12-2022

Learning dynamic multimodal implicit and explicit networks for multiple financial tasks

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. Learning dynamic multimodal implicit and explicit networks for multiple financial tasks. (2022). *Proceedings of the 2022 IEEE International Conference on Big Data (Big Data), Osaka, Japan, December 17-20*. 825-834.

Available at: https://ink.library.smu.edu.sg/sis_research/8325

Learning Dynamic Multimodal Implicit and Explicit Networks for Multiple Financial Tasks

Gary Ang
School of Computing and Information Systems
Singapore Management University
Singapore
gary.ang.2019@phdcs.smu.edu.sg

Ee-Peng Lim
School of Computing and Information Systems
Singapore Management University
Singapore
eplim@smu.edu.sg

2022 IEEE International Conference on Big Data (Big Data) | 978-1-6654-8045-1/22/$31.00 ©2022 IEEE | DOI: 10.1109/BigData55660.2022.10020722

***Abstract*—Many financial forecasting deep learning works focus on the single task of predicting stock returns for trading with unimodal numerical inputs. Investment and risk management however involves multiple financial tasks - forecasts of expected returns, risks and correlations of multiple stocks in portfolios, as well as important events affecting different stocks - to support decision making. Moreover, stock returns are influenced by large volumes of non-stationary time-series information from a variety of modalities and the propagation of such information across inter-company relationship networks. Such networks could be explicit - observed co-occurrences in online news; or implicit - inferred from time-series information. Such networks are often dynamic, i.e. they evolve across time. Therefore, we propose the Dynamic Multimodal Multitask Implicit Explicit (DynMIX) network model, which pairs explicit and implicit networks across multiple modalities for a novel dynamic self-supervised learning approach to improve performance across multiple financial tasks. Our experiments show that DynMIX outperforms other state-of-the-art models on multiple forecasting tasks, and investment and risk management applications.**

***Index Terms*—Graph neural networks, transformers, attention mechanisms, time-series forecasting, networks, multimodality, embeddings, finance**

I. Introduction

Many financial forecasting deep learning works focus on predicting stock returns to support stock trading decisions with unimodal numerical inputs. To support investment and risk management decisions in financial institutions such as banks, forecasts of the means, volatilities (risks) and correlations of returns of multiple stocks (i.e. multiple variables) in portfolios over future time horizons are required. Forecasting important events that affect companies can also help investment and risk managers understand market dynamics and support decision making [1]. Hence, forecasting models need to support a *multitask multivariate setting*, as illustrated by important questions associated with different forecasting tasks shown in Figure 1. These multitask multivariate forecasts can also be utilized in important applications, such as portfolio allocation and forecasting portfolio Value-at-Risk (VaR) [2]. Financial forecasting is challenging due to the low signal-to-noise ratios and non-stationary nature of financial time-series [3]. Stock returns are influenced by dynamic time-series information from *multiple modalities*, e.g., numerical prices, textual news, categorical events information. Propagation of different information across different inter-company relationship networks also influence stock returns. Such inter-company relationship networks could be based on *explicit* relationships, such as observed co-occurrences in online news or businesses; or based on *implicit* relationships inferred from different financial time-series. Such relationship networks are often *dynamic*, i.e. they evolve across time. Forecasting models that can effectively capture such multimodal time-series information and jointly model both dynamic explicit and implicit networks can help address the aforementioned challenges and improve forecasting accuracy.

Classical methods [4] commonly used for financial forecasting are not designed for multitask multimodal settings, nor network information. Deep learning architectures have been applied to time-series forecasting [5], but are mostly designed for unimodal numerical inputs and single forecasting tasks, and do not capture inter-company relationships. Spatio-temporal network models [6], [7] capture relationships between different time-series but most of them are designed for static explicit networks. Dynamic network models [8] are designed for dynamic explicit networks. Such spatio-temporal and dynamic network models are not designed for networks with nodes having multimodal financial time-series attributes. Some works adopt a multitask approach to forecast financial time-series for trading [9]–[11], but do not model multimodal time-series and dynamic network information.

Hence, in this paper, we propose the **Dyn**amic **M**ultimodal Multitask **I**mplicit E**x**plicit (**DynMIX**) network model framework, as shown in Figure 1. DynMIX captures multivariate and multimodal time-series information, together with dynamic explicit inter-company relationships. To address non-stationary time-series distributions and inter-series correlations, DynMIX learns temporal representations, and also discovers dynamic implicit inter-company relationships that are paired with dynamic explicit networks as different views for dynamic self-supervised learning (SSL) to align and regularize representations across different modalities and networks. DynMIX generates forecasts of means, volatilities, correlations of returns of multiple stocks, as well as forecasts of multiple events (i.e. multilabel prediction task) affecting multiple companies over a future time horizon to support investment and risk management decision-making. Our key contributions are as

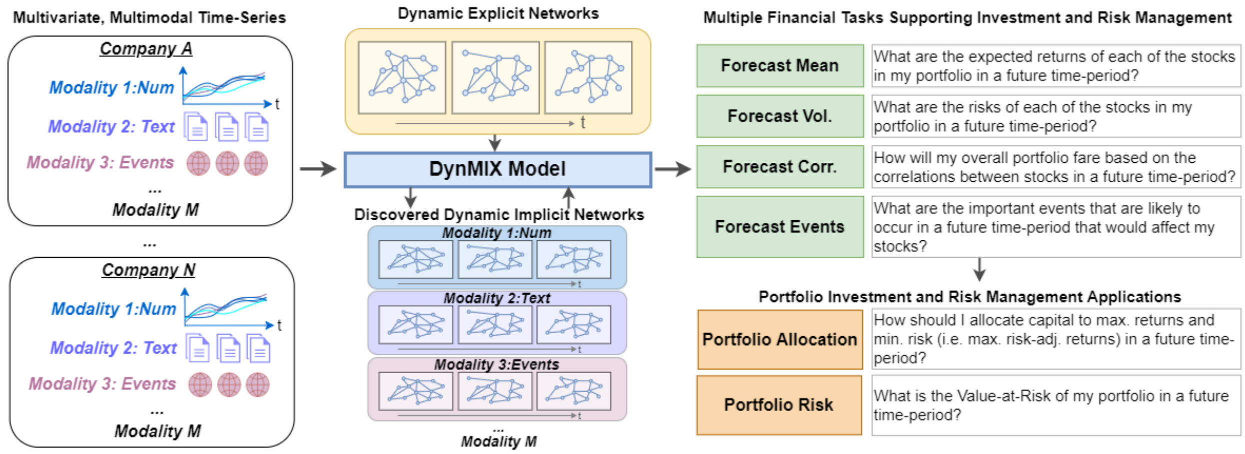

Fig. 1. Model Framework: DynMIX captures multivariate multimodal time-series information and dynamic explicit inter-company relationships as shown on the left. DynMIX also discovers dynamic implicit inter-company relationships from different time-series information and pairs them with explicit networks as different views for dynamic self-supervised learning (SSL) (details in Figure 3). DynMIX generates multiple financial forecasts to support investment and risk management decision-making, as well as industry applications, as shown on the right.

follows:

- To our knowledge, DynMIX is the first model designed to capture multivariate and multimodal time-series information, as well as dynamic explicit and implicit relationship networks for multiple financial forecasting tasks;
- DynMIX learns temporal representations, and also discovers dynamic implicit relationship networks from multimodal time-series and pairs them with dynamic explicit networks as different views for a novel dynamic SSL approach to address non-stationary time-series distributions and correlations;
- We demonstrate that DynMIX outperforms state-of-the-art baselines on multiple forecasting tasks and real-world investment and risk management applications, particularly on volatility and correlation forecasting tasks, and portfolio allocation and VaR applications.

II. Related Work

Financial Time Series Forecasting: Classical methods [4] that are commonly applied to financial time-series forecasting are not designed for multimodal or network information. Deep learning models utilized for time-series forecasting include feed-forward networks [12], convolutional neural networks [13] and recurrent neural networks (RNN) [14], [15]. A review of these works can be found in [5]. Most of these models [12], [16] focus on capturing and predicting numerical information in a single-task setting, and are not designed for unstructured textual information or multimodal information. Time-series Transformer (**TST**) [17] is a recent model based on the transformer encoder architecture designed for numerical time-series information. TST does not model multimodal and network information. Recent works have studied the use of textual news information [18], [19] for financial forecasting. **FAST** [18] uses Time-aware LSTMs [20] to encode textual news information. Similar to TST, FAST also does not capture multimodal nor network information.

