Advance Project 1 – Final Report

Mentor – Professor Azeez Bhavnagarwala

Author - Vishu Shah, vms8225

ABSTRACT

The objective of predicting stock market trends and patterns is to enable investors to make informed investment decisions. Anticipating stock prices or market movements allows investors to make optimal buying or selling decisions, which can increase profits or reduce losses. For optimal results from prediction models, it's important to consider various factors that impact the stock market such as economic indicators, company performance, and geopolitical events. One should also consider the correlation between different stocks for accurate stock prediction. In recent years, graphs have been extensively used to model such correlations. In light of this, various graph based approaches that have been utilized for this purpose are reviewed. The review explores the usefulness and suitability of GCN, GAT, GGNN, HGNN, and knowledge graph methods and their variations in the context of the stock market. As a result, various concerns and obstacles commonly encountered in numerous studies are identified. The objective of predicting stock market trends and patterns is to enable investors to make informed investment decisions. Our work is an extension of an existing approach of stock ranking based on predicted return ratios using a modified version of Graph Convolutional Networks (GCN) which incorporate temporal data. Applying discretization to sequential embeddings of technical indicators, we improve the investment return ratio, a metric important from an investor's point of view.

List of Figures

List of Tables

3.1	Results	

CONTENTS

CHAPTER 1: INTRODUCTION

This section presents the core concepts involved in our project. First we give a brief introduction on the stock market, and its various factors influencing it. Next, prediction in the stock market is explained, followed by explanation of graphs for the stock market. Finally, we explain our research objectives and the scope of this project.

1.1 Introduction to Stock Market

The stock market is a public marketplace where various financial securities, including stocks, are bought and sold. This allows individuals and organizations to invest in public companies and trade ownership stakes with other investors. It is an important economic indicator and is monitored by investors, traders, analysts, researchers, and policymakers [1]. The market enables companies to raise capital and investors to potentially earn profits through buying and selling stocks. The value of publicly traded companies is determined by the trading decisions of investors. The stock market is very volatile and is influenced by numerous factors, so investors need to make informed decisions. One way to achieve this is by using stock market prediction techniques, which involve forecasting the future performance of the stock market or specific stocks using various tools and methods. Investors and traders use these predictions to make investment decisions based on their expectations of how the market or a particular stock will perform in the future. The aim of stock market prediction is to identify trends and patterns that can help investors make informed investment decisions, such as determining the best time to buy or sell shares to potentially maximize profits or minimize losses.

1.2 Prediction in Stock Market

The stock market is very volatile as it is affected by a multitude of factors, and in order to reap the benefits of the market, investors need to make informed decisions. One of the prominent methods for this purpose is to make use of stock market prediction techniques. Stock market prediction refers to the practice of using various techniques and tools to forecast the future performance of the stock market or specific stocks. Investors and traders use stock market predictions to make investment decisions based on their expectations of how the market or a particular stock will perform in the future. The goal of stock market prediction is to identify trends and patterns that can help investors make informed investment decisions. By predicting market movements or stock prices, investors can determine the best time to buy or sell shares, which can potentially lead to higher profits or minimize losses.

Machine learning is a powerful tool that has been increasingly used in the stock market to predict future market trends and make informed investment decisions. Machine learning methods involve using algorithms and statistical models to analyze large amounts of historical data and make predictions about future stock prices. Machine learning models may use a variety of inputs, including historical stock prices, financial data, news articles, and other sources of information. These models are designed to identify patterns and relationships in the data that humans might not be able to detect.

A more advanced and complex subset of machine learning, deep learning, is well-suited for the purpose of stock market prediction. Deep learning provides several advantages over traditional machine learning algorithms, such as better feature extraction, learning complex relationships among features, better adaptability to non-linear data, and more. Above mentioned capabilities of deep learning depict the usefulness of the same for identifying complex patterns present in non-linear stock data and making accurate predictions.

To get the best possible result out of these models, one needs to take into account the variety of factors affecting the stock market. Some of the factors affecting the stock market include economic indicators, company performance, geopolitical events, investor sentiment, trading strategies, and market structure. A major factor other than previously mentioned that needs to be considered for

stock prediction, is the correlation between various stocks. This correlation among stocks can arise if the stocks belong to a common sector of the economy, the stocks are subsidiary stocks of a common parent company or if the company is a major supplier of the other company. This type of relationship between different stock entities is crucial in predicting the future price of stocks. Graphs are suitable data structures to store such relations that are non-euclidean data types. The research considering graphs in stock price prediction tasks is on the rise in the past few years.

