2023 International Conference on Communications, Computing, Cybersecurity, and Informatics (CCCI) | 979-8-3503-0256-1/23/$31.00 ©2023 IEEE | DOI: 10.1109/CCCI58712.2023.10290816

A Graph-based Network with Attention for Stock Price Prediction

1st Haiwei Fu
CentraleSupelec
Paris, France
haiwei.fu@student-cs.fr

***Abstract*—In today's world, graph-based data and models have become increasingly important and indispensable. As a result of graph-based neural networks' excellent ability to process these unstructured data, they have made new breakthroughs both in academia and in other fields. For stock prediction, Graph Neural Networks (GNNs) can bear more information, compared with Recurrent Neural Networks (RNNs), current GNNs assume that stocks and concepts are stationary, lack of dynamic relevance in long time series, and many methods and research fail to extract dynamic aggregated and shared information and feature representation in modeling stock price prediction. We propose a novel graph-based network with attention for stock price prediction (GBNA) to address these issues. To extract aggregated and shared information, we use a more instantiated and parsing method. Furthermore, GBNA can model longer time series information using attention mechanisms in Fully connected layer (Fc layer). Therefore, the novel model GBNA can not only process dynamic information, extract aggregated and shared information, but also longer time series, compared to RNN and LSTM. In addition, GBNA is also better at model features, which means it can extract features and represent them well and has a better fit ability not over smoothing. The experimental results show that our proposed model performs better than any other current methods in stock market data sets. With a better modeling ability, GBNA can predict stock prices more accurately with a higher IC and Rank IC.**

***Index Terms*—graph mining, graph neural network, feature representation, self-attention**

I. Introduction

Graph networks are ubiquitous in the real world, representing social networks, citation networks, chemical molecules, financial networks, and more. Graph Neural Networks (GNNs) are a powerful framework for deep learning in graph-related data.

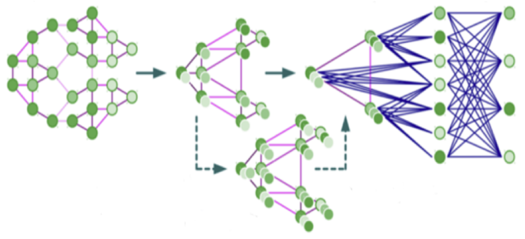

Fig. 1: **The GNNs mechanism**

GNNs can bear more structural information than other Neural Networks (NN), for example, Convolutional Neural Network (CNN) and Recurrent Neural Network (RNN).

In financial domain, stocks are one of the most profitable investment channels. Stock trend forecasting has become a fundamental part of many complex investment strategies in recent years to achieve high yield by investing in it. Currently, most forecasting models are built based merely on information about the stock prices and volume (e.g., opening and closing prices, highest and lowest trading volumes), as they assume that the price of different stocks is independent of each other. Graph Neural Networks (GNNs), which can utilize not only the information of stock price and trading volume, but also the correlation between stocks, has been used by researchers for stock investment and has achieved good results.

Normally, most stocks tend to be highly correlated with each other, researchers can use GNNs to classify factors, measure the momentum or trends, and make stock price prediction. However, in practice, the price trends of different stocks tend to be highly correlated with each other when these stocks bear the shared concept [1]. At the same time, researchers have taken note that many recent research have turned their attention to the use of this information in stock trend prediction, and obtain valuable information in stock concept through linear progresses. Some use the same predefined concept to form a relationship between two stocks and utilize GNNs which its edges can illustrate the relationship of various concepts, so as to establish a more accurate stock trend prediction model [2]–[5]. Despite this, most of the existing GNNs assume that these connections are static, and GNNs propagation mechanism Fig. 1. ignores the dynamic differences of different concepts of financial stock data. On the other hand, the unexpected events that would appear in a certain period of time, such as the Russia-Ukraine crisis, but existing models of GNNs ignore their differences from other such concepts [1].