Network Learning for Financial Time-Series: 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 [21]. Most GNNs are designed for static networks with static node attributes. A few recent works [19], [22], [23] apply GNNs to prediction tasks on financial time-series data involving predefined static networks. Among the GNN models designed for dynamic networks [24], [25], **EvolveGCN** [8] captures dynamic networks by using a RNN to evolve GCN parameters. However, EvolveGCN and these works are not designed for networks with financial time-series node attributes. Spatio-temporal network models [6], [7], [26], [27], primarily used for traffic forecasting, can handle networks where the node attributes are time-series but are designed for pre-defined static networks. Some recent spatio-temporal network models [28] infer implicit relationships between time-series for forecasting. **MTGNN** [29] uses a graph learning layer to learn the underlying network, before applying interleaved temporal convolution modules and graph convolution modules to capture temporal and spatial dependencies. **DYGAP** [30] utilizes attention mechanisms to learn network structures and then applies diffusion convolutional recurrent neural networks [6] with the learned networks as inputs for forecasting financial returns. MTGNN, DYGAP and other works however assume that a single set of relationships applies across the window period, and are not designed for multimodal information nor dynamic explicit and implicit networks. Existing self-supervised learning (SSL) works focus on static networks in unimodal and single task settings [31], [32]. To our knowledge, DDGCL [33] is among the first to propose SSL for dynamic networks, but assumes a smooth evolutionary process for data points that are close in time which is not true for non-stationary financial time-series whose variances can change sharply. DDGCL is also not designed for multimodal information and multiple dynamic networks.

III. Dynamic Multimodal Multitask Implicit Explicit Network Model

Let V denote a set of companies. We first define $X_t^m = [x^m(t-K), ..., x^m(t-1)] \in \mathbb{R}^{|V|\times K\times d^m}$, as a financial time-series information of all companies over K time steps $[t-K, t-1]$ from modality m (out of M modalities) which could be numerical, textual, event or other type. For continuous information such as numerical or encoded textual information, the dimension d^m represents the number of numerical time-series or the encoding dimension for the textual information. For categorical information such as events that are multi-hot encoded, the dimension d^m represents the number of event-types. DynMIX first encodes X^m with a **modality-specific Gated Recurrent Unit (Enc^m)** to obtain $H_t^m = [h^m(t-K), ..., h^m(t-1)] \in \mathbb{R}^{|V|\times K\times d}$ where d is the common dimension shared among all the modalities.

In the **graph discovery** step for each modality m, DynMIX discovers sequences of inter-company implicit relationship networks and applies a sparsification step. This results in inter-company implicit relationship networks $G_t^{m,imp} = [g^{m,imp}(t-K), ..., g^{m,imp}(t-1)] \in \mathbb{R}^{|V|\times|V|\times K}$ for each modality m. With an input series of inter-company explicit relationship networks $G_t^{exp} = [g^{exp}(t-K), ..., g^{exp}(t-1)] \in \mathbb{R}^{|V|\times|V|\times K}$, DynMIX uses the encoded representations $h^m(t-k)$ of the companies' time-series information at time-step $t-k$ and both the explicit and implicit networks, $g^{exp}(t-K)$ and $g^{m,imp}(t-k)$, as inputs to a **dynamic graph encoding** step to generate the companies' explicit and implicit network representations $\tilde{h}^{m,exp}(t-k)$ and $\tilde{h}^{m,imp}(t-k)$ of dimensions d respectively.

In the **attention-based sequential encoding** step, the sequence of explicit and implicit network representations for modality m, $\tilde{H}_t^{m,exp} = [\tilde{h}^{m,exp}(t-K), \cdots, \tilde{h}^{m,exp}(t-1)] \in \mathbb{R}^{|V|\times K\times d}$ and $\tilde{H}_t^{m,imp} = [\tilde{h}^{m,imp}(t-K), \cdots, \tilde{h}^{m,imp}(t-1)] \in \mathbb{R}^{|V|\times K\times d}$, are combined with the temporal representations $P_t = [p(t-K), \cdots, p(t-1)] \in \mathbb{R}^{|V|\times K\times d}$ learnt by a **time vectorization module** from the corresponding timestamps $T_t \in \mathbb{R}^{|V|\times K\times d^{time}}$ in the window based on the timestamps' day, day of week, and week of year, and projected to the same dimension d as the companies' network representations. The temporal representations P_t capture time-series patterns such as linear and non-linear trends and periodicity, which enable the subsequent time-sensitive attention-based sequential encoding of the sequence of explicit and implicit network representations, resulting in $Z_t^{m,exp} = [z^{m,exp}(t-K), \cdots, z^{m,exp}(t-1)] \in \mathbb{R}^{|V|\times K\times d}$ and $Z_t^{m,imp} = [z^{m,imp}(t-K), \cdots, z^{m,imp}(t-1)] \in \mathbb{R}^{|V|\times K\times d}$. Next, we apply a **dynamic alignment and regularization (DynAR)** step to address non-stationary time-series distributions and correlations with dynamic SSL. DynMIX then uses attention mechanisms in the **multimodal fusion** module to fuse the resultant explicit and implicit network representations $Z_t^{m,exp}$ and $Z_t^{m,imp}$ for each of the modalities m based on learnt importances. After fusing explicit and implicit network representations across M modalities, we obtain $Z_t = [z(t-K), \cdots, z(t-1)] \in \mathbb{R}^{|V|\times K\times d}$. The last hidden state $z(t-1)$ is used to generate both the backcast of the financial price-related input data, and the forecasts of the means, volatilities and correlations of financial returns over the future horizon of L time-steps, i.e. means, volatilities and correlations of $Y_t^{returns} = [y^{returns}(t), ..., y^{returns}(t+L)]$, where $y^{returns}(t) = (price(t) - price(t-1))/price(t-1)$ is the percentage return at time step t, and $price(t)$ is the stock price at time step t. $z(t-1)$ is also used to generate the forecasts of important events in the future horizon of L time-steps. Figures 2 and 3 provide an overview of the architecture of DynMIX and the dynamic SSL approach respectively, which we elaborate on below.

***Graph Discovery**:* From the sequential encodings of the time-series information $H_t^m \in \mathbb{R}^{|V|\times K\times d}$, the graph discovery step discovers implicit relationship networks using the dot-product attention mechanism [34]. Unlike MTGNN [29], DYGAP [30] and other related works [28] which return a single network for the window period of length K, our graph discovery module returns multiple implicit relationship networks $g^{m,imp}(t-k)$'s, one for each time-step $t-k$. Discovering multiple implicit networks allows DynMIX to model the non-stationary nature of evolving inter-company relationships and correlations. We first apply shared linear layers to generate queries $\mathcal{Q}_t^{m,imp}$ and keys $\mathcal{K}_t^{m,imp}$ from the hidden representations H_t^m: $\mathcal{Q}_t^{m,imp} = Linear_{Q-NET}(H_t^m)$; $\mathcal{K}_t^{m,imp} = Linear_{K-NET}(H_t^m)$. A $|V|\times|V|\times K$ attention weight tensor $AW_t^{m,imp}$ can then be computed as the dot-product of $\mathcal{Q}_t^{m,imp}$ and $\mathcal{K}_t^{m,imp}$. To allow richer inter-series interactions to be learnt across time-steps [35], we add a learnable inner weight tensor $W \in \mathbb{R}^{K\times d\times d}$:

$$AW_t^{m,imp} = tanh(\frac{\mathcal{Q}_t^{m,imp} \cdot W \cdot \mathcal{K}_t^{m,imp\intercal}}{\sqrt{d}}) \tag{1}$$

