Guided Attention Multimodal Multitask Financial Forecasting with Inter-Company Relationships and Global and Local News

Gary Ang
Singapore Management University
gary.ang.2019@phdcs.smu.edu.sg

Ee-Peng Lim
Singapore Management University
eplim@smu.edu.sg

Abstract

Most works on financial forecasting use information directly associated with individual companies (e.g., stock prices, news on the company) to predict stock returns for trading. We refer to such company-specific information as local information. Stock returns may also be influenced by global information (e.g., news on the economy in general), and inter-company relationships. Capturing such diverse information is challenging due to the low signal-to-noise ratios, different time-scales, sparsity and distributions of global and local information from different modalities. In this paper, we propose a model that captures both global and local multimodal information for investment and risk management-related forecasting tasks. Our proposed Guided Attention Multimodal Multitask Network (GAME) model addresses these challenges by using novel attention modules to guide learning with global and local information from different modalities and dynamic inter-company relationship networks. Our extensive experiments show that GAME outperforms other state-of-the-art models in several forecasting tasks and important real-world application case studies.

1 Introduction

Forecasting stock prices or returns is an important task in trading. Such forecasts can also be used in investment and risk management applications such as portfolio allocation and risk forecasting. Stock returns in financial markets are influenced by large volumes of textual information from diverse sources, e.g., news, blogs, social media. Such textual information can be directly associated with a specific company (*local*), e.g, a company's CEO stepping down; or relevant to multiple companies (*global*), e.g., disruptions in supply chains due to export curbs in key countries, airline industry bankruptcies. In this paper, articles with company tags are treated as local information. All articles are treated as global information as any article could be potentially relevant to a company.

Direct and indirect relationships between companies also serve as channels through which the effects of information from both global and local textual and numerical information propagate and influence stock returns, e.g., a disruption in company A could affect all its suppliers; a scandal involving company A's CEO may affect company B if the CEO is a member of company B's board. We illustrate such diverse information and effects in Figure 1.

Apart from low signal-to-noise ratios in financial time-series due to market forces, there are other challenges in modeling such diverse information. Time scales of information from different modalities are of different granularity, e.g., numerical financial information may be available daily, while publication of financial text happens at irregular times. Companies' local financial news are typically sparse and long-tailed, e.g., a company may not be in the news for an extended period of time, but suddenly becomes the focus of many news reports in a short period due to a scandal. Local textual information may also be noisy with regards to its relevance to the company's stock returns, e.g., a news article on a company's HR practices may have little effect on its stock returns, whereas a news article on a sector's outlook can have a significant effect on the company's stock returns even without any mention of the company.

More research on financial forecasting is required to address such challenges. Most existing works model financial information of a single modality (Ding et al., 2015; Ziniu et al., 2018; Du and Tanaka-Ishii, 2020; Sawhney et al., 2021b), and do not model the effects of inter-company relationships. Some works (Feng et al., 2019; Xu et al., 2021; Sawhney et al., 2021a) model both unimodal financial information and the effects of inter-company relationships. There are however few

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 1: Long Papers, pages 6313 - 6326

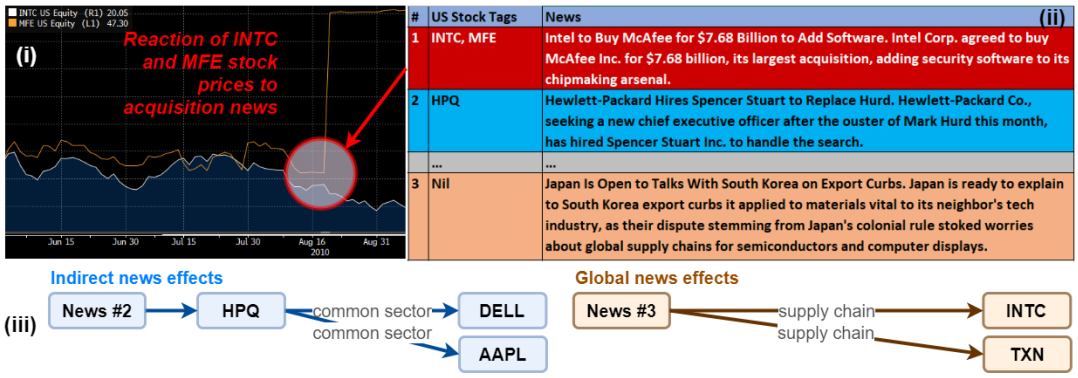

Figure 1: (i) reaction of stock prices to news; (ii) news articles 1 and 2 as local text information associated with specific companies; all news articles 1 to 3 as global text information potentially relevant to any company; (iii) textual news effects propagate through different types of inter-company relationships.

works capturing multimodal financial information and inter-company relationships (Ang and Ee-Peng, 2021; Sawhney et al., 2020b,a). Ang and Ee-Peng (2021) utilizes both numerical and global textual information, as well as inter-company relationships but does not address challenges related to capturing global and local multimodal information. Most works also focus on a single task - forecasting stock returns for trading. Another equally important set of forecasting tasks which has many investment and risk management applications involves similar challenges. It involves a multivariate multitask setting, where there is a need to manage the returns and risks of financial portfolios that comprise many stocks (multivariate), and make investment and risk decisions based on multiple forecasts (multitask): forecast stock i) mean returns and ii) risks (volatilities) over a future horizon to balance potential returns and risks when making investment decisions, as well as forecast iii) correlations between stocks in portfolios over a future horizon.

To address financial data challenges in multitask settings, we propose the Guided Attention Multimodal Multitask Network (GAME) model. Our key idea is to use attention to guide learning between information from different sources and modalities. GAME incorporates several important components: i) guided latent cross-attention learning between modalities of different time-scales and sparsity; ii) graph-guided representation learning based on inter-company relationships with dynamic weights learnt from multimodal information; and iii) guided cross-attention learning between global and local information. GAME is trained on multiple tasks - forecasting means, volatilities and correlations over a future horizon, which could be used for portfolio allocation and risk management. While existing works for financial forecasting capture either local or global information, or network information with either global or local information, GAME jointly captures global, local and network information. Compared to existing works that utilize transformers for time-series forecasting (Zerveas et al., 2021), GAME proposes novel cross-attention mechanisms that enable i) more effective modelling of local information of different lengths and granularity from different modalities by first encoding such information to a common latent representation; and ii) the extraction of global information by leveraging such local information. Hence, 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 for multivariate multitask financial forecasting;
- We propose an attention-based module that encodes multimodal information of different sequence lengths and time granularity to a latent representation space for efficient mutually guided cross-attention learning;
- We design a graph encoding module that uses inter-company relationships to propagate multimodal information across companies; and dynamically updates relationship weights with learnt importances;
- We design an attention-based module that uses cross-attention between local and global information to guide learning of relevant global information;

- We train the model on multiple forecasting tasks to lower the risk of over-fitting, and demonstrate the effectiveness of GAME 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.

