3-2023

# Investment and risk management with online news and heterogeneous networks

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. Investment and risk management with online news and heterogeneous networks. (2023). *ACM Transactions on the Web*. 17, (2), 1-24.
**Available at:** https://ink.library.smu.edu.sg/sis_research/7899

# Investment and Risk Management with Online News and Heterogeneous Networks

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

Stock price movements in financial markets are influenced by large volumes of news from diverse sources on the web, e.g., online news outlets, blogs, social media. Extracting useful information from online news for financial tasks, e.g., forecasting stock returns or risks, is, however, challenging due to the low signal-to-noise ratios of such online information. Assessing the relevance of each news article to the price movements of individual stocks is also difficult, even for human experts. In this article, we propose the Guided Global-Local Attention-based Multimodal Heterogeneous Network (GLAM) model, which comprises novel attention-based mechanisms for multimodal sequential and graph encoding, a guided learning strategy, and a multitask training objective. GLAM uses multimodal information, heterogeneous relationships between companies and leverages significant local responses of individual stock prices to online news to extract useful information from diverse global online news relevant to individual stocks for multiple forecasting tasks. Our extensive experiments with multiple datasets show that GLAM outperforms other state-of-the-art models on multiple forecasting tasks and investment and risk management application case-studies.

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

Additional Key Words and Phrases: Graph neural networks, transformers, attention mechanisms, time-series forecasting, networks, multimodality, embeddings, finance, natural language processing

**ACM Reference format:**
Gary Ang and Ee-Peng Lim. 2023. Investment and Risk Management with Online News and Heterogeneous Networks. *ACM Trans. Web* 17, 2, Article 8 (March 2023), 24 pages.
https://doi.org/10.1145/3532858

## 1 INTRODUCTION

Financial forecasting tasks, i.e., forecasting financial time-series such as stock returns or volatilities at the next timestep or some time horizon in the future, are more challenging than other forecasting tasks due to the low signal-to-noise ratios and the non-stationary nature of financial time-series distributions and inter-series relationships [19]. Financial time-series are also influenced by diverse sources of information, such as local (company-specific) stock prices, global (non-company-specific) news, and network effects due to inter-company relationships, i.e., networks comprising

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' address: G. Ang (corresponding author) and E.-P. Lim, Singapore Management University, 80, Stamford Road, Singapore 178902, Singapore; emails: gary.ang.2019@phdcs.smu.edu.sg, eplim@smu.edu.sg.

1559-1131/2023/03-ART8 $15.00
https://doi.org/10.1145/3532858

company nodes with inter-company relationships forming edges between the company nodes. In this article, we address these challenges and propose a model designed for multiple financial forecasting tasks.

While structured numerical information has traditionally been used as information for forecasting stock price movements, the influence of unstructured textual news information on stock price movements has also been studied in recent works [1, 11, 23, 43, 55]. These works show that both traditional and online textual news contain valuable information that can be used to forecast stock price movements. Extracting useful information from online news for financial tasks is, however, more challenging than from structured numerical information or traditional textual news information for a number of reasons.

First, sources of online news are more diverse, ranging from online news outlets to blogs and social media, with vastly different writing styles. Second, the signal-to-noise ratios of online news is lower. A larger proportion of online news is of low quality or false, as most online news are not subject to the same editorial processes and controls as traditional news. Finally, the relevance of each online news article to individual stock prices is difficult to determine, even for human experts, due to the high volume, velocity, and variety of online news.

To address the above challenges, this article introduces several *key ideas* premised on the following observations relating to online news information, financial time-series, and inter-company networks:

- First, unlike a piece of structured information such as stock price and trading volume histories, which are explicitly associated with specific companies and thus *local* in nature, a piece of unstructured online news may not be associated with specific companies but relevant to multiple companies, an entire industry sector, or the entire market. Hence, we consider these online news *global.* For example, a published news article on disruptions in the operations of a main semiconductor supplier can affect many other companies not only in the semiconductor sector, but also other companies in the downstream industries such as automobiles and computing equipment. Hence, global information could influence local information. This degree of global information's influence differs based on the relevance and quality of global information, e.g., high-quality news on a company that is verified to be true will have a more sustained effect on the stock prices of the company than low-quality news that turns out to be false. Local information may also influence global information, e.g., a sustained decline in stock prices or trading volumes of a key company in a property sector could lead to news articles discussing the vulnerabilities of the entire sector. Such *mutual effects of local information and global information* could be leveraged for extraction of relevant information.
- Both global and local information often come from different modalities, e.g., local numerical price-related information and global textual news information, as mentioned earlier. As the evolution of local stock-specific information from one modality could provide important signals that can be leveraged to extract relevant global information from another modality, a *multimodal approach* to modeling them is thus necessary.
- Next, the ex-post effects of local information can provide valuable signals that can be utilized to extract relevant global information and address the low signal-to-noise ratio, e.g., a personal scandal related to the CEO of Company A might not be related to its core business, but could spark an online-led boycott of Company A's products and lead to a significant decline in its stock price. Such *significant ex-post responses of local information*, i.e., stock prices, therefore provide an important set of signals that can be isolated to learn the relevance of different global information to each company.
- Fourth, different types of inter-company relationships and linkages capture different influences among them. For example, an online negative news article about a Company B could

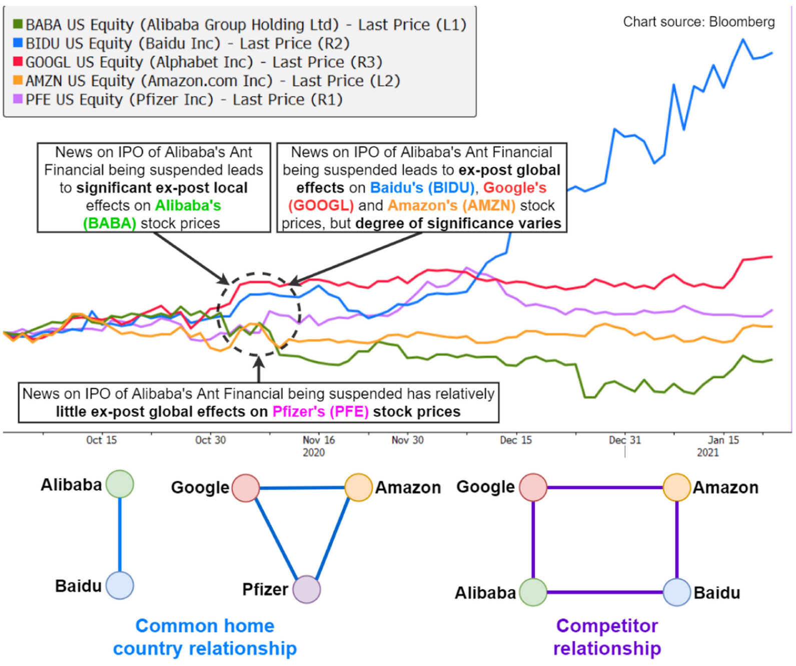

Fig. 1. Motivating example: Effects of news relating to Alibaba's Ant Financial's initial public offering (IPO) being suspended on the immediate stock prices of different companies varies based on heterogeneous relationships. The significance and duration of such effects also varies based on the news content and such heterogeneous relationships.

have negative effects on the stock prices of all its suppliers and customers, but positive effects for its known competitor; a regulatory crackdown on Company C based in a country due to a shift in government policy could affect companies of the same industry in the same country negatively but have positive effects for companies of the same industry in other countries. Such *heterogeneous relationships* between companies, both direct and indirect, can provide structure for learning the relevance of different online news to individual stock prices.

We illustrate these observations with an example in Figure 1. News on the suspension of Alibaba's Ant Financial's initial public offering (IPO) in October 2020 led to a significant decline in the stock price of Alibaba (green line) after that. Stock prices of its *competitors*—Baidu (blue line), Amazon (orange line), and Google (red line)—in the technology sector rise, but to varying degrees after the event. Baidu, being a *competitor* to Alibaba in the *same home country*, i.e., China, appears to increase the most (see the different types of relationships in Figure 1). The longer-term effect of news on the suspension of Alibaba's Ant Financial's IPO on the stocks prices of the different companies also varies. The decline in Alibaba's stock price and rise in Baidu's stock price are more sustained, whereas the changes in stock prices of Google and Amazon level off after November 2020. Stock prices of Pfizer (purple line), a pharmaceutical company without direct links to Alibaba, is relatively unaffected by this news.

In our literature survey, we have noticed several limitations of existing works. Most existing works in this area model financial information of a single modality [13, 23, 43] and do not model the mutual relevance of information from different modalities, or the effects of heterogeneous inter-company relationships. Some works [17, 55] model both unimodal financial information and the effects of inter-company relationships, but not multimodal information. Ang and Lim [1] utilizes multimodal numerical and global textual information as well as inter-company relationships but does not capture the heterogeneity of inter-company relationships. All these works also do not explicitly isolate the more significant ex-post responses of stock prices to learn the relevance of different global information to each company.

Most existing works also focus on forecasting stock prices or returns for trading decisions at the next timestep and do not study the effect of online news on the dynamics of stock prices over a longer future horizon, covering multiple timesteps [23, 43]. Hence, another set of *key ideas* that motivates our article relates to the need for a *multivariate multitask setting* for forecasting the dynamics of stock prices over a longer future horizon. Such a setting is important for investment and risk management applications such as portfolio management and risk forecasting. Investment and risk managers are not only interested in investment returns but also investment risks and other aspects of investment and risk management. Investment and risk managers make investment and risk management decisions over a longer term horizon and are hence also interested in the expected returns and risks (volatilities) of stocks over a longer term horizon, rather than just stock price movements at the next timestep. Investment and risk managers also manage large numbers of stocks in portfolios and are hence also interested in how changes in correlations between stocks affect the overall returns of the portfolio. Therefore, financial forecasting for investment and risk management naturally involves a multivariate multitask setting, where there is a need to manage the returns and risks of financial portfolios that comprise many stocks, and forecast stock mean returns and risks over a future horizon to balance potential returns and risks when making investment decisions, as well as forecast correlations between stocks in portfolios over a future horizon. Designing a model that can be used in a multivariate multitask setting has other potential advantages, as it could enable complementary information from other variables and related tasks to be used to improve overall forecasting performance, and also lower the risk of over-fitting on any one task. Such forecasts can also be utilized in portfolio allocation optimization [35] and Value-at-Risk (VaR) [32] forecasting applications.

To address the above-mentioned challenges in utilizing global online news information in a multivariate multitask setting for investment and risk management based on these key ideas, we propose the **G**uided Global-**L**ocal **A**ttention-based **M**ultimodal Heterogeneous Network **(GLAM)** model. GLAM incorporates several important components: (i) a time-sensitive global-local transformer to learn relevant global online text information and sequentially encode multimodal information (i.e., time-series stock prices and time-stamped news articles); (ii) an attention-based heterogeneous network encoder to leverage heterogeneous inter-company relationships and future correlations; coupled with (iii) auxiliary channels for guided learning from significant ex-post effects of online news. GLAM is trained in a multivariate setting on multiple tasks—forecasting means, volatilities, and correlations over a future horizon. We also demonstrate how such forecasts could be used for portfolio allocation optimization and risk management applications in case-studies. Our key contributions are as follows:

- To our knowledge, this is the first work to propose a model for capturing global and local information from multiple modalities and heterogeneous networks for multivariate multitask financial forecasting tasks for investment and risk management applications.
- We propose a time-sensitive global-local transformer module that encodes sequences of global textual information, local numerical information, and associated time features jointly and extracts the relevant global information.
- To improve extraction of relevant global information, we couple the global-local transformer module with auxiliary channels that enable the significant changes in stock dynamics to be isolated for guiding the learning of global information relevant to each company.

