Exploring Graph Neural Networks for Stock Market Prediction

Kialan Pillay[1,2[0000−0002−2535−4505]] and Deshendran Moodley[1,2[0000−0002−4340−9178]]

[1] University of Cape Town, 18 University Avenue, Rondebosch, Cape Town, 7700, South Africa
[2] Centre for Artificial Intelligence Research, 18 University Avenue, Rondebosch, Cape Town, 7700, South Africa
pllkia010@myuct.ac.za and deshen@cs.uct.ac.za

Abstract. Stock markets are dynamic systems that exhibit both intra-share and inter-share temporal dependencies. The recent emergence of spatial-temporal graph neural networks allows for modelling multivariate time series data generated from systems with explicit or latent structure. This research addresses two weaknesses in the literature by investigating the applicability of spatial-temporal graph neural networks for Johannesburg Stock Exchange price prediction and the suitability of a correlation matrix to encode prior structural information for forecasting tasks. Our results demonstrate that the evaluated graph neural network techniques achieve suitable performance for forecasting tasks over variable prediction horizons. However, the results exemplify that pair-wise linear correlations do not accurately capture latent stock market dependencies, and further, a correlation matrix is an unsuitable encoding of prior information.

Keywords: Graph neural networks · Correlation matrix · Johannesburg Stock Exchange · Price prediction

1 Introduction

A number of novel Deep Neural Networks (DNN) structures have emerged recently to model complex and dynamic systems. Examples of such systems are traffic flow (congestion) in a city, weather and stock markets. Typical characteristics are high frequency and noisy observations from multiple sensors, possibly at different locations, with complex and often latent temporal dependencies, both within and between observational variables. The prominent approaches incorporate a graph neural network (GNN) to capture spatial and inter-variable dependencies. Each variable is typically represented as a node in a graph that captures the intra-variable dynamics of the variable, whilst the inter-variable dynamics are captured by weighted edges that reflect the strength of the dependencies between variables [19].

Thus a variable (node) has strong connections (links) to variables that are affected by changes in its values and weaker links to variables that are not affected by changes in its values. In this way, a node captures local dynamics (intra-variable), while the overall graph structure captures the global (inter-variable) dynamics of the system. The widely used application is predicting traffic flow at different points in a city traffic network. The inter-variable dependencies for this case are spatial relations between traffic flow at different points in a city's traffic network. These approaches are referred to as spatial-temporal GNNs (ST-GNNs). One of the challenges with ST-GNNs is that the spatial dependencies can be dynamic.

Typically, the topology of a road network is supplied as prior knowledge in the form of the adjacency matrix in the GNN. Early DNN approaches used a static graph structure which was provided to the network as prior knowledge before training. However, spatial dependencies in dynamic systems can change and evolve. Graph WaveNet (GWN) [19] was one of the first approaches to implement a dynamic adjacency matrix which learnt and adapted the spatial dependencies as it evolved in the data. One of the newest proposals that have emerged is Spectral-Temporal Graph Neural Network (StemGNN) [1]. StemGNN differs from GWN as it is a purely data-driven approach and requires no prior knowledge about dependencies. It incorporates intra-variable and inter-variable dependencies jointly in the spectral domain and is intended to automatically learn dependencies across multiple variables across time, irrespective of whether the dependency is spatial or not.

StemGNN was found to outperform GWN on nine benchmark datasets representing problems in the traffic, energy and electrocardiogram domains. However, none of the datasets tested was in financial domains, and specifically for share price prediction in a stock market. StemGNN was not tested against MTGNN [18], an evolution of the GWN architecture which yields better performance than GWN on traffic flow prediction. Stock markets are complex, highly dynamic and often erratic systems that are difficult to predict. In this paper, we evaluate and compare StemGNN, GWN and MTGNN for share price prediction on the Johannesburg Stock Exchange (JSE). While StemGNN, as a purely data-driven approach, has outperformed other techniques for capturing spatial and temporal dependencies in diverse domains, it is unclear whether can adequately deal with the complex temporal dependencies associated with share price movements in the stock market. We take a data-driven approach and formulate share price prediction as a multivariate problem using the daily closing prices of shares for the graph nodes and attempt to capture the inter-share dependencies.

2 Background and Related Work

2.1 Problem Formulation

A time series is a sequence of real-valued observations ordered in time. Formally, a univariate time series is a set of random variables $\{X_t, t \in T\}$, where $T = \{1, 2, ..., M\}$. A multivariate time series $\mathbf{X} \in R^{N \times M}$ is defined as a set of N univariate series.

Mathematically, a graph is a pair $G = (V, E)$, where V denotes the set of nodes and E the set of edges $e = (v, u)$. A feature vector X_v is associated with each node $v \in V$. The neighborhood of a node is defined as $N(v) = \{v \in V \mid (v, u) \in E\}$. The adjacency matrix $\mathbf{A} \in R^{N \times N}$ is a mathematical representation of a graph G, with $A_{ij} > 0$ for $(v_i, v_j) \in E$, $A_{ij} = 0$ for $(v_i, v_j) \notin E$ and N denotes the number of nodes. We formulate the stock market prediction problem as a multivariate spatial-temporal graph, where each node v represents a stock in the market. The feature vector X_v corresponds to a univariate times series which is the daily closing price of an individual share. The edges between nodes are weighted and represent the strength of any latent inter-share dependencies over a given time interval.

Node degree is the number of connected node edges. *Betweenness centrality* is a measure of a node's occurrence frequency on the pair-wise geodesic. *Closeness centrality* is a measure of a node's average inverse distance to all other nodes. *Transitivity* is a measure of interconnected neighbourhood nodes.

Multivariate single-step forecasting is the task of predicting the value of a single future set of daily share prices (node values) in a spatial-temporal graph conditional on the historical observations. Let $\mathbf{z_t} \in R^N$ denote a N-dimensional variable at time t. Given $\mathbf{X} = \{\mathbf{z_1}, \mathbf{z_2}, ..., \mathbf{z_t}\}$, the aim is to predict the single-step-ahead vector of node values $\mathbf{Y} = \{\hat{z}_{t+1}\}$.

