3-2024

Temporal implicit multimodal networks for investment and risk management

Meng Kiat Gary ANG
Singapore Management University, gary.ang.2019@phdcs.smu.edu.sg

Ee-peng LIM
Singapore Management University, eplim@smu.edu.sg

Citation

ANG, Meng Kiat Gary and LIM, Ee-peng. Temporal implicit multimodal networks for investment and risk management. (2024). *ACM Transactions on Intelligent Systems and Technology*. 15, (2), 1-25.
Available at: https://ink.library.smu.edu.sg/sis_research/8745

Temporal Implicit Multimodal Networks for Investment and Risk Management

GARY ANG and EE-PENG LIM, Singapore Management University, Singapore, Singapore

Many deep learning works on financial time-series forecasting focus on predicting future prices/returns of individual assets with numerical price-related information for trading, and hence propose models designed for univariate, single-task, and/or unimodal settings. Forecasting for investment and risk management involves multiple tasks in multivariate settings: forecasts of expected returns and risks of assets in portfolios, and correlations between these assets. As different sources/types of time-series influence future returns, risks, and correlations of assets in different ways, it is also important to capture time-series from different modalities. Hence, this article addresses financial time-series forecasting for investment and risk management in a multivariate, multitask, and multimodal setting. Financial time-series forecasting, however, is challenging due to the low signal-to-noise ratios typical in financial time-series, and as intra-series and inter-series relationships of assets evolve across time. To address these challenges, our proposed Temporal Implicit Multimodal Network (TIME) model learns implicit inter-series relationship networks between assets from multimodal financial time-series at multiple time-steps adaptively. TIME then uses dynamic network and temporal encoding modules to jointly capture such evolving relationships, multimodal financial time-series, and temporal representations. Our experiments show that TIME outperforms other state-of-the-art models on multiple forecasting tasks and investment and risk management applications.

CCS Concepts: • **Computing methodologies** → **Neural networks**; **Artificial intelligence**; **Knowledge representation and reasoning**; **Information extraction;**

Additional Key Words and Phrases: Time-series, forecasting, graphs, graph neural networks, finance, multi-modality

ACM Reference Format:
Gary Ang and Ee-Peng Lim. 2024. Temporal Implicit Multimodal Networks for Investment and Risk Management. *ACM Trans. Intell. Syst. Technol.* 15, 2, Article 38 (March 2024), 25 pages. https://doi.org/10.1145/3643855

1 INTRODUCTION

Prior works on financial time-series forecasting with deep learning methods [19, 32, 64] mostly focus on predicting asset (e.g., stock) prices or returns at a future time-step based on time-series information from a single modality (usually numerical price-related information) to support

This research is supported by the National Research Foundation, Singapore under its Strategic Capabilities Research Centres Funding Initiative. Gary Ang is supported by a Monetary Authority of Singapore Postgraduate Scholarship. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not reflect the views of National Research Foundation, Singapore, nor the Monetary Authority of Singapore.

Authors' addresses: G. Ang and E.-Peng Lim, Singapore Management University, 80 Stamford Road Singapore, 178902, Singapore; e-mails: gary.ang.2019@phdcs.smu.edu.sg, eplim@smu.edu.sg.

ACM 2157-6904/2024/03-ART38
https://doi.org/10.1145/3643855

trading decisions and applications. The models proposed in these works are usually designed for *univariate, single task, and/or uni-modal settings*.

Investment and risk managers make *investment and risk management* decisions over time horizons for portfolios comprising multiple assets. To support such decisions, there is a need to forecast expected *returns* and *risks* (volatilities) of *multiple assets* in portfolios, as well as *correlations* between these assets over a selected time horizon. Capturing the interactions between expected returns, risks, and correlations of multiple assets in portfolios in a *multitask* setting is important as these target variables may evolve in a related manner. For example, risk (volatility) in equity asset prices is known to be asymmetric as volatility tends to be higher when returns decline than when returns rise [9]. Correlations between different assets may also increase during periods of high volatility and steep declines in returns and come back down when volatility is low and returns are stable or rising [34].

As multiple sources and types of information can influence future returns, risks, and correlations of multiple assets in a future time horizon in different ways, there is a need to capture information from *multiple modalities*, e.g., numerical price-related information and textual media information [39]. Therefore, financial forecasting for investment and risk management naturally involves a *multivariate, multitask, and multimodal setting*—forecasting expected returns and risks of multiple assets, and correlations between these assets in portfolios over a future time horizon, based on information from multiple modalities. Such forecasts are also necessary for important investment and risk management applications: portfolio allocation optimization [52] and *Value-at-Risk (VaR)* [45] forecasting. Aside from supporting investment and risk management decisions and applications, capturing information from multiple variables (corresponding to different assets) and modalities across multiple tasks may also improve forecasting performance as it enables the forecasting model to leverage information across different variables, modalities, and tasks and prevents overfitting on any one variable, modality, or task.

For either univariate-unimodal-single task or multivariate-multimodal-multitask settings, financial time-series forecasting is challenging due to the inherently low signal-to-noise ratios and the non-stationary nature of financial time-series distributions and their relationships [26]. Figure 1 illustrates the non-stationary nature of such inter-series relationships between multiple assets over time. We see that the inter-series relationships between assets over the window period $[0,3t]$ (see Figure 1(b)) differ from inter-series relationships in three different sub-window periods (see Figures 1(c)–1(e)), $[0,t]$, $[t,2t]$, and $[2t,3t]$, highlighting the importance of modeling evolving inter-series relationships. In this article, we address investment and risk management requirements and the challenges in financial time-series forecasting by designing a model that can (1) *be used in multivariate, multitask, and multimodal settings, which can enable complementary signals from different variables, tasks, and modalities to be used to improve overall forecasting performance*, and (2) *adaptively capture both evolving intra-series patterns and inter-series relationships to address the non-stationary nature of financial time-series information.*

Different classical methods [10, 21, 75] have been used to forecast financial returns, risks of financial returns, and financial correlations. These methods, however, are not designed for multitask settings and cannot capture information from multiple modalities, particularly unstructured information such as text. Classical models also typically adopt fixed distributional assumptions based on domain knowledge, which may not be suitable for modeling evolving intra-series patterns and inter-series relationships. Various deep learning architectures, such as convolutional and recurrent neural networks and transformers, have been applied to time-series forecasting [23, 33, 43, 57, 61, 74]. However, most of these models are designed for univariate, unimodal settings and/or single forecasting tasks. They also do not model evolving inter-series relationships in a multivariate setting. The aforementioned works on models for time-series forecasting focus on

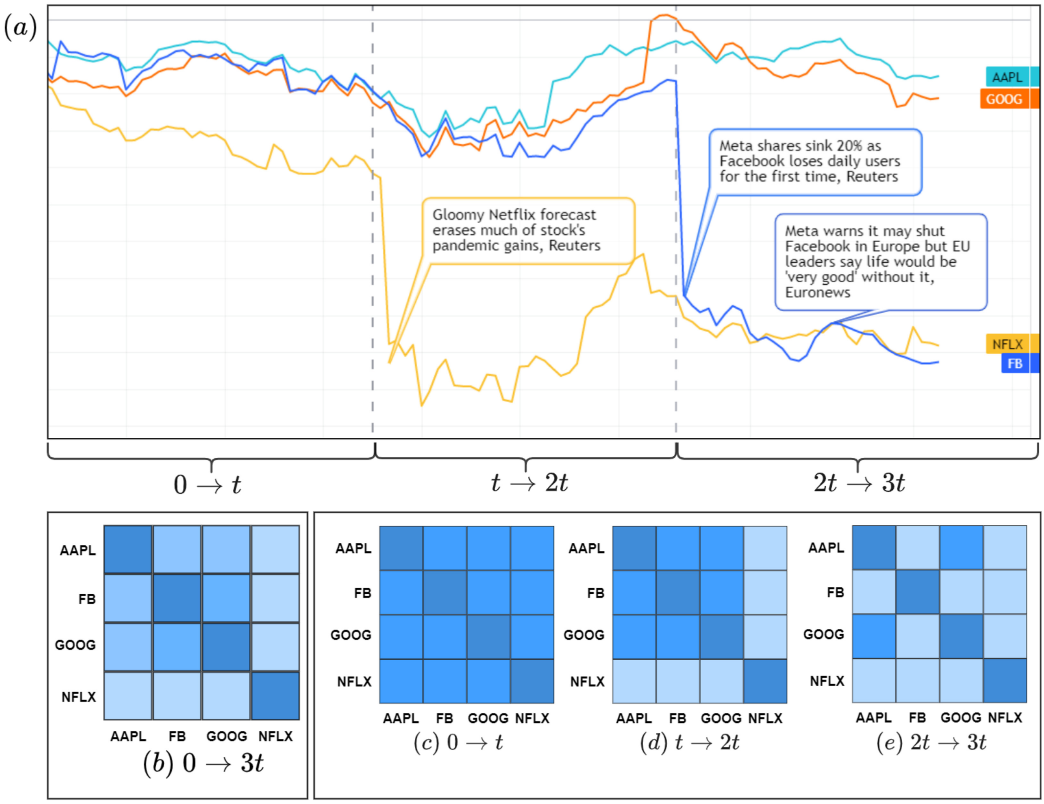

Fig. 1. Non-stationary inter-series relationships/correlations between assets. (a) shows the evolution of stock prices for Facebook (FB), Apple (AAPL), Netflix (NFLX), and Google (GOOG) over a window period, along with key news events that could potentially affect the returns, volatilities, and correlations of stock prices. Inter-series relationships/correlations over the whole window period shown in (b) differ from that in the three sub-window periods shown in (c), (d), and (e), illustrating the need to capture such evolving inter-series relationships.

the batch learning setting, which assume that training data are available *a priori* [63]. There have also been works on models for time-series forecasting that focus on the online learning setting, designed for time-series data that arrive in a stream over time [2, 46, 49, 62, 63], but such works generally do not focus on modeling evolving inter-series relationships in a multivariate setting. Spatio-temporal network models [15, 41, 89, 93] capture relationships between different time series but require explicit spatial relationship networks as input and assume that such networks are static. Dynamic network models [29, 30, 59, 84, 85] can be used for networks that evolve across time but similarly require explicit networks as inputs. Recently, spatio-temporal network models that infer implicit networks have been proposed [5, 12, 16, 38, 82, 96], but they do not infer implicit networks at multiple time-steps to address the non-stationary nature of inter-series relationships. Such spatio-temporal and dynamic network models are also not designed for networks where nodes have multimodal financial time-series as attributes. Some works adopt a multimodal multitask approach to forecast financial time-series for trading [67, 87]. However, such works do not address investment and risk management requirements as they either forecast the same variable over different time horizons, i.e., homogeneous forecasting tasks, or only forecast prices and volatilities. Furthermore, these works do not adaptively capture both evolving intra-series patterns and inter-series relationships.