The HIST framework [1] solves such problem. The HIST is a new framework both mine the stocks' aggregated and shared information and hidden concepts simultaneously, and it shows good results in experimental real-world data. Based on this, we make use of HIST, and we use self-attention at the Fc layer of HIST, so that the model GBNA can have more merits than other models. Our methods are introduced below: to solve the static property of GNNs on dynamic information, emergent information, hidden layer superposition and aggregated information, based on GNNs propagation mechanism, we make use of HIST model, and adopted a more interpretative and instantiated method. By passing parameters layer by layer, differentiate the filtered out information. At the same time, we built a doubly residual architecture [6] to extract the aggregated and shared information of each layer. To extract the features and present feature representation, we add self-attention to the the novel model GBNA's Fc layer of the model [7]. That is, our model can have a better ability of extracting features and a better present of feature representation ability. Moreover, the model GBNA can have a better modeling of longer time series, resulting in a higher accuracy. We use real financial market stock data to evaluate our model. Experimental results show that our model has better performance.

The main contributions of this paper are:

- The new model proposed by us can effectively been used for a longer time series than other RNNs models, for example, RNN, LSTM, etc, which has a better ability of data memorization.
- The new model, which can have a better ability of extract features and ability of present representation features, could help us better dig the valuable information, and do feature representation learning.
- The GBNA model which has a self-attention, can save time and computing source, have an efficient parallel computing. Moreover, we infer that with the self-attention, the result can have a better fit, rather than over smoothing that often happen in a graph networks.
- We run experiments with real-world financial data, and the results of the new model show that it performs well on both performance and experimental outcomes.

II. Related Work

In financial field, we usually use technical analysis, fundamental side analysis and event-driving to predict stocks price. GNNs predict stocks price is combined with technical analysis and factors of fundamental side analysis.

Some papers also research on the extract aggregated and shared information especially in market finance domain [8]. Based on this, this paper propose that we also need to focus on building neural networks for market finance domain, etc [9].

For graph networks, or graph neural networks, self-attention is used in graph in prediction [10], [11], and also in dynamic graph [12], etc.

In addition, there are many different methods to use in prediction and GNNs in financial markets. In terms of using weights, factors and GNNs, there also have some papers focus on weighted and factors related models, the main methods are embedding [13], or through distinguish with its target information [14], etc. For stock prediction, in this paper, the author uses Graph Attention Network (GAT) [15], in this paper, the author uses Nature Language Processing (NLP) with factors [16], in this paper, the author focus on spillover factors and create a novel neural network [17], in this paper, the author also use attention mechanism with GNNs to deal with the lag-effects in time series, and then optimise them [18], and in this paper, the author recently by using ChatGPT to inform the stock pricing prediction [19].

For all this, we do not find any papers that have solved both in extract aggregated and shared information and using self-attention to promote model. So, we think we are the pioneer one.

The HIST model [1] deals with the problems that we have mentioned before. So, our work based on this, to promote it by adding self-attention in the HIST model.

III. Preliminaries

1. Graph and GNNs. Graph Neural Networks (GNNs) is a set of deep learning methods that work in the graph domain. Those networks have recently been used in a number of areas including: optimization, recommendation systems, and computer vision. It can also be used to model large-scale systems, such as social networks, financial networks, protein-protein interaction networks, knowledge maps, and other research areas. Different from other data type such as images, therefore, it can use for node classification, link prediction and clustering, etc. There are several types of neural networks of graph: GCN, GAE, GATs, GGNN and so on. For the GNNs model, we must know both the node features X, and graph (nodes and edges). Especially for GCN [20], we have an adjacency matrix A, a degree matrix $\widetilde{D}$, and also have a Laplacian Transform, get $\widetilde{A}$ it is like a weighted mean.

$$\widehat{A} = (\widetilde{D}^{-\frac{1}{2}}\widetilde{A}\widetilde{D}^{-\frac{1}{2}})X, \tag{1}$$

The formula of Graph convolution network is:

$$H^{(l+1)} = \widetilde{D}^{-\frac{1}{2}}\widetilde{A}\widetilde{D}^{-\frac{1}{2}}H^{(l)}W^{(l)}. \tag{2}$$

GCN and ChebNet are typical applications of spectral domain convolution, However, the adjacency matrix of the directed graph is an asymmetric matrix, so the Laplacian matrix cannot be decomposed spectral at this time, so we need to redefine the adjacency relation, or use other forms of GCN to dig the relation on the topological graph. While spatial convolution operates on the neighborhood of a node, and the feature representation of the node is obtained by aggregating k-hop neighbors from the center node. Spatial convolution is simpler and more efficient than spectral convolution, like GraphSAGE and GATs are typical representatives of spatial convolution (GCN variant).

