Association, Additive genetic interaction

defined by inequality, and Suppressive genetic interaction defined by inequality. Here we only use its first three layers.

Sacchpomb (SA): This multiplex network consists of 4092 nodes and seven types of edges, indicating the genetic interactions in *Saccharomyces Pombe*. These types of edges are Direct interaction, Colocalization, Physical association, Suppressive genetic interaction defined by inequality, Synthetic genetic interaction defined by inequality, Additive genetic interaction defined by inequality, and Association.

Table 1. Statistics of datasets.

Dataset	Nodes	Edges	Layers
CS	61	620	5
CKM	246	1551	3
TF	1564	30,882	2
FF	6407	30,882	3
RA	2640	4229	3
SA	4092	63,677	7

4.2. Experimental Settings

For each dataset, we selected one layer as the target layer and used the remaining layers as auxiliary layers. For the target layer, 10% of edges were randomly selected as the test set, another 10% as the validation set, and the remaining edges were used for training. We employed early stopping to prevent overfitting and to save training time and computational resources. Layers with extremely low density are unlikely to contain meaningful patterns for learning, so we restricted the evaluation to layers with adequate connectivity. Specifically, for the RA dataset, due to the extreme sparsity of other layers, we only used the first three layers (i.e., Physical association, Direct interaction, and Colocalization) for link prediction; for the other datasets, link prediction was performed on all layers. All experiments were conducted using an NVIDIA RTX 4060 GPU, with Python 3.8.13 and PyTorch 2.1.2.

4.3. Evaluation Metrics

We use four widely used metrics to evaluate performance, including area under the curve (AUC) [45], F1 score [46], AP [47], and Accuracy [48]. For each metric, a higher value indicates better performance. A brief description of these metrics is provided below.

AUC: It is the area under the ROC curve, which is an important measure of the model's ability to classify. It is essentially defined as the probability that the model assigns a higher probability score to a randomly selected positive instance than to a randomly selected negative one.

F1: It is a metric used to comprehensively evaluate a model's precision and recall in classification tasks especially in scenarios with class imbalance. The equation of F1 metric is shown as:

$$F_1 = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}}. \quad (12)$$

AP: It evaluates model performance by computing the area under the precision–recall curve, summarizing the trade-off between precision and recall at different thresholds.

Accuracy: It measures the proportion of correctly predicted links over all candidate links, providing an overall assessment of prediction correctness in link prediction tasks.

4.4. Baseline Methods

In order to conduct more comprehensive experiments, we compare the proposed LAT-GCN with classical link prediction methods and state-of-the-art link prediction methods.

NMF [49]: It learns potential relationships between nodes by decomposing the adjacency matrix into two low-rank non-negative matrices.

MGCN [50]: It is a multi-dimensional graph convolutional network framework that captures interactions both within and across layers.

LATGCN-VAR (VAR): It is a variant of our LATGCN, in which the layer importance score is uniformly set to $\frac{1}{K}$, where K is the total number of layers in the multiplex network.

MNERLP-MUL (MNER) [51]: It is a state-of-the-art link prediction method for multiplex networks. It employs an aggregation model to encode information from different layers into a summarized weighted static network. This process considers the relative densities of each layer and integrates both local and global structural information, based on the correlations among merged nodes and edges.

HOPLP-MUL (HOP) [52]: It is a state-of-the-art link prediction method for multiplex networks. It aggregates information from multiple layers into a single weighted static network by considering the relative density of each layer. The method then iteratively estimates link probabilities by exploring longer paths between nodes, thereby capturing higher-order structural information.

Layer-aware Graph neural network (LAGNN) [53]: It is a state-of-the-art link prediction method for multiplex networks, which aggregates the adjacency matrices of different layers and measures the inter-layer similarity.

4.5. Parameter Sensitivity Analysis

Since the size of the layer attention dimension affects the quality of aggregated neighborhood information, and the quality of neighborhood information further influences the richness of feature representation captured by the model, we investigate the performance of the proposed LATGCN under different layer attention dimensions in link prediction tasks to identify the optimal dimension size. To ensure generalization, we conduct experiments on six real-world multiplex network datasets, where 10% of the edges in the target layer are randomly selected as the test set, another 10% to form the validation set, and the remaining edges are used for training. For overall performance evaluation, we compute the average metrics across all layers under different attention dimensions for each dataset.