Multivariate multi-step forecasting is the task of predicting a sequence of daily share price values conditional on the historical observations. Given observed values $\mathbf{X} = \{\mathbf{z_1}, \mathbf{z_2}, ..., \mathbf{z_t}\}$, the aim is to predict the sequence $\mathbf{Y} = \{\mathbf{\hat{z}_{t+1}}, \mathbf{\hat{z}_{t+2}}, ..., \mathbf{\hat{z}_{t+H}}\}$, where H is the next H time-steps, termed the prediction horizon.

2.2 Deep Neural Network Price Prediction

The majority of recent research investigating financial time series forecasting has focused on recurrent neural network (RNN) or Long short-term memory (LSTM) models [13]. The literature has also illustrated the increased prevalence of hybrid models that combine RNNs with an LSTM or RNNs combined with convolutional neural networks (CNN). In an applied portfolio management context, Ta et al. [15] and Wang et al. [17] apply an LSTM to identify investable financial securities based on the forecasted price, using a composite of price and fundamental data. Building on this methodology, Ma et al. [9] diverge by testing the predictive performance of DNNs using only historical raw price data. Ensemble or hybrid models have also been explored and are shown to perform better than standalone models [16]. DNNs have been used for both single- [3] and multi-step [7] [2] forecasting tasks. However, multi-step forecasting tasks involve additional complexity due to error accumulation. Error accumulation is the propagation of past error into future predictions [4], thereby decreasing predictive accuracy.

2.3 Spatial-Temporal Graph Neural Networks

Spatial-temporal GNNs are explicitly designed to model spatial and temporal dependencies in a system. Spatial and temporal components extract underlying patterns from the data in the corresponding domains. The spatial relations are reflected by the graph structure, with dynamic node-level inputs exhibiting one-dimensional temporal dependencies [19]. Spatial-temporal graph convolutional networks [21] [14] are CNN-based architectures frequently applied to spatial-temporal graph modelling tasks. However, Wu et al. state that previously introduced spatial-temporal GNNs are not suitable for modelling multivariate time series due to two distinct factors: unavailability of prior information and the sub-optimality of the predefined graph [18]. Prior knowledge of the spatial dependencies is further assumed, and the models rely on a predefined fixed graph structure for training. This approach is unsuitable for stock market systems with complex dependencies and latent structure. Graph WaveNet (GWN) [19], MT-GNN [18] and Spectral-Temporal Graph Neural Network (StemGNN) [1] are architectures that overcome the aforementioned limitations.

A graph convolution [5] component is used in all three techniques to extract spatial structure. However, in contrast to GWN and MTGNN, StemGNN extracts dependencies in the spectral domain. Both GWN and MTGNN utilise a temporal convolution network component to learn temporal dependencies within the time series.

GWN, MTGNN and StemGNN are all able to adaptively learn the graph structure without the provision of prior knowledge, i.e. providing an initial graph structure that specifies known dependencies. Both MTGNN and GWN outperforms other spatial-temporal GNNs that are initialised with fixed graph structures. However, GWN's performance decreases without prior structural information. A notable limitation of StemGNN is that in its explicit focus on a purely data-driven approach, it fails to accommodate available prior knowledge, whilst GWN and MTGNN are flexible and accept prior knowledge by initialising the adjacency matrix. A summary analysis of these three ST-GNN techniques is shown in table 1.

Table 1. Comparison of selected graph neural network architectures

Architecture	Tested GNNs	Temporal Component	Spatial Component	Datasets (MAPE%)	Strengths	Weaknesses
Graph WaveNet [19]	DCRNN STGCN	Gated Temporal (Gated TCN) Convolution	Graph Convolution (GCN)	Traffic (8.23%)	Learns graph structure adaptively Accepts prior information	Sub-optimal adaptive structure Unproven generality
MTGNN [18]	DCRNN STGCN GWN GMAN MRA-BGCN	Temporal Convolution	Graph Convolution (GCN)	Traffic (5.18%) Electricity (-) Solar (-) Forex (-)	Learns graph structure adaptively Accepts prior information Proven generality	Outperformed on multi-step tasks
StemGNN[1]	DCRNN STGCN GWN	Spectral-Sequential Cell (Spe-Seq Cell)	Spectral Graph Convolution (Spectral GCN)	Traffic (6.46%) Electricity (14.77%) Solar (11.55%) ECG (10.58%)	Learns graph structure adaptively Interpretable adaptive graph Proven generality	Cannot accept prior information

GWN, MTGNN and StemGNN have achieved state-of-the-art performance for single-step and multi-step forecasting task performance on the evaluated datasets. Whilst MTGNN and StemGNN have been tested across diverse domains applications and are intended for general multivariate problems, GWN is only evaluated on traffic flow datasets. None of the techniques has been applied to stock market prediction. However, MTGNN has been evaluated on a foreign exchange rate time series data set. Both MTGNN and StemGNN were shown to outperform GWN on the traffic flow datasets. Notably, MTGNN was not one of the techniques evaluated in the StemGNN study [1], so it is unclear how MTGNN will perform relative to StemGNN.

2.4 Graph Neural Networks for Stock Market Prediction

Whilst the ST-GNNs mentioned above have not explicitly been applied to the share price prediction there have been some isolated studies which have explored graph neural networks for capturing external information. Li et al. [8] propose an LSTM Relational Graph Convolutional Network (LSTM-RGCN) to explore the impact of overnight news on the opening prices of shares on the Tokyo Stock Exchange (TSE). The RCGN component extracts spatial dependencies, whilst an LSTM functions as a news text encoder and dynamic information propagation mechanism between RGCN layers. In their graph structure, each stock is a node, and the stock nodes are connected by the pair-wise inter-share correlations filtered by a threshold. The authors formulate the problem as movement classification problem instead of a price prediction problem. While LSTM-RGCN outperforms the selected baseline models, it produces a relatively poor classification accuracy rate of 57.53% compared to a random model that achieved 50.55%. Their proposed LSTM-RGCN focuses on the representation and impact of overnight news on the overnight share price movement and not multi-step price prediction.