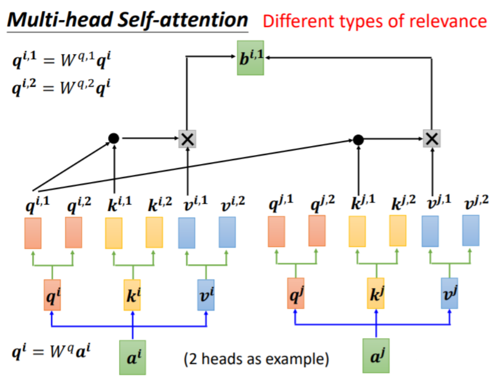
Multi-head Self-attention
Different types of relevance
$q^{i,1} = W^{q,1}q^i$
$q^{i,2} = W^{q,2}q^i$
$b^{i,1}$
$q^{i,1}$ $q^{i,2}$ $k^{i,1}$ $k^{i,2}$ $v^{i,1}$ $v^{i,2}$ $q^{j,1}$ $q^{j,2}$ $k^{j,1}$ $k^{j,2}$ $v^{j,1}$ $v^{j,2}$
q^i k^i v^i q^j k^j v^j
$q^i = W^q a^i$
a^i
(2 heads as example)
a^j

Fig. 2: **Interpretation of self-attention**

2. Self-attention. Self-attention comes from Transformer and it is used to process sequence data. First, it is to calculate the similarity between query and each key to get the weight. Common similarity functions include dot product, concatenation, perceptron, etc. Then is to normalize these weights using a softmax function. At final, the weight and the corresponding key, value are weighted and summed to get the final attention. Fig. 2. In this paper, we also use Multi-head Self-attention in the GBNA model, because of the calculate efficiency and saving running time.

3. Definitions. *Stock price* For data download from Qlib, we mainly have price data include opening price, closing price, volume weighted average price (VWAP), volumns, etc, and we use the closing price in our work.

Stock factors For stock factors, we also regard them as the stock concepts, they are from the stock of company where is attached to, like the industry, the business, the upstream or downstream. We have a pre-defined factors part, which is the factors that have been defined by human. Meanwhile, we have a hidden factors part, which is the factors that hidden in the hidden layer. And we use the HIST model to extract the aggregated and shared information of the hidden layer.

