Forecasting by Learning Dynamic External Influence

Anonymous ACL submission

Abstract

Although there has been tremendous progress in the development of learning methods for forecasting, these approaches learn solely from historical data and overlook the external factors. This work, for the first time, addresses the problem of forecasting under dynamic external influences by proposing learning mechanisms that not only learn from historical trends but also incorporate external knowledge available in the form of temporal knowledge graphs. Because no such datasets or temporal knowledge graphs are available, we study this problem using stock market data and develop comprehensive temporal knowledge graph datasets. In our proposed approach, we model relations in external temporal knowledge graphs as events in a Hawkes process on graphs and use techniques from heterogeneous graph attention networks and time-aware translation-based knowledge graph embeddings. Through extensive experiments, we demonstrate that the learned dynamic representations effectively rank stocks by returns across multiple holding periods, outperforming the related baselines on relevant metrics. While we developed specific datasets on financial markets to showcase our results, the methods are general and can be applied to different fields.

1 Introduction

Developing learning algorithms for event forecasting is crucial for creating AI systems with diverse applications, such as in financial markets and healthcare. Recent progress in these areas has been notable, and methods involving a combination of powerful sequence models, such as transformers, and elegant probabilistic models, such as temporal point processes, have made enormous progress (Mei and Eisner, 2017; Gracious and Dukkipati, 2025; Gracious et al., 2025). However, nearly all existing methods rely almost exclusively on historical patterns and fall short of adequately accounting for or integrating time-varying external factors that influence the outcomes.

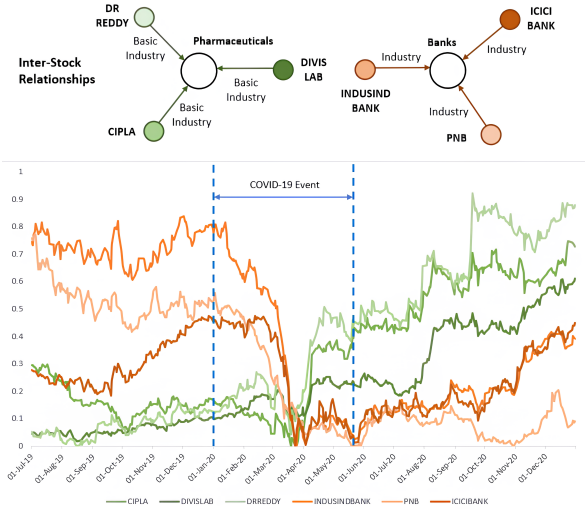

Figure 1: Interdependence of stock trends between stocks in the same class during COVID-19 (January–May 2020, NSE). Pharmaceutical stocks (CIPLA, DIVISLAB, DRREDDY) trended positively, whereas banking stocks (PNB, INDUSINDBANK, ICICIBANK) declined.

In real-world forecasting scenarios, future events are rarely determined solely by past observations. Instead, they are influenced by heterogeneous external factors that evolve continuously over time and interact through complex relational structures. For instance, stock price variations are affected by a complex interplay of factors that extend beyond mere historical patterns of stock prices. These elements can be shaped by external factors, such as prevailing economic conditions, shifts in government policy, and incidents of conflict. Additionally, these factors can be dynamic, with some changing rapidly, others slowly, and others frequently and infrequently.

To study this problem, we introduce a learning mechanism **FLExI** (**F**orecasting by **L**earning **Ex**ternal **I**nfluence, a learning mechanism that cap-

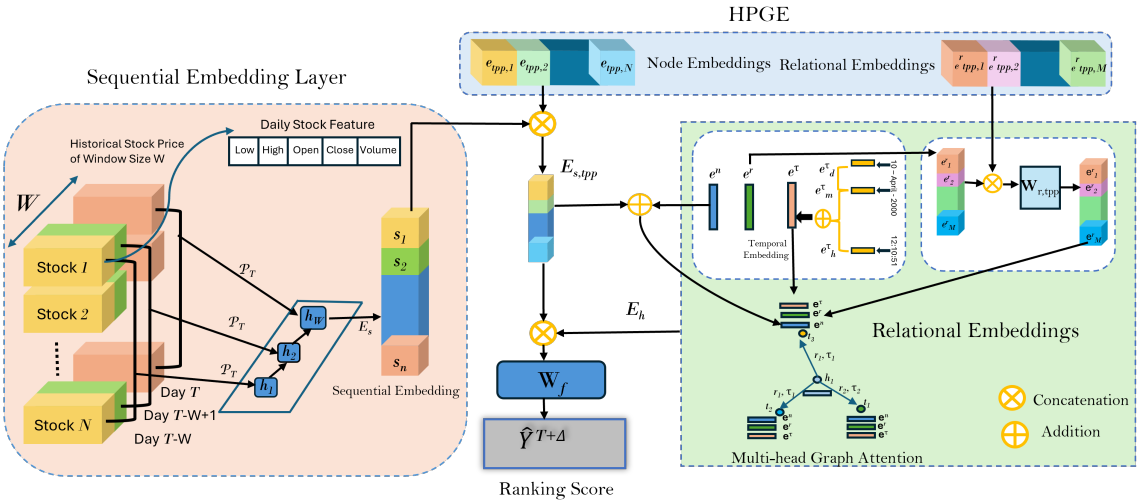

Figure 2: Proposed FLExI framework. Transformer Encoder learns sequential representations from historical stock price sequences. Hawkes Process models dynamic temporal relational events to learn temporal process embeddings. Relational Encoder further captures structural and temporal dependencies using HEAT-based graph attention with time-aware relation embeddings. The sequential, temporal-process, and relational embeddings are fused and passed to the prediction layer to estimate future stock rankings over the forecasting horizon from T to $T + \Delta$.