- We design a heterogeneous network encoding module that uses different types of inter-company relationships to propagate multimodal sequential information across companies. The heterogeneous network encoding module also leverages correlation forecasts to improve parameter learning.
- We train the model on multiple forecasting tasks to lower the risk of over-fitting and demonstrate the effectiveness of GLAM on forecasting tasks and real-world applications against state-of-the-art baselines on real-world datasets.

## 2 RELATED WORK

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

### 2.1 Stock Price Forecasting Using Time Series Modeling

Classical methods, which include univariate Autoregressive Integrated Moving Average (ARIMA) [47], Generalized AutoRegressive Conditional Heteroskedastic (GARCH) [5]; and multivariate Vector Auto-Regressive (Vector AR) [34] and Dynamic Conditional Correlation (DCC)-GARCH [14] models are commonly applied to time-series forecasting. However, such classical methods are designed for numerical data but not unstructured textual information.

To learn time-series information in a data-driven manner and to capture other types of information, deep learning models have been increasingly applied to time-series forecasting. They include feed-forward networks [8, 10, 11, 36, 58], convolutional neural networks [2, 6, 38, 51], recurrent neural networks [18, 31, 33, 41, 42], and transformers [53, 60]. A detailed review of these works can be found in Faloutsos et al. [16], Jiang [24], Lim and Zohren [30], Özen et al. [37], Petropoulos et al. [40], Torres et al. [46]. **TST** [60] is a recent model based on the transformer encoder architecture. It is, however, designed for local numerical information. A number of recent works have studied the use of textual information from traditional and online news [1, 11, 13, 23, 43, 55] for financial forecasting. Most of these works [23, 43] utilize news articles that have been manually tagged to specific companies as inputs, i.e., local textual information. **FAST** [43] is a recent model that uses Time-aware LSTMs [3] to encode sequences of local textual news information. **HAN** [23] utilizes attention mechanisms to learn the importance of each local news article and each timestep. **SE** [13] is a recent model that does not manually assign company tags to news articles but instead uses the dot product of stock embeddings and news representations to extract relevant global news via a data-driven approach. SE utilizes bidirectional GRUs to encode unimodal textual information but does not capture numerical information or inter-company relationships.

### 2.2 Heterogeneous Network Learning and Stock Price Forecasting

Graph Neural Networks (GNN) compose messages based on network features and propagate them to update the embeddings of nodes and/or edges over multiple neural network layers [20]. In particular, Graph Convolutional Network (GCN) [27] aggregates features of neighboring nodes and normalizes aggregated representations by node degrees. Graph Attention Network (GAT) [50] assigns neighboring nodes with different importance weights during aggregation. Such GNNs are designed for homogeneous networks with static node attributes and cannot be directly applied to heterogeneous networks where attributes are evolving time series.

GNNs have also been applied to heterogeneous networks. Relational Graph Convolutional Networks (RGCN) [44] and Graph Convolutional Matrix Completion [48] use multiple GCNs to encode embeddings of multiple adjacency matrices, one for each edge type, before aggregating them. Heterogeneous Graph Attention Network [52] and General Attributed Multiplex Heterogeneous Network [7] use multiple GNN-based layers to encode networks formed from different metapaths

[12] before using an attention mechanism to aggregate the embeddings. **HGT** [22] uses attention mechanisms to encode sub-graphs with different node and edge-types iteratively. Similarly, such GNNs are designed for heterogeneous networks with static node attributes and cannot be directly applied to heterogeneous networks where attributes are evolving time series. GNN models have been designed for spatio-temporal networks where the nodes have time-varying attributes [9, 28, 59, 61]. However, these models are designed for traffic-related numerical information and not suitable for networks where the node attributes are multimodal financial time series.

A few recent works extend different GNNs to prediction tasks on financial time-series data. **RSR** [17] uses LSTM to generate representations for local numerical time-series stock prices before feeding the latter to learn stock embeddings in a heterogeneous network using a GCN-based model but does not consider global textual information. Reference [55] captures heterogeneous relationships using a RGCN-based model but is designed for forecasting with different types of company announcements and not news. Our earlier work **KECE** [1] captures numerical and global textual information and uses a GAT-based model to capture homogeneous inter-company relationships but does not capture the heterogeneity of inter-company relationships. KECE also uses the dot product of stock embeddings and news representations to extract relevant global news information, rather than attention mechanisms based on transformers proposed in this work.

In **general**, these related **forecasting** and **network learning** works are also not designed for the multitask setting of forecasting means, volatilities, and correlations of stock returns that are important for investment and risk management applications. They are designed for single tasks, and either predict prices or returns or price movement directions for trading applications. Further, they do not isolate the significant ex-post changes in local stock dynamics to improve the learning of global information relevant to each company.

## 3 GUIDED GLOBAL-LOCAL ATTENTION-BASED MULTIMODAL HETEROGENEOUS NETWORK MODEL

GLAM represents companies in a network $G = (V, E, X)$, where $V$ represents a set of company nodes, $E$ consists of edges of $R$ different relationship types, i.e., $E = E_1 \cup \cdots \cup E_R$, $X$ represents sequences of multimodal numerical and textual attributes. In this article, we utilize heterogeneous relationships between companies extracted from Wikidata knowledge graphs. Other inter-company relationships, e.g., based on domain knowledge, can also be used, but will be explored in future work. Given a timestep $t$, we define the local numerical features of a company $v_j$, $X_j^{num,local}(t)$, to be the sequence of numerical price-related data associated with $v_j$ over a window of $K$ timesteps up to $t-1$, i.e., $X_j^{num,local}(t) = [x_j^{num,local}(t-K), \ldots, x_j^{num,local}(t-1)]$. We use $X^{num,local}(t)$ to represent the $|V| \times K$ matrix containing the $X_j^{num,local}(t)$ of all companies. The pre-encoded textual news features that are global in nature and not associated with any company are denoted as $X^{txt,global}(t) = [x^{txt,global}(t-K), \ldots, x^{txt,global}(t-1)]$ over the same window period $[t-K, t-1]$, with varying number of news articles $|N_{t-k}|$ at each timestep $t-k$ in $[t-K, t-1]$. Alternative local and global inputs, e.g., local social media information such as tweets from the company's social media account, and global economic indicators, e.g., gross domestic product of countries of the company's key markets, are also possible within our framework, but we focus on global online news and local stock-price related numerical information in this article and will explore other inputs in future work.

As shown in Figure 2, the Guided Global-Local Transformer (GLT) module in GLAM first utilizes local numerical information $X^{num,local}(t)$ (over the time window $[t-K, t-1]$) to extract and sequentially encode relevant global textual information $X^{txt,global}(t)$ in the same time window. The resultant sequence of representations for each company $(H_i(t)[t-K], \ldots, H_i(t)[t-1])$ are

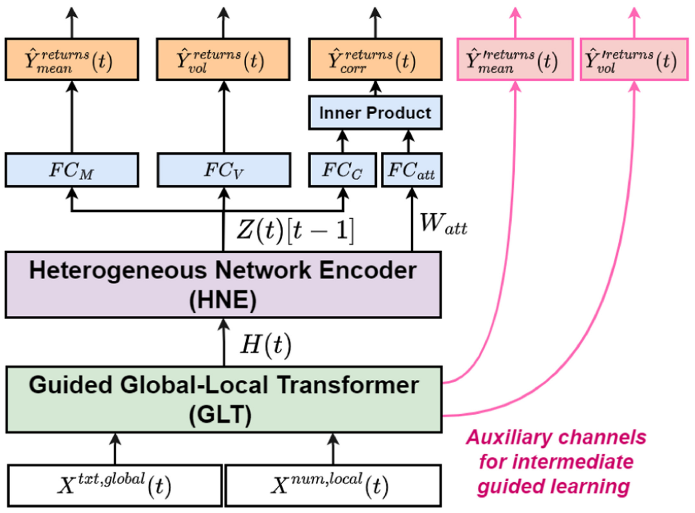
$\hat{Y}^{returns}_{mean}(t)$
$\hat{Y}^{returns}_{vol}(t)$
$\hat{Y}^{returns}_{corr}(t)$
$\hat{Y}'^{returns}_{mean}(t)$
$\hat{Y}'^{returns}_{vol}(t)$
Inner Product
$FC_M$
$FC_V$
$FC_C$
$FC_{att}$
$Z(t)[t-1]$
$W_{att}$
Heterogeneous Network Encoder (HNE)
$H(t)$
Guided Global-Local Transformer (GLT)
$X^{txt,global}(t)$
$X^{num,local}(t)$
Auxiliary channels for intermediate guided learning

Fig. 2. Architecture of GLAM. Detailed Architecture of Guided Global-Local Transformer (GLT) and Heterogeneous Network Encoder (HNE) modules are depicted in Figures 3 and 4, respectively.

then used as inputs to a Heterogeneous Network Encoding (HNE) module that captures the heterogeneous relationships between companies. GLAM finally generates forecasts of means, volatilities, and correlations of financial returns of each company $v_i$ over a selected future horizon of $[t, t+K']$. These financial returns are denoted as

$$Y^{returns}_i(t) = [y^{returns}_i(t), \ldots, y^{returns}_i(t+K')],$$

where $y^{returns}_i(t) = (price_i(t) - price_i(t-1))/price_i(t-1)$ and $price_i(t)$ denote the percentage return and stock price of $v_i$ at time window $[t, t+K']$ and timestep $t$, respectively. We also use $Y^{returns}(t)$ and $y^{returns}(t)$ to denote the percentage return of all companies at time window $[t, t+K']$ and timestep $t$, respectively. To facilitate GLT and HNE in learning the relevant global textual information and important heterogeneous relationships, we add **auxiliary channels for intermediate guided learning** to the GLAM model. The auxiliary channels associated with GLT utilize intermediate forecasts of the most significant means and volatilities of $Y^{returns}(t)$ across all stocks to guide learning of GLT parameters. HNE is also guided by the learning of an inner weight $W_{att}$ that is utilized in both the heterogeneous network encoding as well as the forecasts of the correlations of $Y^{returns}(t)$. We further elaborate on the GLAM modules, auxiliary channels, and training objectives below.

## 3.1 Guided Global-Local Transformer

Transformers [49] were originally proposed for natural language applications but have since been extended to time-series forecasting [29, 53, 60]. Such time-series transformer works [29, 53, 60] use local information (usually numerical), i.e., information directly associated with specific companies, or other variables. To extract and learn global textual information relevant to each company, we design the Guided Global-Local Transformer (GLT), that differs from these past transformer-related works in a number of important aspects, which we elaborate on below.

As shown in Figure 3, the GLT module in Figure 3(i) comprises multiple GLT layers shown in detail in Figure 3(ii). We start from the GLT layer as shown in Figure 3(ii).