1.3 Graph Neural Networks for Prediction in Stock Market

Graph data structures consist of nodes and edges and are used to represent relationships between real-world entities. These graphs, known as knowledge graphs, are utilized in various fields, including finance, to store specific types of information. Ontologies provide the foundation for the formal semantics of knowledge graphs, allowing for the retrieval of implicit knowledge through logical inferences. Knowledge graphs can integrate data from diverse sources with different structures and extract implicit knowledge that can reveal correlations among different stocks. However, processing such data for deep learning applications is challenging. Graph Neural Networks were developed [3], to integrate unstructured data from graphs and deep learning techniques, proving to be effective in node classification, graph classification, and link prediction in domains such as the stock market. GNNs are helpful in aggregating vast amounts of information present for different types of relationships, making them useful in predicting stock prices or market movements.

Stock market analysis relies heavily on continuous data to assess the performance and behavior of stocks and the overall market. Continuous data provides a detailed and granular representation of key market variables. However, continuous data presents challenges in terms of handling, analysis, and the application of certain algorithms. Discretization of data is a necessary step in data analysis when dealing with continuous or numeric values. By discretizing the data, we convert the continuous values into discrete categories or intervals, allowing for easier interpretation and processing. Discretization helps simplify data representation, address outliers and noise, and enables the use of algorithms that require categorical inputs. It plays a crucial role in data preprocessing, enhancing the effectiveness of subsequent analysis and modeling tasks.

Discretization can also help capture non-linear relationships between variables. By dividing continuous variables into meaningful intervals, the model can better capture patterns and relationships that may not be apparent in the original continuous form.

1.4 Research Objectives

The goal of this project is to investigate the use of graph-based approaches for predicting stock market trends and patterns. By analyzing various factors that impact the stock market, such as economic indicators, company performance, and geopolitical events, we aim to develop accurate prediction models that enable investors to make informed investment decisions. We will specifically focus on the correlation between different stocks, as this plays a crucial role in accurate stock prediction. We will review the usefulness and suitability of various graph-based approaches, including GCN, GAT, GGNN, HGNN, and knowledge graph methods, and their variations in the context of the stock market. Through our research, we aim to identify and address various concerns and obstacles commonly encountered in numerous studies, thereby enhancing the caliber and reliability of future research endeavors in this field. Ultimately, we will perform stock ranking based on predicted return ratios to evaluate the effectiveness of our prediction models. The results of this project can provide valuable insights to investors, traders, analysts, and researchers, and contribute to the development of more reliable prediction models for the stock market. Finally, we perform stock ranking based on return ratios which are predicted using a proposed novel approach to improve existing results.

1.5 Scope

The scope of the project is to conduct a literature review for “Stock market prediction using graph neural networks”. After providing a thorough analysis of the retrieved papers, open challenges, and future direction are discussed. After identifying potential gaps between the research, a novel approach for improving the existing results is proposed. It involves extending an existing approach using a modified version of GCNs which incorporates temporal data for prediction. Applying the concept of discretization to sequential embeddings of technical indicators, we significantly

improve the investment return ratio, an important metric indicative of an investment potential of a model's output.

CHAPTER 2: METHODOLOGY

This work is an extension of the paper published by [25], where the authors utilized a temporal graph-based approach for predicting the stock rankings. The time-aware embedding is utilized to encode temporal information into their relation-strength function. In this study, a temporal graph convolution is proposed for relational stock ranking based on criteria such as return ratio. The data used in the study is from NASDAQ and NYSE. In addition to historical data, 5-, 10-, 20-, and 30-day moving averages are used in the study. The stock relation data includes 112 and 130 types of sector-industry relations for NASDAQ and NYSE, respectively, and 42 and 32 types of relations for NASDAQ and NYSE are collected from the Wiki company-based relation. The proposed models are Relational Stock Ranking Explicit (RSR_E) and Relational Stock Ranking Implicit (RSR_I), and the baseline models are Rank LSTM, GBR, and GCN.

The authors made use of sequential embedding derived from the hidden representation of the LSTM layer. Along with normalized historical data as input to the model, wiki relation embedding and sector-industry embedding as adjacency matrices are utilized. Two types of modelings are proposed in the research: 1) implicit, and 2) explicit. The inner product of the sequential embeddings differentiates the modeling procedure during the training phase of the model. In terms of performance using wiki relation is proven to be more effective. The loss function considered is a combination of two losses, 1) regression loss, and 2) ranking loss. Here, the aim of regression loss and ranking loss is to the predicted return ratio and to improve relative ranking among the stocks respectively.

2.1 Data

This section describes the different types of data used and the preprocessing techniques applied to the data before giving it as input to the model.

2.1.1 Stock Exchanges

Stocks from the NYSE (New York Stock Exchange) and NASDAQ stock exchanges in the US stock market were selected for the purpose of this work. Records in the time period of 01/02/2013 to 12/08/2017 were retrieved. NYSE and NASDAQ contain 3,163 and 3,274 stocks respectively. However, only those stocks were selected which passed the following criteria:

- Stocks that were traded for more than 98% of the selected period.
- Stocks that were never traded for less than 5$ per share.