tures external influences using temporal Knowledge Graphs (KG) and learns historical patterns using temporal point processes with explicit start and end timestamps that define event validity intervals. Considering that such specific temporal KGs are not available in the literature, we develop temporal KGs for financial markets for the machine learning community and propose representation learning techniques to establish benchmark results. The temporal KGs we constructed integrated diverse components, including first- and second-order stock relationships, corporate events, macroeconomic indicators, events derived from financial statements, financial ratios disclosed during earnings calls, and analyst sentiments, into a unified temporal relational framework with time-aware validity intervals. The capacity to incorporate entities of varying categories, such as Person, Stock Assets, Products, and Alliance, results in an intrinsically heterogeneous knowledge graph. We refer to these KGs as STOCkg aims to encapsulate all pertinent information related to stock assets in a given stock market.

In summary, our main contributions are as follows.

(1) We formulate a time-series forecasting algorithm that incorporates knowledge of external events influencing the evolution of the series.

(2) We construct **STOCKg**, the first large-scale temporal heterogeneous knowledge graph datasets for forecasting,integrating relevant macroeconomic and financial information related to the NSE and NASDAQ stock markets.

(3) We propose **FLExI**, a learning mechanism that combines transformer-based sequential modeling, heterogeneous graph attention, and Hawkes-process based temporal relational reasoning to jointly learn historical dynamics and external event influences.

(4) We empirically demonstrate through extensive experiments that FLExI consistently outperforms related baselines in ranking stocks across three different holding periods. Additionally, we conduct ablation studies and further experiments to analyze the effectiveness of the different components of our model.

2 Related work

Forecasting has traditionally used time-series models that predict future values solely from historical numerical data, assuming that the target variable evolves as a function of its past. Deep learning techniques using models such as RNNs (Hewamalage et al., 2021), LSTMs (Elsworth and Güttel, 2020; Kong et al., 2025), and transformer-based architectures (Zhou et al., 2021; Liu et al., 2024) enable the capture of long-range dependencies and complex nonlinear patterns that classical models struggle to represent. However, these approaches often assume that all relevant signals are contained within the target series and a limited set of covariates, overlooking the broader context in which the data are generated. This limits their ability to handle situ-

ations in which external events (e.g., pandemics, geopolitical shifts) or inter-entity interactions (e.g., firm-to-firm relationships) cause abrupt or correlated changes in the data. While recent work has begun to incorporate covariates, there remains a gap in systematically modeling rich, temporally dynamic external influences.

To address this gap, recent studies have incorporated unstructured data into forecasting models. These include news articles (Wang et al., 2024a), social media posts (Karlemstrand and Leckström, 2021), financial reports (Zheng et al., 2019), and product reviews (Liu et al., 2023), which offer valuable contextual signals that complement numerical time series. Text-Guided Time Series Forecasting (TGTSF)(Xu et al., 2024) uses textual cues, such as channel metadata or news content, to guide predictions. Con (Chattopadhyay et al., 2025) combines temporal and textual modalities using cross-attention to achieve better results. In a multimodal direction, MM-Forecast (Li et al., 2024) integrates images and other modalities for temporal event prediction. These approaches typically treat unstructured signals as flat text, failing to capture the relationships between entities or how they evolve over time. This limits their ability to reason about structured causal chains or model the temporal unfolding of influences.

Incorporating structural knowledge into learning problems has been studied for natural language problems (Annervaz et al., 2018). Recently, this technique has been applied to forecasting problems. This includes time-varying categorical variables, continuous external signals and relational knowledge captured using graphs and knowledge graphs. For example, TimeXer (Wang et al., 2024b) enhances transformers with patch-wise and variate-wise attention to handle structured exogenous inputs in multivariate time-series data. ChronosX (Arango et al., 2025) fuses historical and future auxiliary inputs using dedicated networks to adapt pretrained models without retraining core models. Beyond tabular features, recent studies have constructed graphs (Cini et al., 2023) and knowledge graphs (Jain et al., 2023) to model the prior knowledge and event dynamics that influence the targets. This structured context supports better generalization across domains and improves interpretability by linking forecasts to causally relevant external signals. However, these efforts are often static or limited to coarse-grained relationships and do not explicitly model evolving temporal interactions between entities.

Our work addresses this gap by constructing temporal knowledge graphs that encode fine-grained, time-indexed external events and using point-process modeling to dynamically represent their influence on forecasting targets.

3 Problem formulation

Consider a forecasting problem on a multivariate time series $\mathcal{P} = (\mathcal{P}_i^\tau : i = 1, \ldots, N)_\tau$, with N values at each time τ in some discrete sequence of time points $\tau_1, \tau_2 \ldots$. At time T, using the history of the series till that time, we wish to forecast $Y^{T+\Delta} = \mathbf{Y}^\Delta \left((\mathcal{P}_i^\tau)_{1 \le i \le N}^{\tau_1 < \tau \le T} \right)$, where $Y^{T+\Delta}$ is some property of the N values $\left(\mathcal{P}_i^{T+\Delta} \right)_{1 \le i \le N}$ of the time series that occur at time $T + \Delta$. For the stock forecasting problem, $\mathcal{P}_i^\tau$ denotes the price of stock asset i at time τ, and $Y^{T+\Delta}$ is the ranking of the stock prices of all assets at time $T + \Delta$.