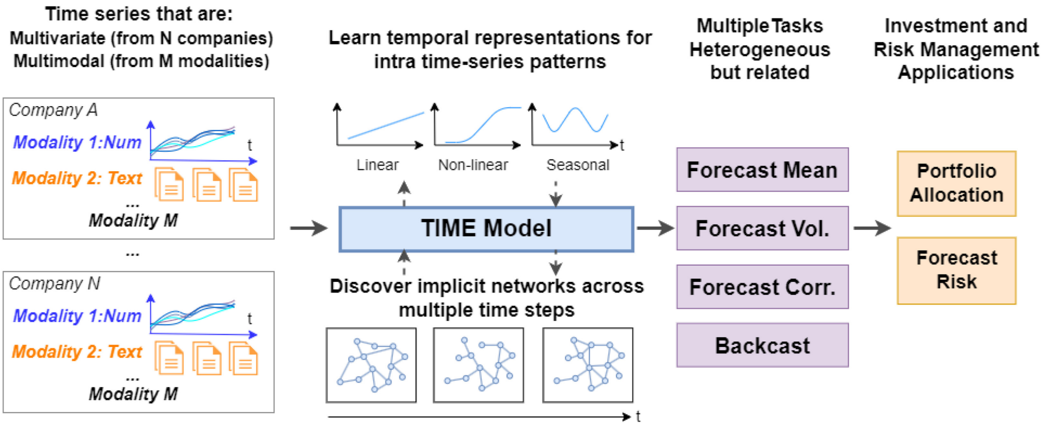
Time series that are:
Multivariate (from N companies)
Multimodal (from M modalities)
Company A
Modality 1:Num
Modality 2: Text
...
Modality M
...
Company N
Modality 1:Num
Modality 2: Text
...
Modality M
t
Learn temporal representations for intra time-series patterns
Linear
Non-linear
Seasonal
TIME Model
Discover implicit networks across multiple time steps
Multiple Tasks Heterogeneous but related
Forecast Mean
Forecast Vol.
Forecast Corr.
Backcast
Investment and Risk Management Applications
Portfolio Allocation
Forecast Risk

Fig. 2. Framework for multivariate, multitask, and multimodal setting: The proposed TIME model is designed to capture information from multiple variables (N companies) and modalities (M sources), e.g., information from numerical price-related and textual news modalities, as shown on the left-hand side of the figure. To adaptively address the evolving temporal characteristics of time-series information and the non-stationary nature of inter-series relationships, the TIME model learns temporal representations for intra-time-series patterns and discovers underlying implicit networks across multiple time-steps for different modalities. The heterogeneous multitask approach of the TIME model generates different types of forecasts: company stock (i) mean returns, (ii) risks (volatilities), and inter-company (iii) correlations over a future horizon to support investment and risk management decision-making, as well as industry applications, as shown on the right-hand side of the figure.

Hence, in this article, we propose the *Temporal Implicit Multimodal Network (TIME)* model framework, as shown in Figure 2. TIME uses multivariate time-series information from different modalities to adaptively discover dynamic implicit relationship networks at different time- steps. It then uses both the multivariate time-series information from different modalities and the discovered implicit networks as input for heterogeneous but related forecasting tasks. These tasks include forecasts of the means, volatilities, and correlations of returns of multiple assets. Modeling implicit relationship networks well in a multivariate setting not only supports the correlation forecasting task but also improves other forecasting tasks across time-series as such relationships influence the evolution of individual time-series. Beyond investment and risk management decisions, forecasts of means, volatilities, and correlations over a future horizon can be used for important industry applications such as portfolio allocation (deciding how to allocate capital between different investment assets) and portfolio risk management (forecasting portfolio VaR) [45]. Our key contributions are as follows:

- To our knowledge, this is the first work to propose the adaptive discovery of *implicit inter-series relationship networks* from multivariate and multimodal financial time-series for multiple financial forecasting tasks to support investment and risk management.
- We design an attention-based module that adaptively discovers implicit relationship networks at multiple time-steps from multimodal financial time-series, which we also leverage to forecast inter-series correlations over a selected future time horizon, and propose the use of a *temporal vectorization module* with multiple functional forms to adaptively capture temporal patterns that are utilized for time-sensitive dot-product attention sequential encoding.
- We train the model on multiple related tasks to leverage complementary information for improving overall forecasting performance and lowering the risk of over-fitting.

- We design a *generalizable model* that can be applied to time-series information from different numbers and types of modalities and generate representations in a shared embedding space.
- We demonstrate the effectiveness of TIME on multiple forecasting tasks against state-of-the-art baselines on real-world datasets. We also show that TIME out-performs these baselines on real-world investment and risk management applications and interpret the implicit relationship networks learned by TIME.

2 RELATED WORK

As this work involves financial time-series forecasting and network learning, we review key related works in these areas.

2.1 Financial Time-series Forecasting

ARIMA [75] is a well-studied classical method for time-series forecasting. It assumes a specific structure for the mean of the underlying stochastic process and is commonly used to forecast financial returns. To forecast risks (i.e., volatility of financial returns), *Generalized AutoRegressive Conditional Heteroskedastic (GARCH)* [10] assumes a specific structure for the variance of the underlying stochastic process. Multivariate versions of these models, e.g., VAR [50], which extends AR, and DCC-GARCH [21], which extends GARCH, have been proposed. In general, classical methods adopt fixed distributional assumptions, which may not always be suitable for evolving intra-series patterns and inter-series relationships. These methods are also not designed for multitask settings, nor information from multiple modalities, particularly unstructured information such as text. To learn temporal patterns in a data-driven manner, deep learning models have been applied to time-series forecasting. They include feed-forward networks [14, 17, 19, 56, 88], convolutional neural networks [6, 11, 58, 78], recurrent neural networks [25, 44, 48, 64, 65], and transformers [42, 55, 73, 79, 80, 90, 91, 95]. A detailed review of these works can be found in [23, 33, 43, 57, 61, 74]. Deep learning models can be designed for univariate or multivariate settings. Models designed for univariate settings [13, 56, 60, 66, 92] model each time-series independently, while multivariate models [25, 35, 40, 41, 65, 69, 83, 89] learn multiple time-series together. Most of these models capture numerical information but not unstructured textual information or information from multiple modalities. [40] captures information from two numeric modalities (media sentiment and price-related data) for financial forecasting. **NBEATS** [56] is an example of a univariate time-series forecasting model designed for numerical time-series information that demonstrated good performance when benchmarked against top classical, deep learning, and classical–deep learning hybrid models from the M4 Competition [51]. NBEATS comprises stacks of fully connected layers with residual connections that can be constrained to decompose forecasts into specific time-series patterns. *Dual-stage attention-based recurrent neural network (DARNN)* [64] is designed for numerical time-series information in a multivariate setting and has shown good performance on financial time-series forecasting. DARNN applies an input attention stage, followed by a temporal attention stage. *Time-series Transformer (TST)* [91] is a recent model based on the transformer encoder architecture designed for numerical time-series information that can be utilized in either univariate or multivariate settings. However, DARNN and most other multivariate models [25, 41, 65, 69, 83, 89] do not address the non-stationary nature of inter-series relationships and are not designed to capture multimodal information. Recent works have studied the use of textual news information [3, 19, 20, 32, 39, 68, 70] for financial forecasting. **FAST** [68] uses time-aware LSTMs [8] to encode textual news information, while *StockEmbed (SE)* [20] uses bidirectional GRUs to encode textual news information. Neither FAST nor SE captures multimodal information or evolving inter-series relationships. As financial time-series forecasting models can be used for quantitative trading applications, an orthogonal but important field is the application of reinforcement learning for quantitative trading [1, 47, 71, 72]. A number of such works utilize neural networks as function approximators [72], which can include neural networks designed for time-series forecasting. In this article, we focus on the more general setting of evaluating TIME for investment and risk management applications, rather than the typical end-to-end setting for reinforcement learning, which involves the use of neural networks within reinforcement learning methods.

2.2 Network Learning for Financial Time-series

Graph neural networks (GNNs) compose messages based on network features and propagate them to update the embeddings of nodes and/or edges over multiple neural network layers [7, 27]. Several GNN-based models have been developed. In particular, Graph Convolutional Network [37] aggregates features of neighboring nodes and normalizes the aggregated representations by the node degrees. GraphSAGE [31] considers mean, LSTM, or pooling aggregation methods and samples a fixed number of neighbors for representation aggregation. Graph Attention Network [77] assigns neighboring nodes with different importance weights during aggregation. Such GNNs are designed for static networks with static node attributes and cannot be applied to networks where the attributes are time-series. They also cannot be used for dynamic networks. A few recent works [3, 4, 24, 53] apply GNNs to prediction tasks on financial time-series data, but they are designed for pre-defined static networks. A number of GNN models have been designed for dynamic networks [29, 30, 59, 84, 85], e.g., by encoding network snapshots and applying a recurrent neural network to the sequence of network snapshot representations. However, they are not designed for networks where the attributes are financial time-series that evolve alongside dynamic networks. They also require network snapshots to be explicitly given as input. Spatio-temporal network models [15, 18, 41, 89, 93, 94], primarily used for traffic forecasting, can handle networks where the node attributes are time-series, e.g., traffic flows at road junctions, but are designed for pre-defined static spatial networks, and hence not able to model evolving inter-series relationships. Some recent spatio-temporal network models [5, 12, 16, 38, 82, 96] infer relationships between time-series for forecasting. **MTGNN** [82] uses a graph learning layer to learn the underlying network, before applying interleaved temporal convolution modules and graph convolution modules to capture temporal and spatial dependencies. However, MTGNN and these other works assume that a single set of relationships applies across the window period, and they are also not designed for multimodal settings. They are also not designed to capture intra-series temporal patterns of financial time-series.

In general, existing methods in financial time-series forecasting models often rely on fixed distributional assumptions, which may not hold due to dynamic market conditions and shifts in intra-series patterns. While deep learning time-series forecasting models can address this issue, most of these models are not designed for multitask or multimodal settings, limiting their utility in the financial domain (as well as other real-world domains), which involves various types of structured and unstructured data. While there have been more recent works in time-series forecasting that focus on multimodal settings, such works are not designed for network information. In the field of network learning, most models are designed for static networks and static attributes. Most dynamic network models require pre-defined network snapshots and do not capture the effects of time-series attributes on network relationships. Recent network learning models that infer and learn network relationships do not capture evolving relationships across time, which is crucial in many domains.

3 TEMPORAL IMPLICIT MULTIMODAL NETWORK MODEL

We formulate the problem as multiple multivariate financial time-series forecasting tasks on dynamic implicit networks. The dynamic implicit networks are sequences of inter-series implicit relationship networks discovered by the proposed TIME model, where the attributes of the asset nodes in the networks are multimodal financial time-series. We denote $X_t^m = [x^m(t-K), \ldots, x^m(t-1)]$, where $X_t^m \in \mathbb{R}^{|V| \times K \times d^m}$, as a sequence of financial time-series information from modality m (out of M different modalities), which could be of numerical, textual, or other type, of dimension d^m over a window of K time-steps for a set of assets V. In the **dynamic implicit network learning and encoding** step for each modality, we first discover temporal sequences of inter-series implicit relationship networks and apply a sparsification step, which enables information propagation between different nodes to be based on the most important implicit relationships and serves as a regularization step to prevent overfitting. This results in inter-series implicit relationship networks $G_t^m = [g^m(t-K), \ldots, g^m(t-1)]$ for each modality m, where $G_t^m \in \mathbb{R}^{|V| \times |V| \times K}$. For each modality m, $g^m(t-k)$ denotes a network $(V, e^m(t-k), h^m(t-k), a^m(t-k))$, where $e^m(t-k)$ represents the most important relational edges between assets discovered at time-step $t-k$; $h^m(t-k)$ $(\in \mathbb{R}^{|V| \times d})$ represents the encoded representations of the assets' time-series information at time-step $t-k$; and $a^m(t-k)$ $(\in \mathbb{R}^{|V| \times |V|})$ represents the weights of the edges $e^m(t-k)$ at time-step $t-k$. Thereafter, for each time-step $t-k$ in the window, TIME uses the encoded representations $h^m(t-k)$ of the assets' time-series information and the discovered network $g^m(t-k)$ as inputs to a dynamic graph convolution step to generate the assets' network representations $\tilde{h}^m(t-k)$ of dimension d. The dynamic graph convolution step is undertaken for each modality separately as the implicit relationships between asset nodes may differ for information from different modalities.

In the **temporal encoding** step, the sequence of network representations $\tilde{H}_t^m = [\tilde{h}^m(t-K), \ldots, \tilde{h}^m(t-1)]$ $(\tilde{H}_t^m \in \mathbb{R}^{|V| \times K \times d})$ are combined with the temporal representations $P_t = [p(t-K), \ldots, p(t-1)]$ $(P_t \in \mathbb{R}^{|V| \times K \times d})$ learned by a *time vectorization module* from the corresponding timestamps $T_t \in \mathbb{R}^{|V| \times K \times d^{time}}$ in the window based on the timestamps' day, day of week, and week of year, and projected to the same dimension d as the assets' network representations. The temporal representations P_t capture time-series patterns such as linear and non-linear trends and periodicity, which enable the subsequent time-sensitive attention-based sequential encoding of the sequence of network representations, resulting in $Z_t^m = [z^m(t-K), \ldots, z^m(t-1)]$ $(Z_t^m \in \mathbb{R}^{|V| \times K \times d})$. Similarly, the temporal encoding step is undertaken for each modality separately as the time-series patterns may differ for information from different modalities. TIME then uses attention mechanisms in the late-stage **multimodal fusion** module to fuse the resultant representations for each of the modalities m based on learned importances. After fusing representations across M modalities, we obtain $Z_t = [z(t-K), \ldots, z(t-1)]$ $(Z_t \in \mathbb{R}^{|V| \times K \times d})$. The last hidden state $z(t-1)$ is used to generate the backcast of the financial price-related input data and forecasts of the means, volatilities, and correlations of financial returns over a selected horizon of L time-steps, i.e., means, volatilities, and correlations of $Y_t^{returns} = [y^{returns}(t), \ldots, y^{returns}(t+L)]$, where $y^{returns}(t) = (price(t) - price(t-1))/price(t-1)$ are the percentage returns, and $price(t)$ is the stock price at time-step t. Figure 3 provides an overview of the architecture of TIME. We elaborate on the steps and modules below.