Time Series Forecasting. Classical methods (Box and Jenkins, 1990; Bollerslev, 1986) are commonly applied to time-series forecasting. However, they are designed for numerical data but not unstructured financial text. Deep learning models have been increasingly applied to time-series forecasting. They include feed-forward networks (Yoojeong et al., 2019; Ding et al., 2015; Oreshkin et al., 2020), convolutional neural networks (Pantiskas et al., 2020; Borovykh et al., 2017; Wan et al., 2019), recurrent neural networks (Flunkert et al., 2020; Qin et al., 2017; Liu et al., 2020), and transformers (Wu et al., 2020; Zerveas et al., 2021). A detailed review of these works can be found in Lim and Zohren (2021); Jiang (2021); Torres et al. (2021).

Time-series Transformer (**TST**) (Zerveas et al., 2021) is a recent model based on the transformer encoder architecture designed for numerical inputs. StockEmbed (**SE**) (Du and Tanaka-Ishii, 2020) is designed for global textual features, while Financial News and Tweet Based Time Aware Network (**FAST**) (Sawhney et al., 2021b) is designed for local textual features. To encode sequences of textual features, SE utilizes bidirectional GRUs, while FAST utilizes Time-aware LSTMs (Baytas et al., 2017). These works are designed for information from a single modality, do not model the effects of company-to-company relationships, and do not address the challenges of capturing global and local multimodal information.

Network Learning. 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 (Gilmer et al., 2017). In particular, Graph Convolutional Network (GCN) (Kipf and Welling, 2017) aggregates features of neighboring nodes and normalizes aggregated representations by node degrees. Graph Attention Network (GAT) (Veličković et al., 2018) assigns neighboring nodes with different importance weights during aggregation. Such GNNs are designed for static networks with static node attributes and cannot be directly applied to networks where attributes are evolving time series.

A few recent works extend GNNs to prediction tasks on financial time-series data (Ang and Ee-Peng, 2021; Feng et al., 2019; Sawhney et al., 2020b,a, 2021a). Relational Stock Ranking (**RSR**) (Feng et al., 2019) uses LSTM to generate output embeddings for numerical time-series data of companies before feeding the latter to learn company embeddings in a network using a GCN-based model, but does not consider textual information. Knowledge Enriched Company Embedding (**KECE**) (Ang and Ee-Peng, 2021) captures numerical and global textual information and uses a GAT-based model to capture inter-company relationships but does not address the challenges of capturing global and local multimodal information. RSR and KECE also do not learn the dynamic importance of inter-company relationships.

3 Guided Attention Multimodal Multitask Network Model

GAME represents companies in a network $G = (V, E, X)$, where V represents a set of company nodes, E represents relationships between companies, X represents sequences of multimodal attributes. Given a time step t, we define numerical features $X_j^{num}(t) = [x_j^{num}(t-K), ..., x_j^{num}(t-1)]$ to be the sequence of numerical price-related data associated with company v_j over a window of K time steps up to $t-1$. Textual news features include local and global textual features, i.e. $X^{txt} = \{X^{txt,loc}, X^{txt,glo}\}$. The pre-encoded local news textual features directly associated with a company v_j within the same window are denoted as $X_j^{txt,loc}(t) = [x_{j,1}^{txt,loc}(t-K), \cdots, x_{j,M}^{txt,loc}(t-K), x_{j,M+1}^{txt,loc}(t-K+1), \cdots, x_{j,S}^{txt,loc}(t-1)]$. $S = M \times K$ and assumes M news articles are captured for each company at each time-step. Where there are less than M articles for any company at any given time-step, we add PAD values of zero to the sequence (Devlin et al., 2019). We denote pre-encoded global news features over the window period K as $X^{txt,glo}(t) = [x^{txt,glo}(t-K), ..., x^{txt,glo}(t-1)]$, with varying number of news articles binned into each time step.