As mentioned earlier, we model the temporal knowledge of external events that can influence this series as a temporal KG. For this, we consider the elements $(\mathcal{P}_i^\tau)_\tau$ of this time series to occur on a set of N nodes of a temporal KG G. While $(\mathcal{P}_i^\tau)$ represents a multivariate series that evolves continuously at every time step, the knowledge graph G has a static component that contains information about all the fixed relations between entities in the graph, and a dynamic component that captures the occurrence of events across different time scales. Each event is represented as an edge between the nodes of the graph, valid for a certain period based on the nature of the event.

To this end, we extended the standard static knowledge graph, consisting of triples of the form (h, r, t), to a temporal knowledge graph by adding a time dimension for the relations, resulting in quintuples of the form $(h, r, t, [\tau_s, \tau_e])$, where h and t are the head and tail entities, respectively, r is the type of relationship between them, and $[\tau_s, \tau_e]$ is the time interval in which the triple remains valid. If the triple is valid for only the time step τ, then it is represented as a quadruple $(h, r, t, [\tau])$.

Thus, the temporal knowledge graph G is a union of subgraphs $G = G_{\tau_0} \cup \bigcup_{j=1}^{T} G_{\tau_j}$, where G_{τ_0} is the static subgraph and G_{τ_j} is the set of relations that are valid at time τ_j. Therefore, the input X^T to our model at time T is of the form $X^T = \left(\mathcal{P}_{1 \le i \le N}^{T-W < \tau \le T}, G_{\tau_0}, G_T \right)$, where $\mathcal{P}_i^{T-W \le \tau \le T}$ is the window of historical stock price data for W previous days of stock i at day T, G_{τ_0} is the non-

temporal positive subgraph of G, and G_{τ_j} is the temporal positive subgraph of G at time τ_j.

The objective is to learn embeddings $\mathcal{E}(X^T)$ that can be used to compute scores $\hat{Y}^{T+\Delta} = \mathbf{Y}^{\Delta}_{\mathbf{W}_f}(\mathcal{E}(X^T))$ to rank the best-performing stocks at a future time $T+\Delta$. The embedding function $\mathcal{E}$ along with the final prediction layer parameters $\mathbf{W}_f$ are learned using an objective that is a combination of knowledge graph embedding consistency loss and ranking losses that align the output with the ground-truth values $\mathcal{P}_i^{T+\Delta}$. The components of this loss function, as well as the detailed equations, are described in Section 4.5 and Appendix D. The learned model is used to predict the top 1 and 5 best-performing stocks Δ time steps in the future. The quality of these predictions is quantified using various relevant metrics, such as return ratios, Sharpe ratios, and accuracy, as described in Appendix E.

4 Proposed Learning Mechanism

The input to the model is X^T at time T consists of $X^T = \{\mathcal{P}_i^{\tau\in(T-W,T]}, G_{\tau_0}, G_T\}$, where $\mathcal{P}_i^{\tau\in(T-W,T]}$ is the window of historical stock price data for W previous days of stock i at day T, G_{τ_0} is the static subgraph of STOCKg, G_{τ_T} is the temporal positive subgraph of STOCKg at time T, and N is the total number of stocks in the dataset.

The objective is to learn an encoder $\mathcal{E}(X^T)$ to obtain output embeddings that can be used to predict the ranking of various stocks at future time-steps. We construct this encoder using multiple components: a Sequential Encoder $\mathcal{E}_s(\mathcal{P}_i^{\tau\in(T-W,T]}) = E_s$ of the stock prices, a set of temporal process embeddings e_{tpp} that are learned to optimally model the events as a heterogeneous Hawkes process, and a Relational Encoder $\mathcal{E}_r(G_{\tau_0}, G_T, E_s, e_{\text{tpp}}) = E_h$. These are then passed through a prediction layer $\mathbf{Y}^{\Delta}_{\mathbf{W}_f}$ to obtain a vector of ranking scores for the stocks at time step $T+\Delta$.

4.1 Sequential Embedding Layer

Given the strong temporal dynamics of stock markets, it is intuitive to regard a stock's historical performance as the most influential factor in predicting its future trend. Therefore, we first apply a sequential embedding layer to capture the sequential dependencies in the historical price data and technical indicators. Owing to the significant advancements and advantages demonstrated by transformers in processing sequential data, we chose to use a transformer model to learn the sequential embeddings. Consequently, we feed the historical time series data $\mathcal{P}_i^T$ of stock i at time step T into the Transformer model, which generates the sequential embedding s_i^T for each stock as $s_i^T = \text{Transformer}\left(\mathcal{P}_i^T\right)$. The Transformer model processes the entire historical sequence simultaneously, effectively capturing long-range dependencies. We employed the encoder-only architecture of the Transformer (Vaswani et al., 2017) to obtain sequential embeddings.