The experimental results for the metrics under different layer attention dimensions are presented in Figure 4. For the AUC metric, the CKM dataset shows an increasing trend as the layer attention dimension increases, reaching its optimal value at 256. In contrast, for the CS, TF, FF, RA, and SA datasets, AUC achieves the best performance at 128, but decreases when the dimension reaches 256, with the most pronounced drop observed in the CS dataset. For the F1 metric, the SA dataset exhibits an increasing trend with larger attention dimensions. For the CS, CKM, TF, FF, and RA datasets, F1 achieves the best performance at 128, the worst at 64, and then declines again at 256, with the CS dataset showing the most significant decrease. For the AP metric, the CKM and SA datasets show improvements with larger attention dimensions, performing worst at 32. For the CS, TF, FF, and RA datasets, AP achieves its highest value at 128 but drops at 256, with the CS dataset again showing a clear decline. For the Accuracy metric, the SA dataset shows improvements with increasing dimensions, while the CS, CKM, FF, and RA datasets achieve the best performance at 128. At 256, Accuracy decreases significantly on the CS dataset, whereas the TF dataset remains unchanged.

From the above analysis, we observe that the SA dataset benefits from larger attention dimensions, as its relatively large scale allows the model to capture richer neighborhood information. Conversely, the CS dataset suffers a significant performance drop at 256 due to its smaller scale, where larger attention dimensions may introduce noise, thereby degrading link prediction performance. Overall, our model achieves the best performance when the

layer attention dimension is set to 128. Therefore, in the subsequent experiments, we fix the attention dimension to 128 to ensure optimal performance.

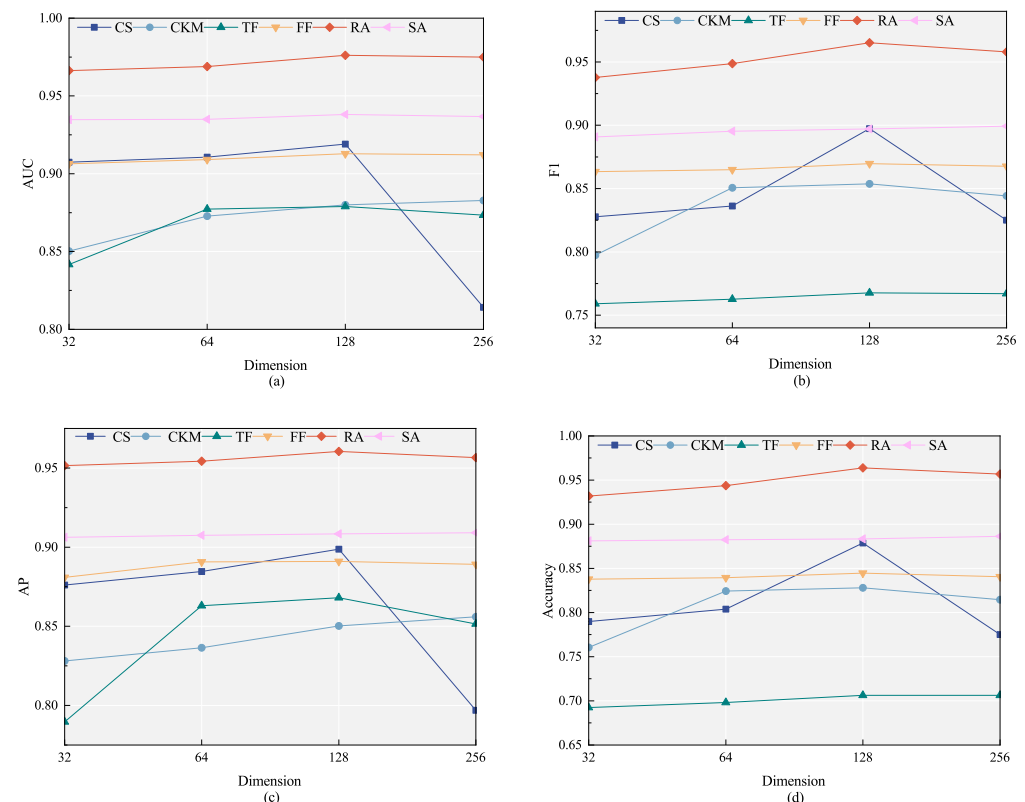


Figure 4. Comparison of metrics under different layer attention dimensions. (a) AUC; (b) F1; (c) AP; (d) Accuracy.

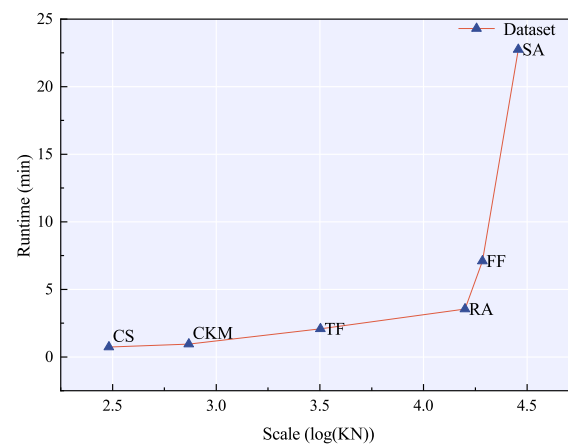
4.6. Comparison of the Link Prediction Performance