3.1 Temporal Implicit Network Learning

The temporal implicit network learning module discovers implicit relationship networks using the dot-product attention mechanism [76]. Unlike MTGNN [82] and other related works [96], which return a single network for the entire window period of length K, our network discovery module returns multiple implicit relationship networks $g(t-k)$s, one for each time-step $t-k$. Discovering multiple implicit networks is important as it allows TIME to model the non-stationary nature of evolving inter-series relationships. We first encode the sequence of financial time-series information from modality m: X_t^m with a modality-specific Gated Recurrent Unit ($SeqEnc^m$) to obtain the hidden representations $H_t^m \in \mathbb{R}^{|V| \times K \times d}$. We then apply shared linear layers to generate queries

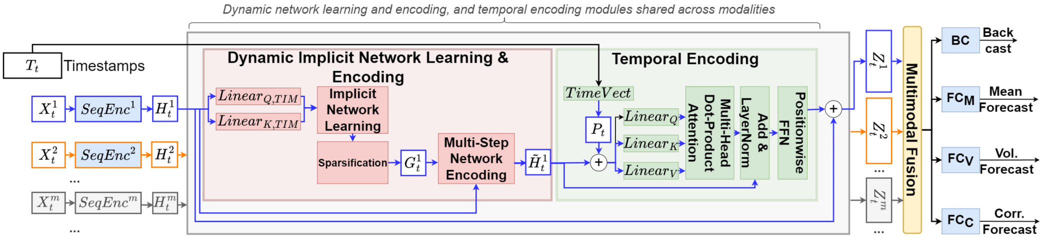

Fig. 3. Model architecture: The proposed model captures time-series features from M different modalities (X_t^m), say numerical or textual as shown in the figure, with their associated timestamps (T_t). We first encode features from a specific modality with a modality-specific sequence encoder ($SeqEnc^m$). The *dynamic implicit network learning and encoding* step then adaptively discovers implicit networks $G_t^m = [g^m(t-K), \ldots, g^m(t-1)]$ between multiple variables V at different time-steps from the encoded sequential representations to address the multivariate setting and the non-stationary nature of inter-series relationships. The *temporal encoding* step then adaptively learns temporal representations from T_t and uses self-attention to jointly encode the implicit network and feature representations in a time-sensitive manner. The *dynamic implicit network learning and encoding* and *temporal encoding* steps are repeated with *shared parameters* for each modality to generate modality-specific representations in a shared embedding space. The representations are then fused with attention mechanisms in the *multimodal fusion* module and used for multiple forecasting tasks.

Q_t^m and keys K_t^m from the hidden representations H_t^m:

$$Q_t^m = Linear_{Q-TIM}\left(H_t^m\right) \tag{1}$$

$$K_t^m = Linear_{K-TIM}\left(H_t^m\right). \tag{2}$$

A $|V| \times |V| \times K$ attention weight tensor AW_t^m can then be computed as the dot-product of Q_t^m and K_t^m. To allow richer inter-series interactions to be learned across time-steps [54], we further add a modality-specific learnable inner weight tensor $W^m \in \mathbb{R}^{K \times d \times d}$:

$$AW_t^m = tanh\left(\frac{Q_t^m \cdot W^m \cdot K_t^{m\intercal}}{\sqrt{d}}\right). \tag{3}$$

To emphasize the most important relationships at each time-step, we apply a *sparsification* step and take the top R relational edges with the highest AW_t^m for each time-step in $[t-K,t-1]$. In this article, we empirically set $R = 20\%$ in our experiments, which works well, and consider other R settings in our ablation study (see Section 5). We shall leave alternative sparsification methods to future work. We then obtain a sequence of sparse inter-series relationship networks, one for each time-step in the window $[t-K, t-1]$. Specifically, the inter-series relationship network is

$$g^m(t-k) = (V, e^m(t-k), h^m(t-k), a^m(t-k)) \tag{4}$$

for $k \in \{1, \ldots, K\}$, where:

- $e^m(t-k)$ represents the top-R (v_i, v_j) edges with the largest $AW_t^m[v_i, v_j, t-k]$ values at time-step $t-k$;
- $h^m(t-k)$ represents the assets encoded by $SeqEnc^m$, $h^m(t-k) = H_t^m[*, t-k]$; and
- $a^m(t-k)$ are either the AW_t^m of the top R edges at time-step $t-k$ or set to zero, i.e.,

$$\begin{aligned} a^m(t-k)[v_i, v_j] \\ = \begin{cases} AW_t^m[v_i, v_j, t-k] & \text{if } (v_i, v_j) \in e^m(t-k) \\ 0 & \text{otherwise.} \end{cases} \end{aligned} \tag{5}$$

We stack the sequence of $a^m(t-k)$'s in G_t^m to obtain the corresponding weighted adjacency tensor:

$$A_t^m \in \mathbb{R}^{|V|\times|V|\times K}, \tag{6}$$

with $A_{t,ij}^m \in \mathbb{R}^K$ representing the weighted relational edges between asset v_i and v_j across the window $[t-K, t-1]$ for modality m, i.e.,

$$A_{t,ij}^m = [a^m(t-K)[v_i, v_j], \dots, a^m(t-1)[v_i, v_j]]. \tag{7}$$

3.2 Dynamic Network Encoding

Next, we utilize the sequential encodings of the time-series information from $SeqEnc^m$, i.e., H_t^m, and the weighted adjacency tensor A_t^m as inputs to a weighted dynamic graph convolution step to generate the dynamic network representations of each of the assets. For an asset v_i, we compute its network representations $\tilde{H}_{t,i}^m \in \mathbb{R}^{K\times d}$ across time-steps in the $[t-K, t-1]$ window by aggregating representations from its neighbors $N_t(v_i)$ based on $A_{t,ij}^m$ as follows:

$$\tilde{H}_{t,i}^m = \sum_{v_j \in N_t(v_i)} \frac{exp(A_{t,ij}^m)}{\sum_{v_{j'} \in N_t(v_i)} exp(A_{t,ij'}^m)} \cdot H_{t,j}^m. \tag{8}$$

We denote the network representations for all assets by $\tilde{H}_t^m \in \mathbb{R}^{|V|\times K\times d}$. Other GNN variants can also be utilized for this graph convolution step, but we adopt this approach for computational efficiency as it allows us to apply the graph convolution on the H_t^m and weighted A_t^m tensors across multiple time-steps in parallel. Using other common GNNs did not yield any improvement in performance.

3.3 Temporal Encoding

Inspired by [28, 36], which proposed general frameworks for learning temporal representations, we introduce a time vectorizer ($TimeVect$) within TIME that is shared across the different modalities. $TimeVect$ takes as input the timestamps from the time-steps in $[t-K, t-1]$ and learns the temporal representations for them. The input timestamps are represented as $T_t \in \mathbb{R}^{|V|\times K\times d^{time}}$, where d^{time} denotes the number of dimensions required for capturing the day of week and week and month of the year of a timestamp. The temporal representations learned by $TimeVect$ are denoted by $P_t \in \mathbb{R}^{|V|\times K\times d}$. Functional forms are combined with learnable weights to adaptively learn and combine periodic and non-periodic components within the multivariate financial time-series. This could also be viewed as a time-sensitive version of positional encodings used in transformers that only deal with sequential positions of word tokens [76]. For each component, we apply linear layers and selected activation functions to T_t. For TIME, the empirically chosen components are $\Phi_1 = Linear(T_t)$; $\Phi_2 = cos(Linear(T_t))$; $\Phi_3 = Sigmoid(Linear(T_t))$; $\Phi_4 = Softplus(Linear(T_t))$, which enable the model to extract linear and non-linear trends, as well as seasonality-based temporal patterns. We then concatenate these components and project them:

$$P_t = Linear([\Phi_1||\Phi_2||\Phi_3||\Phi_4]). \tag{9}$$

In the subsequent transformer-based attention-based sequential encoding step [76], we add the learned temporal representations P_t to the dynamic network representations $\tilde{H}_t^m$ and then apply linear layers shared across different modalities to generate queries, keys, and values:

$$\tilde{Q}_t^m = Linear_Q\left(\tilde{H}_t^m + P_t\right) \tag{10}$$

$$\tilde{K}_t^m = Linear_K\left(\tilde{H}_t^m + P_t\right) \tag{11}$$

$$\tilde{V}_t^m = Linear_V\left(\tilde{H}_t^m + P_t\right). \tag{12}$$

We then apply scaled dot-product attention:

$$\tilde{H}_t'^m = softmax\left(\frac{\tilde{Q}_t^m \cdot \tilde{K}_t^{m\intercal}}{\sqrt{d}}\right)\tilde{V}_t^m, \tag{13}$$

followed by a residual connection with *layer normalization (LayerNorm)* and finally a *feed-forward network (FFN)* shared across different modalities:

$$Z_t^m = FFN(LayerNorm\left(\tilde{H}_t'^m + \tilde{H}_t^m\right). \tag{14}$$

The output of this step is hence

$$Z_t^m = [z^m(t-K), \ldots, z^m(t-1)] \in \mathbb{R}^{|V|\times K\times d}. \tag{15}$$

3.4 Multimodal Fusion

To learn the importance of different modalities, we use attention-based fusion to fuse Z_t^m across M modalities. A non-linear transformation is applied to the representations to obtain scalars

$$s_t^m = W^{(1)} tanh\left(W^{(0)} Z_t^m + b\right), \tag{16}$$

where $W^{(0)}$ and $W^{(1)}$ are learnable weight matrices and b is the non-modality-specific bias vector. Parameters are shared across modalities. We normalize the scalars with a softmax function to obtain the weights β_t^ms, which are used to fuse representations across modalities:

$$\beta_t^m = \frac{exp(s_t^m)}{\sum_{1\le m\le M} exp(s_t^m)} \tag{17}$$

$$Z_t = \sum_{1\le m\le M} \beta_t^m Z_t^m. \tag{18}$$

The output of this step is $Z_t = [z(t-K), \ldots, z(t-1)] \in \mathbb{R}^{|V|\times K\times d}$. We use the last hidden state in the sequence, i.e., $z(t-1)$, for the forecasting step.

3.5 Forecasting and Loss Functions

In the forecasting step, we use fully connected layers to generate the backcast of the numerical price-related input data (say, modality p) and forecasts of the means and volatilities of asset returns over the selected horizon period L:

$$\hat{X}_t^p = BC(z(t-1)) \tag{19}$$

$$\hat{Y}_{mean,t}^{returns} = FC_M(z(t-1)) \tag{20}$$

$$\hat{Y}_{vol,t}^{returns} = FC_V(z(t-1)). \tag{21}$$

TIME can backcast time-series information from multiple modalities. However, we backcast numerical price-related information but not textual information as the multiple tasks in this article focus on forecasts of numerical targets.