4.2 Temporal Process Embeddings

We then consider relations on the dynamic knowledge graph as events of a heterogeneous Hawkes process on the graph and learn node and relation embeddings using the Heterogeneous Hawkes Process for Dynamic Heterogeneous Graph Embedding (HPGE) algorithm (Ji et al., 2021). For event (h, r, t, τ), we consider the intensity function defined as

$$\begin{aligned}\tilde{\lambda}(h,r,t,\tau) =& \mu_r(h,t) \\ &+ \gamma_1 \sum_{r',t',\tau'\in\mathcal{N}_{<\tau}(h)} m_1(h,r',t',\tau') \\ &+ \gamma_2 \sum_{r',h',\tau'\in\mathcal{N}_{<\tau}(t)} m_2(t,r',h',\tau'),\end{aligned}$$

where μ_r is the relation type r-specific base intensity function between h and t, and m_1 and m_2 are the mutual intensities between the entities, summing over all relation types r' in the past neighborhoods of the nodes h and t with respective neighbors t' and h'. Node embeddings $e_{\text{tpp},i}$ and relation embeddings $e^r_{\text{tpp},j}$ are initialized and the base intensity $\mu_r(h,t)$ is taken as

$$\left\|\sigma\left(\mathbf{W}_e \cdot e_{\text{tpp},h}\right) + e^r_{\text{tpp},r} - \sigma\left(\mathbf{W}_e \cdot e_{\text{tpp},t}\right)\right\|^2,$$

where $\mathbf{W}_e$ is a dense layer and the activation function σ is LeakyReLU. The embeddings and weights are optimized by maximizing the intensity function for the positive samples and minimizing the intensity function for the negative samples. The loss function and mutual information terms are defined for the HPGE algorithm (Ji et al., 2021). The node embeddings obtained from the sequential layer and temporal process model are concatenated to create new node embeddings $E_{s,\text{tpp}} = \left(s_i \| e_{\text{tpp},i}\right)_i$.

4.3 Relational Embeddings

To obtain the relational embedding, each entity i in the knowledge graph is assigned a learnable em-

bedding e_i^n randomly initialized from $\mathcal{N}(0,1)$. The node embeddings $E_{s,\text{tpp}}$ are added to the embeddings of the nodes corresponding to the selected stock assets. Together, this creates an entity feature matrix $\mathbf{e}^n = (e_i^n)_{i=1,\ldots,N}$. Similarly, features for the set of relation types are defined as embeddings $\left(e_j^r\right)_{j=1,\ldots,M}$, where M is the number of relation types, and are initialized from $\mathcal{N}(0,1)$. Each of these is concatenated with the corresponding relation embedding $e_{\text{tpp},j}^r$ obtained from the temporal process embedding method and projected to its original dimension to obtain the relational embeddings $\mathbf{e}^r = \mathbf{W}_{r,\text{tpp}} \cdot \left[e_r \| e_{\text{tpp}}^r\right]$.

Additionally, embeddings are defined for the month e_m^τ, day e_d^τ and hour e_h^τ, with weights initialized from $\mathcal{N}(0,1)$. These are then summed to obtain the temporal embedding for the start timestamp τ_s, which is denoted by $\mathbf{e}^\tau$. These node embeddings $\mathbf{e}^n$, along with edge attributes $\mathbf{e}^e = \mathbf{e}^r + \mathbf{e}^\tau$ are passed through two layers of Heterogeneous Edge-featured Graph Attention (HEAT) (Mo et al., 2021) and two linear layers to obtain the final combined relational embedding E_h for all stocks.

4.4 Prediction Layer

The obtained embeddings are passed through a prediction layer to obtain a vector of ranking scores

$$\hat{Y}^{T+\Delta} = \frac{\exp(\mathbf{W}_f(E_{s,\text{tpp}} \| E_h) + b_f)}{\sum_{n=1}^{N} \exp(\mathbf{W}_f(E_{s,\text{tpp}} \| E_h) + b_f)} \quad (1)$$

that acts as the predicted rank of the given stocks in a non-increasing order of their potential return on investment from time T to $T+\Delta$.

4.5 Loss Function

The loss function is defined as the sum of the following four terms:

$$\begin{aligned}\mathcal{L} &= \alpha_1 \mathcal{L}_1(h,r,t,\tau) + \alpha_2 \mathcal{L}_2(\{\hat{Y}\},\{Y\}) \\ &\quad + \alpha_3 \mathcal{L}_3(\{\hat{Y}\},\{D\}) + \alpha_4 \mathcal{L}_4(\{\hat{Y}\},\{R\}). \quad (2)\end{aligned}$$

Here $\mathcal{L}_1(h,r,t,\tau)$ is the Knowledge Graph Embedding loss, which ensures consistency of the learned node and temporal embeddings by minimizing $\mathcal{L}_1(h,r,t,\tau) = \|e_h^n + e^r + e^\tau - e_t^n\|$, $\mathcal{L}_2$ is the listwise loss function ApproxNDCG (Bruch et al., 2019) comparing the actual and predicted ranks, $\mathcal{L}_3$ is a binary cross-entropy loss between the prediction and an indicator variable whether the stock price moved in the upward direction, and finally, $\mathcal{L}_4$ is another cross-entropy loss between the predicted ranking score and an indicator whether the stock is in the top 5 with respect to the IRR metric. The detailed expressions for each of the above losses are provided in Appendix D.