The first criteria was considered with the concern that such stocks may lead to abnormal patterns and the second criteria was selected as selected stocks should not be penny stocks, which are generally deemed to be too risky for general investors. After filtering out stocks, 1,026 and 1,737 stocks remain from NASDAQ and NYSE respectively. Three types of data of these stocks were selected for this work:

- Historical Data
- Wiki-Relation Data
- Sector-Industry Data

2.1.2 Historical Data

Historical data of the selected stocks is extracted from Google Finance and the data contains the open, high, low, close price, and trading volume of each day of the selected time period. Each stock's historical data is contained in a separate .csv (Comma Separated Values) file with the columns being Date-Time, Open, High, Low, Close, Volume.

Preprocessing:

Technical Indicators are then extracted for each stock using just the closing price. The technical indicators are the moving averages of the past 5, 10, 20, and 30 days' closing price. Moving

average of ‘x’ days is just the average of the closing prices of the past ‘x’ days. These technical indicators and the closing price are then normalized using the respective stock’s maximum closing price. This normalized data of each stock is created in a separate csv file which will then be utilized further for extraction of sequential embeddings.

This data is then split into train-val-test on the basis of the number of trading days. Total trading days are 1245, train set contains 756 days, validation set contains 252 days, test set contains 237 days.

EOD_embeddings, mask, ground_truth, and base_price, these four matrices are constructed from the original data. Specifically, in NASDAQ the last sample contains null values, therefore, the last sample is dropped. Here, the null values are represented as -1234. So, the value is replaced with 1.1. If the sample contains a null value for the closing price feature then the corresponding sample of mask is set as 0. The ground truth matrix has a return ratio of the normalized closing price.

2.1.3 Wiki-Relation Data:

Wikidata is a knowledge base containing 42 million entities (eg. companies) and 367 million statements in the form of (subject; predicate; object) (eg. Alphabet Inc.; founded by; Larry Page). From these, the following two categories of relations are obtained:

- First-order: Alphabet Inc. founded by Larry Page is a first-order relation. 5 types of this relation are obtained.
- Second-order: A second-order relation is formed when two subjects in different statements share the same object. 52 types of this relation are obtained.

A pictorial example of the above mentioned relations is given in Fig. 3.1

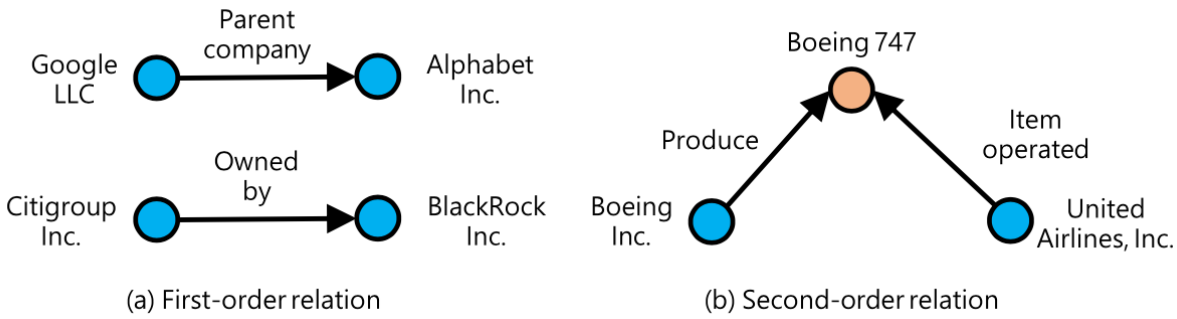

Figure 2.1 Wiki Relation Example

All of these relation types are given an id to represent connection between companies. All of the selected relations types and their respective counts found between companies are stored in a csv file, samples of which can be seen in Fig. 3.2. Rows 1, 2, 3, 6 in the figure represents second-order relations and rows 4, 5 in figure represent first-order relations.

P361_P361 6690

P452_P452 2984

P127_P127 790

P355 2

P155 2

P355_P355 10

Figure 2.2 Sample of wiki relation

A sample of first-order and second-order relation types can be found in tables 3.1 and 3.2 respectively.

Wikidata Relation	Description
P127	Owned by: owner of the subject.
P155	Follows: immediately prior item in a series of which the subject is a part.

Table 2.1 Example of first order wiki relation

	Wikidata Relation	Description
1	R_1 = P31	Instance of: that class of which this subject is a particular example and member.
	R_2 = P366	Use: main use of the subject.
2	R_1 = P31	Instance of: that class of which this subject is a particular example and member.
	R_2 = P452	Industry: industry of company or organization.
3	R_1 = P31	Instance of: that class of which this subject is a particular example and member.
	R_2 = P1056	Product or material produced: material or product produced by an agency.