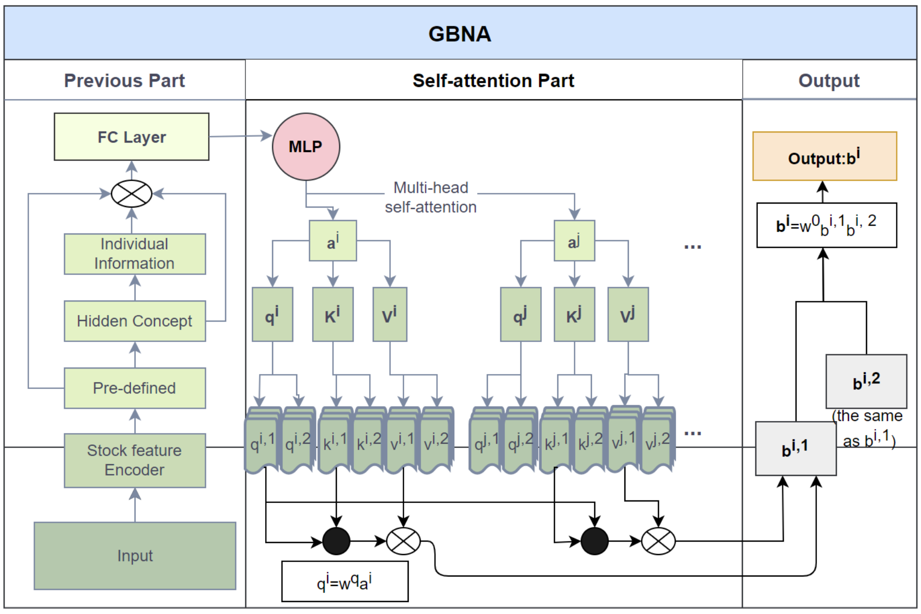

Fig. 3: **GBNA framework**

IV. Methodology

A. General Framework

The GBNA model, see Fig. 3. mainly has two parts. The previous part, which based on HIST model, then we use MLP to map parameters to our next part self-attention, which we select multi-head self-attention to be more efficient. The input data is S^t, the stock features are $S^t = \{s_1^t, s_2^t, \dots, s_n^t\}$ of n stocks and m pre-defined factors (concepts) T_j, $j \in \{1, 2, \dots, m\}$. For previous module, first, the Stock feature encoder, we have a 2-layer GRU to encoder the input data S^t, and for each stock feature s_n^t, to extract the temporal features of each stock. Second, we divide the pre-defined factors with the hidden factors and give the individual information by using doubly residual architecture. Then at this stage, we map the Fc layer perameters to MLP, then to have self-attention in Multi-heads part. Finally, we get the weights of hidden layer of aggregated and shared information with no noise information.

B. Previous Part

We input data S^t, in Stock feature Encoder module, we use a 2-layer Gated Recurrent Unit (GRU) [21] to encoder stock features in well handle the problem of long-term dependencies. Where stock i is in date t, the matrix is $X^{t,0}$, where the i-th row of $X^{t,0}$ is $x_i^{t,0}$.

$$x_i^{t,0} = GRU(s_i^t). \tag{3}$$

And then we follow a doubly residual architecture [6] at the model building at next stage before Fc layer.

1) Pre-defined Concept Module: Firstly, we build a bipartite way with the stocks and the pre-defined factors in the Pre-defined concept module. Secondly, we extract the predefined factors, we initialize the predefined factors with the weighted element-sum of stock embedding. c_i^t is the market capitalization of stock i in date t, and N_k^t is the set of stocks that related to the factors T_k. So, the predefined stock i its weight (concept) T_k is:

$$a_{k_i}^{t,0} = \frac{c_i^t}{\sum_{j \in N_k^t} c_j^t}, \tag{4}$$

After that, we initialize the predefined concept T_k, with the weight $a_{k_i}^{t,0}$ and we get $e_k^{t,0}$:

$$e_k^{t,0} = \sum_{i \in N_k^t} a_{k_i}^{t,0} x_i^{t,0}. \tag{5}$$

Thirdly, we correct the predefined factors using cosine similarity, $v_{k_i}^{t,0}$ between each stock embedding $x_i^{t,0}$, and each predefined concept T_k's initialization $e_k^{t,0}$. Then normalize the cosine similarity $v_{k_i}^{t,0}$ by softmax function, then we get the aggregated weight $a_{k_i}^{t,0}$:

$$v_{k_i}^{t,0} = Cosine(x_i^{t,0}, e_k^{t,0}) = \frac{x_i^{t,0} \cdot e_k^{t,0}}{\|x_i^{t,0}\| \cdot \|e_k^{t,0}\|}, \tag{6}$$

Then, we have:

$$a_{k_i}^{t,0} = \frac{exp(v_{k_i}^{t,0})}{\sum_{j \in S^t} exp(v_{k_j}^{t,0})}. \tag{7}$$

At final, we use LeakyReLU [22] as activation function, where w_e and b_e are the learning parameters:

$$e_k^{t,1} = LeakyReLU(w_e(\sum_{i \in S^t} a_{k_i}^{t,1} x_i^{t,0}) + b_e). \tag{8}$$

2) Hidden Concept Module: For the hidden concept in hidden layer, we mainly need to extract the aggregated and shared information. We have the input $x_t^{t,1}$, and the predefined concept $x_i^{t,0}$.

According to Equation 6, we have got the similarity of the x_i stocks. And for hidden layer we have the same method as Equation 6 and get hidden concepts H_i with its cosine similarity $h_{k_i}^{t,0}$. Then we delete the concepts that not connect with the stocks. After that, we use the same methods as Equation 7 to get the hidden layer concepts.

Moreover, to aggregating the concepts of shared information to stocks, we also compute the cosine similarity, so that we obtain aggregated weight and after do a LeakyReLU.