5 Experiments

Datasets Three datasets are created, containing a subset of stocks from NASDAQ and NIFTY, with datasets NASDAQ100 and S&P500 from the NASDAQ market and NIFTY500 from NSE, using Yahoo Finance. Each dataset consists of historical price movements and a constructed knowledge graph valid for each trading day. The historical price movements include features Close, Open, Low, and High prices, and trading volume for each trading day from 30 January, 2012, to 20 October, 2022. The statistics are presented in Table 2.

The model is given as input a context window of stock data X_T at time T, which consists of the historical stock price $\mathcal{P}^{\tau \in (T-W,T]}$ and the positive subgraph G_T from the knowledge graph at time point T. To avoid data leakage and minimize distribution drift, the values in the historical price window are normalized by dividing them by the last value in the previous window.

To learn embeddings using the Hawkes process, we transformed the data tuples into the format (head, head_type, tail, tail_type, relation, timestamp), with a separate tuple generated for each month within the range of validity of the relation.

Training Setup The value of Δ, the number of days in the future to be considered for prediction, is set as 1, 5, or 20, corresponding to daily, weekly, or monthly holding periods, respectively. We divided the dataset into 24 smaller overlapping datasets to train and evaluate the model separately in different phases by shifting a 600-day sliding window over the entire trading period, starting from a 400-day window. Each phase is divided into a training period of 450 trading days, a validation period of 50 days, and a testing period of 100 days, with the first few phases starting with a 250-day training period. The calculated metrics were obtained by averaging the results of all the test sets across the 24 phases. The metrics reported are described in Appendix E.

Baselines First, we consider a simple transformer (TRANSF) (Vaswani et al., 2017) acting on historical price information $\mathcal{P}$. Next, we consider the Spatio-Temporal Hypergraph Convolution Network (STHGCN) (Sawhney et al., 2020), which uses a gated temporal convolution over hyper-

Δ	Model	NASDAQ100					S&P100					NIFTY500				
		IRR	AIRR	SR	NDCG	ACC	IRR	AIRR	SR	NDCG	ACC	IRR	AIRR	SR	NDCG	ACC
1	TRANSF	0.080	22.498	0.090	0.315	5.556	0.065	17.900	0.030	0.208	1.103	0.098	28.173	-1.020	0.222	1.372
	STHGCN	0.082	22.971	0.117	0.309	4.931	0.052	13.969	-0.020	0.204	0.800	0.069	19.114	0.060	**0.310**	**4.740**
	STHAN-SR	0.069	19.114	0.057	0.308	4.744	0.059	16.013	0.007	0.205	0.914	0.112	32.582	-0.720	0.220	1.433
	GCNKG	0.100	28.741	**0.135**	0.312	5.136	0.070	19.240	0.020	0.207	1.072	0.109	31.538	-1.080	0.218	1.233
	CI-STHPAN	0.105	30.120	0.118	0.355	9.420	0.075	20.410	0.038	0.218	1.680	0.140	42.600	-0.420	0.232	2.980
	FLExI	**0.110**	**31.920**	0.100	**0.397**	**13.150**	**0.080**	**22.320**	**0.059**	**0.228**	**2.300**	**0.170**	**53.420**	**0.050**	0.246	4.630
5	TRANSF	0.422	23.675	0.551	0.316	5.736	0.276	14.882	0.420	0.206	0.942	0.501	28.633	0.071	0.220	1.317
	STHGCN	0.484	27.559	**0.623**	0.318	6.114	0.328	17.961	0.482	0.208	1.242	0.502	28.737	0.049	0.220	1.411
	STHAN-SR	0.489	27.917	0.611	0.318	6.197	**0.363**	**20.028**	**0.517**	0.209	1.217	0.489	27.867	0.031	0.217	1.322
	GCNKG	0.417	23.333	0.545	0.316	5.744	0.308	16.801	0.480	0.206	1.025	0.516	29.757	0.065	0.219	1.356
	CI-STHPAN	0.520	30.180	0.570	0.355	9.240	0.240	12.900	0.330	0.214	1.860	0.585	34.120	0.320	0.223	1.740
	FLExI	**0.620**	**36.540**	0.597	**0.397**	**13.330**	0.180	9.480	0.200	**0.222**	**2.900**	**0.650**	**38.610**	**0.652**	**0.227**	**2.130**
20	TRANSF	1.711	23.829	1.382	0.315	5.950	1.126	15.154	1.142	0.205	1.239	2.310	33.351	1.225	0.221	1.442
	STHGCN	1.718	23.935	1.371	0.314	6.200	1.490	20.460	**1.340**	0.209	1.619	2.358	34.141	1.280	0.221	1.586
	STHAN-SR	**2.013**	28.340	**1.540**	0.316	6.819	1.376	18.846	1.165	0.211	1.567	2.388	34.657	1.330	0.220	1.592
	GCNKG	1.404	19.654	1.126	0.319	6.372	1.168	15.794	1.085	0.206	1.253	**2.470**	**36.06**	1.294	0.220	1.575
	CI-STHPAN	1.720	24.180	0.920	0.355	9.640	1.360	18.740	0.980	0.220	2.420	2.020	28.900	1.560	0.228	2.050
	FLExI	2.000	**28.548**	0.718	**0.391**	**12.990**	**1.560**	**21.530**	0.907	**0.235**	**3.530**	1.550	21.380	**1.880**	**0.236**	**2.670**

Table 1: Top 5 Performance comparison of FLExI against baseline models, reported as averages over all phases of the corresponding datasets NASDAQ100, S&P500, and NIFTY500. Best results for each holding period are shown in **bold**.