To forecast the correlations of asset returns over the horizon period L, we use the weights from the linear layers in the temporal implicit network learning module:

$$Q_{corr,t} = Linear_{Q-TIM}(z(t-1)) \tag{22}$$

$$K_{corr,t} = Linear_{K-TIM}(z(t-1)). \tag{23}$$

This allows what was learned when discovering the inter-series relationships to be leveraged for correlation forecasts:

$$\hat{Y}^{returns}_{corr,t} = FC_C \left(tanh \left(\frac{Q_{corr,t} \cdot K^{\intercal}_{corr,t}}{\sqrt{d}} \right) \right). \tag{24}$$

For $Y^{returns}_t = [y^{returns}(t), \dots, y^{returns}(t+L)]$ over a horizon of L time-steps, the ground-truth labels for means and volatilities are defined as follows:

$$Y^{returns}_{mean,t} = \frac{1}{L} \sum_{l=0}^{L} y^{returns}(t+l) \tag{25}$$

$$Y^{returns}_{vol,t} = \sqrt{\frac{1}{L} \sum_{l=0}^{L} (y^{returns}(t+l) - \mu)^2}, \tag{26}$$

where $\mu = Y^{returns}_{mean,t}$. For correlations between any two assets v_i and v_j:

$$Y^{returns}_{corr,t,ij} = \frac{\sum_{l=0}^{L}(x_i(t+l) - \mu_i)(x_j(t+l) - \mu_j)}{\sqrt{\sum_{l=0}^{L}(x_i(t+l) - \mu_i)^2}\sqrt{\sum_{l=0}^{L}(x_j(t+l) - \mu_j)^2}}, \tag{27}$$

where $x_i(t+l) = y^{returns}_i(t+l)$, $x_j(t+l) = y^{returns}_j(t+l)$. We compute the loss between the forecasts and the respective ground truths defined above with *root mean squared error (RMSE)* and use the total loss as the training objective:

$$\begin{aligned} \mathcal{L}_{total} = {} & \mathcal{L}_{backcast}\left(X^p_t, \hat{X}^p_t\right) + \mathcal{L}_{mean}\left(Y^{returns}_{mean,t}, \hat{Y}^{returns}_{mean,t}\right) \\ & + \mathcal{L}_{vol}\left(Y^{returns}_{vol,t}, \hat{Y}^{returns}_{vol,t}\right) + \mathcal{L}_{corr}\left(Y^{returns}_{corr,t}, \hat{Y}^{returns}_{corr,t}\right). \end{aligned} \tag{28}$$

4 EXPERIMENTS

4.1 Datasets

We conduct experiments with four datasets, comprising textual information of online news articles from two popular financial news portals and numerical information of daily stock market price-related information of two stock markets—NYSE and NASDAQ—from 2015 to 2019.

Textual News Data. The two online news article sources are (1) *Investing (IN)*[1] and (2) *Benzinga (BE)*[2] news datasets. The datasets contain news articles and commentaries collected from IN and BE investment news portals, which are drawn from a wide range of mainstream providers, analysts, and blogs, such as Seeking Alpha. We do not combine the two news datasets as they differ in their coverage of financial news. This also allows us to check the validity of our experimental results across different news datasets. Following [3], we use the Wikipedia2Vec [86] embedding model to pre-encode textual news to capture the rich knowledge present within the Wikipedia knowledge base while offering a relatively compact representation with dimension of 100. Wikipedia2Vec, pretrained with Wikipedia pages of entities and words in these pages, is designed to return representations of similar words and entities close to one another in the representational space. The representation of each news article is the average of Wikipedia2Vec embeddings of words in each news article. It turns out that the Wikipedia2Vec word embedding model produces reasonably good performance compared to other pre-trained text encoders.

[1] Subset extracted from https://www.kaggle.com/gennadiyr/us-equities-news-data

[2] Subset extracted from https://www.kaggle.com/miguelaenlle/massive-stock-news-analysis-db-for-nlpbacktests

Table 1. Overview of Datasets

	IN-NY	IN-NA	BE-NY	BE-NA
No. articles	221,513		1,377,098	
No. assets (stocks)	374	402	2,240	2,514
No. data points	470,118	505,314	2,815,680	3,160,098

Each data point is a set of price information for one stock in a time-step that corresponds to a sliding window $[t - K, t - 1]$ and forecasting window $[t, t + L]$ pair as shown in Figure 4.

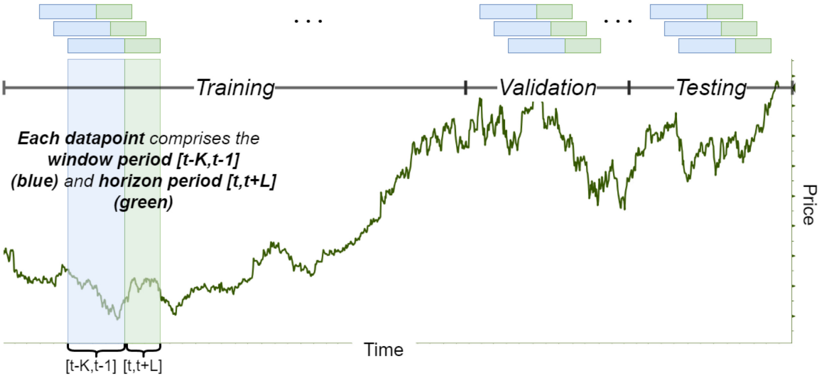
. . .
. . .
Training
Validation
Testing
Each datapoint comprises the window period [t-K,t-1] (blue) and horizon period [t,t+L] (green)
Price
Time
[t-K,t-1]
[t,t+L]

Fig. 4. We adopt a fixed sliding window to extract input numerical stock price and textual news features in the window $[t-K, t-1]$ and return, volatility, and correlation labels in the horizon $[t, t+L]$ to obtain labelled data points, and split these into non-overlapping training, validation, and testing sets.

Numerical Stock Price Data. For the local numerical information, we collected daily stock market price-related information—returns, opening, closing, low and high prices, trading volumes, volume-weighted average prices, and shares outstanding—of the two stock markets, *NYSE (NY)* and *NASDAQ (NA)*, from the Center for Research in Security Prices. We filter out stocks from NY and NA that were not traded in the respective time periods and whose stock symbols were not mentioned in any articles for the respective news article sources.

We combine them into four datasets (two news article sources and two stock markets), covering a different number of assets and news articles as depicted in Table 1. These datasets, spanning 5 years, with more than 1.5 million articles and 2,000 companies, are relatively large and provide strong assurance to our experiment findings, e.g., when compared to recent works [20, 40, 68], which cover fewer than 100 companies. To obtain labelled data samples, we adopt a sliding window approach [81] to extract numerical and textual input features in the window $[t - K, t - 1]$ and returns-related labels, i.e., ground-truth means, volatilities, and correlations of returns in the horizon $[t, t + L]$, as shown in Figure 4. For each of the four datasets, we obtain data across 1,257 time-steps, leading to a total number of data points ranging from 470,118 to 3,160,098 across the four datasets, and divide these samples into non-overlapping training/validation/testing sets in the ratios 0.6/0.2/0.2 for all experiments.

4.2 Tasks and Metrics

We compare TIME with state-of-the-art baselines on three predictive tasks: **forecasting of (1) means, (2) volatilities, and (3) correlations of asset price percentage returns**. We use RMSE, *mean absolute error (MAE),* and *symmetric mean absolute percentage error (SMAPE)* as metrics. RMSE and MAE are common scale-dependent metrics used to evaluate forecasting performance, with RMSE being more sensitive to outliers than MAE. SMAPE is a commonly used scale-independent metric. These metrics are computed as follows:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{|V|}(Y_t^{returns}[i] - \hat{Y}_t^{returns}[i])^2}{|V|}} \tag{29}$$

$$MAE = \frac{\sum_{i=1}^{|V|} |Y_t^{returns}[i] - \hat{Y}_t^{returns}[i]|}{|V|} \tag{30}$$

$$SMAPE = \frac{100\%}{|V|} \sum_{i=1}^{|V|} \frac{|Y_t^{returns}[i] - \hat{Y}_t^{returns}[i]|}{(|Y_t^{returns}[i]| + |\hat{Y}_t^{returns}[i]|)/2}. \tag{31}$$

We choose SMAPE instead of *mean absolute percentage error (MAPE)* as SMAPE gives equal importance to both under- and over-forecasts required in this evaluation context, while MAPE favors under-forecast.

4.3 Baselines and Settings

We compare TIME against **GRU** models and the following state-of-the-art baselines (see Section 2): **NBEATS** [56], **DARNN** [64], **MTGNN** [82], **TST** [91], **FAST**, and **SE** [20]. NBEATS, DARNN, and MTGNN are designed specifically for numerical information, while FAST and SE are designed for textual information. For the more general GRU and TST models, we compare against GRU and TST models with just numerical information as inputs (**GRU-Num** and **TST-Num**) and with concatenated numerical and textual information as inputs (**GRU-NumTxt** and **TST-NumTxt**). We did not compare against classical models as they cannot be adapted to the multitask setting required in our experiments. Instead, NBEATS, a recent state-of-the-art model that already demonstrated good performance when benchmarked against top classical models, is included as one of the baseline models. We add fully connected layers to all baselines for them to forecast means, volatilities, and correlations of asset price percentage returns.

We set the window period K=20 days and horizon period L=10 based on empirical experiments. K=20 corresponds to a trading month, and L=10 days corresponds to a global regulatory requirement for VaR computations, which we will examine in an application case study (see Section 6). Dimensions of hidden representations are fixed at 64 across all models. An Adam optimizer with a learning rate of 1e-3 with a cosine annealing scheduler is used. Models are implemented in Pytorch and trained for a maximum of 50 epochs with early stopping (patience of 5 epochs). We run each experiment 10 times with different random seeds to initialize model parameters and report the averages and standard deviations of results across 10 runs. The TIME model has 3.5e5 parameters and takes around 1 to 3 minutes per training epoch on a 3.60 GHz AMD Ryzen 7 Windows desktop with NVIDIA RTX 3090 GPU and 64 GB RAM.

4.4 Results

Table 2 sets out the results, averaged over 10 runs, of the forecasting experiments on the IN and BE datasets, respectively. In *general*, across all tasks and datasets, TIME out-performs baselines according to most metrics. Other than the narrower performance differences between TIME and baselines on the task of forecasting means for the RMSE metric, the performance differences between TIME and baselines for other tasks (i.e., based on the MAE and SMAPE metric for forecasting means, and all three metrics for forecasting volatilities and correlations) are relatively clear. This suggests that the tasks of forecasting volatilities and correlations are harder than the task of forecasting means,