The outputs are the backcast $\hat{x_i}^{t,0}$ and forecast $\hat{y_i}^{t,0}$ outputs of predefined concept module. We leverage the Adam algorithm [23] as optimizer to loss function to the standard mean squared error (MSE):

$$L = \sum_{t \in \gamma} MSE(p^t, d^t) = \sum_{t \in \gamma} \sum_{i \in S^t} \frac{(p_i^t - d_i^t)^2}{|\mathrm{S^t}|}. \tag{9}$$

Where γ is the set of dates in the training set, S^t is a set of stocks, p_i^t is the prediction and d_i^t is the ground truth.

3) Individual Information Module: The individual information, we input $x_i^{t,2}$ in a Fully connection layer, using LeakyReLU as activate function as:

$$\hat{y}^{t,2} = LeakyReLU(w_f^2 x_i^{t,2} + b_f^2). \tag{10}$$

C. Self-attention Part

At the final NN model Fc layer, we build a self-attention [7] and connect with a 2-layer MLP to map the data, see Fig. 3. The input data is $x_i^{t,2}$, and we map data to our self-attention mechanism.

$$x_i^{t,2} = MLP(s_i^{t,2}). \tag{11}$$

The input $x_i^{t,2}$ include queries and keys of dimension d_k, and values of dimension d_v. We utilise dot products to compute the query with all keys, divide each by $\sqrt{d_k}$ and through a softmax function, then we get values. Often queries is a packed matrix Q, also keys and values are both packed matrix K and V. And we have the outputs:

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V. \tag{12}$$

Instead of performing a single attention function with d_{model}-dimensional queries, keys and values, we use a multi-head self-attention to save time, and run fast concurrently. And study linear progression in d_q, d_k, d_v dimensions. This is a concatenated work in parallel, and with different positions, we get:

$$\begin{aligned} & Multihead(Q, K, V) \\ &= Concat(head_1, \ldots, head_h)W^o \\ where \quad & head_i = (QW_i^Q, KW_i^K, VW_i^V). \end{aligned} \tag{13}$$

Where $W_i^Q \in \mathbb{R}^{d_{model} \times d_k}$, $W_i^K \in \mathbb{R}^{d_{model} \times d_k}$, $W_i^V \in \mathbb{R}^{d_{model} \times d_v}$ are parameter matrix and $W^o \in \mathbb{R}^{hd_v \times d_{model}}$.

In this task, we set $h = 2$, meaning we have 2 heads. And we have d_k = d_v = d_{model} / h = 64.

TABLE I: The results of CSI 100 and CSI 300

Methods	CSI 100					CSI 300				
	IC	Rank IC	Precision			IC	Rank IC	Precision		
	(↑)	(↑)	3	5	10	(↑)	(↑)	3	5	10
MLP	0.071 (4.8e-3)	0.067 (5.2e-3)	56.53 (0.91)	56.17 (0.48)	55.49 (0.30)	0.082 (6e-4)	0.079 (3e-4)	57.21 (0.39)	57.10 (0.33)	56.75 (0.34)
LSTM	0.097 (2.2e-3)	0.091 (2.0e-3)	60.12 (0.52)	59.49 (0.19)	59.04 (0.15)	0.104 (1.5e-3)	0.098 (1.6e-3)	59.51 (0.46)	59.27 (0.34)	58.40 (0.30)
GRU	0.103 (1.7e-3)	0.097 (1.6e-3)	59.97 (0.63)	58.99 (0.42)	58.37 (0.29)	0.113 (1e-3)	0.108 (8e-4)	59.95 (0.62)	59.28 (0.35)	58.59 (0.40)
SFM	0.081 (7.0e-3)	0.074 (8.0e-3)	57.79 (0.76)	56.96 (1.04)	55.92 (0.60)	0.102 (3.0e-3)	0.096 (2.7e-3)	59.84 (0.91)	58.28 (0.42)	57.89 (0.45)
GATs	0.096 (4.5e-3)	0.090 (4.4e-3)	59.17 (0.68)	58.17 (0.52)	57.48 (0.30)	0.111 (1.9e-3)	0.105 (1.9e-3)	60.49 (0.39)	59.96 (0.23)	59.02 (0.14)
ALSTM	0.102 (1.8e-3)	0.097 (1.9e-3)	60.79 (0.23)	59.76 (0.42)	58.13 (0.13)	0.115 (1.4e-3)	0.109 (1.4e-3)	59.51 (0.20)	59.33 (0.51)	58.92 (0.29)
Transformer	0.089 (4.7e-3)	0.090 (5.1e-3)	59.62 (1.20)	59.20 (0.84)	54.80 (0.33)	0.106 (3.3e-3)	0.104 (2.5e-3)	60.76 (0.35)	60.06 (0.20)	59.48 (0.16)
ALSTM+TRA	0.107 (2.0e-3)	0.102 (1.8e-3)	60.27 (0.43)	59.09 (0.42)	57.66 (0.33)	0.119 (1.9e-3)	0.112 (1.7e-3)	60.45 (0.53)	59.52 (0.58)	59.16 (0.43)
HIST	0.120 (1.7e-3)	0.115 (1.6e-3)	61.87 (0.47)	60.82 (0.43)	59.38 (0.24)	0.131 (2.2e-3)	0.126 (2.2e-3)	61.60 (0.59)	61.08 (0.56)	60.51 (0.40)
GBNA	**0.124 (2.0e-3)**	**0.119 (1.6e-3)**	**63.01 (0.42)**	**62.96 (0.43)**	**61.89 (0.34)**	**0.136 (2.3e-3)**	**0.102 (1.9e-3)**	**62.80 (0.48)**	**61.78 (0.39)**	**61.21 (0.32)**