**Time vectorization and projection.** First, we utilize learned time matrices [21, 25] to enable GLT to generate time-sensitive representations. Unlike the usual positional encodings used in

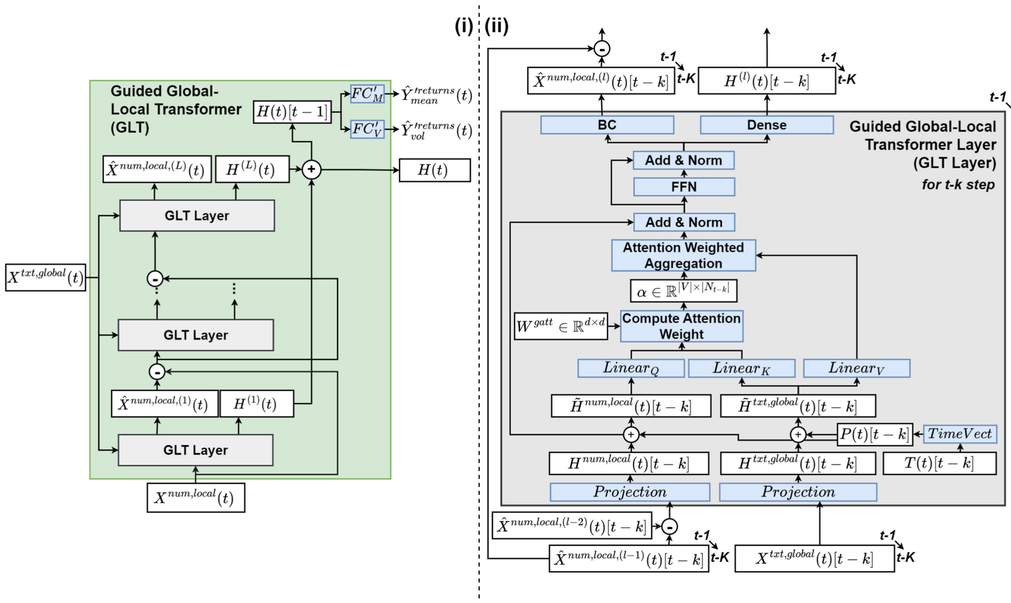
(i)
(ii)
Guided Global-Local Transformer (GLT)
GLT Layer
Guided Global-Local Transformer Layer (GLT Layer) for t-k step
BC
Dense
Add & Norm
FFN
Attention Weighted Aggregation
Compute Attention Weight
Projection
TimeVect
t-1
t-K

Fig. 3. (i) Guided Global-Local Transformer (GLT) comprises multiple GLT layers, which repeatedly extract relevant global representations across all companies with residual local backcasts as inputs and sums them up to obtain the final relevant global representations $H(t)$ for all timesteps in the window $[t-K, t-1]$. (ii) Each GLT layer iteratively extracts relevant global representations across the window from $t-1$ to $t-K$ step in a time-sensitive manner and generates backcasts. Figure (ii) shows the process for the $t-k$ step, which is repeated from the $t-1$ to the $t-K$ step.

transformers, the time matrix $P(t)$ is learned from the set of timestamps $T(t)$ corresponding to the day of week, week and month of year of timesteps $[t-K, t-1]$, as these are most relevant to the respective inputs. The time matrix $P(t) \in \mathbb{R}^{K\times d}$ is learned by combining functional forms and learnable weights. For GLAM, the empirically chosen functional components are $\Phi_1 = sigmoid(Linear(T(t)))$ and $\Phi_2 = cos(Linear(T(t)))$, which enable the model to extract non-linear and seasonality-based temporal patterns in $T(t)$. We then concatenate these matrices $\Phi_1$ and $\Phi_2$ and project them via a linear layer to obtain the time matrix: $P(t) = Linear([\Phi_1 || \Phi_2])$.

Concurrently, we project either (i) the original local numerical features $X^{num,local}(t)$ with dimension $d^{num}$ corresponding to stock market price-related information, say, opening, closing, low, high prices, and trading volumes; or (ii) the residual local numerical features, i.e., the difference between $\hat{X}^{num,local,(l-1)}(t)$ and $\hat{X}^{num,local,(l-2)}(t)$ of the prior layers (which we will elaborate on later in this section) to representations $H^{num,local}(t)$ with dimension $d$. We also project $X^{txt,global}(t)$ with dimension $d^{txt}$ corresponding to the dimensions of average word embeddings of each news article generated with a pre-trained encoder to $H^{txt,global}(t)$ with dimension $d$ again but of different sequence lengths. That is, $H^{num,local}(t) \in \mathbb{R}^{|V|\times K\times d}$, $H^{txt,global}(t) \in \mathbb{R}^{\sum_{k=1}^{K}|N_{t-k}|\times d}$. $H^{txt,global}(t)$ is not specific to any company, and there are varying $|N_{t-k}|$ number of global news representations for each timestep. We broadcast the time matrix $P(t) \in \mathbb{R}^{K\times d}$ by repeating it $|V|$ times, resulting in a $|V|\times K\times d$ matrix, and add it to the local numerical representations to obtain: $\tilde{H}^{num,local}(t) = H^{num,local}(t) + P(t)$. We similarly broadcast the time matrix to match the dimensions of the global news representations, i.e., $\sum_{k=1}^{K}|N_{t-k}| \times d$, and add them to the global news representations to obtain: $\tilde{H}^{txt,global}(t) = H^{txt,global}(t) + P(t)$.

**Cross attention.** Second, in contrast to transformers that apply self-attention to the same sequence of representations that are of equal length, we apply cross-attention between the local numerical representation and $|N_{t-k}|$ global news representations at each timestep $t-k$ in time window $[t-K, t-1]$. We use the local representations $\tilde{H}^{num,local}(t)$ to guide the learning of global textual features from $\tilde{H}^{txt,global}(t)$ relevant to each company $v_i$. As shown in Figure 3(ii), for each timestep in window $[t-K, t-1]$, say, $t-k$, we generate:

$$Q^{num,local}(t)[t-k] = Linear_Q(\tilde{H}^{num,local}(t)[t-k])$$

$$K^{txt,global}(t)[t-k] = Linear_K(\tilde{H}^{txt,global}(t)[t-k])$$

$$V^{txt,global}(t)[t-k] = Linear_V(\tilde{H}^{txt,global}(t)[t-k]).$$

We apply a scaled dot-product *attention weighted aggregation* step that is different from standard transformers [49]:

$$\acute{H}^{txt,global}(t)[t-k] = softmax\left(\frac{K^{txt,global}(t)[t-k] \cdot W^{gatt} \cdot Q^{num,local}(t)[t-k]^{\intercal}}{\sqrt{d}}\right)^{\intercal} \cdot V^{txt,global}(t)[t-k], \tag{1}$$

where $W^{gatt} \in \mathbb{R}^{d \times d}$ is an inner weight shared across all timesteps in window $[t-K, t-1]$ to improve attention extraction of global textual information. The matrix multiplication between $K^{txt,global}(t)[t-k]$, $W^{gatt}$ and $Q^{num,local}(t)[t-k]$, after the scaling by $\sqrt{d}$ and the $softmax$ step, gives us attention weights of dimensions $|N_{t-k}| \times |V|$. We then use the transpose of these attention weights to map the $|N_{t-k}| \times d$ matrix $V^{txt,global}(t)[t-k]$ to $\acute{H}^{txt,global}(t)[t-k]$, which is of $|V| \times d$ dimensions. Across all timesteps $t-k$'s in window $[t-K, t-1]$, we get $\acute{H}^{txt,global}(t) \in \mathbb{R}^{|V| \times K \times d}$. We then apply a series of addition steps, layer normalization, and feed-forward networks as per conventional transformers [49] to $\acute{H}^{txt,global}(t)$ as follows: $H'(t) = LayerNorm(\acute{H}^{txt,global}(t) + \tilde{H}^{num,local}(t))$; followed by $H''(t) = LayerNorm(FFN(H'(t)) + H'(t))$.

**Backcast.** Third, in contrast to transformers layers that only generate representations for subsequent layers to encode or decode, the GLT layer not only generates $H^{(l)}(t) = Dense(H''(t))$ with $H^{(l)}(t) \in \mathbb{R}^{|V| \times K \times d}$ but also generates a backcast of the local numerical information $\hat{X}^{num,local,(l)}(t) = BC(H''(t))$ where $\hat{X}^{num,local,(l)}(t) \in \mathbb{R}^{|V| \times K \times d^{num}}$.

**Residual stacking.** Fourth, as shown in Figure 3(i), instead of the typical encoder-decoder architecture in transformers, we stack multiple GLT layers with a residual connection between the $l$th GLT layer's backcast of the local numerical information: $\hat{X}^{num,local,(l)}(t)$ and the local numerical information used as inputs to the $l$th GLT layer: $\hat{X}^{num,local,(l-1)}(t)$. The difference between the backcast of the local numerical information and the prior local numerical information $\hat{X}^{num,local,(l-1)}(t)$ is used as inputs to the subsequent $l+1$ GLT layer; while the representations of relevant global information $H^{(l)}(t)$ for each of the GLT layers are added to obtain the final $H(t) = \sum_{l=1}^{L} H^{(l)}(t) (\in \mathbb{R}^{|V| \times K \times d})$. This residual stacking architecture is inspired by Reference [36], but to our best knowledge has thus far not been applied for the extraction of relevant global information using transformers. The multiple GLT layers can be viewed as multi-stage extraction of global information that are relevant to each company. The residual connection removes the part of the local information that has already been utilized to extract part of the global information relevant to each company and facilitates the extraction task of the subsequent GLT layers.

**Intermediate forecasts.** Finally, to enable the GLT module to effectively extract global information relevant to each company, we add fully connected layers $FC'_M$ and $FC'_V$ and use the representation of the final step $t-1$ in the window $[t-K, t-1]$ to make intermediate forecasts of the

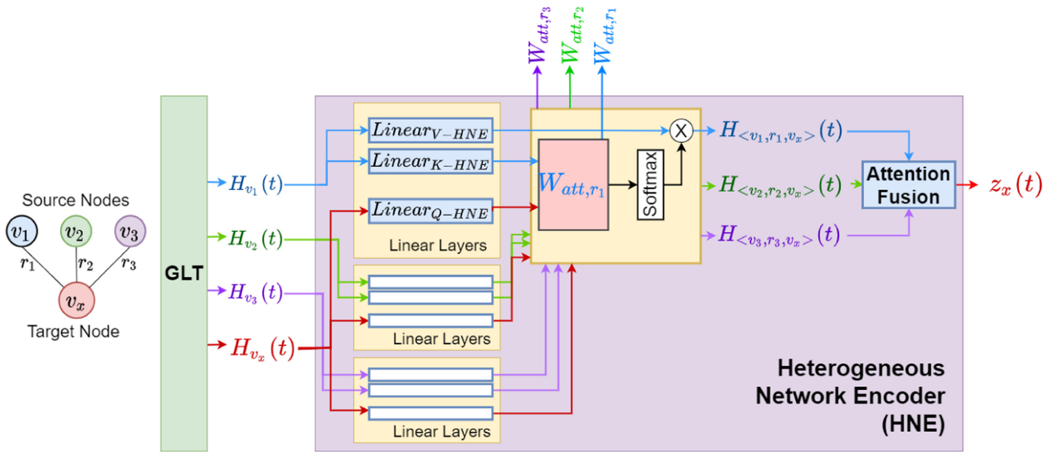

Fig. 4. Heterogeneous Network Encoder (HNE).

most significant means and volatilities, respectively, of $Y^{returns}(t)$ across all stocks as follows:

$$\hat{Y}'^{returns}_{mean}(t) = FC'_M(H(t)[t-1])$$

$$\hat{Y}'^{returns}_{vol}(t) = FC'_V(H(t)[t-1]).$$

The training loss functions for these intermediate forecasts will be elaborated in Section 3.3. The intermediate forecasts are also designed to alleviate the over-smoothing in the subsequent network encoding step [54], which causes representations for all nodes in a network to become very similar to one another and can lead to poorer performance.