Table 2. Forecasting Results

	IN-NY			IN-NA			BE-NY			BE-NA		
	RMSE	MAE	SMAPE	RMSE	MAE	SMAPE	RMSE	MAE	SMAPE	RMSE	MAE	SMAPE
	Mean Forecasting											
GRU-1	0.0653	0.0136	1.2705	0.0259	0.0147	1.2744	0.0768	0.0194	1.3169	0.1937	0.0385	1.3380
TST-1	0.0657	0.0145	1.4798	0.0319	0.0156	1.4492	0.0753	0.0193	1.4898	0.1968	0.0362	1.3724
NBEATS	**0.0651**	0.0136	1.3847	0.0261	0.0155	1.2794	**0.0700**	0.0185	1.4031	**0.1904**	0.0326	1.3555
DARNN	**0.0651**	0.0137	1.3871	0.0262	0.0148	1.3678	0.0724	0.0179	1.3645	0.1950	0.0322	1.3634
MTGNN	0.0652	0.0156	1.2493	0.0428	0.0169	1.3234	0.0703	0.0179	1.3801	0.2323	0.0451	1.3826
FAST	0.0680	0.0148	1.4507	0.0347	0.0174	1.3350	0.0825	0.0199	1.4007	0.1985	0.0395	1.3669
SE	0.0706	0.0201	1.3244	0.0429	0.0233	1.3208	0.0869	0.0226	1.3520	0.1980	0.0410	1.3363
GRU-2	0.0652	0.0140	1.2695	0.0257	0.0146	1.2547	0.0756	0.0192	1.3016	0.1973	0.0368	1.3296
TST-2	0.0656	0.0143	1.4014	0.0329	0.0165	1.2910	0.0768	0.0199	1.4064	0.1962	0.0377	1.3634
TIME	0.0652	**0.0115**	**1.0424**	**0.0231**	**0.0115**	**1.0520**	0.0703	**0.0164**	**1.2696**	0.1929	**0.0320**	**1.2796**
	Volatility Forecasting											
GRU-1	0.1957	0.0437	0.5357	0.0820	0.0463	0.5517	0.2256	0.0556	0.6336	0.5977	0.1137	0.7841
TST-1	0.1909	0.0442	0.5231	0.1012	0.0499	0.5583	0.2383	0.0629	0.6098	0.5928	0.1181	0.6897
NBEATS	0.1571	0.0363	0.4879	**0.0722**	0.0397	0.4921	0.2250	0.0556	0.5917	0.5926	0.1099	0.6862
DARNN	0.1848	0.0381	0.4696	0.0754	0.0409	0.4941	0.2294	0.0594	0.5925	0.5963	0.1171	0.6851
MTGNN	0.1551	0.0414	0.6033	0.1157	0.0577	0.6244	0.2275	0.0561	0.5937	0.5963	0.1189	0.7110
FAST	0.2125	0.0479	0.5623	0.1170	0.0574	0.6272	0.2722	0.0747	0.7218	0.6018	0.1327	0.7737
SE	0.2129	0.0488	0.5758	0.1213	0.0585	0.6270	0.2703	0.0742	0.7044	0.6018	0.1317	0.7102
GRU-2	0.1946	0.0443	0.5595	0.0806	0.0458	0.5484	0.2234	0.0588	0.6453	0.5995	0.1145	0.7672
TST-2	0.1957	0.0450	0.5389	0.1063	0.0541	0.5970	0.2443	0.0662	0.6487	0.5963	0.1282	0.7532
TIME	**0.1550**	**0.0327**	**0.4080**	**0.0722**	**0.0364**	**0.4271**	**0.2200**	**0.0546**	**0.5840**	**0.5922**	**0.1093**	**0.6805**
	Correlation Forecasting											
GRU-1	0.5054	0.4383	1.3498	0.4999	0.4326	1.4708	0.5083	0.4391	1.4381	0.4905	0.4210	1.5441
TST-1	0.5069	0.4414	1.3748	0.4987	0.4319	1.4460	0.5068	0.4391	1.4410	0.4891	0.4205	1.5678
NBEATS	0.5064	0.4395	1.3507	0.4986	0.4322	1.4571	0.5074	0.4387	1.4339	0.4890	0.4202	1.5550
DARNN	0.5069	0.4419	1.3761	0.4991	0.4327	1.4602	0.5083	0.4399	1.4372	0.4897	0.4213	1.5773
MTGNN	0.5110	0.4435	1.3740	0.5002	0.4329	1.4533	0.5085	0.4405	1.4483	0.5035	0.4238	1.5704
FAST	0.5086	0.4436	1.3888	0.4992	0.4328	1.4640	0.5085	0.4407	1.4541	0.4893	0.4207	1.5661
SE	0.5126	0.4431	1.3985	0.5047	0.4348	1.4746	0.5161	0.4433	1.4416	0.4902	0.4198	1.5630
GRU-2	0.5060	0.4391	1.3670	0.5003	0.4321	1.4609	0.5088	0.4387	1.4224	0.4898	0.4209	1.5598
TST-2	0.5063	0.4408	1.3673	0.4989	0.4329	1.4624	0.5068	0.4393	1.4439	0.4894	0.4209	1.5675
TIME	**0.4167**	**0.3396**	**1.0260**	**0.4197**	**0.3472**	**1.1291**	**0.4781**	**0.4075**	**1.3107**	**0.4778**	**0.4062**	**1.4731**

Lower average is better for all metrics. Best and second-best performing models are **boldfaced** and underlined, respectively.

and that TIME performs better on such harder tasks. Among the baselines, NBEATS, which models specific time-series patterns, and MTGNN, which learns the underlying relationships between asset nodes, generally perform better, highlighting the importance of such model features. NBEATS, a univariate model, generally performs better on the means forecasting task when compared to multivariate models such as DARNN and MTGNN, but this difference between the NBEATS model and the DARNN and MTGNN models is less consistent for the volatility and correlation forecasting tasks. This suggests that capturing multivariate information is important for the harder tasks of forecasting volatilities and correlations.

On **forecasting means**, we see that the performance differences between models are the least dispersed, particularly for the RMSE metric. Nonetheless, the differences between TIME and baselines on the MAE and SMAPE metrics are clear, even after taking into account the standard deviations. In particular, TIME enjoys a 16.6% and 16.2% improvement in SMAPE compared with the second-best-performing models on IN-NY and IN-NA, respectively. Among the baselines, NBEATS, DARNN, MTGNN, and GRU-2 show relatively better performance across the four datasets. While the GRU-2 model is a relatively simpler model, its good performance on the task of forecasting means could be due to its use of multimodal (i.e., both numerical and textual) information.

On the task of **forecasting volatilities**, the performance differences between models are more dispersed and the differences in performance between TIME and baselines are more consistent as compared with the task of forecasting means. This could be due to the difficulties that baselines have in adjusting to changes in volatility regimes for different markets, which TIME is able to adjust to due to its ability to capture multivariate and multimodal information and adapt to evolving intra-series patterns and inter-series relationships between assets. Among the baselines, NBEATS, DARNN, and MTGNN again show relatively better performance, similar to what we observed for the task of forecasting means. GRU-2 in this case does not perform as well, possibly due to forecasting volatility being a harder task.

On the task of **forecasting correlations**, the difference in performance between TIME and the baselines is the clearest, as compared to the mean and volatility forecasting tasks. For example, TIME shows 17.5% and 15.8% smaller RMSE compared with the second-best models on IN-NY and IN-NA datasets, respectively. TIME also achieves at least 20% smaller SMAPE than the second-best models on the two datasets. This demonstrates the usefulness of capturing implicit inter-series/asset relationship networks at multiple time-steps—a novel feature of TIME. Among the baselines, NBEATS, TST, and GRU models show relatively better performance, but the gap between these models and the TIME model on the task of forecasting correlations is large relative to other tasks. The performance of these models is also not consistent across the four datasets as we see them performing well only on one or two datasets.

While there are variations in forecasting performances between TIME and baselines across different tasks and datasets, TIME generally achieves consistently good performance across all tasks and datasets. In contrast, the baselines can be seen to perform well on one or two tasks/datasets but perform poorly on other tasks/datasets. For example, while NBEATS performs consistently well on the task of forecasting means, its performance on the tasks of forecasting volatilities and correlations is poorer and less consistent. Performing consistently well on all three tasks is important for investment and risk management, which involves portfolios comprising multiple assets, e.g., a decision on whether to buy or sell a stock in a portfolio depends not only on its mean return in the future but also on how volatile (or risky) the stock will be in the future, and how correlated the stock will be to other stocks in the portfolio in the future (due to diversification considerations). TIME is hence more suited for such applications based on its more consistent and good performance across multiple tasks and datasets as it captures multivariate and multimodal information, as well as implicit inter-series/asset relationship networks at multiple time-steps.

5 ABLATION STUDIES

We conduct ablation studies to evaluate the performance impact on TIME when some model features are removed or simplified or hyper-parameters changed. These include:

- **w/o. TimeVect**: We remove the TimeVect module so that no temporal representation is added to the dynamic network representations $\tilde{H}_t^m$.
- **w. single net**: We take the average of weights across the window $[t-K, t-1]$ to obtain a single implicit network.

Table 3. Ablation Studies on IN-NY Datasets

	RMSE	MAE	SMAPE
	Mean Forecasting		
w/o. TimeVect	0.0716	0.0120	1.0633
w. single net.	0.0653	0.0116	1.0545
w/o. inner wt.	0.0655	0.0116	1.0510
R = 10%	0.0656	0.0117	1.0787
R = 80%	0.0662	0.0117	1.0507
no backcast loss	0.0662	0.0117	1.0573
no mean loss	0.0994	0.0653	1.7131
no vol. loss	0.0656	0.0119	1.1032
no corr. Loss	0.0694	0.0121	1.0813
TIME	**0.0652**	**0.0115**	**1.0424**
	Volatility Forecasting		
w/o. TimeVect	0.1617	0.0341	0.4187
w. single net.	0.1553	0.0338	0.4266
w/o. inner wt.	0.1560	0.0328	0.4137
R = 10%	0.1555	0.0347	0.4416
R = 80%	0.1622	0.0332	0.4185
no backcast loss	0.1558	0.0333	0.4268
no mean loss	0.1561	0.0363	0.4633
no vol. loss	0.2419	0.1051	1.6133
no corr. Loss	0.1599	**0.0327**	0.4096
TIME	**0.1550**	**0.0327**	**0.4080**
	Correlation Forecasting		
w/o. TimeVect	0.4230	0.3475	1.0550
w. single net.	0.4233	0.3465	1.0480
w/o. inner wt.	0.4184	0.3414	1.0328
R = 10%	0.4355	0.3608	1.0913
R = 80%	0.4227	0.3463	1.0440
no backcast loss	0.4223	0.3462	1.0505
no mean loss	0.4422	0.3674	1.1086
no vol. loss	0.4478	0.3733	1.1267
no corr. Loss	0.5467	0.4833	1.9203
TIME	**0.4167**	**0.3396**	**1.0260**

Lower is better for all metrics. Best model(s) are in **bold**; second-best model(s) are underlined.

- **R=10%/R=80%**: Recall that the degree of sparsification R is used for selecting inter-series relationship edges. Instead of the default choice $R = 20\%$, we study the impact of adopting sparser and denser degrees.
- **no backcast loss**: Removal of backcast of numeric price-related input data $\mathcal{L}_{backcast}$ from Equation (28).
- **no mean loss, no vol. loss, no corr. loss**: Removal of $\mathcal{L}_{mean}$, $\mathcal{L}_{vol}$, and $\mathcal{L}_{corr}$, respectively, from Equation (28).

Table 3 shows the results, averaged across 10 runs, of the ablation studies for TIME on the IN-NY datasets. We observe similar sensitivities for the other three datasets. Not utilizing the temporal representation, i.e., **w/o. TimeVect**, shows a significant drop in performance on a number of metrics, demonstrating the importance of capturing intra-series patterns using temporal representations. When we use a single implicit network across the window, i.e., **w. single net.**, we observe poorer performance across all metrics, particularly for the volatility and correlation forecasts, demonstrating the importance of capturing evolving inter-series relationships. When we vary the degree of sparsification, with $R = 10\%$ and $R = 80\%$ (instead of $R = 20\%$ used in our experiments), we see significant variations in performance, particularly for volatility and correlation forecasts, which indicates the importance of the implicit network structural information. When we vary the training objective by either excluding backcast, mean, volatility, or correlation forecast losses (i.e., **no backcast loss, no mean loss, no vol. loss, no corr. loss,** respectively), we see significant drops in performance, even for tasks whose losses were not excluded in the training objective, demonstrating the importance of the multitask setting in improving overall forecasting performance. This suggests that overfitting can be an issue when training on a single task for such financial time-series. The multitask setting that we adopt with these heterogeneous but related forecasting tasks can help to improve overall performance across tasks as it serves as a regularization process to prevent overfitting, and also enables complementary information from other related tasks to be used to improve performance across tasks.

6 CASE STUDIES

In this section, following [4], we apply the model forecasts for two important investment and risk management applications—portfolio allocation and VaR forecasting—to evaluate the quality of TIME's forecasts against the baselines.

6.1 Portfolio Allocation

Portfolio allocation is an important task for many financial institutions. The aim of portfolio allocation is to optimize the proportion of capital invested in each asset in a portfolio by finding an optimal set of weights $\mathbb{W}$ that determine how much capital to invest in each asset, so that portfolio returns can be maximized while minimizing portfolio risk. In this article, we adopt the risk aversion formulation [22] of the mean-variance risk minimization model by [52], which models both portfolio return and risk expressed as mean (μ) and co-variances (Σ) of return, respectively. Under the risk aversion formulation, the classical mean-variance risk minimization model by [52] is re-formulated to maximize the risk-adjusted portfolio return by optimizing the asset allocation $\mathbb{W}$, a $|V|$ dimensional vector:

$$max_{\mathbb{W}}\ (\mathbb{W}^\intercal \mu - \lambda \mathbb{W}^\intercal \Sigma \mathbb{W}), \tag{32}$$

subject to $\mathbb{W}^\intercal \mathbf{1} = 1$. λ, known as the Arrow-Pratt risk aversion index, is used to express an investor's risk preferences and is typically in the range of 2 to 4 [22]. In our experiments, we set $\lambda = 2$. We observe that higher λ values reduce returns across all models, but the relative differences in returns between models generally remain consistent. In this article, we use the forecasted means of asset returns for μ and compute Σ with the forecasted volatilities and correlations of asset returns for the selected horizon period $[t, t + L]$ defined as follows:

$$\tilde{\mu} = \hat{Y}^{returns}_{mean,t} \tag{33}$$

$$\tilde{\Sigma} = D_t \cdot \hat{Y}^{returns}_{corr,t} \cdot D_t, \tag{34}$$

where D_t is the $|V| \times |V|$ diagonal (and thus symmetric) matrix filled with $\hat{Y}^{returns}_{vol,t}$ along the diagonals and 0 otherwise. We choose to forecast correlations of asset returns over the selected horizon period $[t, t + L]$ instead of directly forecasting co-variances as the co-variances need to be *positive semi-definite (PSD)* so that the matrix is invertible [21], which is important for applications such as portfolio optimization. Forecasting co-variances directly does not guarantee PSD. We instead forecast volatilities and correlations separately and compute the co-variance matrix using the forecasted volatilities and correlations.