To emphasize the the most important relationships at each time-step, we apply a *sparsification* step, and take the top R relational edges with the highest $AW_t^{m,imp}$ for each time-step in $[t-K, t-1]$. In this paper, we empirically set $R = 10\%$ in our experiments, and consider other R settings in our ablation study (see Section V). We then obtain a sequence of sparse implicit inter-company relationship networks, one for each time-step in the window $[t-K, t-1]$. Specifically, the implicit inter-company relationship network is: $g^{m,imp}(t-k) = (V, e^{m,imp}(t-k), a^{m,imp}(t-k))$ for $k \in \{1, ..., K\}$ where $e^{m,imp}(t-k)$ represents the top-R (v_i, v_j) edges with the largest $AW_t^{m,imp}[v_i, v_j, t-k]$'s values at time-step $t-k$, and $a^{m,imp}(t-k)$ are either the $AW_t^{m,imp}$ of the top R edges at time step $t-k$, or set to zero, i.e.,

$$a^{m,imp}(t-k)[v_i, v_j] = \begin{cases} AW_t^{m,imp}[v_i, v_j, t-k] & \text{if } (v_i, v_j) \in e^{m,imp}(t-k) \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

We stack the sequence of $a^{m,imp}(t-k)$'s in $G_t^{m,imp}$ to obtain the corresponding weighted adjacency tensor $A_t^{m,imp} \in \mathbb{R}^{|V|\times|V|\times K}$ with $A_{t,ij}^{m,imp} \in \mathbb{R}^K$ representing the weighted

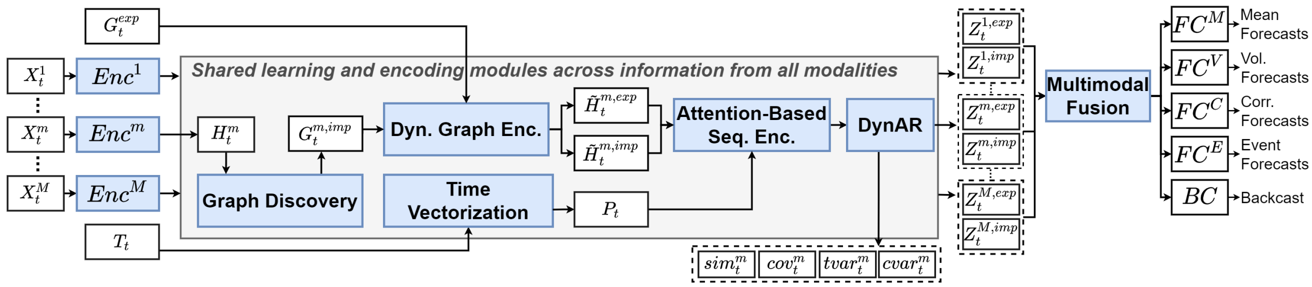

Fig. 2. Model Architecture: DynMIX captures time-series features from M different modalities (X_t^m), e.g., numerical, textual, events; and dynamic explicit networks (G_t^{exp}). Each time-series feature of a modality m is encoded by a *modality-specific encoder (Enc^m)*. The *graph discovery* step discovers dynamic implicit networks for each modality ($G_t^{m,imp}$). The *dynamic graph encoding* step encodes the implicit and explicit networks. The *attention-based sequential encoding* step utilizes temporal representations P_t learnt from time-stamps T_t by a *time vectorization* module to encode the sequence of network and feature representations in a time-sensitive manner. To dynamically align and regularize the explicit and implicit network representations to address non-stationary time-series distributions and correlations, we apply the *dynamic alignment and regularization (DynAR)* step with dynamic SSL (details in Figure 3). The resultant representations $Z_t^{m,exp}$ and $Z_t^{m,imp}$ for each modality m are then fused by the *multimodal fusion* module across M modalities, and used for multiple forecasting tasks.

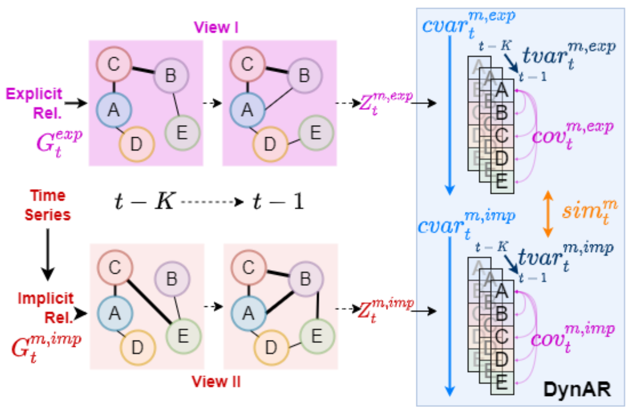

Fig. 3. Dynamic SSL. Dynamic implicit relationship networks discovered from each modality m paired with dynamic explicit networks as different views. The corresponding dynamic explicit and implicit network representations $Z_t^{m,exp}$ and $Z_t^{m,imp}$ are aligned and regularized by utilizing the following loss terms computed in the DynAR step: i) an invariance alignment term, sim_t^m, which minimizes the distance between representations of the same companies; ii) a covariance regularization term, $cov_t^{m,exp/imp}$, which de-correlates representations of different companies; and iii) variance regularization terms, $cvar_t^{m,exp/imp}$ and $tvar_t^{m,exp/imp}$, which maintain a dynamic level of variance in the representations across the company dimension, i.e. companies A to E, and time dimension, i.e. $t-K$ to $t-1$ respectively.

relational edges between company v_i and v_j across the window $[t-K, t-1]$ for modality m, i.e., $A_{t,ij}^{m,imp} = [a^{m,imp}(t-K)[v_i, v_j], \cdots, a^{m,imp}(t-1)[v_i, v_j]]$. We also denote the weighted adjacency tensor for the explicit inter-company relationship network G_t^{exp} as $A_t^{exp} \in \mathbb{R}^{|V|\times|V|\times K}$. No sparsification step is applied so we retain all explicit inter-company relationship information.

Dynamic Graph Encoding: Next, we utilize the sequential encodings of the time-series information H_t^m, and the weighted adjacency tensors for the explicit and implicit inter-company relationship networks, A_t^{exp} and $A_t^{m,imp}$, as inputs to separate weighted dynamic graph convolution steps to generate the dynamic explicit and implicit network representations. For a company v_i, we compute its explicit network representations for the modality m: $\tilde{H}_{t,i}^{m,exp} \in \mathbb{R}^{K\times d}$ across time-steps in the $[t-K, t-1]$ window by aggregating representations from its neighbors $N_t(v_i)$ based on $A_{t,ij}^{exp}$ as follows:

$$\tilde{H}_{t,i}^{m,exp} = \sum_{v_j \in N_t(v_i)} \frac{exp(A_{t,ij}^{exp})}{\sum_{v_{j'} \in N_t(v_i)} exp(A_{t,ij'}^{exp})} \cdot H_{t,j}^m \tag{3}$$

We denote the explicit network representations for all companies by $\tilde{H}_t^{m,exp} \in \mathbb{R}^{|V|\times K\times d}$. We repeat the same steps with $A_t^{m,imp}$ (in place of A_t^{exp}) to obtain the implicit network representations for all companies $\tilde{H}_t^{m,imp} \in \mathbb{R}^{|V|\times K\times d}$. Other GNN-variants can be utilized for this graph convolution step, but we adopt this approach for computational efficiency as it allows us to apply the graph convolution step across multiple time-steps in parallel. Using other common GNNs did not yield any improvement in performance.