## 3.2 Heterogeneous Network Encoder

The heterogeneous network encoder utilizes the heterogeneous relationships between companies to propagate representations between companies based on different relationship types $\{1, \ldots, R\}$. Inspired by References [22, 57], we adapt the scaled dot-product attention module commonly used in transformers [49] for the GNN message-passing framework to design a Heterogeneous Network Encoder (HNE), as shown in Figure 4.

The scaled dot-product attention mechanism used in transformers applied to networks utilizes network structure to compute attention scores between nodes that are neighbors. These attention scores weigh the messages propagated from the source to target nodes for aggregation. Using scaled dot-product attention is more effective than the usual message-passing framework employed in most GNNs, as it allows the model to perform the message composition, propagation, and update steps in the GNN message-passing framework based on the self-discovered relative importance of each neighboring source node and relationship-types.

HNE extracts edges linking neighboring source company nodes $v_s$'s to a target company node $v_x$ as canonical triplets $\langle v_s, r, v_x\rangle$'s (i.e., $v_s, v_x \in V$, $r \in \{1, \ldots, R\}$) from the heterogeneous network. For each canonical triplet, we first utilize linear layers to encode $H(t)$ of the company nodes in time window $[t-K, t-1]$ from the prior GLT module as queries, keys, and values:

$$Q_{v_x}(t) = Linear_{Q-HNE}(H_{v_x}(t))$$

$$K_{v_s}(t) = Linear_{K-HNE}(H_{v_s}(t))$$

$$V_{v_s}(t) = Linear_{V-HNE}(H_{v_s}(t)).$$

We then compute the attention score *AttScore* between a target node $v_x$ and each neighboring source node $v_s \in N(v_x)$ as:

$$AttScore_{\langle v_s, r, v_x \rangle}(t) = softmax_{v_s \in N(v_x)} \left( \frac{Q_{v_x}(t) \cdot W_{att,r} \cdot K_{v_s}(t)^\intercal}{\sqrt{d}} \right), \tag{2}$$

where $N(v_x)$ denotes the neighboring nodes of $v_x$, and $W_{att,r}$ is a $d \times d$ learnable weight matrix. Next, we use the attention score *AttScore* to compute the weighted average of features from all source nodes and use it to update the triplet-specific representation of the target node $v_x$.

$$H_{\langle v_s, r, v_x \rangle}(t) = \sum_{v_s \in N(v_x)} AttScore_{\langle v_s, r, v_x \rangle}(t) \cdot V_{v_s}(t) \tag{3}$$

At this point, we have the embeddings of the target node $v_x$ for each of the canonical triplets or edges connected to neighboring nodes $N(v_x)$. To learn the importance of different edges, we use attention-based fusion. A non-linear transformation is applied to the representations to obtain scalars $s(v_s, r, v_x, t) = W_\omega^{(1)} tanh(W_\omega^{(0)} H_{\langle v_s, r, v_x \rangle}(t) + b_\omega)$, where $W_\omega^{(0)}$ and $W_\omega^{(1)}$ are learnable weight matrices and $b_\omega$ is the bias vector. Parameters are shared across modalities. We normalize the scalars with a softmax function to obtain the weights $\beta_{(v_s, r, v_x)}(t)$'s, which are used to fuse representations across the edges into node $v_x$ as $z_x(t)$'s.

$$\beta_{(v_s, r, v_x)}(t) = \frac{exp(s(v_s, r, v_x, t))}{\sum_{(v_s, r, v_x)} exp(s(v_s, r, v_x, t))} \tag{4}$$

$$z_x(t) = \sum_{(v_s, r, v_x)} \beta_{(v_s, r, v_x)}(t) H_{\langle v_s, r, v_x \rangle}(t) \tag{5}$$

Repeating these steps across all company nodes and edges results in $Z(t) \in \mathbb{R}^{|V| \times K \times d}$. We use the representation of the final step $t-1$ in the window $[t-K, t-1]$, i.e., $Z(t)[t-1] \in \mathbb{R}^{|V| \times d}$ in the subsequent steps. Similarly, across all company nodes and edges, we learn $R$ attention weights $W_{att,r}$, each of dimension $d \times d$, and stack them to get $W_{att}$ of dimension $d \times d \times |R|$.

## 3.3 Forecasting and Loss Functions

We use fully connected layers to generate *final forecasts* of means and volatilities of stock returns over the selected horizon period $[t, t+K']$:

$$\hat{Y}^{returns}_{mean}(t) = FC_M(z_{t-1})$$
$$\hat{Y}^{returns}_{vol}(t) = FC_V(z_{t-1}),$$

where $z_{t-1} = Z(t)[t-1]$. As described in Section 3.2, we stack $R$ attention weights $W_{att,r}$, each of dimension $d \times d$, from HNE to get $W_{att}$ of dimension $d \times d \times |R|$ and forecast correlations of asset returns over the horizon period $[t, t+K']$ as:

$$\hat{Y}^{returns}_{corr}(t) = tanh\left( \frac{z_{t-1} \cdot FC_C(W_{att}) \cdot z_{t-1}^\intercal}{\sqrt{d'}} \right), \tag{6}$$

where $FC_C$ is a fully connected layer that projects $W_{att}$ from dimensions $d \times d \times |R|$ to dimensions $d \times d$.

The final forecasts $\hat{Y}^{returns}_{mean}(t)$, $\hat{Y}^{returns}_{vol}(t)$, and $\hat{Y}^{returns}_{corr}(t)$ are utilized alongside the intermediate forecasts $\hat{Y}'^{returns}_{mean}(t)$ and $\hat{Y}'^{returns}_{vol}(t)$ from GLT, as described in Section 3.1, to learn GLAM's parameters. Doing so enables GLT to learn to extract the relevant global news information well and hence improve the representations that are propagated between companies based on different relationship types in HNE.

The respective ground-truths, i.e., actual means, volatilities, and correlations over the horizon $[t, t+K']$ for each stock, i.e., $v_i$, are computed from the observed stock returns as follows:

$$y^{returns}_{mean,i}(t) = \frac{1}{K'} \sum_{k'=0}^{K'} y^{returns}_i(t+k'), \tag{7}$$

$$y^{returns}_{vol,i}(t) = \sqrt{\frac{1}{K'} \sum_{k'=0}^{K'} (y^{returns}_i(t+k') - \mu_i)^2}, \tag{8}$$

where $\mu_i = y^{returns}_{mean,i}(t)$. For correlations between any two companies $i$ and $j$:

$$y^{returns}_{corr,i,j}(t) = \frac{\sum_{k'=0}^{K'} (y^{returns}_i(t+k') - \mu_i)(y^{returns}_j(t+k') - \mu_j)}{\sqrt{\sum_{k'=0}^{K'} (y^{returns}_i(t+k') - \mu_i)^2}\sqrt{\sum_{k'=0}^{K'} (y^{returns}_j(t+k') - \mu_j)^2}}. \tag{9}$$

For the *main training losses*, we compute losses between the forecasts above and respective ground-truths, i.e., actual means, volatilities, and correlations over the horizon $[t, t+K']$ with root mean squared loss (RMSE) across all companies $v_i \in V$:

$$\begin{aligned} \mathcal{L}_{main} = {} & \mathcal{L}_{RMSE}\left(Y^{returns}_{mean}(t), \hat{Y}^{returns}_{mean}(t)\right) \\ & + \mathcal{L}_{RMSE}\left(Y^{returns}_{vol}(t), \hat{Y}^{returns}_{vol}(t)\right) \\ & + \mathcal{L}_{RMSE}\left(Y^{returns}_{corr}(t), \hat{Y}^{returns}_{corr}(t)\right). \end{aligned} \tag{10}$$

For the *intermediate forecasts*, we filter out subsets of the stocks for each training iteration, one for forecasts of means and one for forecasts of volatilities, i.e., $V'_{mean} \subseteq V$ and $V'_{vol} \subseteq V$, with the most significant ex-post response in the horizon period $[t, t+K']$ to online news in the window period $[t-K, t-1]$. This enables GLT to capture the most significant ex-post effects of online news and improve its extraction and learning of global textual information relevant to each company. We further design an adaptive learning process similar to curriculum learning [45] but undertaken at a much earlier stage of the GLAM model. Curriculum learning is a training strategy that imitates the way humans learn by gradually increasing the difficulty of the data samples used to train a model [4]. In the context of this article, more significant ex-post responses to online news, i.e., higher than average means and volatilities of returns in the horizon period $[t, t+K']$, represent easier training data samples for the extraction of global textual information relevant to specific companies. We use thresholds $\tau_{mean}$ and $\tau_{vol}$ to select companies for $V'_{mean}$ and $V'_{vol}$, respectively. $\tau_{mean}$ and $\tau_{vol}$ are computed as follows:

$$\tau_{mean} = M_{mean} - \gamma \times (M_{mean} - \mu_{mean}), \tag{11}$$

$$\tau_{vol} = M_{vol} - \gamma \times (M_{vol} - \mu_{vol}), \tag{12}$$

where

$$\mu_{mean} = \frac{1}{|V|} \sum_{v_i \in V} y^{returns}_{mean,i}(t),$$

$$\mu_{vol} = \frac{1}{|V|} \sum_{v_i \in V} y^{returns}_{vol,i}(t).$$

The maximum of return means and volatilities $M_{mean}$ and $M_{vol}$ are defined by: $M_{mean} = \max_{v_i \in V} y^{returns}_{mean,i}(t)$ and $M_{vol} = \max_{v_i \in V} y^{returns}_{vol,i}(t)$. The parameter $\gamma = \frac{e}{E} + \eta$ is updated based on the training epochs $e \in \{1, \ldots, E\}$, where $E$ refers to the total number of training epochs. The $\eta$

hyper-parameter is set to a fraction, say, 0.5, which represents a base proportion of the company nodes to be utilized at the start of training.

The subset of company nodes for the most significant ex-post returns means and the subset of company nodes for the most significant ex-post returns volatilities in the horizon period $[t, t+K']$ are hence:

$$V'_{mean} = \left\{ v_i | y^{returns}_{mean,i}(t) > \tau_{mean} \right\}$$
$$V'_{vol} = \left\{ v_i | y^{returns}_{vol,i}(t) > \tau_{vol} \right\}.$$

Hence, for the *auxiliary training losses*, we compute the auxiliary losses for mean and volatilities between the intermediate forecasts from GLT, i.e., $\hat{Y}'^{returns}_{mean}(t)$ and $\hat{Y}'^{returns}_{vol}(t)$ and the ground-truths with RMSE for the subsets of company nodes $v_i \in V'_{mean}$ and $v_i \in V'_{vol}$, respectively:

$$\mathcal{L}_{aux} = \mathcal{L}_{RMSE}\left(Y'^{returns}_{mean}(t), \hat{Y}'^{returns}_{mean}(t)\right) + \mathcal{L}_{RMSE}\left(Y'^{returns}_{vol}(t), \hat{Y}'^{returns}_{vol}(t)\right). \quad (13)$$

GLAM can be trained with the objective of minimizing total main and auxiliary training losses, i.e., a simple addition of $\mathcal{L}_{main}$ and $\mathcal{L}_{aux}$. However, to adaptively balance between these losses, we introduce the $\alpha$ hyper-parameter. The $\alpha$ hyper-parameter balances between the main and auxiliary training objectives with respect to the means and volatilities of returns. A higher weight is placed on the auxiliary training losses $\mathcal{L}_{aux}$ during the initial training epochs to enable the GLT to extract relevant global information well first, before higher weights are placed on the main training losses for forecasts of means and volatilities $\mathcal{L}_{RMSE}(Y^{returns}_{mean}(t), \hat{Y}^{returns}_{mean}(t)) + \mathcal{L}_{RMSE}(Y^{returns}_{vol}(t), \hat{Y}^{returns}_{vol}(t))$ during the later training epochs. The total loss is defined as:

$$\begin{aligned}\mathcal{L}_{total} = \alpha &\left(\mathcal{L}_{RMSE}\left(Y^{returns}_{mean}(t), \hat{Y}^{returns}_{mean}(t)\right) + \mathcal{L}_{RMSE}\left(Y^{returns}_{vol}(t), \hat{Y}^{returns}_{vol}(t)\right)\right) \\ &+ \mathcal{L}_{RMSE}\left(Y^{returns}_{corr}(t), \hat{Y}^{returns}_{corr}(t)\right) + (1-\alpha)\mathcal{L}_{aux}\end{aligned} \quad (14)$$

where $\alpha = \frac{e}{E}$.