As shown in Figure 2, GAME first encodes

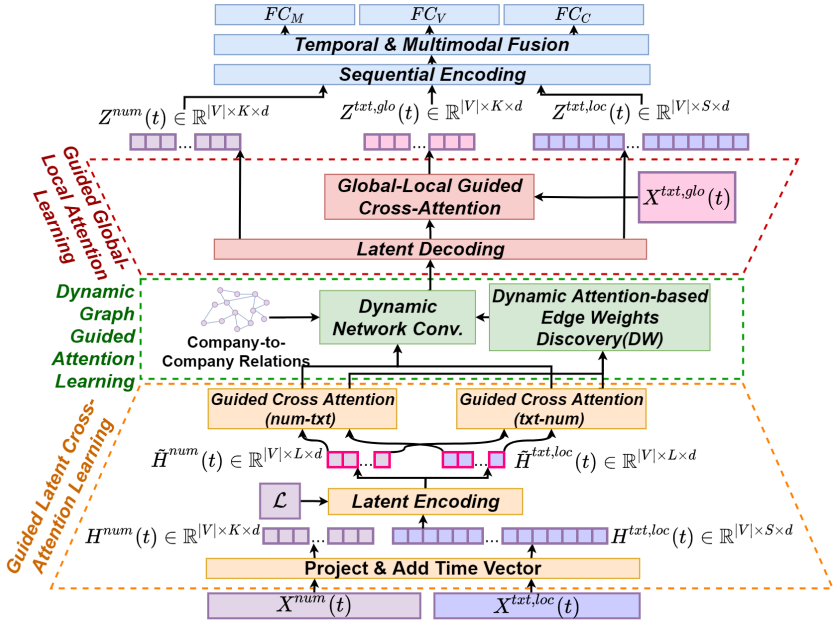

Figure 2: Architecture of GAME

both $X^{num}(t) \in \mathbb{R}^{|V|\times K\times d^{num}}$ and $X^{txt,loc}(t) \in \mathbb{R}^{|V|\times S\times d^{txt}}$, where d^{num}, d^{txt} are embedding dimension sizes, to a common latent sequence length L and dimension d by using latent attention-based encoders inspired by Jaegle et al. (2021a), where $L \ll K$ as part of the *Guided Latent Cross-Attention Learning* step. We introduce guided cross-attention to enable information from one modality to guide the attention learning of another. In the next *Dynamic Graph-Guided Attention Learning* step, representations of both modalities are used to discover and update importance weights of inter-company relationships before applying dynamic graph convolutions. A latent decoder inspired by Jaegle et al. (2021b) then decodes the numerical and local textual representations to the original sequence lengths K and S. In the *Guided Global-Local Attention Learning* step, we use the decoded local representations to guide the attention extraction of the sequence of global textual features relevant to each company v_j. The resultant representations are then combined and sequentially encoded with a transformer, followed by attention-based temporal and multimodal fusion. Finally, GAME generates forecasts of means, volatilities and correlations of financial returns over a selected future horizon of K' timesteps, i.e. the means, volatilities and correlations of $Y^{returns}(t) = [y^{returns}(t), ..., y^{returns}(t + K')]$, where $y^{returns}(t) = (price(t) - price(t - 1))/price(t - 1)$ and $price(t)$ denote the percentage return and stock price at time step t respectively. We further elaborate on GAME modules below.

Guided Latent Cross-Attention Learning. This step addresses the challenge of learning information from modalities of different sequence lengths, degrees of sparsity and distributions, specifically $X^{num}(t) \in \mathbb{R}^{|V|\times K\times d^{num}}$ and $X^{txt,loc}(t) \in \mathbb{R}^{|V|\times S\times d^{txt}}$. For $X^{num}(t)$, we first **project the inputs to common dimension d and add a learnt time vector** (Kazemi et al., 2019; Godfrey and Gashler, 2018). The time vector is learned from the time-stamps $T(t)$ corresponding to the inputs. In this paper, we use day of week, week and month of year for $T^{num}(t)$, and further include seconds of day for $T^{txt,loc}(t)$ as these are most relevant to the respective inputs. The time vector $P^{num}(t) \in \mathbb{R}^{|V|\times K\times d}$ is learned by combining functional forms and learnable weights and could be viewed as a time-sensitive version of positional encodings used in transformers (Vaswani et al., 2017). For GAME, the empirically chosen components used to generate the time vectors are $\Phi_1^{num} = sigmoid(Linear(T^{num}(t)))$ and $\Phi_2^{num} = cos(Linear(T^{num}(t)))$, which enable the model to extract non-linear and seasonality-based temporal patterns. We then concatenate these components and project them: $P^{num}(t) = Linear([\Phi_1^{num}||\Phi_2^{num}])$. The output of the projection and addition of time vectors step is: $H^{num}(t) \in \mathbb{R}^{|V|\times K\times d}$. For the **latent encoding** step, we introduce latent units $\mathcal{L} \in \mathbb{R}^{L\times d}$. We re-

peat $\mathcal{L}$ by $|V|$ times to get $\mathbb{R}^{|V|\times L\times d}$, and apply linear layers to generate queries from $\mathcal{L}$, and keys and values from $H^{num}(t)$. That is, $Q^{num}(t) = Linear_Q(\mathcal{L})$, $K^{num}(t) = Linear_K(H^{num}(t))$, $V^{num}(t) = Linear_V(H^{num}(t))$. We then apply scaled dot-product attention $\tilde{H}^{num}(t) = softmax(Q^{num}(t)\cdot K^{num}(t)^T)V^{num}(t)/\sqrt{d}$. To elaborate, the dot-product between $Q^{num}(t) \in \mathbb{R}^{|V|\times L\times d}$ and $K^{num}(t) \in \mathbb{R}^{|V|\times K\times d}$ gives us attention weights of dimensions $|V| \times L \times K$. We use these attention weights to map $V^{num}(t) \in \mathbb{R}^{|V|\times K\times d}$ to $\tilde{H}^{num}(t) \in \mathbb{R}^{|V|\times L\times d}$. The same set of steps is repeated for $X^{txt,loc}(t)$ to obtain $\tilde{H}^{txt,loc}(t)$. Hence, after the latent encoding step, both $\tilde{H}^{num}(t)$ and $\tilde{H}^{txt,loc}(t)$ have the same sequence length L and dimension d, i.e. $\tilde{H}^{num}(t), \tilde{H}^{txt,loc}(t) \in \mathbb{R}^{|V|\times L\times d}$, and share a common latent space due to the common $\mathcal{L}$.

In the next **guided cross-attention** step, information from each of the modalities guide attention learning of the other. Sharing a common latent space facilitates mutually guided learning between the modalities and is more efficient as $L \ll K \ll S$. For this step, we generate queries, keys, and values from the numerical and local text representations: $\tilde{Q}^{num}(t) = Linear_Q(\tilde{H}^{num}(t))$, $\tilde{K}^{num}(t) = Linear_K(\tilde{H}^{num}(t))$, $\tilde{V}^{num}(t) = Linear_V(\tilde{H}^{num}(t))$, $\tilde{Q}^{txt,loc}(t) = Linear_Q(\tilde{H}^{txt,loc}(t))$, $\tilde{K}^{txt,loc}(t) = Linear_K(\tilde{H}^{txt,loc}(t))$, $\tilde{V}^{txt,loc}(t) = Linear_V(\tilde{H}^{txt,loc}(t))$. Queries of one modality are used to guide the learning of the other modality as follows:

$$\tilde{H}^{num\text{-}txt}(t) = softmax(\frac{\tilde{Q}^{txt,loc}(t)\cdot\tilde{K}^{num}(t)^T}{\sqrt{d}})\tilde{V}^{num}(t) \quad (1)$$

$$\tilde{H}^{txt\text{-}num}(t) = softmax(\frac{\tilde{Q}^{num}(t)\cdot\tilde{K}^{txt,loc}(t)^T}{\sqrt{d}})\tilde{V}^{txt,loc}(t) \quad (2)$$