Attention-based Sequential Encoding: Inspired by [36], [37], which proposed general frameworks for learning temporal representations, we introduce a time vectorization module within DynMIX that is shared across the different modalities. The time vectorization module takes as input the time-stamps from the time steps in $[t-K, t-1]$ and learns their temporal representations. The input time-stamps are represented as $T_t \in \mathbb{R}^{|V|\times K\times d^{time}}$ where d^{time} denotes the number of dimensions required for capturing the day of week, week and month of year of a time-stamp. The temporal representations learnt by the time vectorization module is denoted by $P_t \in \mathbb{R}^{|V|\times K\times d}$. Functional forms are combined with learnable weights to adaptively learn and combine periodic and non-periodic components within the multivariate financial time-series. This could also be viewed as a time-sensitive version of positional encodings used in transformers that only deal with sequential positions of word tokens [34]. For each component, we apply linear layers and selected activation functions to T_t. For DynMIX, the empirically chosen components are: $\Phi_1 = Linear(T_t)$; $\Phi_2 = cos(Linear(T_t))$;

$\Phi_3 = Sigmoid(Linear(T_t))$; $\Phi_4 = Softplus(Linear(T_t))$, which enable the model to extract linear and non-linear trends, as well as seasonality-based temporal patterns. We then concatenate these components and project them: $P_t = Linear([\Phi_1||\Phi_2||\Phi_3||\Phi_4])$. In the subsequent attention-based sequential encoding step [34], we first add the learnt temporal representations P_t to the dynamic explicit network representations $\tilde{H}_t^{m,exp}$, and then apply linear layers shared across different modalities to generate the queries, keys and values - $\tilde{\mathcal{Q}}_t^{m,exp} = Linear_Q(\tilde{H}_t^{m,exp} + P_t)$, $\tilde{\mathcal{K}}_t^{m,exp} = Linear_K(\tilde{H}_t^{m,exp} + P_t)$, $\tilde{\mathcal{V}}_t^{m,exp} = Linear_V(\tilde{H}_t^{m,exp} + P_t)$, We then apply scaled dot-product attention:

$$\tilde{H}_t^{\prime m,exp} = softmax(\frac{\tilde{\mathcal{Q}}_t^{m,exp} \cdot \tilde{\mathcal{K}}_t^{m,exp\intercal}}{\sqrt{d}})\tilde{\mathcal{V}}_t^{m,exp} \quad (4)$$

followed by a residual connection with layer normalization (LayerNorm), and finally a feed-forward network (FFN) shared across different modalities: $Z_t^{m,exp} = FFN(LayerNorm(\tilde{H}_t^{\prime m,exp} + \tilde{H}_t^{m,exp})$. The output of this step is hence $Z_t^{m,exp} \in \mathbb{R}^{|V| \times K \times d}$. We repeat the same steps with the dynamic implicit network representations $\tilde{H}_t^{m,imp}$ to obtain $Z_t^{m,imp} \in \mathbb{R}^{|V| \times K \times d}$.

Dynamic Self-Supervised Learning*:* To align and regularize the diverse representations across financial time-series information from multiple modalities and dynamic networks in an adaptive manner, we propose a novel dynamic SSL approach inspired by [38]. [38] proposes three loss terms: (a) an *invariance* alignment term that minimizes the distance between the same images in different views; (b) a *covariance* regularization term that de-correlates the representations of different images; and (c) a *variance* regularization term that maintains a minimum level of variation across batches of representations so they do not collapse to a mode. Our proposed approach, as shown in Figure 3, is however distinct in a number of ways: (i) Instead of utilizing representations encoded from two views of the same batch of images that are augmented differently, our approach does not require augmentations as we have two views for each modality naturally formed by pairing the dynamic explicit network representations $Z_t^{m,exp}$ with the dynamic implicit network representations $Z_t^{m,imp}$ for each modality m; (ii) the variance term is computed across companies and time-steps instead of batches of images to allow the representations to be sensitive to differences in variances across companies and time-steps; and (iii) instead of a fixed target for the variance term, we adopt a dynamic target, computed based on the standard deviation of actual returns across companies and time-steps, which allows the variances in the representations to adapt to changes in time-series regimes. For the DynAR step, for each time-step $t-k$ and modality m, we first define the *invariance* $sim^m(t-k)$ between $Z_t^{m,exp}$ and $Z_t^{m,imp}$ as the mean-squared Euclidean distance between each pair of $|V|$ explicit and implicit network representation vectors:

$$sim^m(t-k) = \frac{1}{|V|}\sum_{v \in V} \|z_v^{m,exp}(t-k) - z_v^{m,imp}(t-k)\|_2^2 \quad (5)$$

Summed across all time-steps in $[t-K, t-1]$, the total invariance loss term is sim_t^m. Next, for each time-step $t-k$ and modality m, the *covariance* matrix for each of the views is defined as:

$$C^{m,c}(t-k) = \frac{1}{|V|-1}\sum_{v \in V}(z_v^{m,c}(t-k) - \bar{z}^{m,c}(t-k))(z_v^{m,c}(t-k) - \bar{z}^{m,c}(t-k))^\intercal \quad (6)$$

where $c \in \{exp, imp\}$, and $\bar{z}^{m,c}(t-k) = \frac{1}{|V|}\sum_{v \in V} z_v^{m,c}(t-k)$. We take the sum of the off-diagonals and scale them by the dimension of the representation d:

$$cov^{m,c}(t-k) = \frac{1}{d}\sum_{v_i \neq v_j}[C^{m,c}(t-k)]_{v_i,v_j}^2 \quad (7)$$

Summed across all time-steps in $[t-K, t-1]$, and $c \in \{exp, imp\}$, the total covariance loss term is cov_t^m. Finally, the *variance* regularization term has two components that take into account variances across companies and time-steps in an adaptive manner to address the non-stationary nature of financial time-series distributions. This is achieved by using the variances of returns across company nodes γ_η and time-steps γ_τ as dynamic target levels of variances for the representations. While returns and representation values are different quantities, they can be utilized together as the scale of values of representations (due to normalization) and γ_η/γ_τ based on returns (a relative quantity) are similar, and as the aim is to introduce a relative level of variance for regularization to prevent over-fitting and mode collapse. For each time-step $t-k$, modality m and view c, the variance loss term across the company node dimension is defined as:

$$cvar^{m,c}(t-k) = \frac{1}{d}\sum_{f \in d}\max(0, \gamma_\eta(t-k) - S_\eta(z_f^{m,c}, \epsilon)(t-k)) \quad (8)$$

where:

$$S_\eta(z_f^{m,c}, \epsilon)(t-k) = \sqrt{\frac{1}{|V|}\sum_{v \in V}(z_{v,f}^{m,c}(t-k) - \bar{z}_f^{m,c}(t-k))^2 + \epsilon} \quad (9)$$

$$\gamma_\eta(t-k) = \sqrt{\frac{1}{|V|}\sum_{v \in V}(x_v^{returns}(t-k) - \bar{x}^{returns}(t-k))^2} \quad (10)$$

and $\bar{x}^{returns}(t-k) = \frac{1}{|V|}\sum_{v \in V} x_v^{returns}(t-k)$. ϵ is a small scalar preventing numerical instabilities. Summed across all time-steps in $[t-K, t-1]$ and $c \in \{exp, imp\}$, the variance loss term across the company dimension is $cvar_t^m$. For each company v, modality m and view c, the variance loss term across the time dimension is defined as:

$$tvar_{v,t}^{m,c} = \frac{1}{d}\sum_{f \in d}\max(0, \gamma_{\tau,v} - S_{\tau,v}(z_{v,f}^{m,c}, \epsilon)) \quad (11)$$

where:

$$S_{\tau,v}(z_{v,f}^{m,c}, \epsilon) = \sqrt{\frac{1}{|K|}\sum_{k=1}^{K}(z_{v,f}^{m,c}(t-k) - \bar{z}_{v,f}^{m,c})^2 + \epsilon} \quad (12)$$

$$\gamma_{\tau,v} = \sqrt{\frac{1}{|K|}\sum_{k=1}^{K}(x_v^{returns}(t-k) - \bar{x}_v^{returns})^2} \quad (13)$$

and $\bar{z}_{v,f}^{m,c} = \frac{1}{|K|}\sum_{k=1}^{K} z_{v,f}^{m,c}(t-k)$, $\bar{x}_v^{returns} = \frac{1}{|K|}\sum_{k=1}^{K} x_v^{returns}(t-k)$. Summed across all companies V

and $c \in \{exp, imp\}$, the variance loss term across the time dimension is $tvar_t^m$. Hence, for each modality m, we obtain $\mathcal{L}_{DynAR}^m = \lambda sim_t^m + \omega cov_t^m + \nu(cvar_t^m + tvar_t^m)$, where λ, ω and ν are hyper-parameters. Across M modalities, the loss computed by the DynAR step is $\mathcal{L}_{DynAR}$.