Table 2.2 Example of second order wiki relation

Each company mentioned on wikidata is assigned an id. Selected stocks and their respective ids are stored in a csv file, sample of which can be found in Fig. 3.3

AABA	Q30321616
AAON	Q286214
AAPL	Q312
AAWW	unknown
AAXJ	unknown
AAXN	unknown

Figure 2.3 Sample of ticker, id file

Using these previously mentioned company and relation type ids, and statements from wikidata, a JSON file is formed containing each company's relation with other companies in the following format:

{ 'company **i**' :

{ 'company **j**' : [list of relations], ...}, ...}

Here, the list of relations contain all the relation types by which company 'j' is related with company 'i'. A sample from the created JavaScript Object Notation (JSON) file can be seen in the Fig. 3.4.


```
"Q5969562": {
  "Q20022777": [
    ["P17", "P17"],
    ["P31", "P31"],
    ["P414", "P414"]
  ],
  "Q6648755": [
    ["P31", "P31"],
    ["P414", "P414"]
  ],
```


Figure 2.4 Example of wiki connections data

Using the previously mentioned data, embeddings are generated representing these wiki relations. For this, a three-dimensional numpy array is created of shape (x, x, y) where,

- x is the number of tickers in the selected stock exchange
- y is the number of connection types found for tickers of the selected stock exchange plus a value for self-connection, out of the total 57 connection types

Generation of wiki relation embeddings can be better explained with the help of an example. Refer Fig. 3.5 which contains stats for NASDAQ wiki relations.

```
Number of tickers:  1026
Number of tickers with Wikidata ID:  512
Number of connection types:  57
Number of connection items:  511
Number of valid connections:  42
Shape of wiki relation embeddings:  (1026, 1026, 43)
```

Figure 2.5 NASDAQ wiki relation stats

From the Fig. 3.5, it can be seen that few of the selected stocks do not have an ID on the wiki knowledge base, in this case, the relation embeddings of these tickers will have 0 as its value. Those stocks that are related to each other by a specific connection type will have that specific relation type entry in the matrix as 1. So, in the end, the values in the wiki relation embeddings will be either 1 or 0.

2.1.4 Sector-Industry Relations:

The NASDAQ and NYSE stock exchanges are considered for generating sector-industry relation embeddings. The stock from NASDAQ and NYSE are classified as per the sectors on the NASDAQ Inc website. The sectors are further classified into industries according to the authors. Custom industries were created which are mapped to the underlying companies. The resultant industries for NASDAQ and NYSE are 112 and 130 accordingly. The curious thing here is that less than 10% of stock pairs with at least one type of industry in both stock markets, resulting in a sparse matrix for the stock exchanges.

Figure 2.6 Sector Industry examples

Depicted here is the screenshot of the JSON file provided by the authors, where the industries and the stocks related are presented as key-value pairs.

Preprocessing:

For the sole purpose of mapping the stock tickers to the respective stock based on the industries, one hot-encoding matrix was formed. One hot encoding is an identity matrix with the same dimension as the ticker size. The same sample from the one hot encoding is assigned to the respective companies in the same industry. The companies which do not belong to any of the industries are assigned the self-loop as all the stocks are connected to themselves.

The mask of relation data is constructed such that when multiplied with the original data only the values of focus are resulted. For creating the mask null values are replaced with -1e9 and otherwise, the values are 0.

2.2 Sequential Embeddings

A stock's historical movement is considered to play a crucial role in predicting its future trend due to stock markets' strong temporal dynamics. In order to capture the sequential dependencies in the historical data and incorporate them in the prediction model, sequential embeddings are generated. Recurrent Neural Networks (RNNs) have been effective in processing sequential data and have shown promising results in stock prediction research. Out of the various RNN models, such as vanilla RNN, Long Short Term Memory (LSTM), and Gated Recurrent Unit (GRU), LSTM is utilized owing to its ability to capture long-term dependencies found in data, a property essential for stock prediction. This is because various factors such as profit-loss reports, annual reports, interest rate rise, and many more have long-term effects on stock prices.

Sequential embeddings are generated from technical indicators extracted from historical data (refer section 3.1.2) using LSTM. So, the input feature matrix for the LSTM is three dimensional with the shape (X, Y, Z) where:

- X represents the number of tickers
- Y represents the window size or the number of days used for prediction
- Z represents the number of input features (which will be five in this case as four moving averages and closing price)

The LSTM then outputs a matrix of shape (X, Y, W) where W represents the number of hidden neurons and X, Y are the same as before. The sequential embeddings to be utilized for the purpose of model training is then extracted by selecting the matrix in the last hidden state of the output which will be the matrix at the last index of Y dimension.

```
device name: /cpu:0
ground truth =  (1026, 1)
mask =  (1026, 1)
feature =  (1026, 4, 5)
base price =  (1026, 1)
all one =  (1026, 1)
outputs =  (1026, 4, 64)
seq emb =  (1026, 64)
prediction =  (1026, 1)
return ration =  (1026, 1)
```