To comprehensively compare the performance of different methods, we compare the time complexity of the NMF, MGCN, MNER, HOP, LAGNN, and our LATGCN methods. As shown in Table 2, where r is the factorization dimension of NMF, m_1 and m_2 are the hidden layer dimension and the projection matrix dimension in the MGCN model, respectively. d_0 , d_1 , and d_2 are the input layer, hidden layer, and output layer dimensions of the two-layer GCN in LAGNN, respectively. D_{avg} is the average of the node degree; N is the number of nodes in MNER and HOP. E_{max} is the maximum value of the number of edges in each layer of the multiplex network, and F represents the maximum dimensionality of the feature vectors in LATGCN. Obviously, NMF, MGCN, MNER, HOP and LAGNN exhibit significantly higher time complexity than LATGCN on larger datasets with more nodes, particularly when the number of nodes is larger than the maximum number of edges per layer. Moreover, under identical conditions, MNER demonstrates a higher time complexity compared to HOP.

To better analyze how the time complexity of the proposed LATGCN changes with dataset scale, we compare its actual runtime across datasets of different scale, as shown in Figure 5. In this experiment, the number of training epochs is fixed at 500, the layer attention dimension is set to 128. We compare the overall runtime of LATGCN on multiplex networks of varying scale. As shown in Figure 5, the horizontal axis represents the dataset scale. The dataset scale is measured using $\log(KN)$, where K is the number of layers in multiplex network, N is the number of nodes in multiplex network. The vertical axis represents runtime, measured in minutes. The results show that the runtime of LATGCN increases with dataset scale, particularly when both the number of nodes and the number of layers are large.

Table 2. Comparison of time complexity.

Algorithm Name	Time Complexity
NMF	$O(K N ^{2r})$
MGCN	$O(K^2 m_1 ^2m_2)$
MNER	$O(N ^2D_{avg}^2)$
HOP	$O(N ^2D_{avg})$
LAGNN	$O(Nd_1(d_0 + d_2))$
LATGCN	$O(KNF)$ s.t. $M < E_{max} \leq N$ $O(KE_{max}F)$ s.t. $E_{max} > N$

**Figure 5.** Overall runtime of LATGCN with varying dataset scale.

Next, the performance of the above methods is compared in the link prediction task. For each dataset, 10% of the edges in the target layer are randomly removed to form the test set, another 10% to form the validation set, and the remaining edges are used for training. Link prediction is conducted independently on each layer of the multiplex networks, and the corresponding performance results are presented in the following section.

As shown in Table 3, the AUC values achieved by our method are generally higher than those of the baseline methods across all layers, except for the first layer of the CS dataset and the entire TF dataset. Similarly, Table 4 shows that our method outperforms the baseline methods in terms of F1 scores on all layers, except for the first layer of the CS dataset and the sixth layer of the SA dataset.

As for the AP metric, Table 5 shows that our proposed LATGCN method achieves superior results across all datasets, particularly on the TF dataset. As for the Accuracy metric, Table 6 shows that our proposed LATGCN method achieves superior results across all datasets, especially for the third layer of the CS dataset.

Although the AUC metrics of MNER and HOP are higher than that of our method in the TF dataset, the time complexity of them is higher for larger datasets such as the TF dataset. Although our method LATGCN performs slightly worse than MNER and HOP on the TF dataset, it requires less running time on larger datasets, thereby reducing the overall computational cost. Furthermore, as shown in Tables 4–6, LATGCN achieves much higher F1, AP, and Accuracy metrics on the TF dataset, indicating that it can better balance precision and recall, especially in the larger datasets. The higher accuracy reflects the LATGCN's capability of making more precise and consistent link predictions. More notably, as shown in Table 5, LATGCN achieves significantly higher AP metrics than VAR on the TF dataset, indicating the superior effectiveness of our proposed layer attention mechanism. This improvement is mainly attributed to the fact that when the number of layers in a multiplex network is small, our layer attention mechanism assigns relatively

larger importance scores to each layer, thereby enabling more effective aggregation of information across layers. On the TF dataset, the number of layers in the multiplex network is two, so LATGCN performs better when the number of layers is less.

Table 3. Comparison of AUC on each layer of the six datasets.