## 4 EXPERIMENTS

### 4.1 Datasets

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

The two online news article sources are: (i) Investing news datasets (**IN**)[1]; and (ii) Benzinga news datasets (**BE**).[2] The datasets contain news articles and commentaries collected from Investing and Benzinga investment news portals, which are drawn from a wide range of mainstream providers, analysts, and blogs, such as Seeking Alpha and Zacks.

For the local numerical information, we collected daily stock market price-related information—opening, closing, low and high prices, and trading volumes—of the two stock markets—NYSE (**NY**) and NASDAQ (**NA**)—from the Center for Research in Security Prices. We filter out stocks from NYSE and NASDAQ that are not traded in the respective time periods and whose stock symbols are not mentioned in any articles for the respective news article sources. We could have included articles not covering these stocks, as GLAM is able to extract relevant global textual information, but we restrict the choice of stocks and articles for a fair comparison with previous models (e.g.,

[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.

FAST [43] and HAN [52]) that are designed to only capture local textual news information, i.e., news information associated with specific companies in the target company list.

Following the earlier works [1, 17], we utilize relationships between companies extracted from Wikidata knowledge graphs for the inter-company relationships of the company network $G = (V, E, X)$ from Wikidata dumps dated January 7, 2019. Wikidata is chosen, as it is one of the largest and most active collaboratively constructed KGs. Companies such as Google, Apple, and Microsoft are present within the Wikidata KG as entities, and relationships between them, e.g., *Alphabet as a parent company of Google*, and *Apple and Microsoft belong to the same industry sector*, can be extracted from Wikidata. We adopted five first-order relationship types and 52 second-order relationship-types identified by Feng et al. [17] to extract inter-company relationships from the Wikidata dumps dated January 7, 2019. First-order relationship-type refers to a relationship extracted directly from a knowledge graph relation, e.g., *parent organization* relation in Wikidata knowledge graph where company A is the parent of company B. Second-order relationship-type involves the use of two Wikidata relations to create an inter-company relationship. For example, a second-order relationship of company $A$ sharing key management with another company $B$ can be constructed from two Wikidata relations: $A$ having a board member $M$ and $B$ having $M$ as its chief executive officer. Another second-order relationship of companies $A$ and $B$ belonging to the same industry sector can be constructed from two Wikidata relations: $A$ in industry $I$ and $B$ in industry $I$, too. The earliest Wikidata dumps were from 2014. We used Wikidata dumps from January 7, 2019, and not earlier, as we found that knowledge graphs extracted from earlier Wikidata dumps were too sparse to be useful for our experiments. We did not use more recent Wikidata dumps to avoid overlap with the time window of testing data sets. Following Reference [1], we also use a pre-trained Wikipedia2Vec [56] embedding model to pre-encode textual news to capture the rich knowledge present within the Wikipedia knowledge base, as it offers a relatively compact representation with dimension of 100, while giving reasonably good performance compared to other pre-trained encoders, as it captures knowledge-based semantics. The representations of each news article are the average word embeddings of each news article generated with the pre-trained Wikipedia2Vec embedding model. Wikipedia2Vec generates representations of words and entities based on corresponding pages in Wikipedia, placing similar words and entities close to one another in the representational space.

The coverage of these datasets—across five years, with more than 1.5M articles and 2,000 companies, and more than 50 types of inter-company relationships—is extensive and provides strong assurance to our experiment findings. We combine them into four datasets (across two news article sources and two stock markets), each of which covers different number of companies, number of relationship-types, and news sources, as depicted in Table 1. The number of companies included in these datasets is relatively large or comparable to most other related works, e.g., References [13, 43] cover less than 100 companies, [1, 17] cover around 1,000–2,000 companies, while Reference [23] similarly covers more than 2,000 companies.

To obtain the labelled data samples, we adopt a sliding window approach [53] to extract the numerical and textual input features in the window $[t - K, t - 1]$ and returns-related labels, i.e., the ground-truth means, volatilities, and correlations of returns in the horizon $[t, t + K']$. For each of the four datasets, we obtain a total of 1,257 data samples and divide these samples into non-overlapping training/validation/testing sets in the ratios 0.7/0.15/0.15 for all experiments, as shown in Figure 5.

## 4.2 Tasks and Metrics

We compare GLAM with state-of-the-art baselines on three predictive tasks: **forecasting of (i) means, (ii) volatilities, and (iii) correlations of stock price percentage returns**. We use

Table 1. Overview of Datasets

| | IN-NY | IN-NA | BE-NY | BE-NA |
|---|---|---|---|---|
| No. articles | 221,513 | | 1,377,098 | |
| No. company nodes | 374 | 402 | 2,240 | 2,514 |
| No. relationship-types | 58 | 36 | 46 | 34 |
| No. relationships | 3,255 | 1,511 | 6,436 | 4,986 |

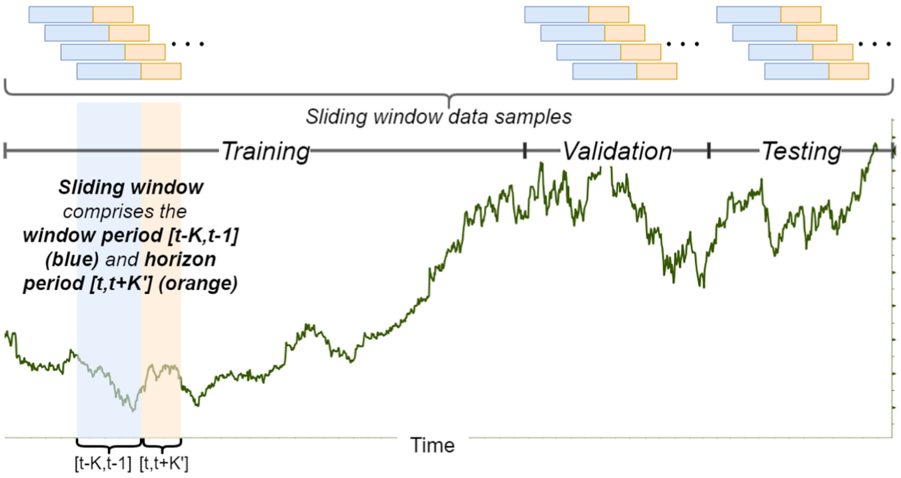
Sliding window data samples
Training
Validation
Testing
Sliding window comprises the window period [t-K,t-1] (blue) and horizon period [t,t+K'] (orange)
Price
Time
[t-K,t-1]
[t,t+K']

Fig. 5. Using a fixed sliding window to extract input features in the window $[t-K, t-1]$ and labels in the horizon $[t, t+K']$ to obtain labelled data samples and splitting into training, validation, and testing sets.

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 defined as:

$$SMAPE = \frac{100\%}{n} \sum_{i=1}^{n} \frac{|Y_i^{returns}(t) - \hat{Y}_i^{returns}(t)|}{(|Y_i^{returns}(t)| + |\hat{Y}_i^{returns}(t)|)/2}, \tag{15}$$

where $n$ is the number of observations. 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 GLAM against the classical **Vector AR** [34] model that captures numerical information and makes forecasts in an auto-regressive manner, as well as state-of-the-art baselines (see Section 2): **TST** [60], which captures numerical information with a conventional transformer encoder; **HAN** [52], which captures local textual information with two sets of attention mechanisms; **SE** [13], which captures global textual information with bidirectional GRUs; **FAST** [43], which captures local textual information with Time-Aware LSTMs[3]; **RSR** [17], which captures numerical information and heterogeneous inter-company relationships with a GCN-based model; **KECE** [1], which captures numerical, global textual information, and homogeneous inter-company relationships with a GAT-based model. We also adapt HGT [22] for time-series attributes by adding a GRU at the start to first encode numerical time-series information before network encoding with HGT.

Table 2. Overview of Data that Can Be Captured by GLAM Model and Baselines

| Model | Vector AR | TST | HAN | SE | FAST | RSR | KECE | GRU-HGT | GLAM |
|---|---|---|---|---|---|---|---|---|---|
| Homogeneous Network | × | × | × | × | × | ✓ | ✓ | ✓ | ✓ |
| Heterogeneous Network | × | × | × | × | × | ✓ | × | ✓ | ✓ |
| Local information | ✓ | ✓ | ✓ | × | ✓ | ✓ | ✓ | ✓ | ✓ |
| Global information | × | × | × | ✓ | × | × | ✓ | × | ✓ |

Homogeneous networks are a special case of heterogeneous networks, hence GLAM and baseline models that can capture heterogeneous networks can also be utilized to capture homogeneous networks.

This model is referred to as **GRU-HGT**. Table 2 provides an overview of the data captured by GLAM and the baselines. For every deep learning baseline, we add a fully connected layer to forecast means, volatilities, and correlations of percentage stock returns. For the classical Vector AR model, we compute the return forecasts in the horizon in an auto-regressive manner and compute the means, volatilities, and correlations of these forecasts.

We set the default window and horizon periods $K = 20$ and $K' = 10$ days based on experiments with different periods $K, K' \in \{5, 10, 20, 60\}$, which correspond to a trading week, fortnight, month, and quarter. Differences in performance between GLAM and baselines were generally consistent across all window and horizon periods. $K = 20$ corresponds to a trading month, and $K' = 10$ days corresponds to a global regulatory requirement for VaR computations, which we examined in a subsequent set of case-studies (see Section 7). Across all models, dimensions of hidden representations are fixed at 100 and two layers ($L = 2$) utilized, where applicable. $\eta$ for GLAM is set to 0.5 based on experiments with different $\eta \in \{0.25, 0.5, 0.75\}$. An Adam [26] optimizer with a learning rate of 1e-3 with a cosine annealing scheduler is used. Models are implemented in Pytorch [39] and trained for 100 epochs on a 3.60 GHz AMD Ryzen 7 Windows desktop with NVIDIA RTX 3090 GPU and 64 GB RAM. Training GLAM, which has around 6e5 to 8e5 parameters (depending on the datasets), takes around 12 to 16 hours.

## 5 FORECASTING RESULTS

Table 3 sets out the results of the forecasting experiments. For each metric, we indicate the best results in boldface and underline the second best results.

On the task of **forecasting means**, GLAM clearly outperforms all baselines. The dispersion in model performances for IN datasets is more narrow than for BE datasets. TST, RSR, and KECE, which utilize numerical information, generally tend to perform better than the models that only utilize textual information.