Dynamic Graph-Guided Attention Learning. We then utilize inter-company relationships E to guide learning. While these relationships do not frequently change (e.g., common sector relationships), their importances vary across time. Hence, we discover dynamic relationship weights with the **dynamic attention-based edge weights discovery (DW)** module. We concatenate and project $\tilde{H}^{num\text{-}txt}(t)$ and $\tilde{H}^{txt\text{-}num}(t)$ with a linear layer to obtain: $\tilde{H}(t) = Linear[\tilde{H}^{num\text{-}txt}(t)||\tilde{H}^{txt\text{-}num}(t)]$. We then generate: $Q^{DW}(t) = Linear_{Q\text{-}DW}(\tilde{H}(t))$; $K^{DW}(t) = Linear_{K\text{-}DW}(\tilde{H}(t))$. To learn the importance of inter-company relationships in a dynamic manner, we compute attention weights:

$$W_{att}(t) = tanh(Q^{DW}(t)\cdot W_{DW}\cdot K^{DW}(t)^T/\sqrt{d}) \quad (3)$$

where $W_{DW} \in \mathbb{R}^{L\times d\times d}$. As we carry out this operation in the latent space with dimension L, $W_{att}(t) \in \mathbb{R}^{|V|\times|V|\times L}$. We then repeat the adjacency matrix corresponding to the inter-company relationships E by L times to get $A(t) \in \mathbb{R}^{|V|\times|V|\times L}$ and compute the Hadamard product between $A(t)$ and $W_{att}(t)$: $\tilde{A}(t) = A(t) \odot W_{att}(t)$. This results in the weighted adjacency tensor $\tilde{A}(t) \in \mathbb{R}^{|V|\times|V|\times L}$ with $\tilde{A}_{ij}(t) \in \mathbb{R}^L$ representing the weighted relational edges between asset i and j across latent dimension L. Next, in the **dynamic network convolution** step, we utilize the encoded company representations $\tilde{H}(t)$ and the weighted adjacency tensor $\tilde{A}(t)$ as inputs to a weighted dynamic graph convolution step to encode network representations of companies. For company v_i, we compute its network representations $Z_i(t) \in \mathbb{R}^{L\times d}$ across L dimension by aggregating representations from its neighbors $N(i,t)$ based on $\tilde{A}_{i,j}(t), j \in V$:

$$Z_i(t) = \sum_{j\in N(i,t)} \frac{exp(\tilde{A}_{ij}(t))}{\sum_{j'\in N(i,t)} exp(\tilde{A}_{ij'}(t))}\cdot\tilde{H}_j(t) \quad (4)$$

Across all assets, we obtain $Z(t) \in \mathbb{R}^{|V|\times L\times d}$. We adopt this approach instead of other GNNs for computational efficiency as it allows us to apply graph convolution across multiple dimensions in parallel.

Guided Global-Local Attention Learning. We then apply **latent decoding** to decode the representation $Z(t)$ from the latent dimension L to the original sequence length K and S for the numerical and local text modalities respectively. To decode the numerical information, the numerical representations after the projection and addition of time vectors $H^{num}(t)$ are used as queries to decode the keys and values of the representation $Z(t)$. We generate: $Q^{num}_{dec}(t) = Linear_Q(H^{num}(t))$, $K^{num}_{dec}(t) = Linear_K(Z(t))$, $V^{num}_{dec}(t) = Linear_V(Z(t))$, and apply scaled dot-product attention:

$$Z^{num}(t) = softmax(\frac{Q^{num}_{dec}(t)\cdot K^{num}_{dec}(t)^T}{\sqrt{d}})V^{num}_{dec}(t) \quad (5)$$

To elaborate, the dot-product between $Q^{num}_{dec}(t) \in \mathbb{R}^{|V|\times K\times d}$ and $K^{num}_{dec}(t) \in \mathbb{R}^{|V|\times L\times d}$ gives us attention weights of dimensions $|V| \times K \times L$. We then use these attention

weights to map $V_{dec}^{num}(t) \in \mathbb{R}^{|V| \times L \times d}$ to $Z^{num}(t) \in \mathbb{R}^{|V| \times K \times d}$. Similarly, to decode the local textual representation, the queries of the local textual representations after the projection and addition of time vectors $H^{txt,loc}(t)$ are used to decode the keys and values of $Z(t)$. We generate: $Q_{dec}^{txt,loc}(t) = Linear_Q(H^{txt,loc}(t))$, $K_{dec}^{txt,loc}(t) = Linear_K(Z(t))$, $V_{dec}^{txt,loc}(t) = Linear_V(Z(t))$, and again apply scaled dot-product attention:

$$Z^{txt,loc}(t) = softmax(\frac{Q_{dec}^{txt,loc}(t) \cdot K_{dec}^{txt,loc}(t)^T}{\sqrt{d}})V_{dec}^{txt,loc}(t) \tag{6}$$

resulting in $Z^{txt,loc}(t) \in \mathbb{R}^{|V| \times S \times d}$. The **global-local guided cross-attention** step uses the decoded $Z^{num}(t)$ to guide the learning of global textual features relevant to each company v_j from $X^{txt,glo}(t)$. We utilize $Z^{num}(t)$ instead of $Z^{txt,loc}(t)$ as we extract global textual features for each time-step $t-k$ in window K rather than S. $Z^{num}(t)$ also contains information relating to $Z^{txt,loc}(t)$ due to the prior guided latent cross-attention learning step. For each time step $t-k$ in window K, we generate $Q^{txt,glo}(t-k) = Linear_Q(Z^{num}(t-k))$, $K^{txt,glo}(t-k) = Linear_K(X^{txt,glo}(t-k))$, $V^{txt,glo}(t-k) = Linear_V(X^{txt,glo}(t-k))$. We apply scaled dot-product attention: $Z^{txt,glo}(t-k) = softmax(K^{txt,glo}(t-k) \cdot W^{txt,glo} \cdot Q^{txt,glo}(t-k)^T/\sqrt{d})^T \cdot V^{txt,glo}(t-k)$ where $W^{txt,glo} \in \mathbb{R}^{d \times d}$ is an inner weight shared across all time steps $t-k$ to improve attention extraction of global textual information. Across the window period, we get $Z^{txt,glo}(t) \in \mathbb{R}^{|V| \times K \times d}$.