Dataset	Layer	NMF	MGCN	MNER	HOP	LAGNN	VAR	LATGCN
CS	1	0.6634	0.8500	0.9500	0.9500	0.7900	0.8425	0.8825
	2	0.8194	0.9527	0.9231	0.9231	0.8343	0.9527	0.9645
	3	0.7500	1.0000	0.3333	0.3333	1.0000	1.0000	1.0000
	4	0.6406	0.8889	0.7778	0.6667	0.8395	0.9012	0.9136
	5	0.6717	0.8200	0.7500	0.7500	0.7450	0.8210	0.8350
CKM	1	0.6583	0.8290	0.5111	0.4889	0.8201	0.8353	0.8405
	2	0.7139	0.8543	0.4800	0.5000	0.8169	0.9169	0.9237
	3	0.6695	0.8689	0.4186	0.4186	0.8131	0.8624	0.8758
TF	1	0.3623	0.7963	1.0000	1.0000	0.8401	0.7401	0.8862
	2	0.3695	0.8200	1.0000	1.0000	0.8200	0.7479	0.8717
FF	1	0.6842	0.9516	0.2586	0.2414	0.7561	0.9862	0.9917
	2	0.5408	0.8331	0.7917	0.6073	0.8647	0.8830	0.9014
	3	0.6945	0.7756	0.8055	0.5274	0.8101	0.8325	0.8455
RA	1	0.4712	0.9217	0.2410	0.2086	0.7609	0.9328	0.9537
	2	0.6330	0.9481	0.0202	0.0303	0.7241	0.9579	0.9746
	3	0.6364	1.0000	0.1667	0.3333	0.8166	1.0000	1.0000
SA	1	0.7868	0.9201	0.3060	0.2537	0.8213	0.9409	0.9660
	2	0.7027	0.9798	0.3243	0.2432	0.8424	0.9887	0.9994
	3	0.6940	0.9150	0.6476	0.6160	0.7036	0.9384	0.9410
	4	0.6227	0.7602	0.4424	0.4131	0.8008	0.8636	0.8893
	5	0.9211	0.8160	0.6299	0.6417	0.8814	0.9231	0.9350
	6	0.6551	0.8000	0.7705	0.7262	0.7637	0.8413	0.8606
	7	0.8095	0.9427	0.3636	0.4545	0.9106	0.9513	0.9757

Table 4. Comparison of F1 on each layer of the six datasets.

Dataset	Layer	NMF	MGCN	MNER	HOP	LAGNN	VAR	LATGCN
CS	1	0.7391	0.7170	0.8740	0.8720	0.7179	0.7843	0.8263
	2	0.7826	0.7429	0.8867	0.8916	0.7742	0.7879	0.8966
	3	0.6667	0.8571	0.6603	0.6626	0.7500	0.8571	1.0000
	4	0.5714	0.7200	0.8323	0.7895	0.7619	0.9365	0.9474
	5	0.7556	0.7037	0.7461	0.7372	0.7619	0.8011	0.8163
CKM	1	0.7321	0.7258	0.7298	0.7212	0.7037	0.8174	0.8348
	2	0.7705	0.7333	0.7138	0.7261	0.7010	0.8730	0.8837
	3	0.7308	0.7576	0.6925	0.6933	0.7007	0.8214	0.8430
TF	1	0.2640	0.7038	0.5000	0.5000	0.7133	0.7557	0.7648
	2	0.2859	0.7260	0.5000	0.5000	0.7377	0.7515	0.7705
FF	1	0.5000	0.9752	0.6293	0.6207	0.6393	0.9612	0.9883
	2	0.5850	0.8185	0.8115	0.7910	0.7490	0.8185	0.8254
	3	0.7396	0.7945	0.7833	0.7574	0.7306	0.7870	0.7953
RA	1	0.3349	0.7240	0.5776	0.5970	0.6652	0.9195	0.9406
	2	0.0784	0.6988	0.5094	0.5144	0.6625	0.9455	0.9550
	3	0.4286	0.6667	0.5833	0.6666	0.6667	1.0000	1.0000
SA	1	0.6765	0.7228	0.6527	0.6265	0.7679	0.9239	0.9330
	2	0.4898	0.8200	0.6621	0.6216	0.7391	0.9524	0.9880
	3	0.3392	0.7480	0.7803	0.7644	0.6232	0.8882	0.8990
	4	0.5016	0.6892	0.6939	0.6797	0.7401	0.8186	0.8358
	5	0.8645	0.6875	0.8138	0.8198	0.8136	0.8724	0.8770
	6	0.5535	0.7227	0.8350	0.8318	0.6997	0.7849	0.7884
	7	0.5000	0.8214	0.6817	0.7272	0.8400	0.9412	0.9588

Table 5. Comparison of AP on each layer of the six datasets.