Multimodal Fusion: To learn the importance of different modalities, we use attention-based fusion to fuse the aligned and regularized $Z_t^{m,exp}$ and $Z_t^{m,imp}$ across M modalities. A non-linear transformation is applied to the representations to obtain scalars $s_t^{m,exp} = W^{(1)}tanh(W^{(0)}Z_t^{m,exp} + b)$ and $s_t^{m,imp} = W^{(1)}tanh(W^{(0)}Z_t^{m,imp} + b)$, where $W^{(0)}$ and $W^{(1)}$ are learnable weight matrices and b is the bias vector. Parameters are shared across modalities, as well as explicit and implicit networks. We normalize the scalars with a softmax function to obtain the weights $\beta_t^{m,exp}$ and $\beta_t^{m,imp}$, which are used to fuse representations across modalities and networks:

$$\beta_t^{m,exp} = \frac{exp(s_t^{m,exp})}{\sum_{1 \le m \le M} exp(s_t^{m,exp})} \tag{14}$$

$$\beta_t^{m,imp} = \frac{exp(s_t^{m,imp})}{\sum_{1 \le m \le M} exp(s_t^{m,imp})} \tag{15}$$

$$Z_t = \sum_{1 \le m \le M} \beta_t^{m,exp} Z_t^{m,exp} + \sum_{1 \le m \le M} \beta_t^{m,imp} Z_t^{m,imp} \tag{16}$$

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

Forecasting and Loss Functions: In the forecasting step, we use fully connected layers to generate the backcast of the numerical price-related input data (say, modality p) and forecasts of the means and volatilities of company stock returns over the selected horizon period L, as well as forecasts of events across the selected horizon period L normalized with a softmax operation:

$$\hat{X}_t^p = BC(z(t-1)) \tag{17}$$

$$\hat{Y}_{mean,t}^{returns} = FC_M(z(t-1)) \tag{18}$$

$$\hat{Y}_{vol,t}^{returns} = FC_V(z(t-1)) \tag{19}$$

$$\hat{Y}_t^{events} = softmax_{e \in d^{events}}(FC_E(z(t-1))) \tag{20}$$

where d^{events}, the dimension of event information, represents the number of event-types, and the softmax operation normalizes the output of FC_E across the event dimension. While DynMIX can backcast time-series information from multiple modalities, we backcast numerical price-related information as most of the multiple tasks in this paper focus on forecasts of numerical targets. To forecast correlations of company stock returns over the horizon period L, we use the weights from the linear layers in the graph discovery step: $\mathcal{Q}_{corr,t} = Linear_{Q-NET}(z(t-1)), \mathcal{K}_{corr,t} = Linear_{K-NET}(z(t-1))$. This allows what was learnt when discovering the dynamic implicit inter-company relationships to be leveraged for correlation forecasts:

$$\hat{Y}_{corr,t}^{returns} = FC_C(tanh(\frac{\mathcal{Q}_{corr,t} \cdot \mathcal{K}_{corr,t}^{\intercal}}{\sqrt{d}})) \tag{21}$$

The ground-truth labels for means, volatilities and correlations of returns in the horizon L, i.e. $Y_{mean,t}^{returns}$, $Y_{vol,t}^{returns}$ and $Y_{corr,t}^{returns}$ respectively, are computed following [39]. We use the frequency distribution of the events in the horizon of L time-steps as the ground-truth event labels for training, defined as:

$$Y_t^{events} = softmax_{e \in d^{events}} \sum_{l=0}^{L} y^{events}(t+l) \tag{22}$$

We compute the loss between the forecasts and the respective ground-truths with root mean squared loss (RMSE), and use the total loss (including the loss from the DynAR step $\mathcal{L}_{DynAR}$ defined earlier) as the training objective:

$$\begin{aligned}\mathcal{L}_{total} = \mathcal{L}_{DynAR} + \mathcal{L}_{backcast}(X_t^p, \hat{X}_t^p) \\ + \mathcal{L}_{mean}(Y_{mean,t}^{returns}, \hat{Y}_{mean,t}^{returns}) + \mathcal{L}_{vol}(Y_{vol,t}^{returns}, \hat{Y}_{vol,t}^{returns}) \\ + \mathcal{L}_{corr}(Y_{corr,t}^{returns}, \hat{Y}_{corr,t}^{returns}) + \mathcal{L}_{event}(Y_t^{events}, \hat{Y}_t^{events})\end{aligned} \tag{23}$$

IV. Experiments

Datasets: We conduct experiments with four datasets, comprising textual information of online news articles from two financial news portals, numerical information of daily stock market price-related information of companies listed on two stock markets, as well as dynamic explicit inter-company relationship networks of these companies and time-series of events. The frequency of all information is daily and spans Apr. 2015 to Dec. 2019.

Online news data. The two online news article sources are: i) Investing (**IN**)[1]; and ii) Benzinga (**BE**)[2] which covers news articles drawn from a wide range of mainstream providers, analysts and blogs. Following [19], we use a pre-trained Wikipedia2Vec [40] 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. The representation of each news article is the average of embeddings of words in each news article for each day generated with the pre-trained Wikipedia2Vec embedding model. Other approaches for encoding will be explored in future work.

Company's numerical data. We collected daily stock market price-related information - returns, opening, closing, low & high prices, trading volumes, volume-weighted average prices, shares outstanding - of the two stock markets - NYSE (**NY**) and NASDAQ (**NA**) - from the Center for Research in Security Prices. We filter out companies or stocks 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.

Dynamic explicit network and event data. We obtain dynamic explicit inter-company relationship networks and time-series of events from the Global Database of Events, Language

[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

and Tone (GDELT) Global Knowledge Graphs (GKG) [41]. GDELT is a research collaboration that monitors newspapers of 65 different languages globally and is updated every 15 minutes, resulting in around 2.5TB of data annually [41]. GKG processes GDELT data and extracts entities such as organizations and events [42]. We retrieve GKG data via BigQuery from the public GKG dataset on the Google Cloud Platform. To extract inter-company relationship networks and obtain time-series of events associated with each company from GKG, we build a look-up table of stock symbols of companies in NY and NA with variations of their names; and use Levenshtein Distance [43] to calculate similarities between the different names of organizations in the GKG dataset with the names in the look-up table, and tag the GKG record with the corresponding stock symbol if the similarity score is above a threshold of 0.75, determined empirically. The relatively high threshold of 0.75 allows us to reduce the noisiness of the processed dataset and obtain samples that have been tagged at high confidence levels. The dynamic explicit inter-company relationship networks are constructed based on co-occurrences of tagged companies in GKG records, aggregated on a daily basis. The time-series of events associated with each tagged company is also aggregated on a daily basis and transformed into a multi-hot vector, representing the daily frequencies of events for each company. Examples of events extracted from GKG include *ECON_IPO* and *HEALTH_PANDEMIC*, which represent the company being affected by an economic-related initial public offer listing event, or a health-related pandemic event respectively. The four datasets obtained by combining IN and BE online news data with companies' stock data from NYSE (NY) and NASDAQ (NA) with the dynamic explicit networks and events from GKG are shown in Table I. We adopt a sliding window approach [44] to extract input features in the window $[t-K, t-1]$ and returns-related labels, i.e. ground-truth means, volatilities and correlations of returns, as well as event labels, in the horizon $[t, t+L]$. We obtain 1196 data samples (or time steps) for each dataset, and divide them into non-overlapping training/validation/testing sets in the ratios 0.7/0.15/0.15.

Tasks and Metrics: We compare DynMIX with baselines on four predictive tasks: forecasting of i) means, ii) volatilities, iii) correlations of stock price percentage returns; and iv) events over the horizon L. We use RMSE, mean absolute error (MAE) and symmetric mean absolute percentage error (SMAPE) as metrics for forecasts of means, volatilities and correlations [45]. 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 that gives equal importance to both under- and over-forecasts required in this evaluation context. For forecasts of events, we use Normalized Discounted Cumulative Gain (NDCG), a metric used for evaluating ranking quality, to compare the forecasted event labels ranked by the output values after the softmax operation (i.e. ordered based on the output of Equation 20) with the ground-truth event labels ranked by the frequency of the events in the horizon period (i.e. ordered based on the output of Equation 22). We choose NDCG as analysts are interested in more important events that rank higher based on frequencies. Other metrics such as F1 do not distinguish between importances of different events, e.g., between an event that occurred once or 20 times.