-	NASDAQ100	S&P500	NIFTY500
#Assets	83	442	336
#Days	2800	2800	2800

Table 2: Statistics of the three datasets - NASDAQ100, S&P500 and NIFTY500.

graphs constructed from stock asset sector classifications, and the Spatio-Temporal Hypergraph Attention Network (STHAN-SR) (Sawhney et al., 2021), which models stock dependencies via hypergraphs to capture higher-order relations, with a hyperedge indicating the presence of a first- or second-order relation. The channel-independent spatiotemporal hypergraph pretrained attention network (CI-STHPAN) (Xia et al., 2024) is a two-stage stock selection model that pretrains a transformer on stock time-series data and incorporates a hypergraph attention network to model inter-stock relations. It constructs channel-independent dynamic hypergraphs using DTW-based similarity and is fine-tuned with a ranking-based objective for stock-selection. Finally, we consider GCNKG, a two-layer GCN that considers a homogeneous directed graph version of the current time snapshot of the constructed knowledge graphs to obtain node embeddings. These are then concatenated with the sequential embeddings and used to calculate prediction scores. The loss function used for training the baselines is a linear combination of the pair-wise ranking loss (6) and direction loss (9) defined in Appendix D. We do not include LLMs as baselines because they cannot effectively model large temporal knowledge graphs owing to context length limitations and the costs associated with long-context inputs. Moreover, they lack explicit mechanisms for structured, time-aware relational reasoning required for the task.

Overall Performance. Tables 1 and 8 (Appendix) compare FLExI with the five baseline methods across all holding periods ($\Delta \in \{1, 5, 20\}$). FLExI consistently achieves the strongest or near-strongest ranking quality (NDCG/ACC) and investment returns (IRR/AIRR) across both the NASDAQ and NSE markets. Purely sequential models, such as TRANSF, perform competitively when trends are dominated by historical momentum but degrade under evolving external events. Graph-based baselines (STHGCN, STHAN-SR, and GCNKG) capture cross-asset dependencies but rely on static or weakly time-aware relational structures. In contrast, FLExI jointly models temporal external events, temporal KG relations, and price dynamics, resulting in more robust and stable forecasting performance across short-, medium-, and long-term horizons.

6 Ablations and Additional Experiments

Impact of Sliding window. Our method integrates historical stock prices with temporal events and relationships to forecast future trends. To analyze the effect of the historical context length, we varied the sliding window size W used in the sequential embedding layer. Moderate window sizes generally yield stronger results, whereas very large values

of W lead to a gradual decline in the return ratio. This behavior suggests that excessively long price histories may introduce outdated or noisy information that dilutes relevant temporal signals and negatively affects performance.

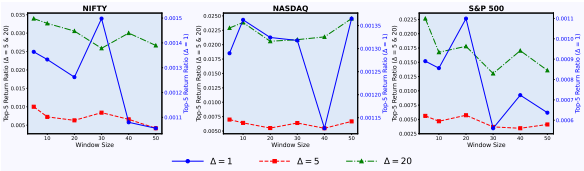

Figure 3: Effect of sliding window size W on Top-5 return ratio across different holding periods Δ.

Impact of Holding Period (Δ) and Knowledge Graph Integration. Figure 4 shows that the return ratios improve with increasing holding periods, indicating stable performance across different horizons. In addition, models with knowledge graph integration consistently outperform those without KG in both return ratio and accuracy across all datasets. The growing performance gap for larger Δ further demonstrates the importance of external relational information for robust long-term forecasting.

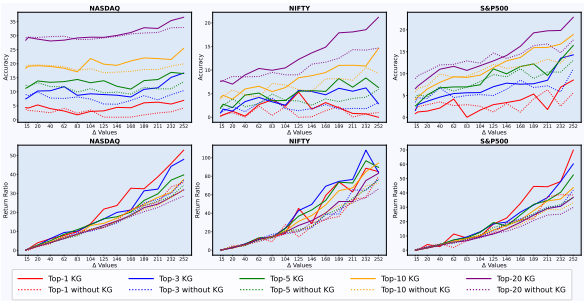

Figure 4: Analysis of accuracy and return ratio across varying holding periods Δ for each dataset, comparing models with and without knowledge graph integration.

We wish to study the contributions of the individual components of FLExI through multiple ablation variants. In **WOTPP**, we remove the temporal process embeddings and retain only the sequential and relational embeddings. In **LSTM***, the transformer-based sequential encoder in the WOTPP is replaced with an LSTM. The **WOSEQ** variant removes the sequential embedding updates used in the relational encoder, together with the temporal process embeddings. Finally, **WOHK** removes both temporal process embeddings and heterogeneous feature transformation. The detailed results in Table 14 show that each component consistently contributes to improving the forecasting performance.

We further evaluate the robustness of FLExI under longer training, validation, and testing horizons, where market drift and evolving external events introduce additional forecasting challenges. To analyze this effect, we conducted experiments with larger temporal splits for each phase. The results in Table 15 (Appendix) show that performance gradually decreases under longer horizons due to increasing distribution drift; however, FLExI consistently maintains positive Sharpe ratios, indicating performance superior to risk-free returns.
We include an additional analysis of metric variance across phases and the effect of trading the stock at best and worst case times during the day in Appendix F Table 9.