TABLE II: Ablation study for effect with predefined information

Initial	Correct	Hidden	Individual	100 IC	100 R IC	100 Pre	300 IC	300 R IC	300 Pre
√	-	-	-	0.087	0.084	57.40	0.101	0.094	57.89
√	√	-	-	0.099	0.096	58.14	0.112	0.106	58.74
-	-	√	-	0.097	0.096	58.05	0.110	0.104	58.46
√	√	√	-	0.111	0.107	58.76	0.120	0.113	59.55
√	√	√	√	0.120	0.115	59.53	0.131	0.126	60.50

V. Experiment

A. Data sets

Data sets: We use data sets that are come from the Qlib platform [24]. We have CSI 100 and CSI 300, which are represented stocks in China A stocks market. They are the largest 100 and 300 stocks in the China A share market. The CSI 100 represents the performances of most influential large-cap A-shares market, the CSI 300 represents the overall performance of China A-share market.

Features: The Alpha360 is a algorithmic system, which can calculate the features as the factors[1] . It has 6 stocks data in daily, which are opening price, closing price, highest price, lowest price, volume weighted average price (VWAP) and trading volume. We use the features of stocks in CSI 100 and CSI 300 both from 01/01/2007 to 12/31/2020, and split them by time to obtain training set (from 01/01/2007 to 12/31/2014), validation set (from 01/01/2015 to 12/31/2016), and test set (from 01/01/2017 to 12/31/2020).

[1]We download Alpha360 the complete factor system from Qlib. https://github.com/microsoft/qlib

B. Metrics

In the experiments, we choose IC (Information Coefficient), Rank IC and Precision as metrics to evaluate the results of stock price prediction. The larger the IC factor, the stronger the stock picking ability. The maximum value of IC is 1, indicating that the stock selection factor is 100 accurate, corresponding to the stock with the highest ranking score, and the selected stock will increase the most in the next adjustment cycle. On the contrary, if the IC value is -1, the biggest decline in the next adjustment cycle, and is a completely inverse indicator. The most useless IC value is 0 or close to 0, which means that the factor has no predictive power for the stock. If the IC value is positively high, that means the stock price would be more correlated and influenced more, or vice versa.

C. Experiment Setup

Baselines: We use those methods as the baseline:

- MLP [25] MLP is a common neutral network, multiple data sets are mapped to a data set, to do linear classification and prediction.
- LSTM [26] LSTM is a RNNs neural network, and it better in dealing with gradient descent of the long time series.
- GRU [27] GRU also dealing with memory of long time series, different from LSTM, GRU use different algorithms.
- SFM [28] The SFM mixed with LSTM and Fourier transform, in order to forward decomposed frequency spectrum state.
- GATs [29] GATs is a GNNs model, using masked self-attention to do inductive learning.
- ALSTM [30] ALSTM uses linear combination, so it can not only see the state 'one step' ahead, but also see the further state ahead.
- Transformer [7] Compared with RNN network structure, its advantage is that it can be calculated in parallel.
- ALSTM+TRA [31] This model is to model multiple stock trading patterns, and improve information coefficient (IC).
- HIST [1] HIST can adequately mine the concept-oriented aggregated and shared information from predefined concepts and hidden concepts.