Matsunaga et al. [10] explore known inter-company relationships for stock prediction on the Japanese Nikkei 225. They incorporate prior knowledge of supplier relations between companies in a knowledge graph and combine GCNs and an LSTM layer to form a Temporal Graph Convolution. Both studies propose and evaluate architectures customised for their applications and do not provide any performance comparisons with ST-GNN approaches. Sawhney et al. [12] propose a similar custom GCN+LSTM framework but construct a predefined stock market hypergraph that reflects complex dependencies instead of a simple graph. However, the authors formulate the problem as a ranking problem and not multi-step price prediction. Furthermore, the hypergraph is not learnt adaptively, requiring a significant amount of expert domain knowledge which restricts its application.

3 Experimental Design

In this research, we evaluate and compare the performance of three state of the art ST-GNN approaches, GWN [19], MTGNN [18], and StemGNN [1] for single-step and multi-step prediction of the closing prices of shares in the Top 40 Index on the Johannesburg Stock Exchange (JSE). Our objective was to capture the latent and dynamic dependencies between different shares on the JSE and evaluate the impact of this on price prediction. In order to effectively compare the three ST-GNN models, we followed the original experimental setups and configuration settings specified by the authors as closely as possible.

In order to apply ST-GNN approaches we formulate the problem as a flow problem, where a daily price movement in certain shares triggers a chain of daily price movements in other shares over a number of days. Our objective was to discover complex, non-linear latent dependency chains between different shares in the market. Both GWN and MTGNN allow for the specification of an initial adjacency matrix. Inspired by the approach taken by Li et al. [8] we use a correlation matrix to encode pair-wise dependencies to prime the adjacency matrices of GWN and MTGNN. The correlation matrix (Figure 3) is calculated using the historical daily close prices of the 30 companies in the dataset and initialises the adjacency matrix of GWN and MTGNN before training. The static correlation matrix represents pair-wise linear dependencies between shares in the market and serves as a basic initial starting point for representing more complex non-linear dependency chains. We thus adopt a purely data-driven approach and do not incorporate any additional prior knowledge like external news events [8] or supplier relations between companies [10].

3.1 GNN Models

The configuration details of the three GNN models are briefly described below.

Graph WaveNet GWN is a hybrid architecture for spatial-temporal graph modelling. The GWN architecture consists of temporal convolution (Gated TCN) and graph convolution (GCN) modules. The GCN module contains a self-adaptive adjacency matrix that requires no prior information and is learnt using stochastic gradient descent. The Gated TCN module consists of one-dimensional convolutions that capture long-range temporal sequences [11] whilst avoiding the vanishing gradient problem [6]. The GCN module extracts node-level dependencies using neighbourhood feature aggregation. The model outputs the predicted sequence over the entire prediction horizon H instead of iteratively generating H conditioned predictions. Following the authors' [19] adjacency matrix configuration results, we evaluate a double transition matrix plus adaptive adjacency matrix using the structural information initialisation against the adaptive-only adjacency matrix.

MTGNN MTGNN is a hybrid architecture designed with an explicit focus on multivariate time series forecasting. MTGNN can accommodate unavailable prior information through an adaptive adjacency matrix, although the model does not update the structure during training. MTGNN consists of a distinct graph learning layer that extracts the adjacency matrix. The GCN module utilises a neighbourhood aggregation strategy [20] to learn node-level spatial dependencies and the TCN module extracts temporal dependencies. MTGNN avoids the vanishing gradient problem [6] by including residual and skip intraand inter-layer connections. The model is trained using a curriculum learning strategy that splits the input into subgroups. Curriculum learning locates optimal local minima by training the algorithm on a single-step forecasting task first and subsequently increases the prediction horizon at each iteration [18]. We evaluate a predefined static graph structure against the adaptive-only adjacency matrix.

StemGNN StemGNN is a hybrid architecture for multivariate time series forecasting that captures inter-series correlations and temporal dependencies in the spectral domain. The spectral GCN component analogously extracts dependencies in the spectral rather than spatial domain. Spectral-Temporal GNN models are trained on the spectral representation of the graph [22] using graph signal processing techniques. A graph signal x is transformed into the spectral domain by a Fourier transform $\mathcal{F}$, a convolution operator is applied to the spectral signal and the inverse Fourier transform $\mathcal{F}^{-1}$ is applied to transform the signal into its original representation [22]. StemGNN consists of a latent correlation layer to automatically learn correlations between time series without a predefined graph structure to generate an adjacency matrix. For our evaluation, StemGNN is not initialised with prior structural information.

3.2 Data

The performances of the GNN models are compared on daily close price data for FTSE/JSE Top 40 Index constituent shares from 18 May 2009 to 20 July 2021. The Top 40 Index contains the 40 largest JSE-listed companies by market capitalisation. Market capitalisation is the current value of all outstanding shares. Companies listed after 2012 are excluded such that sufficient training data is available. The final dataset consisted of 30 nodes (stocks) and 3146 samples. For GWN and MTGNN, the data is further pre-processed following Wu et al. [19] and Wu et al. [18] to generate a four-dimensional dataset. The dataset is augmented by aggregating samples into windows of a specified size to construct synthetic features. The data is standardised using Z-score normalisation that removes the mean and re-scales to unit variance. The dataset is split in chronological order to preserve temporal dependencies with 60% for training, 20% for validation and 20% for testing (2019 - 2021).