7 Counterfactual Interpretability: How External Relations Influence Forecasting

A key advantage of FLExI is that its predictions are auditable; forecasting decisions can be traced back to temporally grounded and typed knowledge graph (KG) relations. To evaluate whether the learned relational attention is faithful to the model's forecasting behavior, we perform counterfactual relational interventions by removing selected relation types while keeping the historical-price trajectories fixed. If the learned attention is meaningful, removing highly attended relations should induce a larger degradation in the forecasting performance.

7.1 Relation Faithfulness

We systematically evaluated the importance of external relations by removing one relation type at a time from the temporal KG and measuring the degradation in forecasting performance for counterfactual analysis. Specifically, for each relation type r, we construct a counterfactual KG $\mathcal{G}^{-r}$ by removing all instances of relation r while preserving sequential historical observations. We then measured the changes in the Investment Return Ratio (IRR), NDCG, and ranking accuracy relative to the original model trained on the full KG.
Table 3 shows that removing highly attended relations consistently causes larger forecasting degradation. In particular, event-driven relations such as `DECLARES_DIVIDEND` and financial trend indicators such as `INCREASE_YoY` produce the largest drop in return prediction quality, indicating that the learned attention distributions are aligned with the model's forecasting results.

Relation Type	Attn	ΔIRR ↓	ΔNDCG ↓
DECLARES_DIVIDEND	0.31	0.08	0.03
INCREASE_YoY	0.23	0.06	0.02
INDUSTRY	0.22	0.04	0.01
SUBSIDIARY	0.18	0.03	0.01
CEO	0.07	0.01	0.002

Table 3: Faithfulness analysis under counterfactual relation removal. Higher-attention relations induce larger forecasting degradation when removed.

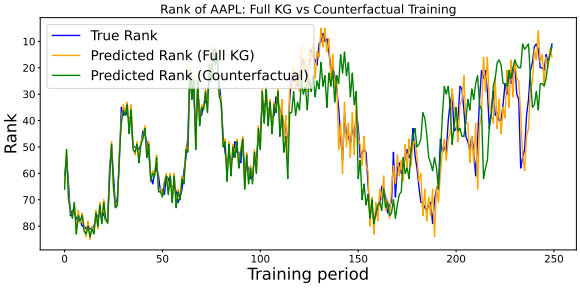

Figure 5: Rank trajectory of AAPL under full KG and counterfactual KG training. Performance degrades after removing highly informative external relations.

7.2 NASDAQ Case Study: *Apple Inc. (AAPL)*

We further illustrate the behavior of FLExI through a detailed case study of Apple Inc. (AAPL) within the NASDAQ STOCKg graph. AAPL participates in heterogeneous, time-aware relations that capture both structural dependencies and event-driven external influences. Under the original full KG, FLExI consistently ranks AAPL among the Top-1 and Top-5 stocks across the daily, weekly, and monthly holding horizons. To study influential external relations, we removed two key relation types during training: DECLARES_DIVIDEND, representing short-term corporate events, and INCREASE_YoY, representing longer-term financial growth trends. After removing these relations, the forecasting performance degrades consistently across multiple holding periods. For longer horizons ($\Delta = 20$), the IRR decreases from **0.73** for the full KG to **0.67** for the counterfactual KG. Figure 5 shows the ranking trajectories. Before the intervention, both models behaved similarly; however, after relation removal, the counterfactual model progressively diverged and produced inaccurate rankings.

Attention-Based Explanation. Figure 6 visualizes the relation-level attention learned by the HEAT attention layers. Under the full KG, attention is concentrated on dividend-related corporate actions and financial growth indicators near event

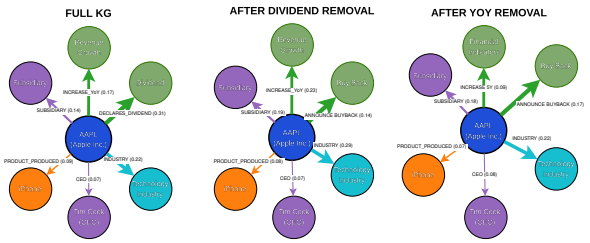

Figure 6: Top attended relations for AAPL before and after counterfactual intervention. Attention shifts from informative event-driven relations after relation removal.

timestamps. After removing DECLARES_DIVIDEND and INCREASE_YoY, attention shifts towards other relations that are comparatively less informative for return forecasting.

This attention shift aligns with the observed degradation in return ratios, suggesting that FLExI relies on temporally localized external relational events for forecasting rather than treating contextual information as unstructured, auxiliary features.

Discussion. These results suggest that FLExI's forecasting improvements arise not only from a stronger representational capacity but also from integrating temporally grounded external knowledge. Different relationship types contribute at different temporal scales, whereas counterfactual interventions provide evidence regarding which external relationships most strongly influence forecasting decisions.

A detailed analysis of the time complexity of the model is presented in Appendix B.

8 Conclusion

This study, for the first time, addresses forecasting problems by incorporating dynamic external influences, which is a departure from the predominant focus on historical data in existing techniques. We constructed temporal KGs called STOCKg to capture the temporal and heterogeneous nature of event data. Our main contributions are the development of learning mechanisms using heterogeneous graph attention networks, temporal process models, and time-aware translation-based knowledge graph embedding techniques to address forecasting with external influences. Although we demonstrated our results on the stock market data, the proposed learning algorithms are sufficiently versatile to be applied to forecasting problems in various fields.