Figure 2.7 Dimension of NASDAQ input and output data

2.3 Temporal Graph Convolution

The Temporal Graph Convolution (TGC) is a machine learning module that is used to predict the future trends and patterns in the stock market. It takes two types of embeddings as input: sequential embedding and relational embedding. The sequential embedding is a vector representation of the historical data for each stock in the selected market. The shape of this embedding is (M, Q), where M is the number of stocks in the market, and Q is the length of the embedding obtained from the LSTM model. The relational embedding is an adjacency matrix that shows the relationship between the stocks in the market. The shape of this embedding is (M, M, P), where M is the number of stocks in the market, and P is the number of relations selected.

The main task of the TGC module is to learn the relation weights and update the sequential embedding based on these weights. The TGC module captures both temporal and relational information among the selected stocks using a time-sensitive relation-strength function. To model the interactions between two stocks, the TGC module uses two types of time-sensitive relation-strength functions: explicit and implicit modeling. In explicit modeling, the relation-strength function is based on the explicit time difference between the two stocks. In contrast, the implicit

modeling approach considers the strength of the relationship between two stocks to be a function of the historical correlation between their sequential embeddings.

Overall, the TGC module is a powerful tool for predicting future trends in the stock market, as it captures both the temporal and relational aspects of the data. By using the TGC module, investors can make informed investment decisions and maximize their profits while minimizing their losses.

2.3.1 Explicit Modeling

$$g\left(\mathbf{a_{ji}}, \mathbf{e_i^t}, \mathbf{e_j^t}\right) = \underbrace{\mathbf{e_i^{t}}^T \mathbf{e_j^t}}_{\text{similarity}} \times \underbrace{\phi(\mathbf{w}^T \mathbf{a_{ji}} + b)}_{\text{relation importance}}$$

Explicit modeling is a technique employed in the TGC module to capture the impact of the relationship between two stocks. This modeling is based on two crucial factors: the similarity between the stocks at the current timestep and the importance of the relation between them. The rationale behind this is that when two stocks exhibit high similarity, it is more likely that their relations will affect their prices in the near future. Therefore, estimating similarity is vital to forecast stock prices accurately.

To estimate similarity, an inner product is used between sequential embeddings, which is inspired by its effectiveness in modeling the similarity between two entities or embeddings in Collaborative Filtering. The use of an inner product provides an efficient way to measure similarity without introducing additional hyperparameters or computations. The resulting similarity value acts as a measure of the strength of the relation between the two stocks.

The second factor in explicit modeling is a nonlinear regression model that estimates the importance of the relations between the stocks. Each element in the weight matrix w corresponds to a relation in general, while the bias term b represents the global influence of all relations. By using a nonlinear regression model, the TGC module can capture complex and nonlinear interactions between the stocks. In contrast to traditional linear models, nonlinear models can capture more complex relationships between the input and output variables.

The explicit modeling technique, which combines similarity measurement and nonlinear regression, is highly interpretable. This interpretability allows investors and researchers to gain insights into the contribution of the similarity and relation importance factors in the prediction of stock prices. Therefore, the explicit modeling technique is an essential tool for modeling the relationship between two stocks in the TGC module.

2.3.2 Implicit Modeling

$$g\left(\mathbf{a_{ji}}, \mathbf{e_i^t}, \mathbf{e_j^t}\right) = \phi\left(\mathbf{w}^T \left[\mathbf{e_i^t}^T, \mathbf{e_j^t}^T, \mathbf{a_{ji}}^T\right]^T + b\right)$$

Implicit modeling is another approach used in the TGC module for capturing temporal and relational information among stocks in the stock market. Unlike explicit modeling, this approach does not require explicitly defined weights for the relation-strength function. Instead, a fully connected layer is used to estimate the relation strength by taking both the sequential embeddings and the relation vector as inputs. This layer has learnable model parameters w and b, and an activation function ϕ, similar to the explicit modeling approach. However, the output of this layer is normalized using a softmax function, which adds more non-linearities to the process.

In contrast to explicit modeling, where the relation strength is determined based on similarity and relation importance terms, the implicit modeling approach captures the interaction between two stocks using the learned parameters. This allows the model to implicitly model the relation strength and its temporal evolution over time, without the need for explicitly defined weights. Due to its implicit nature, we refer to this approach as Implicit Modeling.