3.3 Implementation

All three GNN models were implemented in PyTorch, based on the authors' implementations. Each experiment is conducted on an Apple MacBook Pro with an Intel(R) Core(TM) i5-8257U CPU @ 1.4 GHz. Following Cao et al. [1], the GNN models and LSTM are trained using RMSProp optimiser and Mean Squared Error loss function for 50 epochs. The initial learning rate is set to 0.001 with a decay and dropout rate of 0.05. All other hyperparameters are configured as reported by the authors [19] [18] [1] for accurate comparative analysis. We also selected the same error metrics to evaluate out-of-sample performance, i.e. Mean Absolute Percentage Error (MAPE), Mean Absolute Error (MAE), and Root Mean Squared Error (RMSE). For multi-step forecasting, these metrics are averaged over H steps corresponding to the prediction horizon and all nodes. Each experiment evaluated over 5 different training/test runs and the denormalised mean metric value is reported.

3.4 Baseline

Several baseline models were implemented and tested for comparison. The first is an LSTM with $l = 2$ hidden layers and $N = 500$ hidden layer nodes. A grid search of $l \in [1, 3]$ and $N \in [50, 500]$ was conducted to locate the optimal hyperparameter configuration.

The last LSTM layer is connected to a fully connected neural network layer and outputs a forecast over the prediction horizon for a single node, in contrast to the GNN models that produce multi-node predictions for the entire horizon in a single run. The LSTM is trained independently for each node and the mean metric value is reported.

For single-step forecasting, a Huber regressor, Ridge regressor, SVR ($C = 10, \epsilon = 0.03$) and last-value model are selected in addition to the LSTM model to produce baseline performance results. For any time series $X = \{x_1, x_2, x_3, ...x_t\}$, a last-value model outputs the last sequential observation x_t as the forecasted series value x_{t+1}. The baseline models are implemented using the PyTorch and sklearn machine learning libraries.

4 Results

4.1 Single-Step Forecasting

Table 2 shows the performance of GWN, MTGNN and StemGNN and the baseline models on a single-step forecasting task for 30-, 60- and 120-day window sizes. Regular trading days for the Johannesburg Stock Exchange are Monday through Friday, and thus the selected window sizes correspond to 6, 12 and 24 trading weeks.

Table 2. Single-step forecasting performance comparison of GNNs and baseline models

	30-Day Window			**60-Day Window**			**120-Day Window**		
	MAPE(%)	**MAE**	**RMSE**	**MAPE(%)**	**MAE**	**RMSE**	**MAPE(%)**	**MAE**	**RMSE**
Last-Value	12.56	1096.77	1672.06	12.98	1062.39	1459.65	12.92	1082.40	1493.29
Huber	12.83	**771.57**	**1129.51**	12.82	**804.25**	**1161.12**	13.46	905.29	1291.06
Ridge	12.18	801.95	1161.86	13.08	817.01	1178.17	12.45	**858.87**	**1230.04**
SVR	32.45	1396.30	2305.08	42.58	1824.44	2966.35	91.52	2602.45	3947.69
LSTM	27.96	2449.67	3431.34	29.07	2602.87	3409.67	43.69	3135.29	3977.02
Graph WaveNet	10.44	3072.72	8496.79	24.35	5432.03	15325.71	32.90	7603.66	20206.61
MTGNN	**6.76**	1478.92	3511.12	**9.06**	1953.99	4871.51	**10.62**	2472.13	6857.16
StemGNN	18.01	8415.90	21715.64	23.62	9329.78	21636.29	23.25	9249.32	21583.35

The results illustrate that MTGNN outperforms both GWN and StemGNN by a substantial margin across all metrics for all window sizes. In addition, the choice of input window size affects performance with performance degrading with larger window sizes. The degradation is much more pronounced for GWN. Whilst GWN outperforms StemGNN for a 30-day window, the results indicate a sharp decline in GWN's performance as the window size is doubled from 30 to 60.

MTGNN is by far the best performing model using a 30-day input window amongst both the ST-GNNs and the baseline models. It achieves an approximately 3% lower MAPE compared to the next best ST-GNN, Graph WaveNet, and a 5% lower MAPE score compared to the best baseline model, i.e. Ridge regressor.

4.2 Multi-Step Forecasting

Table 3 compares the performance of GWN, MTGNN and StemGNN and baseline LSTM on a multi-step forecasting task for 20-, 40- and 60-day window sizes and 5-, 10- and 20-day close price prediction horizons.

Table 3. Multi-step forecasting performance comparison of GNNs and baseline LSTM model across different window sizes

20-Day Window	**5-Day Horizon**			**10-Day Horizon**			**20-Day Horizon**		
	MAPE(%)	**MAE**	**RMSE**	**MAPE(%)**	**MAE**	**RMSE**	**MAPE(%)**	**MAE**	**RMSE**
LSTM	47.74	3168.03	**4411.10**	53.65	3729.25	**5157.98**	70.02	4928.60	**6656.45**
Graph WaveNet	**17.87**	**2554.10**	6430.45	**17.27**	**2716.00**	6819.69	**12.17**	**3147.66**	8852.29
MTGNN	216.97	29217.75	54875.26	258.05	31484.04	54808.86	232.43	32251.45	63249.00
StemGNN	24.96	10327.52	24464.46	26.64	10907.52	25375.75	30.92	11992.33	27384.19
40-Day Window									
LSTM	45.96	3207.62	**4363.04**	53.72	3712.97	**5037.78**	71.10	5224.54	**6676.38**
Graph WaveNet	**12.44**	**3960.92**	11642.37	**15.35**	**4370.69**	12882.04	25.69	**5022.74**	13472.62
MTGNN	122.63	19905.99	42729.60	152.71	23566.66	49893.09	182.12	29028.59	62179.24
StemGNN	19.78	8640.64	21504.16	19.94	85f76.14	21305.53	**21.97**	9287.20	22693.34
60-Day Window									
LSTM	46.39	**3241.21**	**4204.87**	56.91	**3916.42**	**5053.54**	65.50	**4563.63**	**5884.99**
Graph WaveNet	25.82	5535.79	15284.90	26.23	5792.11	16287.67	28.82	5998.07	16441.86
MTGNN	269.24	32627.72	59139.12	324.88	34788.54	54687.03	269.80	30684.99	51937.85
StemGNN	**23.10**	9198.82	21732.27	**25.55**	10797.06	26811.51	**23.51**	9604.94	22622.16
Best Model	**Graph WaveNet**			**Graph WaveNet**			**Graph WaveNet**		
Best MAPE	**12.44%**			**15.35%**			**12.17%**		

When comparing MAPE scores, GWN significantly outperforms MTGNN, StemGNN and the LSTM for the tested hyperparameters. Whilst MTGNN was the best performer on a single-step forecasting task, it fared very poorly on multi-step forecasts. Even though StemGNN's performance is much closer to GWN's performance than MTGNN, there is still overall a substantial and clear performance difference between the two approaches.