Dataset	Layer	NMF	MGCN	MNER	HOP	LAGNN	VAR	LATGCN
CS	1	0.6025	0.8184	0.0534	0.0524	0.8170	0.8474	0.8817
	2	0.8078	0.9491	0.0441	0.0471	0.8361	0.9541	0.9682
	3	0.7500	1.0000	0.0408	0.0625	1.0000	1.0000	1.0000
	4	0.6052	0.8182	0.0337	0.0372	0.8649	0.8042	0.8394
	5	0.6018	0.7941	0.0337	0.0316	0.6836	0.7833	0.8040
CKM	1	0.5959	0.7736	0.0154	0.0162	0.7838	0.7845	0.7936
	2	0.6380	0.8411	0.0158	0.0180	0.8132	0.9034	0.9139
	3	0.6045	0.8347	0.0184	0.0193	0.8363	0.8398	0.8430
TF	1	0.3961	0.7508	0.0012	0.0012	0.8549	0.6546	0.8874
	2	0.3998	0.7874	0.0012	0.0012	0.8361	0.6634	0.8488
FF	1	0.6842	0.9756	0.0100	0.0114	0.7992	0.9842	0.9909
	2	0.5850	0.7727	0.0014	0.0037	0.8827	0.8718	0.8831
	3	0.6081	0.6932	0.0028	0.0085	0.7872	0.7852	0.7990
RA	1	0.4627	0.8867	0.0002	0.0011	0.7734	0.8918	0.9227
	2	0.6544	0.9205	0.0004	0.0006	0.7644	0.9583	0.9587
	3	0.6364	1.0000	0.0103	0.0186	0.8626	1.0000	1.0000
SA	1	0.7910	0.9265	0.0074	0.0062	0.8348	0.9067	0.9322
	2	0.7027	0.9797	0.0166	0.0140	0.8671	0.9901	0.9994
	3	0.5683	0.8964	0.0006	0.0006	0.7320	0.8981	0.9096
	4	0.5166	0.7388	0.0015	0.0014	0.8156	0.8204	0.8424
	5	0.8821	0.8370	0.0084	0.0087	0.8762	0.8768	0.8876
	6	0.5368	0.7543	0.0028	0.0043	0.7865	0.7924	0.8138
	7	0.8095	0.9474	0.0049	0.0127	0.9236	0.9583	0.9741

Table 6. Comparison of Accuracy on each layer of the six datasets.

Dataset	Layer	NMF	MGCN	MNER	HOP	LAGNN	VAR	LATGCN
CS	1	0.6842	0.6250	0.1901	0.1241	0.7250	0.7250	0.7898
	2	0.7917	0.6538	0.1449	0.1459	0.7308	0.7308	0.8846
	3	0.7500	0.8333	0.0258	0.0219	0.6667	0.8333	1.0000
	4	0.6250	0.6111	0.0715	0.0334	0.7222	0.9322	0.9445
	5	0.7105	0.6000	0.1028	0.0700	0.7500	0.7556	0.7750
CKM	1	0.6591	0.6458	0.0132	0.0193	0.6667	0.7813	0.8021
	2	0.7143	0.6491	0.0144	0.0105	0.7456	0.8596	0.8684
	3	0.6667	0.6863	0.0122	0.0107	0.5980	0.7887	0.8138
TF	1	0.3544	0.5851	0.0012	0.0012	0.6301	0.6899	0.7012
	2	0.3580	0.6343	0.0012	0.0012	0.7068	0.6832	0.7114
FF	1	0.6667	0.9758	0.0048	0.0081	0.6452	0.9608	0.9884
	2	0.5627	0.7864	0.0342	0.0293	0.7159	0.7867	0.7927
	3	0.6941	0.7503	0.0298	0.0358	0.7136	0.7427	0.7527
RA	1	0.4765	0.6226	0.0029	0.0022	0.5000	0.9139	0.9371
	2	0.5204	0.5932	0.0001	0.0002	0.5045	0.9455	0.9546
	3	0.6364	0.5001	0.0026	0.0417	0.5000	1.0000	1.0000
SA	1	0.7519	0.6301	0.0050	0.0029	0.7692	0.9201	0.9290
	2	0.6622	0.7805	0.0076	0.0046	0.7073	0.9512	0.9879
	3	0.5165	0.6719	0.0140	0.0173	0.6578	0.8802	0.8902
	4	0.5550	0.5921	0.0024	0.0033	0.7460	0.8051	0.8122
	5	0.8700	0.5829	0.0210	0.0170	0.8052	0.8667	0.8708
	6	0.5982	0.6307	0.0116	0.0229	0.7241	0.7321	0.7369
	7	0.6667	0.7916	0.0467	0.0106	0.8333	0.9375	0.9570

5. Conclusions

In this paper, we propose a new deep learning algorithm, LATGCN, to address the multiplex network link prediction problem. First, an embedding representation extraction

module is employed to capture node features. Second, a layer attention mechanism module synthesizes the information from each layer of a multiplex network using a layer importance score, which effectively captures the contribution of nodes in other layers to those in the target layer. Based on the learned contributions, the model assigns optimal weights to the embedding representations of each layer, enabling a more effective aggregation of global topological information in the multiplex network. Finally, we use an embedding representation fusion module to merge the local embedding representation and the global embedding representation. We conducted extensive experiments on six public multiplex network datasets, and the results validate the superiority of LATGCN. In addition, we demonstrate that failing to effectively synthesize embedding representations across layers degrades model performance, whereas LATGCN overcomes this limitation and achieves strong results on real multiplex network datasets. Although LATGCN shows strong performance in multiplex network link prediction, it still has several limitations. In particular, the model may incur high computational and memory costs when applied to very large-scale multiplex networks.