Baselines and Settings: We compare DynMIX against a **GRU** model, and the following state-of-the-art baselines (see Section II): **TST** [17], a transformer encoder designed for time-series information; **FAST** [18] that captures textual information with Time-Aware LSTMs [20]; **MTGNN** [29] and **DYGAP** [30] that learn implicit networks from time-series data; and **EvolveGCN-H** and **EvolveGCN-O** [8], two variants of a dynamic network model. FAST and DYGAP were designed for financial time-series. For all baselines, we concatenate numerical price-related, textual news-related, and categorical event information as inputs for a fair comparison, and add fully-connected layers to all baselines to forecast means, volatilities and correlations of stock price percentage returns, as well as forecast events. We set the window period K=20 days; and horizon period L=10. K=20 corresponds to a trading month, and L=10 days corresponds to a global regulatory requirement for VaR computations, which we examine in an application (see Section VI). Dimensions of hidden representations are fixed at 64 across all models. We follow [38] and set both λ and ν to 25, and ω to 1. An Adam optimizer with a learning rate of 1e-3 and a cosine annealing scheduler are used. Models are implemented in Pytorch and trained for 50 epochs. The DynMIX model has 9e5 parameters and takes 10-20 minutes per training epoch on a 3.60GHz AMD Ryzen 7 Windows desktop with NVIDIA RTX 3090 GPU and 64GB RAM.

Results: Tables II and III set out the results of the forecasting experiments on the IN and BE datasets. Across all tasks and datasets, DynMIX out-performs baselines for most metrics. For the task of forecasting **means**, the performance differences between DynMIX and baselines is clearer for the larger BE datasets than the IN datasets. Among baselines, TST and MTGNN designed for time-series information show relatively better performance. On the tasks of forecasting **volatilities** and **correlations**, the differences in performance between DynMIX and baselines are clearer as compared with forecasting means across all datasets. There is also greater dispersion and variation in performances across baselines, which indicates the greater difficulty of these tasks due to non-stationary time-series distributions and inter-series correlations, and demonstrates the usefulness of DynMIX's key features - learning temporal representations, discovery of implicit networks and dynamic SSL. On the task of forecasting **events**, DynMIX also out-performs most baselines across most datasets, and we also observe greater dispersion and variation in performances of baselines on this task.

V. Ablation Studies

Table IV shows the results of the ablation studies for DynMIX on the IN-NY dataset. We observe similar sensitivities for

TABLE I
DATASET OVERVIEW. FOR NUM. EDGES, AVG. EDGE WEIGHTS OF EXPLICIT NETWORKS, AND EVENT DENSITY, I.E. NUM. OF EVENT LABELS PER COMPANY PER LABEL TYPE, WE SHOW AVG. AND STD. DEVIATIONS ACROSS 1196 DAILY NETWORK SNAPSHOTS/DAYS.

	IN-NY	**IN-NA**	**BE-NY**	**BE-NA**
Num. articles	189,917		1,295,491	
Num. stocks	336	371	1,693	1,705
Num. edges	2,212±1,574	1,217±387	4,913±2,887	3,973±1,749
Avg. edge weights	3,547±2,911	3,901±2,218	1,537±1,386	1,541±1,448
Num. event types	2,177	2,135	2,182	2,186
Event density	0.45±0.16	0.33±0.13	0.09±0.03	0.09±0.03

TABLE II
MEAN, VOLATILITY, CORRELATION FORECAST RESULTS. LOWER BETTER FOR ALL METRICS. BEST MODEL(S) IN BOLD; SECOND-BEST MODEL(S) UNDERLINED FOR THIS AND SUBSEQUENT TABLES.

	IN-NY			**IN-NA**			**BE-NY**			**BE-NA**		
	RMSE	MAE	SMAPE	RMSE	MAE	SMAPE	RMSE	MAE	SMAPE	RMSE	MAE	SMAPE
	Mean Forecasts											
GRU	0.0744	0.0144	1.4579	0.0356	0.0175	1.4801	0.1341	0.0343	1.4357	0.6119	0.0851	1.4587
TST	0.0742	0.0140	1.3844	0.0335	0.0155	1.3631	0.1286	0.0251	1.3139	0.5373	0.0637	1.5380
FAST	0.0742	0.0141	1.3511	0.0362	0.0164	1.7424	0.1464	0.0260	1.3403	0.6358	0.0673	1.3801
MTGNN	0.0712	0.0139	1.3002	0.0314	0.0149	1.4843	0.1386	0.0338	1.5168	0.4844	0.0609	1.3218
DYGAP	0.0723	0.0146	1.4430	0.0353	0.0157	1.4074	0.1430	0.0263	1.4411	0.6361	0.0670	1.3520
EVOLVEGCN-H	0.0743	0.0143	1.5058	0.0394	0.0176	1.3273	0.1470	0.0288	1.3540	0.6444	0.0823	1.4600
EVOLVEGCN-O	0.0750	0.0153	1.4372	0.0354	0.0158	1.2921	0.1453	0.0268	1.3679	0.6325	0.0726	1.4014
DYNMIX	**0.0539**	**0.0111**	**1.1693**	**0.0244**	**0.0128**	**1.2080**	**0.1058**	**0.0194**	**1.2391**	**0.4164**	**0.0471**	**1.2337**
	Volatility Forecasts											
GRU	0.2331	0.0507	0.6244	0.1188	0.0599	0.6384	0.4181	0.1049	1.1047	1.9152	0.2418	0.9816
TST	0.2330	0.0483	0.5578	0.1109	0.0559	**0.6046**	0.3974	0.0894	0.7063	1.7047	0.2133	0.8463
FAST	0.2332	0.0485	0.5595	0.1244	0.0605	0.6338	0.4688	0.0990	0.7521	1.9993	0.2316	0.7935
MTGNN	0.2011	0.0529	0.6206	0.1077	0.0565	0.6178	0.4122	0.0919	0.7488	1.5066	0.2023	0.7860
DYGAP	0.2224	0.0497	**0.5475**	0.1234	0.0606	0.6359	0.4542	0.0928	0.7028	2.0003	0.2410	0.8589
EVOLVEGCN-H	0.2331	0.0485	0.5602	0.1253	0.0619	0.6470	0.4688	0.1012	0.7670	2.0097	0.2547	0.8832
EVOLVEGCN-O	0.2328	0.0491	0.5831	0.1246	0.0607	0.6370	0.4656	0.1010	0.7646	1.9970	0.2419	0.8616
DYNMIX	**0.1491**	**0.0410**	0.5483	**0.0870**	**0.0510**	0.6086	**0.3284**	**0.0763**	**0.7011**	**1.2956**	**0.1667**	**0.7227**
	Correlation Forecasts											
GRU	0.5361	0.4671	1.5484	0.5163	0.4425	1.4072	0.5341	0.4600	1.4455	0.5180	0.4416	1.4193
TST	0.5145	0.4498	1.4646	0.5069	0.4400	1.4802	0.5235	0.4540	1.4553	0.5099	0.4374	1.4661
FAST	0.5087	0.4414	1.3395	0.5099	0.4394	1.4239	0.5246	0.4546	1.4533	0.5112	0.4380	1.4374
MTGNN	0.5215	0.4558	1.5012	0.5167	0.4438	1.4078	0.5288	0.4576	1.4505	0.5268	0.4398	1.5195
DYGAP	0.5415	0.4562	1.4074	0.5074	0.4362	1.4192	0.5125	0.4443	1.4460	0.5060	0.4346	1.4738
EVOLVEGCN-H	0.5104	0.4448	1.3996	0.5058	0.4390	1.4681	0.5192	0.4520	1.4842	0.5270	0.4426	1.5217
EVOLVEGCN-O	0.5118	0.4466	1.4194	0.5064	0.4390	1.4564	0.5228	0.4536	1.4592	0.5033	0.4348	1.5197
DYNMIX	**0.4272**	**0.3551**	**1.0844**	**0.4515**	**0.3810**	**1.2256**	**0.4898**	**0.4173**	**1.3042**	**0.4848**	**0.4095**	**1.3461**

TABLE III
EVENT FORECAST RESULTS (NDCG). HIGHER BETTER.