Most interestingly is that increasing the prediction horizon does not always result in performance degradation. For example, the best overall prediction performance of GWN is on 20-day forecasts and not on 5-day forecasts. This result could be because GWN captures longer-term dependencies more accurately than StemGNN.

All techniques yield the best 5-day and 10-day predictions on a 40-day input window with GWN producing the best performance of 12.44% and 15.35% respectively. For 20-day predictions, GWN also yields the best overall performance (12.17%) but on a 20-day input window. However, StemGNN outperforms GWN on 60-day window sizes. The results do not demonstrate a clear relation between window size and prediction error.

StemGNN demonstrates inferior performance on the shortest window size, producing superior MAPE scores with a 40-day input window. An outlier is its performance for a 40-day window and 20-day horizon, where it outperforms GWN and MTGNN as measured by MAPE. Furthermore, StemGNN's performance is relatively stable as the prediction horizon increases for both 40- and 60-day windows across all metrics for the tested horizon lengths.

4.3 Correlation Matrix Impact

We then tested the impact of prior knowledge, i.e. the inclusion of the correlation matrix on multi-step prediction performance. Table 4 shows the performance of GWN and MTGNN on the 5- and 10-day multi-step forecasting task for 20- and 40-day input window sizes, with and without the inclusion of the correlation matrix to initialise the GWN and MTGNN adjacency matrices.

Table 4. Multi-step forecasting performance comparison of GWN and MTGNN with predefined adjacency matrix across different window sizes

20-Day Window		**5-Day Horizon**			**10-Day Horizon**		
	Adjacency Matrix	**MAPE(%)**	**MAE**	**RMSE**	**MAPE(%)**	**MAE**	**RMSE**
Graph WaveNet	Adaptive-only	17.87	**2554.10**	**6430.45**	17.27	**2716.00**	**6819.69**
Graph WaveNet	Forward-backward-adaptive	**15.80**	3237.76	9103.10	**16.59**	3348.49	9104.82
MTGNN	Adaptive-only	216.97	29217.75	54875.26	258.05	31484.04	54808.86
MTGNN	Predefined	214.05	28885.62	54690.14	237.09	30302.03	55173.77
40-Day Window							
Graph WaveNet	Adaptive-only	**12.44**	**3960.92**	**11642.37**	**15.35**	**4370.69**	**12882.04**
Graph WaveNet	Forward-backward-adaptive	22.25	5150.09	14683.01	20.43	5089.13	14254.43
MTGNN	Adaptive-only	122.63	19905.99	42729.60	152.71	23566.66	49893.09
MTGNN	Predefined	118.53	20303.73	44546.03	192.05	26726.98	52491.44
Best Model		**Graph WaveNet Adaptive**			**Graph WaveNet Adaptive**		
Best MAPE		**12.44%**			**15.35%**		

The MAPE scores indicate that the provision of initial structural information does not improve predictive performance. For a 20-day window, the GWN with the forward-backward-adaptive adjacency matrix obtains a marginally higher MAPE score. However, doubling the window size significantly degrades the GWN performance when initialised with the correlation matrix. This may be attributed to the temporal divergence between the longer input sequence length and the shorter-term dependencies captured by the correlation matrix.

The impact of prior information on MTGNN is inconclusive. There is a substantial performance improvement when using a 40-day horizon. The predefined MTGNN configuration yields the best 5-day predictions with a 40-day input window, while the best 10-day predictions are produced by the adaptive-only MTGNN with a 40-day input window.

4.4 Inter-Share Dependency Analysis

We then compared the inter-share dependencies and graph structure produced by the three ST-GNNs. GWN and MTGNN produce a graph structure that is highly dissimilar to that represented by the correlation matrix. Comparing the adaptive adjacency matrices (Figure 4 in the appendix), GWN extracts a sparse graphical structure, in contrast to the dense graph learnt by MTGNN. There

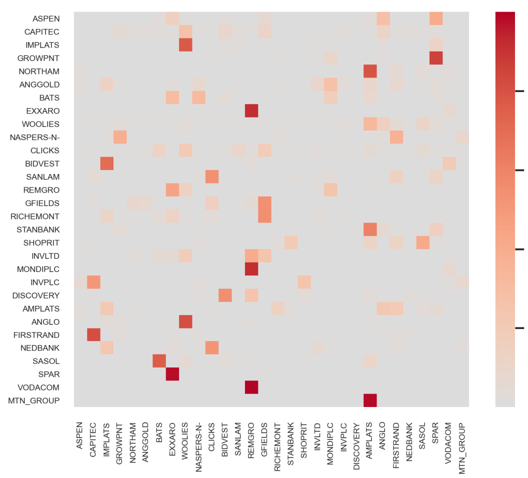

Fig. 1. GWN adaptive adjacency matrix

are no identifiable commonalities between the two matrices, nor is one matrix a more fine-grained representation of the extracted dependencies. In contrast to GWN and MTGNN, StemGNN appears to elevate intra- rather than inter-node correlations. StemGNN learns only trivial positive correlations (self-correlation), and extracts weak negative correlations between other pairs.