2.4 Loss Function

$$l(\hat{\mathbf{r}}^{t+1}, \mathbf{r}^{t+1}) = \left\|\hat{\mathbf{r}}^{t+1} - \mathbf{r}^{t+1}\right\|^2 + \alpha \sum_{i=0}^{N} \sum_{j=0}^{N} \max\left(0, -\left(\hat{r}_i^{t+1} - \hat{r}_j^{t+1}\right)\left(r_i^{t+1} - r_j^{t+1}\right)\right)$$

Our proposed model is optimized by an objective function that includes two different types of losses: pointwise regression loss and pairwise ranking-aware loss. The ground-truth ranking scores, denoted as r^{t+1}, and the predicted ranking scores, denoted as $\hat{r}^{t+1}$, are combined in this objective function. A hyperparameter, α, is used to balance the impact of these two types of loss.

Our focus is on identifying the most profitable stock for trading, therefore, we use the 1-day return ratio of a stock as the ground-truth value rather than the normalized price used in previous research.

The first term in our objective function is the regression term that punishes the difference between the predicted ranking scores and the ground-truth scores. The second term is the pairwise max-margin loss, which aims to ensure that the predicted ranking scores of a pair of stocks have the same relative order as the ground-truth scores. This loss has been widely used in various applications such as recommendation systems and knowledge-based completion and has demonstrated strong performance in ranking tasks.

By minimizing our proposed combined loss function, the predicted ranking scores are encouraged to be close to both the absolute values of the return ratios and the relative orders of the return ratios among stocks. This facilitates investors in making better investment decisions. The correct relative order of stocks helps to select the investment targets, which are the top-ranked stocks. Additionally, the accurate prediction of the return ratio facilitates the timing of investment since the top-ranked stocks are valuable targets only when the return ratios are expected to largely increase.

2.5 Proposed Approach

As discussed earlier, discretization enhances the effectiveness of modeling tasks, and also helps better capture non-linear relationships. Keeping this in mind, the sequential embeddings generated using LSTM were discretized. As relational embeddings are already in discrete form, and the model also utilizes sequential embeddings as part of its input, sequential embeddings were chosen for discretization.

The values of sequential embeddings were rounded up to the third decimal point. Then, the values were classified according to the following equation:

```
\begin{equation}
   emb[x_{ij}] =
```

```
    \begin{cases}
        \hspace{0.3cm}1 & x_{ij} \le 1 \hspace{0.3cm} \& \hspace{0.3cm} x_{ij} > 0.5\\
        \hspace{0.3cm}0.5 & x_{ij} \le 0.5 \hspace{0.3cm} \& \hspace{0.3cm} x_{ij} > 0\\
        \hspace{0.3cm}0 & x_{ij} = 0 \\
        -0.5 & x_{ij} < 0 \hspace{0.3cm} \& \hspace{0.3cm} x_{ij} \ge -0.5\\
        -1 & x_{ij} < -0.5 \hspace{0.3cm} \& \hspace{0.3cm} x_{ij} \ge -1\\

    \end{cases}
\end{equation}
```

Here, after the discretization values are classified into five distinct values, namely: -1,-0.5,0,0.5,1. Fig 3.8, represents the proposed approach for a 3x3 embedding. The result of the approach gives the same matrix with classified values.

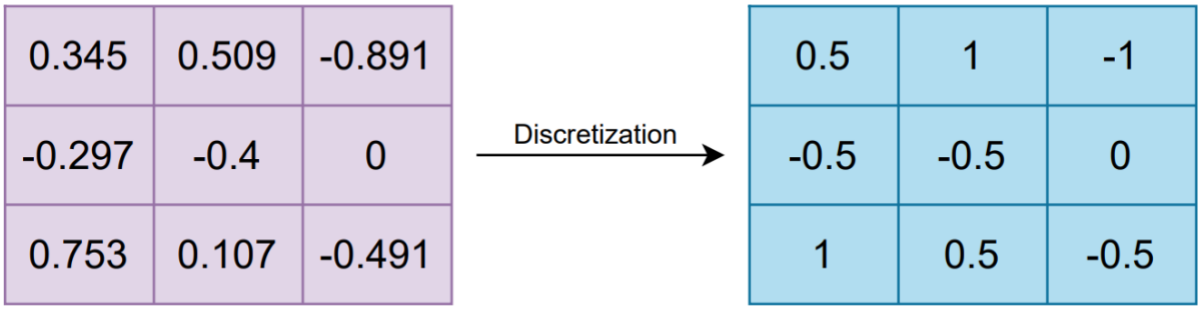

Figure 2.8 Data Discretization

2.6 Evaluation Metrics

The objective of the study is to predict the return ratio of stocks accurately and rank them appropriately. To evaluate the performance of the model, three metrics are employed - Mean

Squared Error (MSE), Mean Reciprocal Rank (MRR), and the cumulative investment return ratio (IRR).

MSE is a widely used metric in regression tasks and is calculated over all stocks on each trading day within the testing period. The lower the value of MSE, the better the performance of the model.