D. Results

The results, see TABLE I, shows the GBNA and baseline of other models' results. GBNA get the higher IC and Rank IC than other models in the baseline. Also has the higher results than other models. GBNA is a graph based neural network, not only can extract dynamic aggregated and shared information but also can have a better feature representation ability, resulting in higher precision than other current used models. it also proved that our model has a higher IC, Rank IC and a higher precision accuracy which outperforms other recently using models from the baselines. The experiments using real-world data sets also show the effective of our model in practical.

E. Ablation Study

We provide an ablation study and study the effect of our model comparing to other models that from the baselines. The method is that we remove some of the parts of the predefined concept and test the results, if our model have the effects. We then test the initial predefined concept with the correct predefined concept after model training, see in TABLE II. If we remove some parts, like Correct or/and Individual part, the performance will be sensitively influenced in the IC, Rank IC and Precision for CSI 100 and CSI 300 stocks. In the TABLE II, it shows correctly that individual information have been extract from the model, and the progress can effect the IC, Rank IC, as well as effect the Precision for both CSI 100 and CSI 300 stocks sensitively. That means our experiment can make some degree of influence and effect on the results. Moreover, the sensitivity of some information would make more effect, like Correct, Hidden and Individual, and not only the one kind of information could make the effect, but also in a mixed way.

VI. Conclusion

In this work, we propose a novel model GBNA, it is a graph based model that with self-attention. Based on our experiment and results, it can have a better modeling capability of extracting aggregated and shared information and better at feature representation. Moreover, it has a better modeling ability in having a longer time series, can compute parallel, and we infer it also can solve over smoothing problem that usually happen in graph networks and have a better fit. Different from other models. Based on this, it also proved that our model has a higher IC, Rank IC and a higher precision accuracy which outperforms other recently used models from the baselines. The experiments using real-world data sets also show the effective of our model in practical. For further work, we plan to expend our model to a wider range of domains, such as biology, chemistry, biochemistry, bio-medicine, gene, etc, to mining a more valuable representation features of the structure, with longer time series, to generalize our model that suit for different tasks of variable domains.

References

[1] W. Xu, W. Liu, L. Wang, Y. Xia, J. Bian, J. Yin, and T.-Y. Liu, "Hist: A graph-based framework for stock trend forecasting via mining concept-oriented shared information," *arXiv preprint arXiv:2110.13716*, 2021.

[2] F. Feng, X. He, X. Wang, C. Luo, Y. Liu, and T.-S. Chua, "Temporal relational ranking for stock prediction," *ACM Transactions on Information Systems (TOIS)*, vol. 37, no. 2, pp. 1–30, 2019.

[3] W. Li, R. Bao, K. Harimoto, D. Chen, J. Xu, and Q. Su, "Modeling the stock relation with graph network for overnight stock movement prediction," in *Proceedings of the twenty-ninth international conference on international joint conferences on artificial intelligence*, 2021, pp. 4541–4547.

[4] R. Kim, C. H. So, M. Jeong, S. Lee, J. Kim, and J. Kang, "Hats: A hierarchical graph attention network for stock movement prediction," *arXiv preprint arXiv:1908.07999*, 2019.

[5] R. Akita, A. Yoshihara, T. Matsubara, and K. Uehara, "Deep learning for stock prediction using numerical and textual information," in *2016 IEEE/ACIS 15th International Conference on Computer and Information Science (ICIS)*. IEEE, 2016, pp. 1–6.

[6] B. N. Oreshkin, D. Carpov, N. Chapados, and Y. Bengio, "N-beats: Neural basis expansion analysis for interpretable time series forecasting," *arXiv preprint arXiv:1905.10437*, 2019.

[7] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[8] S. Xiang, D. Cheng, C. Shang, Y. Zhang, and Y. Liang, "Temporal and heterogeneous graph neural network for financial time series prediction," in *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, 2022, pp. 3584–3593.