Node colour is a gradient scale of node degree, whilst node size illustrates betweenness centrality. Figure 2 illustrates that there are multiple nodes with a relatively high degree which is indicative of salient stocks. For GWN, there are fewer highly connected stocks as expected, with the analysis indicating that Anglo American Platinum, Capitec, and AngloGold Ashanti are the dominant stocks. Furthermore, the GWN graph contains a few influential nodes, with Anglo American Platinum, Capitec, and AngloGold Ashanti providing maximal

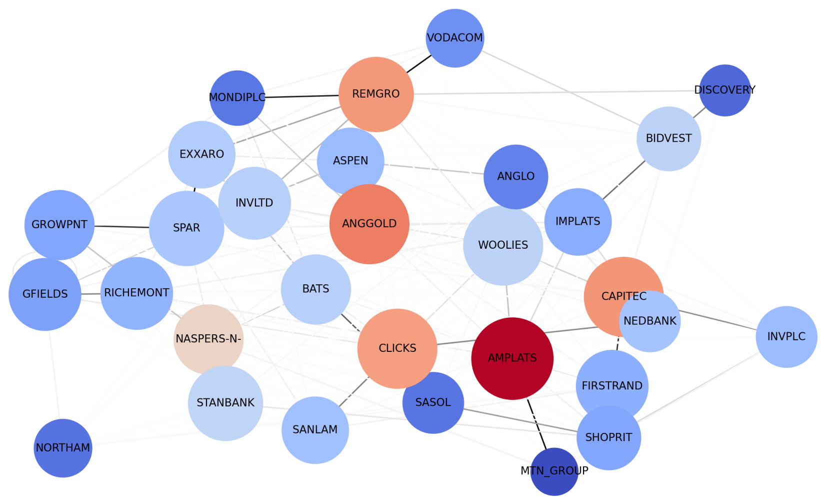

Fig. 2. GWN adaptive graph

influence on other graph nodes through strategic placement. This aligns with the results of bi-clustering (Figure 3 in the appendix) performed on the correlation matrix, which finds two compact clusters of highly correlated stocks.

The correlation network in Figure 5 illustrates that Discovery, Remgro and AngloGold Ashanti are both the most influential and salient stocks in the network. There are several nodes on the edge of the network that are not influential nor important.

5 Discussion and Conclusions

Graph WaveNet [19], MTGNN [18] and StemGNN [1] are spatial-temporal GNN architectures that are able to adaptively learn spatial dependencies during training without requiring prior knowledge. While these techniques have been shown to outperform other methods for traffic flow prediction and other applications in the energy and health domains, it is unclear whether they can achieve similar performance for stock market prediction. This study evaluated and compared GWN, MTGNN and StemGNN for single-step and multi-step forecasting tasks on the JSE. For single-step forecasting, MTGNN outperformed both GWN and StemGNN, achieving a significant increase in predictive accuracy across varying windows. However, MTGNN yields very poor performance for multi-step forecasting and substantially poorer performance to GWN. This differs from the findings of the original study [18] which showed better performance than GWN on two traffic flow benchmark data sets.

Our results show that while MTGNN is able to accurately predict share prices in the next step, it is not suitable for forecasting a sequence of daily closing prices. MTGNN's performance may be attributable to its curriculum learning strategy. Curriculum learning initially trains the algorithm on a single-step forecasting task, predicting the next point and then iterates through this process to predict subsequent points. While this may be well suited to spatial-temporal flow problems where the next point can be a suitable starting point for identifying a trajectory, inter-share dependencies in the stock market have more complex relationships which can pan out in an erratic fashion over multiple time steps. For multi-step forecasting tasks, GWN is the clear winner. It achieved the highest predictive accuracy over all the tested window sizes and horizon lengths. The size of the input window had a substantial impact on performance. All techniques produced the best performance the best MAPE scores over a 40-day input window. This corresponds to the previous two-months daily closing prices. This may be the optimal input sequence length for latent inter-share temporal effects to unfold within the JSE. However, GWN with a 20-day input window and 20-day prediction horizon produced the highest performance overall, marginally higher than GWN with a 40-day input window and 5-day horizon. This is an interesting phenomenon and requires further exploration.

Even though StemGNN yielded good results compared to MTGNN, MAPE scores were approximately 10% lower than those reported on the electricity, ECG and solar datasets and in almost all cases it was outperformed by GWN. Similarly, comparing the best GWN performance score to the metrics reported for traffic network forecasting, we find that the approximately 10% decrease in MAPE is realistic given the dynamic and erratic intra- and inter-share dependencies exhibited within a stock market and the inherent complexity in extracting and representing these dependencies.

Our results demonstrate that an ST-GNN model, specifically GWN, has high potential for predicting future daily closing price values over short- and medium-term prediction horizons, but this requires further investigation. Furthermore, this study also showed that some ST-GNN approaches have broader applicability to a wider range of complex and dynamic systems, especially systems where there is no explicit spatial property. In this study, the graph is used to represent and learn the internal dynamics of the Johannesburg Stock Exchange. The nodes are used to capture intra-share dependencies, while the graph structure (links) capture inter-share dynamics. The extracted GWN adaptive graph produces an approximate representation of inter-share dependencies in the JSE Top 40. This work provides a foundation for future investigation of ST-GNNs for real-world applications of share price forecasting.

Based on the results of the correlation matrix experiment, we found that the *a priori* inclusion of multivariate correlations produced a negligible impact on model predictive accuracy. While GWN appears to have performed well without prior structural information, this requires further exploration, e.g. the sector and industry groupings for each share may contain important structural information.

6 Limitations and Future Work

The results of this work present several opportunities for further investigation. The selected models were evaluated on a single dataset. Obtaining empirical results on data from different stock exchanges allows for a broader assessment of model generalisability and identifying potential commonalities and discrepancies in market dynamics. The methodology was restricted to evaluating ST-GNNs using only their original hyperparameter configurations. Future research can include hyperparameter tuning for stock market data which may yield higher predictive performance. In addition, testing prior structural information sources other than a correlation matrix, or feature combinations thereof, presents an avenue for future study. The hypergraph proposed by Sawhney et al. [12] is a promising starting point.

7 Acknowledgements

This research is wholly funded by the National Research Foundation (Grant Number MND200411512622).