Sequential Encoding and Fusion. Transformer encoders (Vaswani et al., 2017) are then used to encode the resultant sequence of representations: $Z^{num\prime}(t) = \text{TransformerEnc}(Z^{num}(t))$, $Z^{txt,loc\prime}(t) = \text{TransformerEnc}(Z^{txt,loc}(t))$, and $Z^{txt,glo\prime}(t) = \text{TransformerEnc}(Z^{txt,glo}(t))$. The transformer encoded sequence of representations are combined with temporal attention fusion, which weights contributions of each time step $t-k$ based on its importance. A non-linear transformation is applied to the respective representations, say $Z^{num\prime}(t-k)$, to obtain scalar $\alpha(t-k)$ for each time step $t-k$ in the window of t: $\alpha(t-k) = W_\tau^{(1)} tanh(W_\tau^{(0)} Z^{num\prime}(t-k) + b_\tau)$, where $W_\tau^{(0)}$ and $W_\tau^{(1)}$ are learnable weight matrices and b_τ is the bias vector. We normalize each $\alpha(t-k)$ to obtain the weights: $\beta(t-k) = \frac{exp(\alpha(t-k))}{\sum_{k=1}^{K} exp(\alpha(t-k))}$. We then fuse the sequence of representations: $Z^{num\prime\prime}(t) = \sum_{k=1}^{K} \beta(t-k) Z^{num\prime}(t-k)$, where $Z^{num\prime\prime}(t) \in \mathbb{R}^{|V| \times d}$. This temporal attention fusion step is repeated across K time-steps for $Z^{txt,glo\prime}(t)$ to obtain $Z^{txt,glo\prime\prime}(t) \in \mathbb{R}^{|V| \times d}$ and across S time-steps for $Z^{txt,loc\prime}(t)$ to obtain $Z^{txt,loc\prime\prime}(t) \in \mathbb{R}^{|V| \times d}$. The representations from the three modalities are then fused with multimodal attention fusion. We denote each of the modalities as r, for a total of $R = 3$ modalities for the numerical, local textual and global textual modalities respectively. A non-linear transformation is applied to the representations to obtain scalars $s(r) = W_\omega^{(1)} tanh(W_\omega^{(0)} \bar{Z}^{r\prime\prime}(t) + b_\omega)$, where $W_\omega^{(0)}$ and $W_\omega^{(1)}$ are learnable weight matrices and b_ω is the bias vector. Parameters are shared across modalities. We normalize the scalars with a softmax function to obtain the weights: $\beta_r = \frac{exp(s(r))}{\sum_{r=1}^{R} exp(s(r))}$, which are used to fuse representations across the three modalities: $Z'''(t) = \sum_{r=1}^{R} \beta_r Z^{r\prime\prime}(t)$, where $Z'''(t) \in \mathbb{R}^{|V| \times d}$.

Table 1: Overview of datasets

	IN-NY	IN-NA	BE-NY	BE-NA
No. articles	221,513		1,377,098	
No. companies	374	402	2,240	2,514
No. relationships	3,255	1,511	6,436	4,986

Forecasting and Loss Functions. We use fully connected layers to generate forecasts of means and volatilities of stock returns over the selected horizon period $[t, t+K']$: $\hat{Y}_{mean}^{returns}(t) = FC_M(Z'''(t))$; and $\hat{Y}_{vol}^{returns}(t) = FC_V(Z'''(t))$. To forecast correlations of asset returns over the horizon period $[t, t+K']$, we use weights from linear layers in DW: $Q_{corr}(t) = Linear_{Q\text{-}DW}(Z'''(t))$; $K_{corr}(t) = Linear_{K\text{-}DW}(Z'''(t))$. This allows what was learnt in the DW step to be utilized here: $\hat{Y}_{corr}^{returns}(t) = FC_C(tanh(\frac{Q_{corr}(t) \cdot K_{corr}(t)^T}{\sqrt{d'}}))$. We then compute losses between the forecasts above and respective ground-truths, i.e. actual means, volatilities and correlations over the horizon $[t, t+K']$ (see Appendix A.2 for ground-truth definitions) with root mean squared loss (RMSE), and use total losses as the training objective:

$$\begin{aligned} \mathcal{L}_{total} = &\mathcal{L}_{mean}(Y_{mean}^{returns}(t), \hat{Y}_{mean}^{returns}(t)) \\ &+ \mathcal{L}_{vol}(Y_{vol}^{returns}(t), \hat{Y}_{vol}^{returns}(t)) \\ &+ \mathcal{L}_{corr}(Y_{corr}^{returns}(t), \hat{Y}_{corr}^{returns}(t)) \end{aligned} \tag{7}$$

We do not weight the losses differently as we want the model to perform equally well on all three tasks.

Table 2: Forecast Results. Lower better for all metrics. Best model(s) in bold; second-best model(s) underlined.

	IN-NY			IN-NA			BE-NY			BE-NA		
	RMSE	MAE	SMAPE	RMSE	MAE	SMAPE	RMSE	MAE	SMAPE	RMSE	MAE	SMAPE
	Means											
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
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
GAME	**0.0491**	**0.0107**	**1.2022**	**0.0222**	**0.0118**	**1.2295**	**0.0487**	**0.0119**	**1.4449**	**0.1130**	**0.0209**	**1.4866**
	Volatilities											
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
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
GAME	**0.1436**	**0.0414**	**0.6113**	**0.0833**	**0.0501**	**0.6528**	**0.1631**	0.0594	**1.0156**	**0.3589**	**0.0949**	**1.1179**
	Correlations											
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
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
GAME	**0.4024**	**0.3247**	**1.1239**	**0.4169**	**0.3437**	**1.2327**	**0.3355**	**0.2377**	**1.5857**	**0.3079**	**0.1989**	**1.7146**

4 Experiments

Datasets. We conduct experiments with four datasets, comprising global and local textual information of news articles from financial news portals - Investing news (**IN**) and Benzinga news (**BE**); and numerical information of daily stock market price-related information of two stock markets - NYSE (**NY**) and NASDAQ (**NA**) from 2015 to 2019. The coverage of these datasets - across five years, more than 1.5m articles and 2,000 companies - is more extensive than most existing works and provides strong assurance to our experiment findings. Following Ang and Ee-Peng (2021), we utilize relationships between companies extracted from Wikidata knowledge graphs for the inter-company relationships E from Wikidata dumps dated 7 Jan. 2019. 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 (first-order), both Apple and Microsoft are producing computer hardware (second-order), can be extracted from Wikidata. We use a pre-trained Wikipedia2Vec (Yamada et al., 2020) model to pre-encode textual news to capture the rich knowledge present within the Wikipedia knowledge base (see Table 1 and Appendix A.1 for more details on datasets).

Tasks and Metrics. We compare GAME 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 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 scale-independent metric that gives equal importance to under- and over-forecasts required in our evaluation context (see Appendix A.3 for more details on SMAPE). Datasets are divided into non-overlapping training/validation/test sets in the ratios 0.7/0.15/0.15 for experiments.