This application can be viewed as a predictive task as we use the input information from the window period $[t-K, t-1]$ to make forecasts of the mean (μ) and correlation (Σ) of asset returns over the future horizon $[t, t+L]$, which are in turn used to determine the asset allocation weights $\mathbb{W}^{forecast}$. The realized returns of the investment portfolio constructed according to $\mathbb{W}^{forecast}$ are computed as $E^{forecast} = \mathbb{W}^{forecast\intercal} R^{real}$, where R^{real} is a vector of realized percentage asset returns over the future horizon.

Instead of model forecasts, the classical approach, which we use as a *naive* approach in this article, uses the mean of percentage asset returns based on historical returns over a selected period μ, and the historical co-variances of the same set of historical returns Σ as inputs to Equation (32) to obtain the asset allocation weights ($\mathbb{W}^{naive}$), which we can use to compute the portfolio returns $E^{naive} = \mathbb{W}^{naive\intercal} R^{real}$. Actual returns of a portfolio of assets depend on the time-series and time periods under consideration. Hence, for better comparability, we evaluate the performance of TIME and baselines relative to the classical/naive approach by computing the ratio $\mathcal{R} = E^{forecast}/E^{naive}$. Given that the aim is to maximize portfolio returns while minimizing portfolio risk (volatility), we also compute the respective risk-adjusted realized portfolio returns over the future horizon $[t, t+L]$: $E^{forecast\prime} = \frac{E^{forecast}}{\sigma^{forecast}}$, where $\sigma^{forecast}$ is the corresponding volatility of the portfolio constructed using $\mathbb{W}^{forecast}$ in the future horizon $[t, t+L]$, $E^{naive\prime} = \frac{E^{naive}}{\sigma^{naive}}$, and σ^{naive} is similarly the volatility of the portfolio constructed using $\mathbb{W}^{naive}$ in the future horizon $[t, t+L]$. These are used to compute the risk-adjusted ratio $\mathcal{R}' = E^{forecast\prime}/E^{naive\prime}$ to further evaluate the performance of TIME and baselines. In both cases, portfolio return volatility (i.e., $\sigma^{forecast}$ and σ^{naive}) is defined as one standard deviation of the respective portfolio returns over the future horizon $[t, t+L]$ and is computed as $\sigma = \sqrt{\mathbb{W}^{\intercal}\Sigma\mathbb{W}}$, where Σ are the co-variances of realized percentage asset returns of the respective portfolios over the same future horizon.

For this application, the datasets are similarly divided into non-overlapping training/validation/testing sets in the ratios 0.6/0.2/0.2 as described in Section 4.1, and we evaluate performance based on the averages of $\mathcal{R}$ and $\mathcal{R}'$ across the testing set.

6.2 Value-at-Risk

VaR [45] is a key measure of risk used in financial institutions for the measurement, monitoring, and management of financial risk. Financial regulators require important financial institutions such as banks to measure and monitor their VaR over a L=10 day horizon and maintain capital based on this VaR as loss buffers. VaR measures the loss that an institution may face in the predefined horizon with a probability of p%. For example, if the 10-day 95% VaR is \$1,000,000, it means that there is a p = 5% probability of losses exceeding \$1,000,000 over a 10-day horizon.

VaR can be computed as a multiple of the portfolio's volatility:

$$VaR(p) = -\phi^{-1}(p) \times \sigma, \tag{35}$$

where σ is the portfolio volatility, and ϕ is the inverse cumulative distribution function of the standard normal distribution, for example, if $p = 5\%$, then $\phi^{-1}(p) = 1.645$. Whenever realized portfolio losses (i.e., negative realized portfolio returns $E^{realized}$) are greater than the forecasted VaR, it is regarded as a *VaR breach*, i.e., $E^{realized} \leq VaR(p)$.

For this application, the portfolio is constructed based on the approach described for the portfolio allocation application at each time-step. This mimics a real-world scenario where financial institutions continually update their portfolios based on market conditions. To evaluate

Table 4. Applications: Higher Is Better for $\mathcal{R}/\mathcal{R}'$; Lower Is Better for % Br

	IN-NY			IN-NA		
	% Br.	$\mathcal{R}$	$\mathcal{R}'$	% Br.	$\mathcal{R}$	$\mathcal{R}'$
GRU-1	5.1%	1.7	1.2	3.4%	1.9	0.9
TST-1	11.7%	1.2	1.1	3.4%	1.0	1.0
NBEATS	5.1%	1.4	1.5	2.4%	1.7	0.8
DARNN	8.5%	1.7	1.1	4.7%	1.3	0.8
MTGNN	6.3%	1.8	1.1	9.4%	0.6	0.4
FAST	17.9%	0.1	0.1	13.7%	0.3	0.4
SE	12.6%	0.1	0.1	5.7%	0.3	0.5
GRU-2	6.3%	**2.5**	1.7	3.4%	1.8	0.6
TST-2	13.5%	1.3	1.2	5.3%	0.8	1.1
TIME	**4.3%**	**2.5**	**2.2**	**2.2%**	**2.7**	**2.9**

the baseline models, we use the forecasted portfolio volatility $\tilde{\sigma} = \sqrt{\tilde{\Sigma}}$, where $\tilde{\Sigma}$ is computed using the forecasted volatilities and correlations of asset returns as defined in Equation (34). Similar to the portfolio allocation application, this can also be viewed as a predictive task as we are using input information from the window period $[t-K, t-1]$ to make forecasts over the future horizon $[t, t+L]$ and using these forecasts to determine the VaR in the future horizon. We evaluate model performances based on percentage of VaR breaches (% Br.), i.e., the percentage of losses in the testing set that led to 95% VaR breaches (using the same training/validation/testing sets as described in Section 4.1). Models that are able to make accurate forecasts of VaR should have a lower percentage of VaR breaches (% Br.). We choose the 95% threshold for our experiments as it is a common confidence level used by banks to monitor their risks.

We run each of these portfolio allocation and VaR measurement experiments 10 times with the models trained earlier for the forecasting tasks and report the averages of results across the 10 runs. We conduct and report experiments on the IN-NY and IN-NA datasets with fewer assets as a smaller pool of potential assets usually presents a greater challenge for these two applications by limiting potential returns and risk diversification.

6.3 Experiment Results

Table 4 depicts results for the IN-NY and IN-NA datasets for the VaR application. On the **portfolio allocation** application, portfolios constructed using the forecasts from TIME achieve better relative performance on the return ratio ($\mathcal{R}$) and risk-adjusted return ratio ($\mathcal{R}'$) for both datasets. Similarly, on the **VaR** application, TIME also out-performs the baselines, with a lower percentage of VaR breaches (% Br.). For both applications, we observe significant variance in performance for the baselines, with a number of baselines showing ratios of less than 1 (i.e., performing worse than the naive approach) or high levels of percentage of VaR breaches, demonstrating the difficulty of these applications. The performances of NBEATS, MTGNN, and GRU-2 are the closest to TIME, indicating the importance of capturing intra-series patterns, implicit inter-series relationships, and multimodality, respectively.

7 INTERPRETABILITY

Being able to interpret underlying implicit relationship networks discovered by TIME and utilized for forecasting can support further analysis by investment and risk managers. In this section, we show how implicit relationship networks across multiple modalities discovered by TIME can be

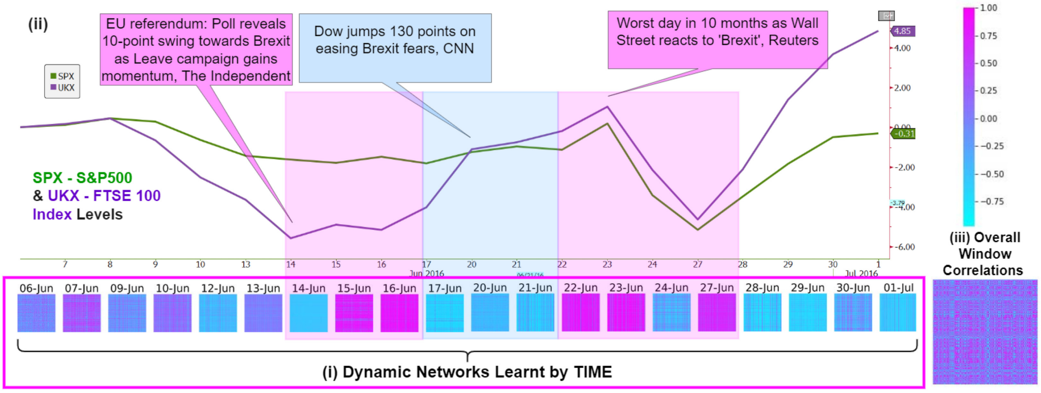

Fig. 5. (i) is a sequence of heatmaps visualizing the dynamic networks, representing inter-series relationships between assets, learned by TIME. (ii) contextualizes these dynamic networks by mapping them to changes in market prices and news sentiment in the lead-up to and after the announcement of the Brexit referendum on June 23, 2016. (iii) visualizes the overall correlations between assets over the entire window period. As shown in the legend on the right, pink indicates more positive attention weights (more correlated or similar), while cyan indicates more negative attention weights (less correlated or dissimilar). We can see that TIME captures rich evolving inter-series relationships between assets at multiple time-steps as shown in (i), which would not have been captured with the usual approach of modeling a single implicit network or overall correlations over the entire window period as shown in (iii).

extracted and demonstrate the importance of capturing evolving inter-series relationships with a case study. As described in Section 3.1, TIME learns modality-specific AW_t^m from the encoded financial time-series information. AW_t^m represent weighted inter-series relationships between assets learned by TIME for modality m. To visualize AW_t^m across M modalities, we utilize the multimodal fusion weights β_t^m learned by TIME, as described in Section 3.4, to obtain $AW_t = \sum_{m \in M} \beta_t^m AW_t^m$. The resultant $AW_t \in \mathbb{R}^{|V| \times |V| \times K}$ represents fused inter-series relationships between assets learned by TIME in an adaptive fashion, which we can analyze to interpret how underlying relationships between assets evolved in different window periods.

To illustrate the inter-series relationships learned by TIME, we use a case study of a period in June 2016 leading up to and after the announcement of the Brexit referendum, i.e., the results of the vote on whether the United Kingdom would exit the European Union. There were significant swings in market and public sentiment during that short period, corresponding to changes in numerical price-related and textual news information, respectively. Hence, it allows us to observe the evolving inter-series relationships between assets due to these changes that were learned by TIME. Figure 5(i) visualizes the dynamic networks representing inter-series relationships between assets learned by TIME. Figure 5(ii) shows a context of changes in market prices and news sentiment in the lead-up to and after the announcement of the Brexit referendum on June 23, 2016. Figure 5(iii) shows the overall correlations between assets over the whole window period. We observe that the marked swings in market prices and news sentiment due to changes in expectations on the outcome of the Brexit referendum were reflected clearly in the dynamic networks learned by TIME. Periods of fear due to heightened expectations of Brexit happening (June 14–16, 2016) and when Brexit materialized (June 22–27, 2016) correspond to high inter-series relationship weights in the dynamic networks learned by TIME. TIME was also sensitive to the easing of such fears in the intervening period (June 17–21, 2016), as we see lower inter-series relationship weights in the dynamic networks learned by TIME during that period. In contrast, the usual approach of modeling a single implicit network or overall correlations over the entire window period, as shown in Figure 5(iii), would not have captured such rich evolving inter-series relationships.