On the tasks of **forecasting volatilities**, GLAM again outperforms baselines on most metrics. Compared to the task of forecasting means, the dispersion in model performance for forecasting volatilities is more significant, which could be due to the task of forecasting volatilities being more difficult and requiring textual information to be captured more effectively. Hence, we observe baselines such as SE, FAST, and KECE, which capture textual information performing better on this task. This could be due to textual news information containing information that may be indicative of subsequent periods of increased volatility.

On the task of **forecasting correlations**, GLAM outperforms baselines most significantly. This is likely due to its utilization of heterogeneous network information. We similarly see RSR and KECE, baselines that utilize network information, performing better than baselines here. GRU-HGT, which also utilizes network information, does not perform as well, but this could be due to its inability to capture sequential information effectively with the simple GRU extension.

In **general**, GLAM outperforms all baselines by a significant margin on all tasks. Different baselines perform better on different tasks based on the nature of information that they capture.

Table 3. Forecast Results

| | **IN-NY** | | | **IN-NA** | | | **BE-NY** | | | **BE-NA** | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | RMSE | MAE | SMAPE | RMSE | MAE | SMAPE | RMSE | MAE | SMAPE | RMSE | MAE | SMAPE |
| | **Means** | | | | | | | | | | | |
| Vector AR | 0.1509 | 0.0413 | 1.7718 | 0.0618 | 0.0372 | 1.4357 | 0.0765 | 0.0201 | 1.6669 | 0.1526 | 0.0282 | 1.5673 |
| TST | 0.0689 | 0.0133 | 1.4860 | 0.0323 | 0.0148 | 1.3349 | 0.0662 | 0.0142 | 1.5216 | 0.1521 | 0.0288 | 1.5511 |
| HAN | 0.0689 | 0.0134 | 1.6905 | 0.0322 | 0.0144 | 1.3704 | 0.0664 | 0.0158 | 1.5238 | 0.1499 | 0.0271 | 1.5560 |
| SE | 0.0689 | 0.0134 | 1.4646 | 0.0323 | 0.0144 | 1.3919 | 0.0676 | 0.0158 | 1.5952 | 0.1666 | 0.0339 | 1.5445 |
| FAST | 0.0689 | 0.0134 | 1.4442 | 0.0329 | 0.0162 | 1.2921 | 0.0663 | 0.0144 | 1.5285 | 0.1496 | 0.0276 | 1.5599 |
| RSR | 0.0690 | 0.0135 | 1.3785 | 0.0327 | 0.0156 | 1.3128 | 0.0664 | 0.0163 | 1.5050 | 0.1499 | 0.0300 | 1.5581 |
| KECE | 0.0688 | 0.0134 | 1.4014 | 0.0324 | 0.0152 | 1.2965 | 0.0662 | 0.0152 | 1.6411 | 0.1465 | 0.0301 | 1.6610 |
| GRU-HGT | 0.0689 | 0.0134 | 1.3853 | 0.0322 | 0.0147 | 1.4299 | 0.0663 | 0.0145 | 1.5311 | 0.1498 | 0.0287 | 1.5540 |
| GLAM | **0.0491** | **0.0107** | **1.2031** | **0.0223** | **0.0116** | **1.2408** | **0.0487** | **0.0118** | **1.4449** | **0.1125** | **0.0208** | **1.4959** |
| | **Volatilities** | | | | | | | | | | | |
| Vector AR | 0.3605 | 0.0871 | 0.7569 | 0.1163 | 0.0542 | 0.6956 | 0.2287 | 0.0662 | 1.0481 | 0.4775 | 0.1183 | 1.1401 |
| TST | 0.2177 | 0.0482 | 0.6225 | 0.1155 | 0.0587 | 0.6773 | 0.2202 | 0.0627 | 1.0181 | 0.4827 | 0.1200 | 1.1521 |
| HAN | 0.2180 | 0.0486 | 0.6226 | 0.1154 | 0.0579 | 0.6719 | 0.2223 | 0.0663 | 1.0361 | 0.4814 | 0.1174 | 1.1682 |
| SE | 0.2175 | 0.0485 | 0.6319 | 0.1148 | 0.0558 | **0.6547** | 0.2245 | 0.0688 | 1.0209 | 0.4795 | 0.1143 | 1.1286 |
| FAST | 0.2179 | 0.0485 | 0.6228 | 0.1145 | 0.0561 | 0.6638 | 0.2217 | 0.0633 | 1.0260 | 0.4789 | 0.1155 | 1.1594 |
| RSR | 0.2181 | 0.0487 | 0.6232 | 0.1161 | 0.0590 | 0.6830 | 0.2240 | 0.0724 | 1.0488 | 0.4818 | 0.1253 | 1.1748 |
| KECE | 0.2177 | 0.0483 | 0.6239 | 0.1193 | 0.0651 | 0.7167 | 0.2186 | **0.0591** | 1.0486 | 0.4619 | 0.1005 | 1.1545 |
| GRU-HGT | 0.2176 | 0.0483 | 0.6270 | 0.1145 | 0.0577 | 0.6699 | 0.2232 | 0.0684 | 1.0343 | 0.4807 | 0.1174 | 1.1374 |
| GLAM | **0.1435** | **0.0414** | **0.6117** | **0.0835** | **0.0507** | 0.6556 | **0.1632** | 0.0601 | **1.0170** | **0.3578** | **0.0946** | **1.0836** |
| | **Correlations** | | | | | | | | | | | |
| Vector AR | 0.7443 | 0.5903 | 1.5869 | 0.7122 | 0.5704 | 1.6241 | 0.5306 | 0.3265 | 1.7658 | 0.4540 | 0.2534 | 1.8834 |
| TST | 0.4953 | 0.4222 | 1.5009 | 0.4913 | 0.4184 | 1.5402 | 0.3899 | 0.2768 | 1.7220 | 0.3379 | 0.2177 | 1.8082 |
| HAN | 0.4943 | 0.4222 | 1.4861 | 0.4914 | 0.4185 | 1.5356 | 0.3902 | 0.2777 | 1.7243 | 0.3386 | 0.2174 | 1.7986 |
| SE | 0.5090 | 0.4308 | 1.5456 | 0.4980 | 0.4208 | 1.5167 | 0.4023 | 0.2844 | 1.7224 | 0.3395 | 0.2212 | 1.7854 |
| FAST | 0.4958 | 0.4223 | 1.5035 | 0.4917 | 0.4176 | 1.5056 | 0.3882 | 0.2752 | 1.7198 | 0.3371 | 0.2167 | 1.7996 |
| RSR | 0.4927 | 0.4200 | 1.4299 | 0.4940 | 0.4201 | 1.5145 | 0.3903 | 0.2780 | 1.7233 | 0.3398 | 0.2206 | 1.7943 |
| KECE | 0.4958 | 0.4227 | 1.5165 | 0.4916 | 0.4184 | 1.5268 | 0.3891 | 0.2617 | 1.7070 | 0.3381 | 0.2186 | 1.8005 |
| GRU-HGT | 0.4965 | 0.4234 | 1.5287 | 0.4933 | 0.4194 | 1.5193 | 0.3872 | 0.2770 | 1.7290 | 0.3395 | 0.2231 | 1.7874 |
| GLAM | **0.4025** | **0.3248** | **1.1221** | **0.4169** | **0.3437** | **1.2332** | **0.3355** | **0.2382** | **1.5844** | **0.3060** | **0.1979** | **1.7018** |

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

Performance differences between GLAM and baselines are more significant for the larger BE datasets than for the IN datasets due to the larger volume of news textual information. The differences in performances between GLAM and baselines are more pronounced for volatilities and correlations forecasting than means forecasting, as these are harder tasks that require the model to capture global textual news information and the propagation of news effects between companies based on heterogeneous relationships, which are key features of the GLAM model.

## 6 ABLATION STUDIES

To further substantiate the earlier findings, we conduct ablation studies on GLAM. Table 4 shows the results of ablation studies for GLAM on IN-NY and IN-NA for forecasting means, volatilities, and correlations. We choose IN-NY and IN-NA datasets for ablation studies to represent the two markets, as the difference in the number of relationship-types and number of relationships is the most distinct between these two datasets using the same source of news. The impact of different changes to the GLAM model vary for different tasks but are generally consistent across both datasets.

When we do not capture heterogeneous network information and exclude the HNE module, i.e., (**GLT only ($L$ = 2)**), the drop in performance for the correlation forecasting task is the most significant, which highlights the importance of capturing heterogeneous network information effectively. Nonetheless, we can also see that the GLT module alone (with $L = 2$ as set in the GLAM model) is already able to outperform most of the baselines on the tasks of forecasting means and

Table 4. Ablation Studies

| | **IN-NY** | | | **IN-NA** | | |
|---|---|---|---|---|---|---|
| | **RMSE** | **MAE** | **SMAPE** | **RMSE** | **MAE** | **SMAPE** |
| | **Means** | | | | | |
| GLT only ($L$ = 2) | 0.0508 | 0.0109 | 1.2046 | 0.0244 | 0.0135 | 1.3423 |
| GLT only ($L$ = 1) | 0.0516 | 0.0125 | 1.2102 | 0.0256 | 0.0145 | 1.3446 |
| GLT only ($L$ = 3) | 0.0508 | 0.0108 | 1.2093 | 0.0241 | 0.0134 | 1.3413 |
| w/o. inner $W_{att}$ | **0.0491** | **0.0107** | 1.2042 | 0.0225 | 0.0118 | 1.2415 |
| w/o. GLT guided learning loss | 0.0518 | 0.0126 | 1.2054 | 0.0243 | 0.0126 | 1.2568 |
| w. subset final forecasts | 0.0510 | 0.0117 | 1.2091 | 0.0342 | 0.0203 | 1.3221 |
| w. $\eta$ = 0.25 | 0.0525 | **0.0107** | 1.2034 | 0.0226 | **0.0116** | 1.2494 |
| w. $\eta$ = 0.75 | 0.0527 | **0.0107** | 1.2045 | **0.0223** | 0.0117 | 1.2447 |
| w/o. adaptive $\alpha$ | 0.0511 | 0.0117 | 1.2082 | 0.0232 | 0.0128 | 1.2457 |
| w. mean loss | 0.0505 | 0.0113 | 1.2039 | 0.0226 | 0.0123 | 1.2463 |
| w. vol. loss | 0.1420 | 0.0479 | 1.5869 | 0.1594 | 0.1327 | 1.8501 |
| w. corr. loss | 1.4864 | 1.4801 | 1.9908 | 0.1154 | 0.0790 | 1.7290 |
| **GLAM** | **0.0491** | **0.0107** | **1.2031** | **0.0223** | **0.0116** | **1.2408** |
| | **Volatilities** | | | | | |
| GLT only ($L$ = 2) | 0.1492 | 0.0418 | 0.6196 | 0.0854 | 0.0528 | 0.6616 |
| GLT only ($L$ = 1) | 0.1495 | 0.0420 | 0.6194 | 0.0863 | 0.0534 | 0.6626 |
| GLT only ($L$ = 3) | 0.1482 | **0.0413** | 0.6199 | 0.0854 | 0.0523 | 0.6613 |
| w/o. inner $W_{att}$ | 0.1435 | 0.0414 | **0.6117** | 0.0836 | 0.0509 | 0.6566 |
| w/o. GLT guided learning loss | 0.1450 | 0.0425 | 0.6199 | 0.0844 | 0.0520 | 0.6567 |
| w. subset final forecasts | 0.1446 | 0.0421 | 0.6125 | 0.1042 | 0.0887 | 0.8508 |
| w. $\eta$ = 0.25 | 0.1450 | 0.0414 | 0.6123 | **0.0835** | 0.0508 | 0.6569 |
| w. $\eta$ = 0.75 | 0.1450 | 0.0414 | 0.6123 | **0.0835** | 0.0508 | 0.6559 |
| w/o. adaptive $\alpha$ | 0.1445 | 0.0424 | 0.6127 | 0.0843 | 0.0524 | 0.6643 |
| w. mean loss | 0.3040 | 0.0692 | 0.8391 | 0.1423 | 0.0750 | 0.8220 |
| w. vol. loss | 0.1436 | 0.0414 | 0.6197 | 0.0836 | 0.0508 | 0.6558 |
| w. corr. loss | 1.0126 | 0.9547 | 1.9990 | 0.1426 | 0.0719 | 0.8651 |
| **GLAM** | **0.1435** | 0.0414 | **0.6117** | **0.0835** | **0.0507** | **0.6556** |
| | **Correlations** | | | | | |
| GLT only ($L$ = 2) | 0.5036 | 0.4264 | 1.5320 | 0.4914 | 0.4185 | 1.5392 |
| GLT only ($L$ = 1) | 0.5033 | 0.4265 | 1.5376 | 0.4919 | 0.4190 | 1.5392 |
| GLT only ($L$ = 3) | 0.5046 | 0.4266 | 1.5382 | 0.4912 | 0.4188 | 1.5381 |
| w/o. inner $W_{att}$ | 0.4050 | 0.3275 | 1.1371 | 0.4199 | 0.3555 | 1.2612 |
| w/o. GLT guided learning loss | 0.4050 | 0.3271 | 1.1337 | 0.4193 | 0.3461 | 1.2382 |
| w. subset final forecasts | 0.4030 | 0.3257 | 1.1235 | 0.4452 | 0.3780 | 1.3376 |
| $\eta$ = 0.25 | 0.4038 | 0.3260 | 1.1496 | 0.4198 | 0.3466 | 1.2359 |
| $\eta$ = 0.75 | 0.4037 | 0.3259 | 1.1491 | **0.4169** | **0.3437** | 1.2336 |
| w/o. adaptive $\alpha$ | 0.4035 | 0.3268 | 1.1231 | 0.4173 | 0.3452 | 1.3307 |
| w. mean loss | 0.5223 | 0.4437 | 1.9759 | 0.5110 | 0.4350 | 1.9537 |
| w. vol. loss | 0.5224 | 0.4437 | 1.9797 | 0.5108 | 0.4348 | 1.9576 |
| w. corr. loss | 1.2713 | 1.1697 | 1.6481 | 0.4175 | 0.3442 | 1.2418 |
| **GLAM** | **0.4025** | **0.3248** | **1.1221** | **0.4169** | **0.3437** | **1.2332** |