References

1. Cao, D., Wang, Y., Duan, J., Zhang, C., Zhu, X., Huang, C., Tong, Y., Xu, B., Bai, J., Tong, J., et al.: Spectral temporal graph neural network for multivariate time-series forecasting (2021)
2. Chen, K., Zhou, Y., Dai, F.: A lstm-based method for stock returns prediction: A case study of china stock market. In: 2015 IEEE International Conference on Big Data (Big Data). pp. 2823–2824. IEEE (2015)
3. Chen, L., Qiao, Z., Wang, M., Wang, C., Du, R., Stanley, H.E.: Which artificial intelligence algorithm better predicts the chinese stock market? IEEE Access **6**, 48625–48633 (2018)
4. Cheng, H., Tan, P.N., Gao, J., Scripps, J.: Multistep-ahead time series prediction. In: Pacific-Asia Conference on Knowledge Discovery and Data Mining. pp. 765–774. Springer (2006)
5. Fawaz, H.I., Forestier, G., Weber, J., Idoumghar, L., Muller, P.A.: Deep learning for time series classification: a review. Data Mining and Knowledge Discovery **33**(4), 917–963 (2019)
6. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 770–778 (2016)
7. Hiransha, M., Gopalakrishnan, E.A., Menon, V.K., Soman, K.: Nse stock market prediction using deep-learning models. Procedia Computer Science **132**, 1351–1362 (2018)
8. Li, W., Bao, R., Harimoto, K., Chen, D., Xu, J., Su, Q.: Modeling the stock relation with graph network for overnight stock movement prediction. In: Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20. pp. 4541–4547. International Joint Conferences on Artificial Intelligence Organization (7 2020)

9. Ma, Y., Han, R., Wang, W.: Portfolio optimization with return prediction using deep learning and machine learning. Expert Systems with Applications **165**, 113973 (2021)
10. Matsunaga, D., Suzumura, T., Takahashi, T.: Exploring graph neural networks for stock market predictions with rolling window analysis (2019)
11. Oord, A.v.d., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., Kalchbrenner, N., Senior, A., Kavukcuoglu, K.: Wavenet: A generative model for raw audio (2016)
12. Sawhney, R., Agarwal, S., Wadhwa, A., Derr, T., Shah, R.R.: Stock selection via spatiotemporal hypergraph attention network: A learning to rank approach. Proceedings of the AAAI Conference on Artificial Intelligence **35**(1), 497–504 (May 2021), https://ojs.aaai.org/index.php/AAAI/article/view/16127
13. Sezer, O.B., Gudelek, M.U., Ozbayoglu, A.M.: Financial time series forecasting with deep learning: A systematic literature review: 2005–2019. Applied Soft Computing **90**, 106181 (2020)
14. Song, C., Lin, Y., Guo, S., Wan, H.: Spatial-temporal synchronous graph convolutional networks: A new framework for spatial-temporal network data forecasting. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 34, pp. 914–921 (2020)
15. Ta, V.D., Liu, C.M., Tadesse, D.A.: Portfolio optimization-based stock prediction using long-short term memory network in quantitative trading. Applied Sciences **10**(2), 437 (2020)
16. Tealab, A.: Time series forecasting using artificial neural networks methodologies: A systematic review. Future Computing and Informatics Journal **3**(2), 334–340 (2018)
17. Wang, W., Li, W., Zhang, N., Liu, K.: Portfolio formation with preselection using deep learning from long-term financial data. Expert Systems with Applications **143**, 113042 (2020)
18. Wu, Z., Pan, S., Long, G., Jiang, J., Chang, X., Zhang, C.: Connecting the dots: Multivariate time series forecasting with graph neural networks. In: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. pp. 753–763 (2020)
19. Wu, Z., Pan, S., Long, G., Jiang, J., Zhang, C.: Graph wavenet for deep spatial-temporal graph modeling. In: Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19. pp. 1907–1913. International Joint Conferences on Artificial Intelligence Organization (7 2019)
20. Xu, K., Hu, W., Leskovec, J., Jegelka, S.: How powerful are graph neural networks? In: 7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019. OpenReview.net (2019)
21. Yu, B., Yin, H., Zhu, Z.: Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting. In: Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18. pp. 3634–3640. International Joint Conferences on Artificial Intelligence Organization (7 2018)
22. Zhou, J., Cui, G., Hu, S., Zhang, Z., Yang, C., Liu, Z., Wang, L., Li, C., Sun, M.: Graph neural networks: A review of methods and applications. AI Open **1**, 57–81 (2020)