In future work, we will extend our framework to link prediction tasks for multiplex signed networks, which will be a challenging research direction for link prediction in multiplex networks.

Author Contributions: Conceptualization, M.Y.; methodology, M.Y.; funding acquisition, M.Y.; resources, M.Y.; supervision, M.Y.; formal analysis, Y.H.; validation, Y.H.; software, Y.H.; visualization, Y.H.; investigation, Y.H.; writing—original draft preparation, M.Y. and Y.H.; writing—review and editing, M.Y. and Y.H. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the National Natural Science Foundation of China under Grant No. 62402327, Natural Science Foundation Joint Fund of Liaoning Province under Grant No. 2023-BSBA-244, and Liaoning Provincial Department of Education funding for research projects under Grant No. LJ212410142053.

Data Availability Statement: This study analyzed existing data that are publicly available. These data were derived from the following resources available in the public domain: CS-Aarhus, CKM-Physicians-Innovation, Rattus, and Sacchpomb (<https://manliodedomenico.com/data.php>, accessed on 7 July 2024), Twitter-Foursquare (<https://github.com/shivansh-mishra/linkpredict-multiplex-layer>, accessed on 5 July 2024), and ff-tw-yt (<https://uuinfofab.github.io/data>, accessed on 10 July 2024).

Conflicts of Interest: The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

1. Chen, D.; Zhang, S.; Zhao, Y.; Xie, M.; Wang, D. An Attribute Graph Embedding Algorithm for Sensing Topological and Attribute Influence. *Mathematics* **2024**, *12*, 3644. [\[CrossRef\]](#)
2. He, Q.; Zhang, S.; Cai, Y.; Yuan, W.; Ma, L.; Yu, K. A Survey on Exploring Real and Virtual Social Network Rumors: State-of-the-Art and Research Challenges. *ACM Comput. Surv.* **2025**, *57*, 1–37. [\[CrossRef\]](#)
3. He, Q.; Zhang, L.; Fang, H.; Wang, X.; Ma, L.; Yu, K.; Zhang, J. Multistage competitive opinion maximization with Q-learning-based method in social networks. *IEEE Trans. Neural Netw. Learn. Syst.* **2024**, *36*, 7158–7168. [\[CrossRef\]](#)
4. Lakshmi, T.J.; Bhavani, S.D. Link prediction approach to recommender systems. *Computing* **2024**, *106*, 2157–2183. [\[CrossRef\]](#)
5. Abduljabbar, D.A.; Hashim, S.Z.M.; Sallehuddin, R. An enhanced evolutionary algorithm for detecting complexes in protein interaction networks with heuristic biological operator. In Proceedings of the Recent Advances on Soft Computing and Data Mining: Proceedings of the Fourth International Conference on Soft Computing and Data Mining (SCDM 2020), Melaka, Malaysia, 22–23 January 2020; Springer: Berlin/Heidelberg, Germany, 2020; pp. 334–345.
6. Wang, H.; Zhang, R.; Cheng, X.; Yang, L. Hierarchical traffic flow prediction based on spatial-temporal graph convolutional network. *IEEE Trans. Intell. Transp. Syst.* **2022**, *23*, 16137–16147. [\[CrossRef\]](#)
7. Dong, S.; Wang, P.; Abbas, K. A survey on deep learning and its applications. *Comput. Sci. Rev.* **2021**, *40*, 100379. [\[CrossRef\]](#)