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

volatilities. We observe similar changes in performance when we do not utilize $W_{att}$ (**w/o. inner** $W_{att}$) as the inner weight when forecasting correlations. While the impact of not utilizing $W_{att}$ is not significant for the tasks of forecasting means and volatilities, there is a material drop in performance on the task of forecasting correlations.

We also explore the effects of changing the number of layers in GLT, i.e., (**GLT only (**$L$ **= 1)** and **GLT only (**$L$ **= 3)**). In general, increasing the number of layers leads to better performance. However, the selected hyper-parameter of $L = 2$ for GLT in GLAM achieves a good balance between model complexity and performance.

The drop in performance when we exclude the guided learning losses (**w/o. GLT guided learning loss**), i.e., excluding $\mathcal{L}_{aux}$ from the training objective, is more apparent for the tasks of forecasting means and volatilities. This demonstrates the importance of the proposed approach of using intermediate forecasts for early guidance when learning GLT parameters, which enables GLT to focus on learning the relevance of global news information by utilizing significant ex-post responses of stock prices to news. An alternative approach would be to compute the losses for the final forecasts only on the subset of stocks with such significant ex-post responses (**w. subset final forecasts**). However, such an approach leads to poorer performance, particularly for the IN-NA dataset. This could be due to the generally more volatile price movements of stocks listed on the NASDAQ market.

We also explore alternative hyper-parameter settings for $\eta$. While the differences in performance for different $\eta$ (**w.** $\eta$ **= 0.25 and** $\eta$ **= 0.75**), are not significant, the chosen $\eta = 0.5$ for GLAM generally gives the best performance across most metrics. We also explore not using the $\alpha$ parameter to adaptively balance between the different losses (**w/o. adaptive** $\alpha$) and find that it leads to worse performance across most metrics.

When we vary the multitask aspect of GLAM by training on mean, volatility, or correlation forecast losses only (i.e., **w. mean loss only, w. vol. loss only, w. corr. loss only**), we see significant drops in performance, even on tasks that correspond to the training loss, e.g., performance of mean forecasts when we train only on mean loss is poorer than when we train GLAM with multiple tasks.

In general, we see that the key features of GLAM work together to enable it to achieve the best performance on the multiple tasks. The GLT module and using intermediate forecasts of means and volatilities for early guidance when learning GLT parameters improves forecasts of means and volatilities, while the HNE module and utilizing $W_{att}$ as an inner weight for correlation forecasts improves performance for the correlation forecasts.

## 7 APPLICATION CASE STUDIES

In this section, we use model forecasts for important investment and risk management applications to evaluate the quality of forecasts.

### 7.1 Portfolio Allocation

Investment portfolio allocation is an important task for many financial institutions. The aim of investment portfolio allocation is to optimize the proportion of capital invested in each stock (also known as asset) in a portfolio, by finding an optimal set of weights $\mathbb{W}$ that determine how much capital to invest in each stock, so portfolio returns can be maximized while minimizing portfolio risk. In this article, we adopt the risk aversion formulation [15] of the mean-variance risk minimization model by Markowitz [35], which models both portfolio return and risk expressed as mean ($\mu$) and co-variances ($\Sigma$) of return, respectively. Under the risk aversion formulation, the classical mean-variance risk minimization model by Markowitz [35] is re-formulated to maximize the risk-adjusted portfolio return by optimizing the asset allocation $\mathbb{W}$, a $|V|$ dimensional vector:

$$max_{\mathbb{W}} \ (\mathbb{W}^\intercal \mu - \lambda \mathbb{W}^\intercal \Sigma \mathbb{W}) \tag{16}$$

subject to $\mathbb{W}^\intercal \mathbf{1} = 1$. $\lambda$, 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 [15]. In our experiments, we set $\lambda = 2$. We observe that higher $\lambda$ 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 $\mu$ and compute $\Sigma$ with the forecasted volatilities and correlations of asset returns for the selected horizon period $[t, t + K']$ defined as follows:

$$\tilde{\mu} = \hat{Y}^{returns}_{mean}(t), \tag{17}$$

$$\tilde{\Sigma} = D(t) \cdot \hat{Y}^{returns}_{corr}(t) \cdot D(t), \tag{18}$$

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 $[t, t + K']$ instead of directly forecasting co-variances as the co-variances need to be positive semi-definite (PSD) so the matrix is invertible [14], 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 multimodal and network information (as applicable) from the window period $[t - K, t - 1]$ to make forecasts of mean ($\tilde{\mu}$) and correlation ($\tilde{\Sigma}$) of asset returns over the future horizon $[t, t + K']$, which are in turn used to determine the asset allocation weights $\mathbb{W}$. $\mathbb{W}$ represents an investment portfolio with returns realized in this future horizon defined as: $E^{real} = \mathbb{W}^{\intercal} R^{real}$, where $R^{real}$ is a vector of realized percentage stock returns over the future horizon.

Given that the aim is to maximize portfolio returns while minimizing portfolio risk (volatility), we choose *risk-adjusted realized portfolio returns* over the future horizon $[t, t+K']$ as the evaluation metric, defined as: $\tilde{E} = \frac{E^{real}}{\sigma^{real}}$, where $\sigma^{real}$ is portfolio return volatility in the future horizon $[t, t + K']$. Portfolio return volatility is defined as one standard deviation of the portfolio returns over the future horizon $[t, t + K']$ and is computed as $\sigma^{real} = \sqrt{\mathbb{W}^{\intercal} \Sigma^{real} \mathbb{W}}$, where $\Sigma^{real}$ are the co-variances of realized percentage stock returns over the same future horizon.

For this application, the datasets are similarly divided into non-overlapping training/validation/testing sets in the ratios 0.7/0.15/0.15, as described in Section 4.1, and we evaluate performance based on the average of the risk-adjusted realized portfolio returns ($\tilde{E}$) across future horizon periods in the testing set.

Table 5 depicts results for the IN-NY and IN-NA datasets for the portfolio allocation application. We see that portfolios constructed using GLAM's forecasts for the IN-NY dataset achieves the highest average risk-adjusted returns of 1.67%, which is 15% better than the second highest results from KECE; and portfolios constructed using GLAM's forecasts for the IN-NA dataset achieves the highest average risk-adjusted returns of 2.32%, which is 56% better than the second highest results from GRU-HGT. Baselines utilizing textual information or inter-company relationships (FAST, RSR, KECE, and GRU-HGT) generally perform better on this application, demonstrating the value of capturing textual and relational information for selecting optimal portfolios.

## 7.2 Value-at-Risk (VaR)

VaR [32] 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 $K' = 10$ day horizon and maintain capital based on this VaR as loss buffers. VaR measures the loss that an institution may face in the pre-defined horizon with a probability of $p$%. For example, if the 10 day 95% VaR is \$1,000,000, then 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:

$$VaR(p) = -\phi^{-1}(p) \times \sigma, \tag{19}$$

Table 5. Portfolio Allocation and VAR

| | IN-NY | | | IN-NA | | |
|---|---|---|---|---|---|---|
| | $\tilde{E}$ | **VaR Breaches** | **% VaR Breaches.** | $\tilde{E}$ | **VaR Breaches** | **% VaR Breaches** |
| Vector AR | 0.61% | <u>22</u> | <u>11.6%</u> | 0.40% | 19 | 10.1% |
| TST | 1.37% | 40 | 21.2% | 0.48% | 32 | 16.9% |
| HAN | 0.66% | 34 | 18.0% | 0.10% | 35 | 18.5% |
| SE | 1.32% | 28 | 14.8% | 0.95% | 28 | 14.8% |
| FAST | 0.64% | 36 | 19.1% | 1.26% | <u>7</u> | <u>3.7%</u> |
| RSR | 1.42% | 46 | 24.3% | 1.21% | 8 | 4.2% |
| KECE | <u>1.45%</u> | 59 | 31.2% | 1.21% | 12 | 6.4% |
| GRU-HGT | 1.10% | 30 | 15.9% | <u>1.49%</u> | 36 | 19.1% |
| GLAM | **1.67%** | **7** | **3.7%** | **2.32%** | **1** | **0.5%** |

Higher better for average risk-adjusted percentage returns $\tilde{E}$. Lower better for number of VaR breaches (VaR Breaches) & percentage of VaR breaches (% VaR Breaches). % VaR Breaches is computed by dividing the number of VaR breaches by the number of data samples in the testing dataset.

where $\sigma$ is the portfolio volatility, and $\phi$ 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 $E^{real}$) is greater than the forecasted VaR, it is regarded as a *VaR breach*, i.e., $E^{real} \leq VaR(p)$.