8 CONCLUSION AND FUTURE WORK

In this article, we proposed TIME, a novel model that self-discovers implicit inter-series relationship networks at multiple time-steps from multimodal time-series data and applies dynamic network learning on such networks for multivariate time-series forecasting on multiple tasks. Based on extensive experiments on three forecasting tasks and two important real-world financial applications across real-world datasets, we show the value of learning implicit inter-series networks at multiple time-steps from time-series data and combining these with learned temporal representations for multiple forecasting tasks.

In future work, we intend to explore combining such learned implicit relationship networks together with pre-defined explicit networks (e.g., from knowledge graphs extracted from Wikidata or economic/financial transaction networks purchased from data providers) on other tasks that could benefit from such information, such as forecasting events, stock returns, and credit ratings of companies, and providing stock recommendations. Further work could also be conducted on learning important implicit relationships to improve model explainability and interpretability. Aside from focusing on designing deep learning models for such dynamic time-series and network information, further work could also focus on how such models could be integrated within end-to-end reinforcement learning frameworks for quantitative trading such as TradeMaster [71].

REFERENCES

[1] Bo An, Shuo Sun, and Rundong Wang. 2022. Deep reinforcement learning for quantitative trading: Challenges and opportunities. *IEEE Intelligent Systems* 37, 2 (2022), 23–26.

[2] Oren Anava, Elad Hazan, Shie Mannor, and Ohad Shamir. 2013. Online learning for time series prediction. In *Conference on Learning Theory*. PMLR, 172–184.

[3] Gary Ang and Ee-Peng Lim. 2021. Learning knowledge-enriched company embeddings for investment management. In *ACM International Conference on AI in Finance*.

[4] Gary Ang and Ee-Peng Lim. 2022. Guided attention multimodal multitask financial forecasting with inter-company relationships and global and local news. In *Annual Meeting of the Association for Computational Linguistics (ACL'22)*.

[5] Lei Bai, Lina Yao, Can Li, Xianzhi Wang, and Can Wang. 2020. Adaptive graph convolutional recurrent network for traffic forecasting. In *International Conference on Neural Information Processing Systems (NIPS'20)*.

[6] Shaojie Bai, J. Zico Kolter, and Vladlen Koltun. 2018. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *CoRR* (2018).

[7] Peter W. Battaglia, Jessica B. Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinícius Flores Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, Çaglar Gülçehre, H. Francis Song, Andrew J. Ballard, Justin Gilmer, George E. Dahl, Ashish Vaswani, Kelsey R. Allen, Charles Nash, Victoria Langston, Chris Dyer, Nicolas Heess, Daan Wierstra, Pushmeet Kohli, Matthew M. Botvinick, Oriol Vinyals, Yujia Li, and Razvan Pascanu. 2018. Relational inductive biases, deep learning, and graph networks. *CoRR* (2018).

[8] Inci M. Baytas, Cao Xiao, Xi Zhang, Fei Wang, Anil K. Jain, and Jiayu Zhou. 2017. Patient subtyping via time-aware LSTM networks. In *International Conference on Knowledge Discovery and Data Mining (KDD'17)*.

[9] Geert Bekaert and Guojun Wu. 2015. Asymmetric volatility and risk in equity markets. *Review of Financial Studies* 13, 1 (2015), 1–42.

[10] Tim Bollerslev. 1986. Generalized autoregressive conditional heteroskedasticity. *Journal of Econometrics* 31, 3 (April 1986), 307–327.

[11] Anastasia Borovykh, Sander Bohte, and Cornelis W. Oosterlee. 2017. Conditional time series forecasting with convolutional neural networks. In *Lecture Notes in Computer Science/Lecture Notes in Artificial Intelligence*. 729–730.

[12] Defu Cao, Yujing Wang, Juanyong Duan, Ce Zhang, Xia Zhu, Congrui Huang, Yunhai Tong, Bixiong Xu, Jing Bai, Jie Tong, and Qi Zhang. 2020. Spectral temporal graph neural network for multivariate time-series forecasting. In *NIPS*.

[13] Cristian Challu, Kin G. Olivares, Boris N. Oreshkin, Federico Garza Ramirez, Max Mergenthaler Canseco, and Artur Dubrawski. 2023. Nhits: Neural hierarchical interpolation for time series forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

[14] Hao Chen, Keli Xiao, Jinwen Sun, and Song Wu. 2017. A double-layer neural network framework for high-frequency forecasting. *ACM Transactions on Management Information Systems* 7, 4, Article 11 (2017), 17 pages.

[15] Jinyin Chen, Xueke Wang, and Xuanheng Xu. 2021. GC-LSTM: Graph convolution embedded LSTM for dynamic link prediction. *Applied Intelligence* 52 (September 2021), 7513–7528.

[16] Rui Cheng and Qing Li. 2021. Modeling the momentum spillover effect for stock prediction via attribute-driven graph attention networks. In *AAAI Conference on AI (AAAI'21).*

[17] Eunsuk Chong, Chulwoo Han, and Frank C. Park. 2017. Deep learning networks for stock market analysis and prediction: Methodology, data representations, and case studies. *Expert Systems with Applications* 83 (2017), 187–205.

[18] Andrea Cini, Daniele Zambon, and Cesare Alippi. 2023. Sparse graph learning from spatiotemporal time series. *Journal of Machine Learning Research* 24, 242 (2023), 1–36.

[19] Xiao Ding, Yue Zhang, Ting Liu, and Junwen Duan. 2015. Deep learning for event-driven stock prediction. In *International Joint Conference on AI (IJCAI'15).*

[20] Xin Du and Kumiko Tanaka-Ishii. 2020. Stock embeddings acquired from news articles and price history, and an application to portfolio optimization. In *Annual Meeting of the Association for Computational Linguistics (ACL'20).*

[21] Robert Engle. 2002. Dynamic conditional correlation: A simple class of multivariate generalized autoregressive conditional heteroskedasticity models. *Journal of Business and Economic Statistics* 20, 3 (2002), 339–350.

[22] F. J. Fabozzi, P. N. Kolm, D. A. Pachamanova, and S. M. Focardi. 2007. *Robust Portfolio Optimization and Management.* Wiley.

[23] Christos Faloutsos, Valentin Flunkert, Jan Gasthaus, Tim Januschowski, and Yuyang Wang. 2020. Forecasting big time series: Theory and practice. In *The Web Conference (WWW'20) Tutorial.*

[24] Fuli Feng, Xiangnan He, Xiang Wang, Cheng Luo, Yiqun Liu, and Tat-Seng Chua. 2019. Temporal relational ranking for stock prediction. *ACM Transactions on Information Systems* 37, 2 (2019), 27:1–27:30.

[25] Valentin Flunkert, David Salinas, and Jan Gasthaus. 2020. DeepAR: Probabilistic forecasting with autoregressive recurrent networks. *International Journal of Forecasting* 36, 3 (2020), 1181–1191.

[26] C. Lee Giles, Steve Lawrence, and Ah Chung Tsoi. 2001. Noisy time series prediction using recurrent neural networks and grammatical inference. *Machine Learning.* 44, 1/2 (2001), 161–183.

[27] Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. 2017. Neural message passing for quantum chemistry. *International Conference on Machine Learning (ICML'17).*

[28] Luke B. Godfrey and Michael S. Gashler. 2018. Neural decomposition of time-series data for effective generalization. *IEEE Transactions on Neural Networks Learning Systems* 29, 7 (2018), 2973–2985.

[29] Palash Goyal, Nitin Kamra, Xinran He, and Yan Liu. 2018. DynGEM: Deep embedding method for dynamic graphs. *CoRR* (2018).

[30] Ehsan Hajiramezanali, Arman Hasanzadeh, Krishna R. Narayanan, Nick Duffield, Mingyuan Zhou, and Xiaoning Qian. 2019. Variational graph recurrent neural networks. In *NIPS.*

[31] William L. Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *NIPS.*

[32] Ziniu Hu, Weiqing Liu, Jiang Bian, Xuanzhe Liu, and Tie-Yan Liu. 2018. Listening to chaotic whispers: A deep learning framework for news-oriented stock trend prediction. In *ACM International Conference on Web Search and Data Mining (WSDM'18).*

[33] Weiwei Jiang. 2021. Applications of deep learning in stock market prediction: Recent progress. *Expert Systems with Applications* 184 (December 2021), 115537.

[34] Leonidas Jr and Italo Franca. 2011. Correlation of financial markets in times of crisis. *Physica A: Statistical Mechanics and Its Applications* 391, 1 (02 2011), 187–208.

[35] Kelvin Kan, François-Xavier Aubet, Tim Januschowski, Youngsuk Park, Konstantinos Benidis, Lars Ruthotto, and Jan Gasthaus. 2022. Multivariate quantile function forecaster. In *Proceedings of the 25th International Conference on Artificial Intelligence and Statistics.*

[36] Seyed Mehran Kazemi, Rishab Goel, Sepehr Eghbali, Janahan Ramanan, Jaspreet Sahota, Sanjay Thakur, Stella Wu, Cathal Smyth, Pascal Poupart, and Marcus A. Brubaker. 2019. Time2Vec: Learning a vector representation of time. *CoRR* (2019).

[37] Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations (ICLR'17).*

[38] Mengzhang Li and Zhanxing Zhu. 2021. Spatial-temporal fusion graph neural networks for traffic flow forecasting. In *AAAI Conference on AI (AAAI'21).*

[39] Qing Li, Yan Chen, Jun Wang, Yuanzhu Chen, and Hsinchun Chen. 2018. Web media and stock markets: A survey and future directions from a big data perspective. *IEEE TKDE* 30, 2 (2018), 381–399.

[40] Qing Li, Jinghua Tan, Jun Wang, and Hsinchun Chen. 2021. A multimodal event-driven LSTM model for stock prediction using online news. *IEEE TKDE* 33, 10 (2021), 3323–3337.

[41] Yaguang Li, Rose Yu, Cyrus Shahabi, and Yan Liu. 2018. Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. In *International Conference on Learning Representations (ICLR'18)*.

[42] Bryan Lim, Sercan Ömer Arik, Nicolas Loeff, and Tomas Pfister. 2021. Temporal fusion transformers for interpretable multi-horizon time series forecasting. *International Journal of Forecasting* 37, 4 (2021), 1748–1764.

[43] Bryan Lim and Stefan Zohren. 2021. Time series forecasting with deep learning: A survey. *Philosophical Transactions of the Royal Society A* 379, 2194 (2021), 20200209.

[44] Bryan Lim, Stefan Zohren, and Stephen Roberts. 2020. Recurrent neural filters: Learning independent Bayesian filtering steps for time series prediction. In *International Joint Conference on Neural Networks (IJCNN'20)*.

[45] Thomas J. Linsmeier and Neil D. Pearson. 2000. Value at risk. *Financial Analysts Journal* 56, 2 (2000), 47–67.

[46] Chenghao Liu, Steven C. H. Hoi, Peilin Zhao, and Jianling Sun. 2016. Online ARIMA algorithms for time series prediction. *Proceedings of the AAAI Conference on Artificial Intelligence* 30, 1 (Feb. 2016). https://doi.org/10.1609/aaai.v30i1.10257

[47] Xiao-Yang Liu, Ziyi Xia, Jingyang Rui, Jiechao Gao, Hongyang Yang, Ming Zhu, Chris Wang, Zhaoran Wang, and Jian Guo. 2022. FinRL-meta: Market environments and benchmarks for data-driven financial reinforcement learning. *SSRN Electronic Journal* (2022).

[48] Yeqi Liu, Chuanyang Gong, Ling Yang, and Yingyi Chen. 2020. DSTP-RNN: A dual-stage two-phase attention-based recurrent neural network for long-term and multivariate time series prediction. *Expert Systems with Applications* 143 (2020), 113082.

[49] Chuan Luo, Sizhao Wang, Tianrui Li, Hongmei Chen, Jiancheng Lv, and Zhang Yi. 2023. RHDOFS: A distributed online algorithm towards scalable streaming feature selection. *IEEE Transactions on Parallel and Distributed Systems* 34, 6 (2023), 1830–1847.