	IN-NY	IN-NA	BE-NY	BE-NA
GRU	0.5909	0.3336	0.3091	0.2336
TST	0.5705	0.3513	0.2870	0.2179
FAST	0.6106	0.3549	0.3074	0.2246
MTGNN	0.5907	**0.3710**	0.2911	0.2317
DYGAP	0.6113	0.3485	0.3069	0.2126
EVOLVEGCN-H	0.4757	0.2743	0.2136	0.1542
EVOLVEGCN-O	0.5717	0.3512	0.2904	0.2193
DYNMIX	**0.6182**	0.3626	**0.3163**	**0.2474**

the other datasets. Omitting the novel dynamic SSL approach (**w/o. dynamic SSL**), i.e. excluding implicit networks as the second view and the DynAR step, leads to a substantial drop in performance across all four tasks. When we set γ_η/γ_τ to a fixed value of 1 per [38] instead of using a dynamic target based on returns variances (**w/o. dynamic var. target**), we also observe a clear decline in performance across all four tasks. When we vary the degree of sparsification by setting R=50% instead of 10% (**R=50%**), we see significant variations in performance, indicating the importance of dynamic implicit network information. Not utilizing the temporal representation generated by the time vectorization module in the attention-based sequential encoding step (**w/o. time vect.**) leads to a greater drop in performance on forecasting means than other tasks. In general, we observe the key features of DynMIX working together to achieve the best performance on the multiple tasks.

TABLE IV
ABLATION STUDIES ON IN-NY DATASET. LOWER BETTER FOR RMSE, MAE, SMAPE, HIGHER BETTER FOR NDCG.

	RMSE	MAE	SMAPE
		Mean Forecasts	
w/o. dynamic SSL	0.0579	0.0124	1.1719
w/o dynamic var. target	0.0569	0.0121	1.1782
R=50%	0.0558	0.0121	1.1789
w/o. time vect.	0.0562	0.0121	1.1759
DynMIX	**0.0539**	**0.0111**	**1.1693**
		Volatility Forecasts	
w/o. dynamic SSL	0.1527	0.0422	0.5544
w/o dynamic var. target	0.1511	0.0420	0.5532
R=50%	0.1501	0.0415	0.5503
w/o. time vect.	0.1495	0.0419	0.5586
DynMIX	**0.1491**	**0.0410**	**0.5483**
		Correlation Forecasts	
w/o. dynamic SSL	0.4973	0.4302	1.3785
w/o dynamic var. target	0.4399	0.3685	1.1225
R=50%	0.4304	0.3590	1.0983
w/o. time vect.	0.4280	0.3562	1.0948
DynMIX	**0.4272**	**0.3551**	**1.0844**
		NDCG	
		Event Forecasts	
w/o. dynamic SSL		0.6059	
w/o dynamic var. target		0.5617	
R=50%		0.5565	
w/o. time vect.		0.5861	
DynMIX		**0.6182**	

TABLE V
APP. HIGHER BETTER FOR $\mathcal{R}$; LOWER BETTER FOR % BR.

	IN-NY		**IN-NA**	
	$\mathcal{R}$	% Br.	$\mathcal{R}$	% Br.
GRU	0.04	34.4%	1.20	2.2%
TST	0.05	22.8%	0.32	2.2%
FAST	0.03	15.6%	0.55	2.8%
MTGNN	0.67	4.4%	0.33	8.9%
DYGAP	0.41	6.7%	0.61	3.9%
EVOLVEGCN-H	0.01	13.9%	0.06	13.9%
EVOLVEGCN-O	0.06	15.6%	0.32	5.0%
DYNMIX	**2.47**	**3.9%**	**3.31**	**1.1%**

VI. APPLICATIONS

Following the methodology in [2], [46], we utilize the model forecasts for two applications to evaluate the quality of DynMIX's forecasts against the baselines. **Portfolio allocation** seeks to optimize the proportion of capital invested in each stock within a portfolio, by finding an optimal set of allocation 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 [46] of the mean-variance risk minimization model by [47] for portfolio allocation. The key inputs to the mean-variance risk minimization model are the *means* and *co-variances* of returns. Under the classical approach, which we use as a *naive* approach in this section, means and co-variances are computed using historical returns. We use the resultant allocation weights $\mathbb{W}^{naive}$ to construct a portfolio of stocks, and compute the risk-adjusted portfolio returns E^{naive} as the realized portfolio returns divided by realized portfolio volatility. For DynMIX and baselines, we use model forecasts to compute the means and co-variances as inputs to the mean-variance risk minimization model. Similarly, we use the resultant allocation weights $\mathbb{W}^{forecast}$ to construct a portfolio of stocks and compute the risk-adjusted portfolio returns $E^{forecast}$ as the realized portfolio returns divided by realized portfolio volatility. Actual returns of a portfolio of stocks depend on the time-series and time periods under consideration. Hence, for better comparability, we evaluate the performance of DynMIX and baselines relative to the naive approach by computing the ratio $\mathcal{R} = E^{forecast}/E^{naive}$. We compute the averages of $\mathcal{R}$ across the testing set (sampled per Section IV). The second application **VaR** [2] is a key measure of risk used in financial institutions. VaR measures the loss that an institution may face in the pre-defined horizon with a probability of $p\%$, e.g., if the 10 day 95% VaR is \$1m, it means that there is a 5% probability of losses exceeding \$1m over a 10 day horizon. Whenever the realized portfolio loss exceeds the forecasted VaR, we call it a VaR breach. We use the model forecasts to compute 10 day 95% VaR forecasts at the portfolio-level, and evaluate model performances based on percentage VaR breaches (% Br.), i.e. the percentage of losses in the testing set that led to VaR breaches. Models that are able to make accurate forecasts of VaR should have lower % Br. We conduct and report experiments on the IN-NY and IN-NA datasets with fewer stocks than the BE-NY and BE-NA datasets (as shown in Table I), as a smaller pool of potential stocks usually presents a greater challenge for these two applications by limiting potential returns and risk diversification. As shown in Table V, on the **portfolio allocation** application, portfolios constructed using the forecasts from DynMIX achieve better relative performance ($\mathcal{R}$) for both datasets. Similarly, on the **VaR** application, DynMIX also out-performs the baselines, with lower percentage VaR breaches (% Br.). For both applications, we observe significant variance in performance for the baselines, with a number of baselines performing worse than the naive approach ($\mathcal{R} < 1$), or high levels of percentage VaR breaches, demonstrating the difficulty of these applications.

VII. CONCLUSION AND FUTURE WORK

In this paper, we propose DynMIX, a novel model that captures multimodal financial time-series information and pairs discovered dynamic implicit inter-company relationship networks with dynamic explicit inter-company relationship networks as different views for a novel dynamic SSL task. Based on extensive experiments conducted on real-world datasets, we show that DynMIX outperforms baselines on forecasting tasks and important real-world financial applications. DynMIX can be applied to information from other modalities (e.g., audio of company earnings calls), and other types of dynamic network information (e.g., inter-company transaction networks). Future work could also explore extending DynMIX for heterogeneous networks.