MRR is used to evaluate the ranking performance of the model. It calculates the average reciprocal rank of the selected stock over the testing days. The value of MRR ranges from 0 to 1, with a higher value indicating better performance. MRR value shows the degree of difference between the predicted rank and the actual rank in ground truth. Eg. for top 1 MRR, if the top ranking stock in prediction is at rank 5 in ground truth data, then the value of MRR will be 1/5.

IRR is the main metric used to evaluate the performance of the model, as it directly reflects the effect of stock investment. It is calculated by summing up the return ratios of the selected stock on each testing day. A higher value of IRR indicates better performance.

For each method, the testing procedure is repeated five times to eliminate the fluctuations caused by different initializations, and the average performance is reported. Therefore, a smaller MSE and larger values of MRR and IRR indicate better performance of the model.

Chapter 3: Results and Analysis

			Sector Industry			Wiki Relation		
			MSE	MRR	IRR	MSE	MRR	IRR

Base Paper	NASDAQ	E	3.83E-04	3.73E-02	1.280	3.79E-04	2.96E-02	1.286
		I	3.78E-04	3.96E-02	1.639	3.79E-04	2.87E-02	0.826
	NYSE	E	2.28E-04	3.26E-02	1.902	2.25E-04	1.90E-02	1.335
		I	2.27E-04	3.02E-02	0.834	2.27E-04	3.40E-02	1.557
Proposed Approach	NASDAQ	E	1.72E-02	2.63E-02	1.486	1.7E-02	3.138E-02	1.288
		I	1.035E-02	2.32E-02	1.751	1.75E-02	3.34E-02	1.322
	NYSE	E	2.427E-02	5.65E-02	2.079	2.30E-02	4.53E-02	1.413
		I	1.743E-02	1.19E-02	0.922	2.26E-02	5.31E-02	2.091

Table 3.1 Results

Several combinations of values were used for discretization to test which combination gives better results. Out of those tested, the values -1, -0.5, 0, 0.5, 1 for discretization provide better results of tested combinations.

The proposed approach demonstrates significant improvements in IRR for both implicit and explicit modeling techniques on the NASDAQ and NYSE stock exchanges.

Specifically, using sector industry relations, the results show a {x}% increase in IRR for implicit modeling and a {x}% increase for explicit modeling on the NASDAQ stock exchange, compared to the base paper's results. When incorporating wiki relations on NASDAQ, even greater improvements in IRR are achieved, with a {x}% increase for implicit modeling and a {x}% increase for explicit modeling compared to the base paper.

Turning to the NYSE, the approach again delivers substantial enhancements to IRR. The use of sector industry relations in NYSE generates a {x}% increase in IRR for implicit modeling and a {x}% increase for explicit modeling, relative to the base paper's findings. Meanwhile,

incorporating wiki relations on NYSE results in an {x}% increase in IRR for implicit modeling and a {x}% increase for explicit modeling, further outperforming the base paper's results.

Overall, these findings suggest that the proposed approach, which leverages sector industry and wiki data relations in addition to other data sources, can effectively improve IRR on both the NASDAQ and NYSE stock exchanges. However, it is evident from the results that while IRR shows significant improvement, MSE and MRR have decreased as well. As IRR is calculated using ground truth return ratios of stocks from predicted rankings and MSE is calculated using predicted return ratios, so, while predicted return ratios may deviate much more than the ground truth, the relative order of stocks obtained from these predicted return ratios are more accurate to the original order. Due to this reason, IRR is higher as profitable stocks are accurately selected from their rankings despite MSE being higher. As the main task is to rank stocks, the results are better suited for the task at hand.

These improvements have important implications for investors and financial analysts seeking to optimize their investment strategies and maximize returns.

Chapter 4: Conclusion and Future Work

This study reviews recent graph-based approaches in stock market prediction (2018-2023). By categorizing articles based on prediction approaches and exploring graph neural network models and knowledge graph methods, the study later focuses on understanding how investors can utilize prediction models to make informed decisions for maximizing investment returns. The model utilized in this paper was developed to perform the task of ranking stocks based on their predicted return ratios. An investor using this model will thus receive a relative ranking of stocks. For evaluating the returns of such prediction models, a metric informing the investor of the best returns is of great importance. In this case, IRR is the metric that is of use to the investor. Thus, an approach to improving IRR would be of benefit to the end user. Keeping this in mind, we propose an approach which improves IRR from the original approach. This proposed approach utilizes the concept of discretization due to benefits discussed earlier. On applying discretization to the sequential embeddings, the proposed approach significantly outperforms the IRR obtained from

the original approach. While MSE increases and MRR decreases, it is more depictive of the model's return ratio prediction performance rather than its ability to provide fruitful investment insights. Thus, from an investor's point of view, the proposed approach's results would be more beneficial in providing returns.