[50] Helmut Lütkepohl. 2011. *Vector Autoregressive Models*. Springer, Berlin.

[51] Spyros Makridakis, Evangelos Spiliotis, and Vassilios Assimakopoulos. 2020. The M4 competition: 100,000 time series and 61 forecasting methods. *International Journal of Forecasting* 36, 1 (2020), 54–74.

[52] Harry Markowitz. 1952. Portfolio selection. *Journal of Finance* 7, 1 (1952), 77–91.

[53] Daiki Matsunaga, Toyotaro Suzumura, and Toshihiro Takahashi. 2019. Exploring graph neural networks for stock market predictions with rolling window analysis. *CoRR* (2019).

[54] Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. 2011. A three-way model for collective learning on multi-relational data. In *International Conference on Machine Learning, (ICML'11)*.

[55] Yuqi Nie, Nam H. Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. 2023. A time series is worth 64 words: Long-term forecasting with transformers. In *International Conference on Learning Representations (ICLR'23)*.

[56] Boris N. Oreshkin, Dmitri Carpov, Nicolas Chapados, and Yoshua Bengio. 2020. N-BEATS: Neural basis expansion analysis for interpretable time series forecasting. In *8th International Conference on Learning Representations (ICLR'20)*.

[57] Serkan Özen, Volkan Atalay, and Adnan Yazici. 2019. Comparison of predictive models for forecasting time-series data. In *International Conference on Big Data Research*.

[58] Leonardos Pantiskas, Kees Verstoep, and Henri E. Bal. 2020. Interpretable multivariate time series forecasting with temporal attention convolutional neural networks. In *IEEE Symposium Series on Computational Intelligence*.

[59] Aldo Pareja, Giacomo Domeniconi, Jie Chen, Tengfei Ma, Toyotaro Suzumura, Hiroki Kanezashi, Tim Kaler, Tao B. Schardl, and Charles E. Leiserson. 2020. EvolveGCN: Evolving graph convolutional networks for dynamic graphs. In *AAAI Conference on AI (AAAI'20)*.

[60] Jigar Patel, Sahil Shah, Priyank Thakkar, and Ketan Kotecha. 2015. Predicting stock market index using fusion of machine learning techniques. *Expert Systems with Applications* 42, 4 (2015), 2162–2172.

[61] Fotios Petropoulos, Daniele Apiletti, Vassilios Assimakopoulos, Mohamed Zied Babai, Devon K. Barrow, Souhaib Ben Taieb, Christoph Bergmeir, Ricardo J. Bessa, Jakub Bijak, John E. Boylan, Jethro Browell, Claudio Carnevale, Jennifer L. Castle, Pasquale Cirillo, Michael P. Clements, Clara Cordeiro, Fernando Luiz Cyrino Oliveira, Shari De Baets, Alexander Dokumentov, Joanne Ellison, Piotr Fiszeder, Philip Hans Franses, David T. Frazier, Michael Gilliland, M. Sinan Gönül, Paul Goodwin, Luigi Grossi, Yael Grushka-Cockayne, Mariangela Guidolin, Massimo Guidolin, Ulrich Gunter, Xiaojia Guo, Renato Guseo, Nigel Harvey, David F. Hendry, Ross Hollyman, Tim Januschowski, Jooyoung Jeon, Victor Richmond R. Jose, Yanfei Kang, Anne B. Koehler, Stephan Kolassa, Nikolaos Kourentzes, Sonia Leva, Feng Li, Konstantia Litsiou, Spyros Makridakis, Gael M. Martin, Andrew B. Martinez, Sheik Meeran, Theodore Modis, Konstantinos Nikolopoulos, Dilek önkal, Alessia Paccagnini, Anastasios Panagiotelis, Ioannis Panapakidis, Jose M. Pavía, Manuela Pedio, Diego J. Pedregal, Pierre Pinson, Patrícia Ramos, David E. Rapach, J. James Reade, Bahman Rostami-Tabar, MichaŁRubaszek, Georgios Sermpinis, Han Lin Shang, Evangelos Spiliotis, Aris A. Syntetos, Priyanga Dilini Talagala, Thiyanga S. Talagala, Len Tashman, Dimitrios Thomakos, Thordis Thorarinsdottir, Ezio Todini, Juan Ramón Trapero Arenas, Xiaoqian Wang, Robert L. Winkler, Alisa Yusupova, and Florian Ziel. 2022. Forecasting: Theory and practice. *International Journal of Forecasting* 38, 3 (January 2022), 705–871.

[62] Quang Pham, Chenghao Liu, Doyen Sahoo, and Steven Hoi. 2021. Contextual transformation networks for online continual learning. In *International Conference on Learning Representations*.

[63] Quang Pham, Chenghao Liu, Doyen Sahoo, and Steven Hoi. 2023. Learning fast and slow for online time series forecasting. In *The 11th International Conference on Learning Representations*. https://openreview.net/forum?id=q-PbpHD3EOk

[64] Yao Qin, Dongjin Song, Haifeng Chen, Wei Cheng, Guofei Jiang, and Garrison W. Cottrell. 2017. A dual-stage attention-based recurrent neural network for time series prediction. In *International Joint Conference on AI (IJCAI'17)*.

[65] Syama Sundar Rangapuram, Matthias W. Seeger, Jan Gasthaus, Lorenzo Stella, Yuyang Wang, and Tim Januschowski. 2018. Deep state space models for time series forecasting. In *NIPS*.

[66] Akhter Mohiuddin Rather, Arun Agarwal, and V. N. Sastry. 2015. Recurrent neural network and a hybrid model for prediction of stock returns. *Expert Systems with Applications* 42, 6 (2015), 3234–3241.

[67] Ramit Sawhney, Puneet Mathur, Ayush Mangal, Piyush Khanna, Rajiv Ratn Shah, and Roger Zimmermann. 2020. Multimodal multi-task financial risk forecasting. In *ACM International Conference on Multimedia (MM'20)*.

[68] Ramit Sawhney, Arnav Wadhwa, Shivam Agarwal, and Rajiv Ratn Shah. 2021. FAST: Financial news and tweet based time aware network for stock trading. In *Conference of the European Chapter of the Association for Computational Linguistics (EACL'21)*.

[69] Rajat Sen, Hsiang-Fu Yu, and Inderjit S. Dhillon. 2019. Think globally, act locally: A deep neural network approach to high-dimensional time series forecasting. In *NIPS*.

[70] Lei Shi, Zhiyang Teng, Le Wang, Yue Zhang, and Alexander Binder. 2019. DeepClue: Visual interpretation of text-based deep stock prediction. *IEEE TKDE* 31, 6 (2019), 1094–1108.

[71] Shuo Sun, Molei Qin, Xinrun Wang, and Bo An. 2023. PRUDEX-compass: Towards systematic evaluation of reinforcement learning in financial markets. *Transactions on Machine Learning Research* 2023 (2023).

[72] Shuo Sun, Rundong Wang, and Bo An. 2023. Reinforcement learning for quantitative trading. *ACM Transactions on Intelligent Systems and Technologies*, Article 44 (March 2023).

[73] Binh Tang and David S. Matteson. 2021. Probabilistic transformer for time series analysis. In *Annual Conference on Neural Information Processing Systems (NIPS'21)*.

[74] José F. Torres, Dalil Hadjout, Abderrazak Sebaa, Francisco Martínez-Álvarez, and Alicia Troncoso. 2021. Deep learning for time series forecasting: A survey. *Big Data* 9, 1 (2021), 3–21.

[75] Granville Tunnicliffe Wilson. 2016. Time series analysis: Forecasting and control. *Journal of Time Series Analysis* 37, 5 (2016), 709–711.

[76] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NIPS*.

[77] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph attention networks. In *International Conference on Learning Representations (ICLR'18)*.

[78] Renzhuo Wan, Shuping Mei, Jun Wang, Min Liu, and Fan Yang. 2019. Multivariate temporal convolutional network: A deep neural networks approach for multivariate time series forecasting. *Electronics* 8, 8 (2019).

[79] Qingsong Wen, Tian Zhou, Chaoli Zhang, Weiqi Chen, Ziqing Ma, Junchi Yan, and Liang Sun. 2022. Transformers in time series: A survey. *CoRR* (2022).

[80] Neo Wu, Bradley Green, Xue Ben, and Shawn O'Banion. 2020. Deep transformer models for time series forecasting: The influenza prevalence case. *CoRR* (2020).

[81] Neo Wu, Bradley Green, Xue Ben, and Shawn O'Banion. 2020. Deep transformer models for time series forecasting: The influenza prevalence case. *CoRR* (2020).

[82] Zonghan Wu, Shirui Pan, Guodong Long, Jing Jiang, Xiaojun Chang, and Chengqi Zhang. 2020. Connecting the dots: Multivariate time series forecasting with graph neural networks. In *International Conference on Knowledge Discovery and Data Mining (KDD'20)*.

[83] Zonghan Wu, Shirui Pan, Guodong Long, Jing Jiang, and Chengqi Zhang. 2019. Graph WaveNet for deep spatial-temporal graph modeling. In *International Joint Conference on AI (IJCAI'19)*.

[84] Chengjin Xu, Mojtaba Nayyeri, Fouad Alkhoury, Jens Lehmann, and Hamed Shariat Yazdi. 2019. Temporal knowledge graph embedding model based on additive time series decomposition. (2019).

[85] Da Xu, Chuanwei Ruan, Evren Körpeoglu, Sushant Kumar, and Kannan Achan. 2020. Inductive representation learning on temporal graphs. In *International Conference on Learning Representations (ICLR'20)*.

[86] Ikuya Yamada, Akari Asai, Jin Sakuma, Hiroyuki Shindo, Hideaki Takeda, Yoshiyasu Takefuji, and Yuji Matsumoto. 2020. Wikipedia2Vec: An efficient toolkit for learning and visualizing the embeddings of words and entities from wikipedia. In *EMNLP: System Demos*.

[87] Linyi Yang, Tin Lok James Ng, Barry Smyth, and Ruihai Dong. 2020. HTML: Hierarchical transformer-based multi-task learning for volatility prediction. In *The Web Conference (WWW'20)*.

[88] Song Yoojeong, Lee Jae Won, and Lee Jongwooy. 2019. A study on novel filtering and relationship between input-features and target-vectors in a deep learning model for stock price prediction. *Applied Intelligence* 49 (2019), 897–911.

[89] Bing Yu, Haoteng Yin, and Zhanxing Zhu. 2018. Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting. In *ICLR*.

[90] Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. 2023. Are transformers effective for time series forecasting? *Proceedings of the AAAI Conference on Artificial Intelligence*.

[91] George Zerveas, Srideepika Jayaraman, Dhaval Patel, Anuradha Bhamidipaty, and Carsten Eickhoff. 2021. A transformer-based framework for multivariate time series representation learning. In *International Conference on Knowledge Discovery and Data Mining (KDD'21)*.

[92] Liheng Zhang, Charu C. Aggarwal, and Guo-Jun Qi. 2017. Stock price prediction via discovering multi-frequency trading patterns. In *International Conference on Knowledge Discovery and Data Mining (KDD'17)*.

[93] Ling Zhao, Yujiao Song, Chao Zhang, Yu Liu, Pu Wang, Tao Lin, Min Deng, and Haifeng Li. 2020. T-GCN: A temporal graph convolutional network for traffic prediction. *IEEE Transactions on Intelligent Transportation Systems* 21, 9 (2020), 3848–3858.

[94] Chuanpan Zheng, Xiaoliang Fan, Cheng Wang, and Jianzhong Qi. 2020. GMAN: A graph multi-attention network for traffic prediction. In *AAAI Conference on AI (AAAI'20)*.

[95] Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. 2021. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *AAAI Conference on Artificial Intelligence*.

[96] Daniel Zügner, François-Xavier Aubet, Victor Garcia Satorras, Tim Januschowski, Stephan Günnemann, and Jan Gasthaus. 2021. A study of joint graph inference and forecasting. *Time Series Workshop, 38th International Conference on Machine Learning (ICML'21)*.

Received 21 April 2023; revised 16 December 2023; accepted 21 January 2024