ACKNOWLEDGMENT

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

[1] S. Deng, H. Rangwala, and Y. Ning, "Understanding event predictions via contextualized multilevel feature learning," in *ACM Int. Conf. on Info. and Knowledge Management*, 2021.
[2] T. J. Linsmeier and N. D. Pearson, "Value at risk," *Financial Analysts Journal*, vol. 56, no. 2, pp. 47–67, 2000.
[3] C. L. Giles, S. Lawrence, and A. C. Tsoi, "Noisy time series prediction using recurrent neural networks and grammatical inference," *Machine Learning.*, vol. 44, no. 1/2, pp. 161–183, 2001.
[4] G. Tunnicliffe Wilson, "Time series analysis: Forecasting and control," *Journal of Time Series Analysis*, vol. 37, 2016.
[5] B. Lim and S. Zohren, "Time series forecasting with deep learning: A survey," *Philo. Trans. of the Royal Society*, vol. 379, no. 2194, 2021.
[6] Y. Li, R. Yu, C. Shahabi, and Y. Liu, "Diffusion convolutional recurrent neural network: Data-driven traffic forecasting," in *Int. Conf. on Learning Representations*, 2018.
[7] T. Mallick, M. Kiran, B. Mohammed, and P. Balaprakash, "Dynamic Graph Neural Network for Traffic Forecasting in Wide Area Networks," in *2020 IEEE Int. Conf. on Big Data*, 2020.
[8] A. Pareja, G. Domeniconi, J. Chen, T. Ma, T. Suzumura, H. Kanezashi, T. Kaler, T. B. Schardl, and C. E. Leiserson, "Evolvegcn: Evolving graph convolutional networks for dynamic graphs," in *AAAI Conf. on AI*, 2020.
[9] L. Yang, T. L. J. Ng, B. Smyth, and R. Dong, "HTML: hierarchical transformer-based multi-task learning for volatility prediction," in *The World Wide Web Conf.*, 2020.
[10] R. Sawhney, P. Khanna, A. Aggarwal, T. Jain, P. Mathur, and R. R. Shah, "Voltage: Volatility forecasting via text audio fusion with graph convolution networks for earnings calls," in *Conf. on Empirical Methods in NLP*, 2020.
[11] R. Sawhney, P. Mathur, A. Mangal, P. Khanna, R. R. Shah, and R. Zimmermann, "Multimodal multi-task financial risk forecasting," in *ACM Int. Conf. on Multimedia*, 2020.
[12] B. N. Oreshkin, D. Carpov, N. Chapados, and Y. Bengio, "N-BEATS: neural basis expansion analysis for interpretable time series forecasting," in *Int. Conf. on Learning Representations*, 2020.
[13] L. Pantiskas, K. Verstoep, and H. E. Bal, "Interpretable multivariate time series forecasting with temporal attention convolutional neural networks," in *IEEE Symp. Series on Computational Intelligence*, 2020.
[14] Y. Qin, D. Song, H. Chen, W. Cheng, G. Jiang, and G. W. Cottrell, "A dual-stage attention-based recurrent neural network for time series prediction," in *Int. Joint Conf. on AI*, 2017.
[15] Y. Liu, C. Gong, L. Yang, and Y. Chen, "DSTP-RNN: A dual-stage two-phase attention-based recurrent neural network for long-term and multivariate time series prediction," *Expert Syst. Appl.*, vol. 143, 2020.
[16] D. Huynh, G. Audet, N. Alabi, and Y. Tian, "Stock Price Prediction Leveraging Reddit: The Role of Trust Filter and Sliding Window," in *2021 IEEE Int. Conf. on Big Data*, 2021.
[17] G. Zerveas, S. Jayaraman, D. Patel, A. Bhamidipaty, and C. Eickhoff, "A transformer-based framework for multivariate time series representation learning," in *ACM Int. Conf. on Knowledge Discovery and Data Mining*, 2021.
[18] R. Sawhney, A. Wadhwa, S. Agarwal, and R. R. Shah, "FAST: financial news and tweet based time aware network for stock trading," in *Conf. of the European Chap. of the Assoc. for Computational Linguistics*, 2021.
[19] G. Ang and E.-P. Lim, "Learning knowledge-enriched company embeddings for investment management," in *ACM Int. Conf. on AI in Finance*, 2021.
[20] I. M. Baytas, C. Xiao, X. Zhang, F. Wang, A. K. Jain, and J. Zhou, "Patient subtyping via time-aware LSTM networks," in *ACM Int. Conf. on Knowledge Discovery and Data Mining*, 2017.
[21] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl, "Neural message passing for quantum chemistry," in *Int. Conf. on Machine Learning*, 2017.
[22] F. Feng, X. He, X. Wang, C. Luo, Y. Liu, and T. Chua, "Temporal relational ranking for stock prediction," *ACM Trans. Inf. Syst.*, vol. 37, no. 2, pp. 27:1–27:30, 2019.
[23] G. Ang and E.-P. Lim, "Guided attention multimodal multitask financial forecasting with inter-company relationships and global and local news," in *Annual Meeting of the Assoc. for Computational Linguistics*, 2022.
[24] P. Goyal, N. Kamra, X. He, and Y. Liu, "Dyngem: Deep embedding method for dynamic graphs," *CoRR*, 2018.
[25] E. Hajiramezanali, A. Hasanzadeh, K. R. Narayanan, N. Duffield, M. Zhou, and X. Qian, "Variational graph recurrent neural networks," in *Annual Conf. on Neural Info. Processing Systems*, 2019.
[26] B. Yu, H. Yin, and Z. Zhu, "Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting," in *Int. Conf. on Learning Representations*, 2018.
[27] Y. Wang, Q. Zheng, J. Ruan, Y. Gao, Y. Chen, X. Li, and B. Dong, "T-EGAT: A Temporal Edge Enhanced Graph Attention Network for Tax Evasion Detection," in *2020 IEEE Int. Conf. on Big Data*, 2020.
[28] D. Zügner, F. Aubet, V. G. Satorras, T. Januschowski, S. Günnemann, and J. Gasthaus, "A study of joint graph inference and forecasting," *Int. Conf. on Machine Learning, Time Series Workshop*, 2021.
[29] Z. Wu, S. Pan, G. Long, J. Jiang, X. Chang, and C. Zhang, "Connecting the dots: Multivariate time series forecasting with graph neural networks," in *ACM Int. Conf. on Knowledge Discovery and Data Mining*, 2020.
[30] A. Uddin, X. Tao, and D. Yu, "Attention based dynamic graph learning framework for asset pricing," in *Int. Conf. on Info. and Knowledge Management*, 2021.
[31] K. Hassani and A. H. K. Ahmadi, "Contrastive multi-view representation learning on graphs," in *Int. Conf. on Machine Learning*, 2020.
[32] P. Velickovic, W. Fedus, W. L. Hamilton, P. Liò, Y. Bengio, and R. D. Hjelm, "Deep graph infomax," in *Int. Conf. on Learning Representations*, 2019.
[33] S. Tian, R. Wu, L. Shi, L. Zhu, and T. Xiong, "Self-supervised representation learning on dynamic graphs," in *ACM Int. Conf. on Info. and Knowledge Management*, 2021.
[34] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Annual Conf. on Neural Info. Processing Systems*, 2017.
[35] M. Nickel, V. Tresp, and H. Kriegel, "A three-way model for collective learning on multi-relational data," in *Int. Conf. on Learning Representations*, 2011.
[36] S. M. Kazemi, R. Goel, S. Eghbali, J. Ramanan, J. Sahota, S. Thakur, S. Wu, C. Smyth, P. Poupart, and M. A. Brubaker, "Time2vec: Learning a vector representation of time," *CoRR*, 2019.
[37] L. B. Godfrey and M. S. Gashler, "Neural decomposition of time-series data for effective generalization," *IEEE Trans. Neural Networks Learn. Syst.*, vol. 29, no. 7, pp. 2973–2985, 2018.
[38] A. Bardes, J. Ponce, and Y. LeCun, "Vicreg: Variance-invariance-covariance regularization for self-supervised learning," in *Int. Conf. on Learning Representations*, 2022.
[39] J. Daníelsson, *Financial Risk Forecasting*. John Wiley & Sons, Ltd, 2012.
[40] I. Yamada, A. Asai, J. Sakuma, H. Shindo, H. Takeda, Y. Takefuji, and Y. Matsumoto, "Wikipedia2vec: An efficient toolkit for learning and visualizing the embeddings of words and entities from wikipedia," in *Conf. on Empirical Methods in NLP, System Demos*, 2020.
[41] GDELT. (2001) The GDELT Project. [Online]. Available: https://www.gdeltproject.org/
[42] K. Leetaru, T. Perkins, and C. Rewerts, "Cultural computing at literature scale," *D-Lib Magazine*, vol. 20, 2014.
[43] RapidFuzz. (2001) Levenshtein distance matching. [Online]. Available: https://maxbachmann.github.io/RapidFuzz/Usage/distance/Levenshtein.html
[44] N. Wu, B. Green, X. Ben, and S. O'Banion, "Deep transformer models for time series forecasting: The influenza prevalence case," *CoRR*, 2020.
[45] R. Godahewa, C. Bergmeir, G. I. Webb, R. J. Hyndman, and P. Montero-Manso, "Monash time series forecasting archive," in *Annual Conf. on Neural Info. Processing Systems*, 2021.
[46] F. Fabozzi, *Robust Portfolio Optimization and Management*. Wiley, 2007.
[47] H. Markowitz, "Portfolio selection," *The Journal of Finance*, vol. 7, no. 1, pp. 77–91, 1952.