8. Najari, S.; Salehi, M.; Ranjbar, V.; Jalili, M. Link prediction in multiplex networks based on interlayer similarity. *Phys. A Stat. Mech. Its Appl.* **2019**, *536*, 120978. [\[CrossRef\]](#)
9. Gao, M.; Jiao, P.; Lu, R.; Wu, H.; Wang, Y.; Zhao, Z. Inductive link prediction via interactive learning across relations in multiplex networks. *IEEE Trans. Comput. Soc. Syst.* **2022**, *11*, 3118–3130. [\[CrossRef\]](#)
10. Lu, R.; Jiao, P.; Wang, Y.; Wu, H.; Chen, X. Layer information similarity concerned network embedding. *Complexity* **2021**, *2021*, 2260488. [\[CrossRef\]](#)
11. Newman, M.E. Clustering and preferential attachment in growing networks. *Phys. Rev. E* **2001**, *64*, 025102. [\[CrossRef\]](#) [\[PubMed\]](#)
12. Dierk, S. The SMART retrieval system: Experiments in automatic document processing—Gerard Salton, Ed. (Englewood Cliffs, NJ: Prentice-Hall, 1971, 556 pp., \$15.00). *IEEE Trans. Prof. Commun.* **1972**, *1*, 17. [\[CrossRef\]](#)
13. Sorensen, T. A method of establishing groups of equal amplitude in plant sociology based on similarity of species content and its application to analyses of the vegetation on Danish commons. *Biol. Skr.* **1948**, *5*, 1–34.
14. Ravasz, E.; Somera, A.L.; Mongru, D.A.; Oltvai, Z.N.; Barabási, A.L. Hierarchical organization of modularity in metabolic networks. *Science* **2002**, *297*, 1551–1555. [\[CrossRef\]](#)
15. Adamic, L.A.; Adar, E. Friends and neighbors on the web. *Soc. Netw.* **2003**, *25*, 211–230. [\[CrossRef\]](#)
16. Zhou, T.; Lü, L.; Zhang, Y.C. Predicting missing links via local information. *Eur. Phys. J. B* **2009**, *71*, 623–630. [\[CrossRef\]](#)
17. Lü, L.; Jin, C.H.; Zhou, T. Similarity index based on local paths for link prediction of complex networks. *Phys. Rev. E Stat. Nonlinear Soft Matter Phys.* **2009**, *80*, 046122. [\[CrossRef\]](#) [\[PubMed\]](#)
18. Aziz, F.; Gul, H.; Muhammad, I.; Uddin, I. Link prediction using node information on local paths. *Phys. A Stat. Mech. Its Appl.* **2020**, *557*, 124980. [\[CrossRef\]](#)
19. Katz, L. A new status index derived from sociometric analysis. *Psychometrika* **1953**, *18*, 39–43. [\[CrossRef\]](#)
20. Perozzi, B.; Al-Rfou, R.; Skiena, S. Deepwalk: Online learning of social representations. In Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, New York, NY, USA, 24–27 August 2014; pp. 701–710.
21. Mikolov, T.; Chen, K.; Corrado, G.; Dean, J. Efficient estimation of word representations in vector space. *arXiv* **2013**, arXiv:1301.3781. [\[CrossRef\]](#)
22. Grover, A.; Leskovec, J. node2vec: Scalable feature learning for networks. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, 13–17 August 2016; pp. 855–864.
23. Cao, S.; Lu, W.; Xu, Q. Grarep: Learning graph representations with global structural information. In Proceedings of the 24th ACM International on Conference on Information and Knowledge Management, Melbourne, Australia, 19–23 October 2015; pp. 891–900.
24. Wang, D.; Cui, P.; Zhu, W. Structural deep network embedding. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, 13–17 August 2016; pp. 1225–1234.
25. Xuan, Q.; Fang, B.; Liu, Y.; Wang, J.; Zhang, J.; Zheng, Y.; Bao, G. Automatic pearl classification machine based on a multistream convolutional neural network. *IEEE Trans. Ind. Electron.* **2017**, *65*, 6538–6547. [\[CrossRef\]](#)
26. Niepert, M.; Ahmed, M.; Kutzkov, K. Learning convolutional neural networks for graphs. In Proceedings of the International Conference on Machine Learning, New York, NY, USA, 19–24 June 2016; pp. 2014–2023.
27. Kipf, T.N.; Welling, M. Semi-supervised classification with graph convolutional networks. *arXiv* **2016**, arXiv:1609.02907.
28. Wang, H.; Wang, J.; Wang, J.; Zhao, M.; Zhang, W.; Zhang, F.; Xie, X.; Guo, M. Graphgan: Graph representation learning with generative adversarial nets. In Proceedings of the AAAI Conference on Artificial Intelligence, New Orleans, LA, USA, 2–7 February 2018; Volume 32.
29. Li, Z.; Zhang, L.; Song, G. Sepne: Bringing separability to network embedding. In Proceedings of the AAAI Conference on Artificial Intelligence, Honolulu, HI, USA, 27 January–1 February 2019; Volume 33, pp. 4261–4268.
30. Yao, Y.; Guo, P.; Mao, Z.; Ti, Z.; He, Y.; Nian, F.; Zhang, R.; Ma, N. Multi-scale contrastive learning via aggregated subgraph for link prediction. *Appl. Intell.* **2025**, *55*, 489. [\[CrossRef\]](#)
31. Su, H.; Li, Z.; Yuan, C.A.; Vladimir, F.F.; Huang, D.S. Variational graph neural network with diffusion prior for link prediction. *Appl. Intell.* **2025**, *55*, 90. [\[CrossRef\]](#)
32. Gallo, L.; Latora, V.; Pulvirenti, A. MultiSAGE: A multiplex embedding algorithm for inter-layer link prediction. *arXiv* **2022**, arXiv:2206.13223.
33. Hamilton, W.; Ying, Z.; Leskovec, J. Inductive representation learning on large graphs. *Adv. Neural Inf. Process. Syst.* **2017**, *30*.
34. Nasiri, E.; Berahmand, K.; Li, Y. A new link prediction in multiplex networks using topologically biased random walks. *Chaos Solitons Fractals* **2021**, *151*, 111230. [\[CrossRef\]](#)
35. Bai, S.; Zhang, Y.; Li, L.; Shan, N.; Chen, X. Effective link prediction in multiplex networks: A TOPSIS method. *Expert Syst. Appl.* **2021**, *177*, 114973. [\[CrossRef\]](#)
36. Shan, N.; Li, L.; Zhang, Y.; Bai, S.; Chen, X. Supervised link prediction in multiplex networks. *Knowl.-Based Syst.* **2020**, *203*, 106168. [\[CrossRef\]](#)