Baselines and Settings. We compare GAME against state-of-the-art baselines (see Section 2): **TST** (Zerveas et al., 2021) that captures numerical information; **SE** (Du and Tanaka-Ishii, 2020) that captures global textual information; **FAST** (Sawhney et al., 2021b) that captures local textual information; **RSR** (Feng et al., 2019) that captures numerical information and inter-company relationships; and **KECE** (Ang and Ee-Peng, 2021) that captures numerical, global textual information and inter-company relationships. We add fully-connected layers to baselines for them to forecast means, volatilities and correlations of percentage stock returns. We set the window period $K = 20$ days; and horizon period $K' = 10$. $K = 20$ corresponds to a trading month, and $K' = 10$ days corresponds to a global regulatory requirement for VaR computations, which we examine in the case-study (in Section 6). Following Sawhney et al. (2021b),

we set M for local news text sequences to be 10. We empirically set L to 16. Dimensions of hidden representations are fixed at 100 across all models. Models are implemented in Pytorch and trained for 100 epochs on a 3.60GHz AMD Ryzen 7 Windows desktop with NVIDIA RTX 3090 GPU and 64GB RAM. Training GAME, which has 1.01e6 parameters, takes around two hours on the IN datasets and nine hours on the BE datasets (see Appendix A.4 for more details on settings).

Results. Table 2 sets out the results of the forecasting experiments. Across all tasks, GAME outperforms all baselines. On the task of **forecasting means**, dispersion in model performances for IN datasets is more narrow than for BE datasets. On the tasks of **forecasting volatilities** and **forecasting correlations**, baseline models (RSR, KECE) that perform better for BE datasets utilize textual and relational information. Performance differences between GAME and baselines are more significant for the larger BE datasets than for the IN datasets due to the larger volume of news textual information. Differences in performances between GAME 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 and local news effects and the propagation of news effects between companies, which are key features of the GAME model.

5 Ablation Studies

Table 3 shows the results of ablation studies for GAME on IN-NY. We observe similar sensitivities for other datasets. When we exclude the guided co-attention module (**w/o. guided co-attn.**), the drop in performance is more significant for volatility and correlation forecasting tasks, while performance decline is more significant for the correlation forecasting task when we exclude the dynamic graph-guided attention module (**w/o. graph-guided enc.**). When we vary the multi-task aspect of GAME 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 GAME with multiple tasks.

Table 3: Ablation Studies

	RMSE	MAE	SMAPE
Means			
w/o. guided co-attn.	0.0510	0.0108	1.2065
w/o. graph-guided enc.	0.0493	0.0110	1.2038
w/o. global-local attn.	0.0519	0.0110	1.2030
w. mean loss	0.0511	0.0110	1.2206
w. vol. loss	0.1087	0.0354	1.5279
w. corr. loss	0.1079	0.0303	1.6232
GAME	**0.0491**	**0.0107**	**1.2022**
Volatilities			
w/o. guided co-attn.	0.1442	0.0424	0.6223
w/o. graph-guided enc.	0.1440	0.0424	0.6232
w/o. global-local attn.	0.1448	0.0426	0.6187
w. mean loss	0.3212	0.2190	1.9977
w. vol. loss	0.1436	0.0414	0.6143
w. corr. loss	0.2399	0.0929	1.9351
GAME	**0.1436**	**0.0414**	**0.6113**
Correlations			
w/o. guided co-attn.	0.4034	0.3251	1.1366
w/o. graph-guided enc.	0.4038	0.3264	1.1243
w/o. global-local attn.	0.4038	0.3254	1.1399
w. mean loss	0.5700	0.4887	1.6854
w. vol. loss	0.5227	0.4439	1.9906
w. corr. loss	0.4027	0.3264	1.1401
GAME	**0.4024**	**0.3247**	**1.1239**

6 Application Case Studies

We use model forecasts for investment and risk management applications to evaluate the quality of forecasts. **Portfolio allocation** optimizes the proportion of capital invested in each stock in a portfolio by finding an optimal set of investment weights $\mathbb{W}$ that maximize portfolio returns while minimizing portfolio risk. We use model forecasts as optimization inputs to find $\mathbb{W}$ that maximizes risk-adjusted returns in a future horizon. **Value-at-Risk (VaR)** (Linsmeier and Pearson, 2000) is a key measure of risk used in financial institutions that measures potential losses in a pre-defined horizon with a probability of $p\%$, e.g., 10 day 95% VaR of \$1m means a 5% probability of losses exceeding \$1m over a 10 day horizon. When realized losses exceed forecasted VaR, we call it a VaR breach. We use model forecasts to compute 10 day 95% portfolio VaR forecasts, and evaluate model performances by the total number of VaR breaches. Details on computation methodologies are provided in Appendix A.5. Table 4 depicts results for the IN-NY/IN-NA datasets. For **portfolio allocation**, portfolios constructed using GAME's forecasts achieve highest average risk-adjusted returns. For **VaR**, GAME out-performs baselines with significantly less VaR breaches. Baselines utilizing textual information or inter-company relationships (SE, FAST, RSR and KECE) generally perform better.

Table 4: Portfolio Allocation and VAR. Higher better for average risk-adjusted returns (%Ret.). Lower better for number of VaR breaches (Br.) & percentage of VaR breaches (%Br.).

	IN-NY			IN-NA		
	%Ret.	**Br.**	**%Br.**	**%Ret.**	**Br.**	**%Br.**
TST	1.37%	40	21.2%	0.48%	32	16.9%
SE	1.32%	28	14.8%	0.95%	28	14.8%
FAST	0.64%	36	19.1%	1.26%	7	3.7%
RSR	1.42%	46	24.3%	1.21%	8	4.2%
KECE	1.45%	59	31.2%	1.21%	12	6.4%
GAME	**1.61%**	**6**	**3.2%**	**2.85%**	**2**	**1.1%**

7 Conclusion and Future Work

In this paper, we designed GAME, a model that captures global and local multimodal information with modules that i) enable mutual guidance between modalities with different time-scales, sparsity and distributions; ii) propagation of multimodal information between companies via real-world relationships with dynamic weights to guide learning; iii) guided attention learning between global and local information to extract relevant global information; and iv) was trained in a multivariate multi-task setting. The model performs strongly on three forecasting tasks and two real-world applications, demonstrating the value of guided attention learning for global and local multimodal information. The datasets used are more extensive than most similar works and provide strong assurance on the validity of the results across different companies and textual information. Future work could extend GAME to capture information from other modalities (e.g., audio, visual), textual sources (e.g., Twitter, Reddit), and inter-company relationships (e.g., DBPedia, GDELT). In relation to the *societal impact* of this work, we see opportunities for GAME to support better investment and risk management decisions, and also benefit a range of real-world applications, such as investment portfolio allocation and risk management, as we demonstrated in our paper. We should however recognize that models such as GAME generate forecasts based on past historical patterns that may not always hold in the future, particularly for non-stationary financial time-series. Hence, model risk management, e.g., monitoring significant changes in input information and model performance, is particularly important to avoid negative impacts, such as investment losses.