A Figures

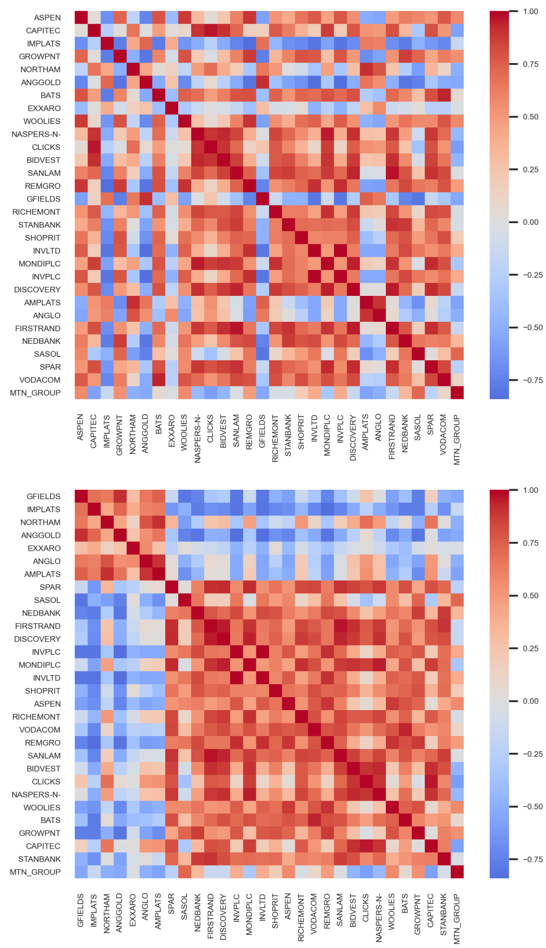

Fig. 3. FTSE/JSE Top 40 Index correlation and bi-clustered matrices

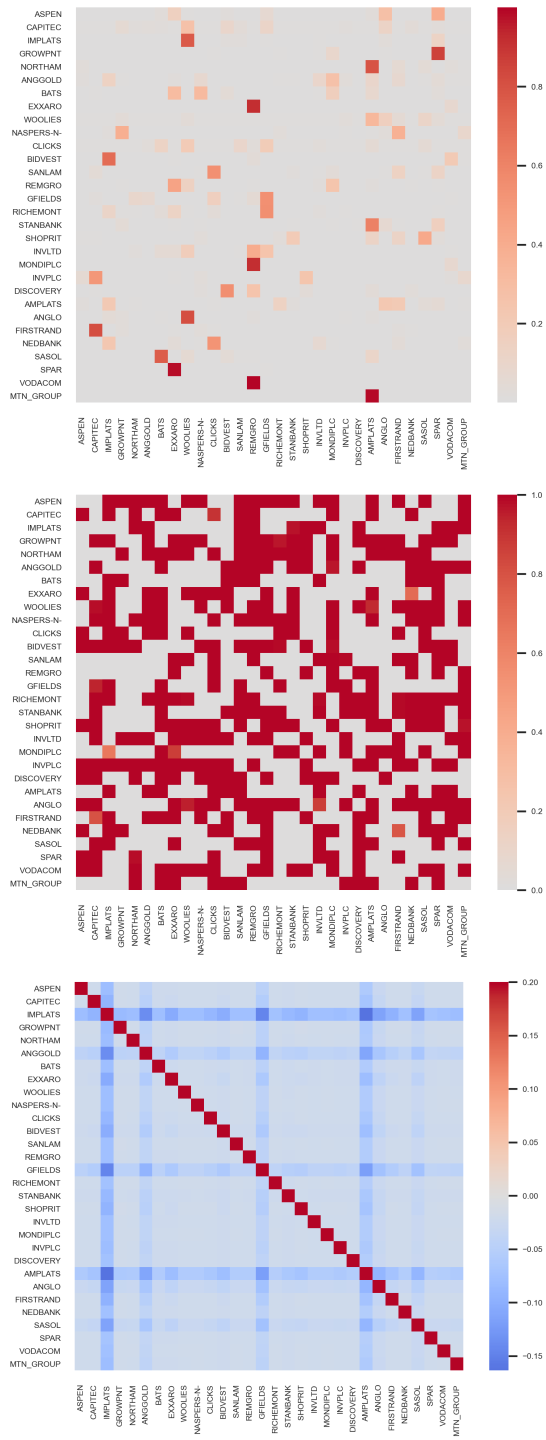

Fig. 4. GWN, MTGNN and StemGNN adaptive adjacency matrices

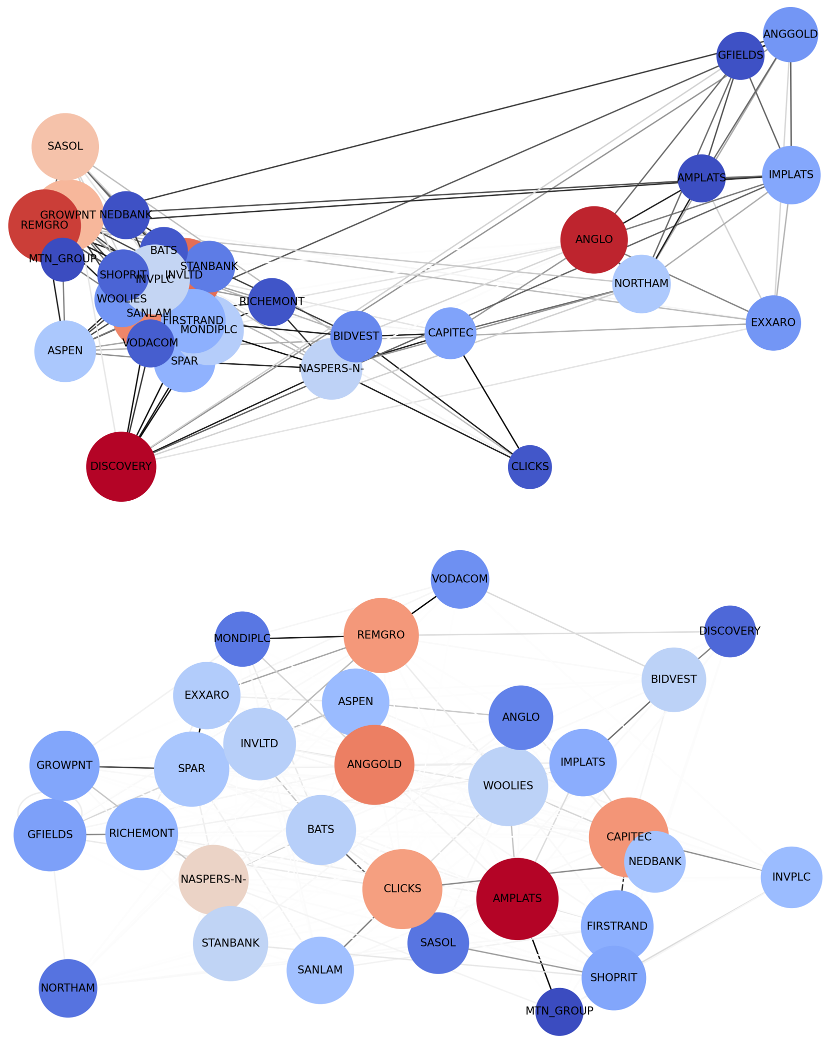

Fig. 5. FTSE/JSE Top 40 Index correlation and GWN adaptive graphs