For this application, the portfolio is constructed based on the approach described for the portfolio allocation application at each timestep. This mimics a real-world scenario where financial institutions continually update their portfolios based on market conditions. To evaluate 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 (18). Similar to the portfolio allocation application, this can also be viewed as a predictive task, as we are using multimodal and network information (as applicable) from the window period $[t-K, t-1]$ to make forecasts over the future horizon $[t, t+K']$ and using these forecasts to determine the VaR in the future horizon. We evaluate model performances by counting the total number of 95% VaR breaches, i.e., where the realized portfolio loss is greater than the forecasted VaR in the testing dataset (using the same training/validation/testing sets as described in Section 4.1). We choose the 95% VaR for our experiments, as it is a common confidence level used by banks to monitor their risks. Models that are able to make accurate forecasts of VaR should have less VaR breaches.

Table 5 depicts results for the IN-NY and IN-NA datasets for the VaR application. We see that GLAM outperforms baselines with significantly less VaR breaches. Similar to the portfolio allocation application, we observe baselines that utilize textual information or inter-company relationships (SE, FAST, RSR, and KECE) generally performing better on this application.

## 8 DISCUSSION

Our experimental results demonstrate the value of the proposed time-sensitive guided global-local transformer in extracting relevant global information for forecasting on multiple tasks. We see that the proposed GLAM model outperforms other baselines such as SE and KECE, which also extract and utilize relevant global textual information. In the ablation studies, we observe that even without the use of the heterogeneous network information and the HNE module, the GLT module itself is already able to perform better than the baselines on the tasks of forecasting means and volatilities. Further, we also show that isolating and utilizing significant ex-post stock price responses to global textual information in the window period improves the extraction of relevant global textual information. We also demonstrated the value of capturing heterogeneous network relationships and using a learned set of attention weights $W_{att}$ for forecasting correlations. The use of attention weights $W_{att}$ enables heterogeneous inter-company relationships to be captured and facilitates better performance on the task of forecasting correlations.

We show that the proposed model features are valuable in investment and risk management applications, which differs from the more common and simpler task of forecasting stock prices or returns for trading decisions. GLAM forecasts the price dynamics of a portfolio of multiple stocks over a longer future horizon, i.e., expected returns, volatilities, and correlations of stocks over a longer term horizon, which are important in enabling investment and risk managers to make effective decisions over a longer term horizon. Importantly, we also see that designing a model that can be used in a multivariate multitask setting for investment and risk management applications has other potential advantages, as forecasting in a multivariate multitask setting enables complementary information from other variables and related tasks to be used to improve overall forecasting performance and also lowers the risk of over-fitting on any one task.

The framework proposed in this article could be potentially extended to capture other information sources, such as other types of global and local information, e.g., local social media information such as tweets from the company's social media account and global economic indicators, e.g., gross domestic product of countries of the company's key markets; as well as other static and dynamic inter-company relationships (i.e., inter-company relationships captured at different timestamps), e.g., from domain experts, DBPedia, GDELT.

## 9 CONCLUSION AND FUTURE WORK

In this article, we designed GLAM, a model that comprises a time-sensitive global-local transformer to learn relevant global online text information with local numerical information and sequentially encode such multimodal information; and an attention-based heterogeneous network encoder to leverage heterogeneous inter-company relationships. Auxiliary channels and an adaptive learning strategy are also utilized in GLAM to facilitate intermediate guided learning of the parameters of the time-sensitive global-local transformer and heterogeneous network encoder modules. The model performs strongly on three forecasting tasks and two real-world applications, demonstrating the value of the proposed model features and learning strategies. The datasets used are extensive and provide strong assurance on the validity of the results across different companies and textual information. Future work could extend GLAM to different types of global and local information, as well as other static and dynamic inter-company relationships.

## REFERENCES

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

[2] Shaojie Bai, J. Zico Kolter, and Vladlen Koltun. 2018. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *Computing Research Repository (CoRR)* abs/1803.01271 (2018).

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

[4] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *International Conference on Learning Representations (ICML'09)*.

[5] Tim Bollerslev. 1986. Generalized autoregressive conditional heteroskedasticity. *J. Economet.* 31, 3 (Apr. 1986), 307–327.

[6] 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.

[7] Yukuo Cen, Xu Zou, Jianwei Zhang, Hongxia Yang, Jingren Zhou, and Jie Tang. 2019. Representation learning for attributed multiplex heterogeneous network. In *ACM International Conference on Knowledge Discovery and Data Mining (KDD'19)*.

[8] Hao Chen, Keli Xiao, Jinwen Sun, and Song Wu. 2017. A double-layer neural network framework for high-frequency forecasting. *ACM Trans. Manag. Inf. Syst.* 7, 4 (2017).
[9] Jinyin Chen, Xuanheng Xu, Yangyang Wu, and Haibin Zheng. 2018. GC-LSTM: Graph convolution embedded LSTM for dynamic link prediction. *Appl. Intell.* 52, 7 (2022), 7513–7528.
[10] 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 Syst. Applic.* 83 (2017), 187–205.
[11] 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).*
[12] Yuxiao Dong, Nitesh V. Chawla, and Ananthram Swami. 2017. MetaPath2vec: Scalable representation learning for heterogeneous networks. In *ACM International Conference on Knowledge Discovery and Data Mining (KDD'17).*
[13] 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).*
[14] Robert Engle. 2002. Dynamic conditional correlation: A simple class of multivariate generalized autoregressive conditional heteroskedasticity models. *J. Bus. Econ. Statist.* 20, 3 (2002), 339–350.
[15] F. J. Fabozzi, P. N. Kolm, D. A. Pachamanova, and S. M. Focardi. 2007. *Robust Portfolio Optimization and Management.* Wiley.
[16] Christos Faloutsos, Valentin Flunkert, Jan Gasthaus, Tim Januschowski, and Yuyang Wang. 2020. Forecasting big time series: Theory and practice. In *the Web Conference.*
[17] Fuli Feng, Xiangnan He, Xiang Wang, Cheng Luo, Yiqun Liu, and Tat-Seng Chua. 2019. Temporal relational ranking for stock prediction. *ACM Trans. Inf. Syst.* 37, 2 (2019), 27:1–27:30.
[18] Valentin Flunkert, David Salinas, and Jan Gasthaus. 2020. DeepAR: Probabilistic forecasting with autoregressive recurrent networks. *Int. J. Forecast.* 36, 3 (2020), 1181–1191.
[19] C. Lee Giles, Steve Lawrence, and Ah Chung Tsoi. 2001. Noisy time series prediction using recurrent neural networks and grammatical inference. *Mach. Learn.* 44, 1/2 (2001), 161–183.
[20] Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. 2017. Neural message passing for quantum chemistry. In *International Conference on Machine Learning (ICML'17).*
[21] Luke B. Godfrey and Michael S. Gashler. 2018. Neural decomposition of time-series data for effective generalization. *IEEE Trans. Neural Netw. Learn. Syst.* 29, 7 (2018), 2973–2985.
[22] Ziniu Hu, Yuxiao Dong, Kuansan Wang, and Yizhou Sun. 2020. Heterogeneous graph transformer. In *the World Wide Web Conference (WWW'20).*
[23] 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).*
[24] Weiwei Jiang. 2021. Applications of deep learning in stock market prediction: Recent progress. *Expert Syst. Applic.* 184 (2021), 115537.
[25] 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. *Computing Research Repository (CoRR)* abs/1907.05321 (2019).
[26] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR'15).*
[27] Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations (ICLR'17).*
[28] 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).*
[29] Bryan Lim, Sercan Ömer Arik, Nicolas Loeff, and Tomas Pfister. 2021. Temporal fusion transformers for interpretable multi-horizon time series forecasting. *Int. J. Forecast.* 37, 4 (2021), 1748–1764.
[30] Bryan Lim and Stefan Zohren. 2021. Time series forecasting with deep learning: A survey. *Philos. Trans. Roy. Societ. A* 379, 2194 (2021), 20200209.
[31] 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.*
[32] Thomas J. Linsmeier and Neil D. Pearson. 2000. Value at risk. *Finan. Anal. J.* 56, 2 (2000), 47–67.
[33] 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 Syst. Appl.* 143 (2020).
[34] Helmut Lütkepohl. 2011. *Vector Autoregressive Models.* Springer Berlin.
[35] Harry Markowitz. 1952. Portfolio selection. *J. Finance* 7, 1 (1952), 77–91.
[36] Boris N. Oreshkin, Dmitri Carpov, Nicolas Chapados, and Yoshua Bengio. 2020. N-BEATS: Neural basis expansion analysis for interpretable time series forecasting. In *International Conference on Learning Representations (ICLR'20).*

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

[38] 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*.

[39] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Z. Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. PyTorch: An imperative style, high-performance deep learning library. In *Annual Conference on Neural Information Processing Systems (NIPS'19)*.

[40] Fotios Petropoulos et al. 2022. Forecasting: Theory and practice. *Int. J. Forecast.* (Jan. 2022).

[41] 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)*.

[42] 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 *Annual Conference on Neural Information Processing Systems (NIPS'18)*.

[43] 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 Assoc. for Computational Linguistics (EACL'21)*.

[44] Michael Sejr Schlichtkrull, Thomas N. Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *the Extended Semantic Web Conference (ESWC'18)*.

[45] Petru Soviany, Radu Tudor Ionescu, Paolo Rota, and Nicu Sebe. 2021. Curriculum learning: A survey. *Int. J. Comput. Vis.* 130, 6 (2022), 1526–1565.

[46] 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.

[47] Granville Tunnicliffe Wilson. 2016. Time series analysis: Forecasting and control. *J. Time Series Anal.* 37 (2016).

[48] Rianne van den Berg, Thomas N. Kipf, and Max Welling. 2017. Graph convolutional matrix completion. *Computing Research Repository (CoRR)* abs/1706.02263 (2017).

[49] 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 *Annual Conference on Neural Information Processing Systems (NIPS'17)*.

[50] 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)*.

[51] 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).

[52] Xiao Wang, Houye Ji, Chuan Shi, Bai Wang, Yanfang Ye, Peng Cui, and Philip S. Yu. 2019. Heterogeneous graph attention network. In *the World Wide Web Conference (WWW'19)*.

[53] Neo Wu, Bradley Green, Xue Ben, and Shawn O'Banion. 2020. Deep transformer models for time series forecasting: The influenza prevalence case. *Computing Research Repository (CoRR)* abs/2001.08317 (2020).

[54] Keyulu Xu, Chengtao Li, Yonglong Tian, Tomohiro Sonobe, Ken-ichi Kawarabayashi, and Stefanie Jegelka. 2018. Representation learning on graphs with jumping knowledge networks. In *International Conference on Machine Learning (ICML'18)*.

[55] Wentao Xu, Weiqing Liu, Chang Xu, Jiang Bian, Jian Yin, and Tie-Yan Liu. 2021. REST: Relational event-driven stock trend forecasting. In *the World Wide Web Conference (WWW'21)*.

[56] 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 *Conference on Empirical Methods in NLP*.

[57] Shaowei Yao, Tianming Wang, and Xiaojun Wan. 2020. Heterogeneous graph transformer for graph-to-sequence learning. In *Annual Meeting of the Association for Computational Linguistics (ACL'20)*.

[58] 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. *Appl. Intell.* 49, 3 (2019), 897–911.

[59] Bing Yu, Haoteng Yin, and Zhanxing Zhu. 2018. Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting. In *International Joint Conference on AI (IJCAI'18)*.

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

[61] 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 Trans. Intell. Transport. Syst.* 21, 9 (2020), 3848–3858.

Received 15 December 2021; revised 28 May 2022; accepted 27 December 2022