Acknowledgements

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.

References

Gary Ang and Lim Ee-Peng. 2021. Learning knowledge-enriched company embeddings for investment management. In *2nd ACM International Conference on AI in Finance, ICAIF*.

Inci M. Baytas et al. 2017. Patient subtyping via time-aware LSTM networks. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

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

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*.

George Edward Pelham Box and Gwilym Jenkins. 1990. *Time Series Analysis, Forecasting and Control*.

Jacob Devlin et al. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT*.

Xiao Ding et al. 2015. Deep learning for event-driven stock prediction. In *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI*.

Xin Du and Kumiko Tanaka-Ishii. 2020. Stock embeddings acquired from news articles and price history, and an application to portfolio optimization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL*.

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

Fuli Feng et al. 2019. Temporal relational ranking for stock prediction. *ACM Trans. Inf. Syst.*, 37(2):27:1–27:30.

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

Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. 2017. Neural message passing for quantum chemistry. In *Proceedings of the 34th International Conference on Machine Learning, ICML*.

Luke B. Godfrey and Michael S. Gashler. 2018. Neural decomposition of time-series data for effective generalization. *IEEE Trans. Neural Networks Learn. Syst.*, 29(7):2973–2985.

Andrew Jaegle et al. 2021a. Perceiver: General perception with iterative attention. In *Proceedings of the 38th International Conference on Machine Learning, ICML*.

Andrew Jaegle et al. 2021b. Perceiver IO: A general architecture for structured inputs & outputs. *CoRR*.

Weiwei Jiang. 2021. Applications of deep learning in stock market prediction: recent progress. *Expert Systems with Applications*, page 115537.

Seyed Mehran Kazemi et al. 2019. Time2vec: Learning a vector representation of time. *CoRR*.

Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *5th International Conference on Learning Representations, ICLR*.

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

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

Yeqi Liu et al. 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.

Harry Markowitz. 1952. Portfolio selection. *The Journal of Finance*, 7(1):77–91.

Boris N. Oreshkin et al. 2020. N-BEATS: neural basis expansion analysis for interpretable time series forecasting. In *8th International Conference on Learning Representations, ICLR*.

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 2020*.

Yao Qin et al. 2017. A dual-stage attention-based recurrent neural network for time series prediction. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI*.

Ramit Sawhney et al. 2020a. Risk forecasting from earnings calls acoustics and network correlations. In *Interspeech 2020, 21st Annual Conference of the International Speech Communication Association*.

Ramit Sawhney et al. 2020b. Voltage: Volatility forecasting via text audio fusion with graph convolution networks for earnings calls. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP*.

Ramit Sawhney et al. 2021a. Exploring the scale-free nature of stock markets: Hyperbolic graph learning for algorithmic trading. In *WWW '21: The Web Conference*.

Ramit Sawhney et al. 2021b. FAST: financial news and tweet based time aware network for stock trading. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, EACL*.

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):3–21.

Ashish Vaswani et al. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems, NIPS*.

Petar Veličković et al. 2018. Graph Attention Networks. In *6th International Conference on Learning Representations, ICLR*.

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).

Neo Wu et al. 2020. Deep transformer models for time series forecasting: The influenza prevalence case. *CoRR*.

Wentao Xu et al. 2021. REST: relational event-driven stock trend forecasting. In *WWW '21: The Web Conference*.

Ikuya Yamada et al. 2020. Wikipedia2vec: An efficient toolkit for learning and visualizing the embeddings of words and entities from wikipedia. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, EMNLP*.

Song Yoojeong, Lee Jae Won, and Lee Jongwooy. 2019. A study on novel filtering and relationship between input-features and target-vectors in a deep learning model for stock price prediction. *Applied Intelligence*.

George Zerveas et al. 2021. A transformer-based framework for multivariate time series representation learning. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*.

Hu Ziniu et al. 2018. Listening to chaotic whispers: A deep learning framework for news-oriented stock trend prediction. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining, WSDM*.

A Appendix

A.1 Datasets

The four datasets (across two news article sources and two stock markets) differ in the companies covered and news sources as depicted in Table 1 in the main paper. The 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.

We also collected daily stock market price-related information - opening, closing, low & high prices, and trading volumes - of 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 not mentioned in any articles for the respective news article sources. GAME can be used for datasets that contain more stocks, i.e., even those that are not mentioned in any news articles, as it captures both global and local textual news information as well as numerical information. However, we restrict the experiments to stocks that are mentioned in the articles for a fair comparison with models such as FAST that are designed to only capture local textual news information, i.e. they cannot capture any news information not associated with any specific companies.

For inter-company relationships, we use Wikidata, 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 (first-order), both Apple and Microsoft are producing computer hardware (second-order), can be extracted from Wikidata. We extracted instances of 57 first and second-order relationship-types identified by (Feng et al., 2019) from the Wikidata dumps dated 7 Jan. 2019. The earliest Wikidata dumps were from 2014. We used Wikidata dumps from 7 Jan. 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 Wikidata dumps that were more recent so that the starting date of the test sets will be after the 7 Jan. 2019 date of the Wikidata dump used to construct the KG.

[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

A.2 Ground-truth Definitions

For $Y^{returns}(t) = [y^{returns}(t), ..., y^{returns}(t + K')]$ over a horizon of K' time-steps, the ground-truth labels for means and volatilities are defined as follows:

$$Y^{returns}_{mean}(t) = \frac{1}{K'} \sum_{k'=0}^{K'} y^{returns}(t + k') \tag{8}$$

$$Y^{returns}_{vol}(t) = \sqrt{\frac{1}{K'} \sum_{k'=0}^{K'} (y^{returns}(t + k') - \mu)^2} \tag{9}$$

where $\mu = Y^{returns}_{mean}(t)$. For correlations between any two assets i and j:

$$Y^{returns}_{corr,ij}(t) = \frac{\sum_{k'=0}^{K'} (x_i - \mu_i)(x_j - \mu_j)}{\sqrt{\sum_{k'=0}^{K'} (x_i - \mu_i)^2} \sqrt{\sum_{k'=0}^{K'} (x_j - \mu_j)^2}} \tag{10}$$

where $x_i = y^{returns}_i(t + k')$, $x_j = y^{returns}_j(t + k')$.