There are still several approaches that can be incorporated to extend this approach. Data sources which influence public opinion and stock prices such as news and social media can also be utilized in the model with the help of sentiment analysis. Hyperparameter tuning is also a major influencing factor for model performance. While, in this case several combinations of hyperparameters were tested for obtaining better performance, many metaheuristic approaches exist which can be applied to this case. The model can also be extended to be utilized for stock exchanges of other nations.

REFERENCES

[1] Srinivasan Palamalai and Karthigai Prakasam. "Stock market development and economic growth in India: An empirical analysis". In: Srinivasan P. and Karthigai P.(2014), Stock Market Development and Economic Growth in India: An Empirical Analysis, International Journal of Finance & Banking Studies 5.3 (2015), pp. 361–375.

[2] Jie Zhou et al. "Graph neural networks: A review of methods and applications". In: AI open 1 (2020), pp. 57–81.

[3] Franco Scarselli et al. "The graph neural network model". In: IEEE transactions on neural networks 20.1 (2008), pp. 61–80.

[4] Wenjun Zhang et al. "Research on Graph Neural Network in Stock Market". In: Procedia Computer Science 214 (2022), pp. 786–792

[5] Yunong Wang, Yi Qu, and Zhensong Chen. "Review of graph construction and graph learning in stock price prediction". In: Procedia Computer Science 214 (2022), pp. 771–778

[6] Weiwei Jiang. "Applications of deep learning in stock market prediction: recent progress". In: Expert Systems with Applications 184 (2021), p. 115537.

[7] Zexin Hu, Yiqi Zhao, and Matloob Khushi. "A survey of forex and stock price prediction using deep learning". In: Applied System Innovation 4.1 (2021), p. 9.

[8] Jinan Zou et al. “Stock Market Prediction via Deep Learning Techniques: A Survey”. In: arXiv preprint arXiv:2212.12717 (2022).
[9] Junhua Gu et al. “Dynamic Correlation Adjacency-Matrix-Based Graph Neural Networks for Traffic Flow Prediction”. In: Sensors 23.6 (2023), p. 2897.
[10] Aditya Grover and Jure Leskovec. “node2vec: Scalable feature learning for networks”. In: Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining. 2016, pp. 855–
864.
[11] Justin Gilmer et al. “Neural message passing for quantum chemistry”. In: International conference on machine learning. PMLR. 2017, pp. 1263–1272.
[12] Will Hamilton, Zhitao Ying, and Jure Leskovec. “Inductive representation learning on large graphs”. In: Advances in neural information processing systems 30 (2017).
[13] Thomas N Kipf and Max Welling. “Semi-supervised classification with graph convolutional networks”. In: arXiv preprint arXiv:1609.02907 (2016).
[14] Petar Velickovic et al. “Graph attention networks”. In: stat 1050.20 (2017), pp. 10–48550.
[15] Nicol`o Navarin, Dinh Van Tran, and Alessandro Sperduti. “Universal readout for graph convolutional neural networks”. In: 2019 International Joint Conference on Neural Networks (IJCNN). IEEE. 2019, pp. 1–7.
[16] Dong-Ming Song et al. “Evolution of worldwide stock markets, correlation structure, and correlation-based graphs”. In: Physical Review E 84.2 (2011), p. 026108.
[17] Wei Bao, Jun Yue, and Yulei Rao. “A deep learning framework for financial time series using stacked autoencoders and long-short term memory”. In: PloS one 12.7 (2017), e0180944
[18] Ashish Vaswani et al. “Attention is all you need”. In: Advances in neural information processing systems 30 (2017).
[19] Xiaohan Zou. “A survey on application of knowledge graph”. In: Journal of Physics: Conference Series. Vol. 1487. 1. IOP Publishing. 2020, p. 01201
[20] Yining Chen et al. “Scalable and Transferable Reinforcement Learning for Multi-Agent Mixed Cooperative–Competitive Environments Based on Hierarchical Graph Attention”. In: Entropy 24.4 (2022), p. 563.
[21] Shuai Zhang et al. “Joint Edge-Model Sparse Learning is Provably Efficient for Graph Neural Networks”. In: arXiv preprint arXiv:2302.02922 (2023).

[22] Xiaoxiao Li et al. “Braingnn: Interpretable brain graph neural network for fmri analysis”. In: Medical Image Analysis 74 (2021), p. 102233.
[23] Qiang Huang et al. “Graphlime: Local interpretable model explanations for graph neural networks”. In: IEEE Transactions on Knowledge and Data Engineer-ing (2022).
[24] Zipeng Liu et al. “Visualizing graph neural networks with corgie: Corresponding a graph to its embedding”. In: IEEE Transactions on Visualization and Computer Graphics 28.6 (2022), pp. 2500–2516
[25] Fuli Feng et al. “Temporal relational ranking for stock prediction”. In: ACM Transactions on Information Sys-tems (TOIS) 37.2 (2019), pp. 1–30