[9] L. Cao, "Ai in finance: challenges, techniques, and opportunities," *ACM Computing Surveys (CSUR)*, vol. 55, no. 3, pp. 1–38, 2022.

[10] S. Zhang, M. Jiang, S. Wang, X. Wang, Z. Wei, and Z. Li, "Sag-dta: prediction of drug–target affinity using self-attention graph network," *International Journal of Molecular Sciences*, vol. 22, no. 16, p. 8993, 2021.

[11] Z. Cheng, C. Yan, F.-X. Wu, and J. Wang, "Drug-target interaction prediction using multi-head self-attention and graph attention network," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 19, no. 4, pp. 2208–2218, 2021.

[12] A. Sankar, Y. Wu, L. Gou, W. Zhang, and H. Yang, "Dysat: Deep neural representation learning on dynamic graphs via self-attention networks," in *Proceedings of the 13th international conference on web search and data mining*, 2020, pp. 519–527.

[13] P. Konar, V. S. Ngairangbam, and M. Spannowsky, "Energy-weighted message passing: an infra-red and collinear safe graph neural network algorithm," *Journal of High Energy Physics*, vol. 2022, no. 2, pp. 1–30, 2022.

[14] J. Li and X. Yao, "Corporate investment prediction using a weighted temporal graph neural network," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 12, no. 6, p. e1472, 2022.

[15] S. Feng, C. Xu, Y. Zuo, G. Chen, F. Lin, and J. XiaHou, "Relation-aware dynamic attributed graph attention network for stocks recommendation," *Pattern Recognition*, vol. 121, p. 108119, 2022.

[16] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *arXiv preprint arXiv:1412.3555*, 2014.

[17] J. Tan, Q. Li, J. Wang, and J. Chen, "Finhgnn: A conditional heterogeneous graph learning to address relational attributes for stock predictions," *Information Sciences*, vol. 618, pp. 317–335, 2022.

[18] D. Cheng, F. Yang, S. Xiang, and J. Liu, "Financial time series forecasting with multi-modality graph neural network," *Pattern Recognition*, vol. 121, p. 108218, 2022.

[19] Z. Chen, L. N. Zheng, C. Lu, J. Yuan, and D. Zhu, "Chatgpt informed graph neural network for stock movement prediction," *arXiv preprint arXiv:2306.03763*, 2023.

[20] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907*, 2016.

[21] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *arXiv preprint arXiv:1412.3555*, 2014.

[22] A. L. Maas, A. Y. Hannun, A. Y. Ng *et al.*, "Rectifier nonlinearities improve neural network acoustic models," in *Proc. icml*, vol. 30, no. 1. Atlanta, Georgia, USA, 2013, p. 3.

[23] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[24] X. Yang, W. Liu, D. Zhou, J. Bian, and T.-Y. Liu, "Qlib: An ai-oriented quantitative investment platform," *arXiv preprint arXiv:2009.11189*, 2020.

[25] I. O. Tolstikhin, N. Houlsby, A. Kolesnikov, L. Beyer, X. Zhai, T. Unterthiner, J. Yung, A. Steiner, D. Keysers, J. Uszkoreit *et al.*, "Mlp-mixer: An all-mlp architecture for vision," *Advances in neural information processing systems*, vol. 34, pp. 24 261–24 272, 2021.

[26] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[27] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *arXiv preprint arXiv:1412.3555*, 2014.

[28] L. Zhang, C. Aggarwal, and G.-J. Qi, "Stock price prediction via discovering multi-frequency trading patterns," in *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, 2017, pp. 2141–2149.

[29] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Lio, Y. Bengio *et al.*, "Graph attention networks," *stat*, vol. 1050, no. 20, pp. 10–48 550, 2017.

[30] F. Feng, H. Chen, X. He, J. Ding, M. Sun, and T.-S. Chua, "Enhancing stock movement prediction with adversarial training," *arXiv preprint arXiv:1810.09936*, 2018.

[31] H. Lin, D. Zhou, W. Liu, and J. Bian, "Learning multiple stock trading patterns with temporal routing adaptor and optimal transport," in *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, 2021, pp. 1017–1026.