A.3 Metrics

SMAPE is defined as:

$$SMAPE = \frac{100\%}{n} \sum_{i=1}^{n} \frac{|Y^{returns}_i(t) - \hat{Y}^{returns}_i(t)|}{(|Y^{returns}_i(t)| + |\hat{Y}^{returns}_i(t)|)/2} \tag{11}$$

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.

A.4 Settings

To train GAME, we chose the 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 GAME and baselines were generally consistent across all window and horizon periods. Hence, we set the window period $K = 20$ days; and horizon period $K' = 10$ as $K = 20$ corresponds to a trading month, and $K' = 10$ days corresponds to a global regulatory requirement for VaR computations, which we examined in the case-study (see Section 6 of the paper). For the latent

$\mathcal{L}$, we chose the latent dimension $L = 16$ based on experiments with different periods $L \in \{8, 16, 32\}$. For $K = 20$ and $K' = 10$, we found that $L = 16$ led to the best overall performance, and enabled more efficient scaled dot-product operations than if we had chosen larger values for L. An Adam optimizer with a learning rate of 1e-3 with a cosine annealing scheduler is used. Models are implemented in Pytorch and trained for 100 epochs on a 3.60GHz AMD Ryzen 7 Windows desktop with NVIDIA RTX 3090 GPU and 64GB RAM. Training GAME, which has 1.01e6 parameters, takes around two hours on the IN datasets and nine hours on the BE datasets.

A.5 Methodology for Application Case Studies

In this section, we describe the detailed methodology for portfolio allocation and VaR risk measurement used in this paper.

A.5.1 Portfolio Allocation Methodology

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 in a portfolio, by finding an optimal set of weights $\mathbb{W}$ that determine how much capital to invest in each stock, so that portfolio returns can be maximized while minimizing portfolio risk. In this paper, we adopt the risk aversion formulation (Fabozzi et al., 2007) of the mean-variance risk minimization model by Markowitz (1952), which models both portfolio return and risk. Under the risk aversion formulation, the classical mean-variance risk minimization model by Markowitz (1952) is re-formulated to maximize the risk-adjusted portfolio return:

$$max_{\mathbb{W}} \ (\mathbb{W}^T \mu - \lambda \mathbb{W}^T \Sigma \mathbb{W}) \qquad (12)$$

subject to $\mathbb{W}^T \mathbf{1} = 1$. λ, known as the Arrow-Pratt risk aversion index, is used to express risk preferences and is typically set from 2 to 4 (Fabozzi et al., 2007). In this paper, we set $\lambda = 2$ for the experiments. Higher values of $\lambda = 2$ will reduce returns across all models, but the relative differences between models were generally consistent. In this paper, we use the forecasted means of asset returns for μ, i.e. $\tilde{\mu} = \hat{Y}_{mean}^{returns}(t)$; and compute Σ with the forecasted volatilities and correlations of asset returns:

$$\tilde{\Sigma} = D(t) \cdot \hat{Y}_{corr}^{returns}(t) \cdot D(t) \qquad (13)$$

where $D(t)$ is the diagonal matrix filled with $\hat{Y}_{vol}^{returns}(t)$ along the diagonals. 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 that the matrix is invertible, which is important for applications such as portfolio allocation. Forecasting co-variances directly does not guarantee PSD, but forecasting volatilities and correlations separately and computing the co-variance matrix using the volatilities and correlations with the formula: $\tilde{\Sigma} = D(t) \cdot \hat{Y}_{corr}^{returns}(t) \cdot D(t)$, where $D(t)$ is the diagonal matrix filled with $\hat{Y}_{vol}^{returns}(t)$ along the diagonals, allows the co-variance matrix to be PSD.

This application can be viewed as a predictive task as we are using the returns from the window period $t - K$ to $t - 1$ to make forecasts over the future horizon t to $t + K'$ and using these forecasts to determine the resultant weights $\mathbb{W}$ to invest in stocks over the future horizon; and then measuring the portfolio returns realized in this future horizon: $E^{real} = \mathbb{W}^T R^{real}$, 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 selected future period as the evaluation metric, defined as: $\tilde{E} = \frac{E^{real}}{\sigma(E^{real})}$, where $\sigma(E^{real})$ is portfolio return volatility, defined as the one standard deviation of the portfolio returns over the same future period. For this application, the datasets are similarly divided into non-overlapping training/validation/test sets in the ratios 0.7/0.15/0.15, and we evaluate performance based on the average of the risk-adjusted realized portfolio returns across the test set.

A.5.2 Value-at-Risk (VaR) Measurement Methodology

VaR 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 10 day horizon and maintain capital based on this VaR as loss buffers. Exchanges may also collect margins from individual investors based on the VaR of their investment portfolios. VaR measures the loss that an institution may face in the pre-defined horizon with a probability of $p\%$, for e.g., if the 10 day

95% VaR is $1,000,000, it means that there is a 5% probability of losses exceeding $1,000,000 over a 10 day horizon. Whenever the realized losses exceed the VaR, it is regarded as a VaR breach. More formally, we define VaR as:

$$Pr[E^{real} \leq -VaR(p)] = p \tag{14}$$

where E^{real} is the realized portfolio value and the minus sign is added to VaR as we are dealing with losses, i.e. the probability of realized portfolio value (i.e. losses) being more negative than negative VaR. For this application, the portfolio is constructed based on the approach described for the portfolio allocation application at each time-step. This mimics a real-world scenario where financial institutions continually update their portfolios based on market conditions. VaR can be computed as a multiple of the portfolio's volatility:

$$VaR = \phi^{-1}(p) \times \sigma \tag{15}$$

where ϕ is the inverse cumulative distribution function of the standard normal distribution, for e.g. if $p = 95\%$ then $\phi^{-1}(p) = 1.645$. In the classical approach, σ is the historical portfolio volatility over a pre-defined window period. To evaluate the baseline models, we instead 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 (13). Similar to the portfolio allocation application, this can also be viewed as a predictive task as we are using the returns from the window period $t-K$ to $t-1$ to make forecasts over the future horizon t to $t + K'$ 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 exceeds the forecasted VaR in the testing dataset (using the same training/validation/test sets as the portfolio allocation application). 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.