37. Wang, C.; Tang, F.; Zhao, X. LPGRI: A global relevance-based link prediction approach for multiplex networks. *Mathematics* **2023**, *11*, 3256. [\[CrossRef\]](#)
38. Velickovic, P.; Cucurull, G.; Casanova, A.; Romero, A.; Lio, P.; Bengio, Y. Graph attention networks. *Stat* **2017**, *1050*, 10–48550.
39. Magnani, M.; Micenkova, B.; Rossi, L. Combinatorial analysis of multiple networks. *arXiv* **2013**, arXiv:1303.4986. [\[CrossRef\]](#)
40. Coleman, J.; Katz, E.; Menzel, H. The diffusion of an innovation among physicians. *Sociometry* **1957**, *20*, 253–270. [\[CrossRef\]](#)
41. Torabi, E.; Ghobaei-Arani, M.; Shahidinejad, A. Data replica placement approaches in fog computing: A review. *Clust. Comput.* **2022**, *25*, 3561–3589. [\[CrossRef\]](#)
42. Yang, R.; Yang, C.; Peng, X.; Rezaeipannah, A. A novel similarity measure of link prediction in multi-layer social networks based on reliable paths. *Concurr. Comput. Pract. Exp.* **2022**, *34*, e6829. [\[CrossRef\]](#)
43. Dickison, M.E.; Magnani, M.; Rossi, L. *Multilayer Social Networks*; Cambridge University Press: Cambridge, UK, 2016.
44. De Domenico, M.; Nicosia, V.; Arenas, A.; Latora, V. Structural reducibility of multilayer networks. *Nat. Commun.* **2015**, *6*, 6864. [\[CrossRef\]](#)
45. Hanley, J.A.; McNeil, B.J. The meaning and use of the area under a receiver operating characteristic (ROC) curve. *Radiology* **1982**, *143*, 29–36. [\[CrossRef\]](#)
46. Schütze, H.; Manning, C.D.; Raghavan, P. *Introduction to Information Retrieval*; Cambridge University Press Cambridge: Cambridge, UK, 2008; Volume 39.
47. Everingham, M.; Van Gool, L.; Williams, C.K.; Winn, J.; Zisserman, A. The pascal visual object classes (voc) challenge. *Int. J. Comput. Vis.* **2010**, *88*, 303–338. [\[CrossRef\]](#)
48. Liben-Nowell, D.; Kleinberg, J. The link prediction problem for social networks. In Proceedings of the Twelfth International Conference on Information and Knowledge Management, New Orleans, LA, USA, 2–8 November 2003; pp. 556–559.
49. Lin, C.J. Projected gradient methods for nonnegative matrix factorization. *Neural Comput.* **2007**, *19*, 2756–2779. [\[CrossRef\]](#)
50. Ma, Y.; Wang, S.; Aggarwal, C.C.; Yin, D.; Tang, J. Multi-dimensional graph convolutional networks. In Proceedings of the 2019 Siam International Conference on Data Mining, SIAM, Calgary, AB, Canada, 2–4 May 2019; pp. 657–665.
51. Mishra, S.; Singh, S.S.; Kumar, A.; Biswas, B. MNERLP-MUL: Merged node and edge relevance based link prediction in multiplex networks. *J. Comput. Sci.* **2022**, *60*, 101606. [\[CrossRef\]](#)
52. Mishra, S.; Singh, S.S.; Kumar, A.; Biswas, B. HOPLP- MUL: link prediction in multiplex networks based on higher order paths and layer fusion. *Appl. Intell.* **2023**, *53*, 3415–3443. [\[CrossRef\]](#)
53. Abedini, M.; Shakibian, H. Inter-layer similarity-based graph neural network for link prediction in social multiplex networks. In Proceedings of the 2024 10th International Conference on Web Research (ICWR), Tehran, Iran, 24–25 April 2024; IEEE: New York, NY, USA, 2024; pp. 76–80.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.