9 Limitations

Despite the strong empirical performance of FLExI, there are a few limitations.

1. Proposed framework depends heavily on the quality and completeness of the constructed temporal knowledge graphs. Missing, noisy, delayed, or incorrectly extracted events from financial news and external sources can propagate errors into learned relational representations and forecasting outputs.

2. Knowledge graph construction pipeline relies partly on rule-based event extraction and predefined relation schemas, which may fail to capture nuanced semantic or causal relationships present in unstructured financial text.

3. Although FLExI models heterogeneous temporal interactions effectively, the computational cost increases with the number of temporal events and relation types, making scalability challenging for extremely large or high-frequency financial markets.

4. Model is evaluated primarily on stock market forecasting datasets from NASDAQ, S&P500, and NIFTY500; therefore, its generalization to other domains with substantially different temporal dynamics remains only partially validated despite preliminary experiments on recommendation and healthcare tasks. In addition, counterfactual interpretability analysis is relation-level and does not yet provide fine-grained causal guarantees regarding individual events or entity interactions.

5. Financial markets are inherently influenced by unpredictable macroeconomic shocks, policy changes, and human behavioral factors that may not be fully represented within the temporal knowledge graph, limiting forecasting robustness under extreme market conditions.

10 LLM Disclosure

Large Language Models (LLMs) were used only for language refinement, proofreading, and improving the readability of the manuscript. All technical ideas, methodology design, experiments, analysis, and conclusions were developed and verified by the authors. The authors take full responsibility for the content of this work.

References

K. M. Annervaz, S. Chowdhury, and A. Dukkipati. 2018. Learning beyond datasets: Knowledge graph augmented neural networks for natural language processing. In *Proceedings of NAACL HLT*, pages 313–322.

Sebastian Pineda Arango, Pedro Mercado, Shubham Kapoor, Abdul Fatir Ansari, Lorenzo Stella, Huibin Shen, Hugo Senetaire, Caner Turkmen, Oleksandr Shchur, Danielle C. Maddix, Michael Bohlke-Schneider, Yuyang Wang, and Syama Sundar Rangapuram. 2025. Chronosx: Adapting pretrained time series models with exogenous variables. *Preprint*, arXiv:2503.12107.

Sebastian Bruch, Masrour Zoghi, Michael Bendersky, and Marc Najork. 2019. Revisiting approximate metric optimization in the age of deep neural networks. In *Proceedings of the 42nd international ACM SIGIR conference on research and development in information retrieval*.

Sameep Chattopadhyay, Pulkit Paliwal, Sai Shankar Narasimhan, Shubhankar Agarwal, and Sandeep P. Chinchali. 2025. Context matters: Leveraging contextual features for time series forecasting. *Preprint*, arXiv:2410.12672.

Edward Choi, Mohammad Taha Bahadori, Jimeng Sun, Joshua Kulas, Andy Schuetz, and Walter Stewart. 2016. Retain: An interpretable predictive model for healthcare using reverse time attention mechanism. *Advances in neural information processing systems*, 29.

Andrea Cini, Ivan Marisca, Daniele Zambon, and Cesare Alippi. 2023. Graph deep learning for time series forecasting. *Preprint*, arXiv:2310.15978.

Steven Elsworth and Stefan Güttel. 2020. Time series forecasting using lstm networks: A symbolic approach. *Preprint*, arXiv:2003.05672.

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

Tony Gracious and Ambedkar Dukkipati. 2025. Dynamic representation learning with temporal point processes for higher-order interaction forecasting. In *Proceedings of 37th AAAI Conference on Artificial Intelligence (AAAI)*.

Tony Gracious, Arman Gupta, and Ambedkar Dukkipati. 2025. Neural temporal point processes for forecasting directional relations in evolving hypergraphs. In *Proceedings of 39th AAAI Conference on Artificial Intelligence (AAAI)*.

Hansika Hewamalage, Christoph Bergmeir, and Kasun Bandara. 2021. Recurrent neural networks for time series forecasting: Current status and future directions. *International Journal of Forecasting*, 37(1):388–427.

Samyak Jain, Manuel Burger, Gunnar Rätsch, and Rita Kuznetsova. 2023. Knowledge graph representations to enhance intensive care time-series predictions. *Preprint*, arXiv:2311.07180.

Yugang Ji, Tianrui Jia, Yuan Fang, and Chuan Shi. 2021. Dynamic heterogeneous graph embedding via heterogeneous hawkes process. In *Machine Learning and Knowledge Discovery in Databases, ECML PKDD 2021*. Springer.

Wang-Cheng Kang and Julian McAuley. 2018. Self-attentive sequential recommendation. In *2018 IEEE international conference on data mining (ICDM)*, pages 197–206. IEEE.

Roderick Karlemstrand and Ebba Leckström. 2021. Using twitter attribute information to predict stock prices. *Preprint*, arXiv:2105.01402.

Yaxuan Kong, Zepu Wang, Yuqi Nie, Tian Zhou, Stefan Zohren, Yuxuan Liang, Peng Sun, and Qingsong Wen. 2025. Unlocking the power of lstm for long term time series forecasting. *Preprint*, arXiv:2408.10006.

Haoxuan Li, Zhengmao Yang, Yunshan Ma, Yi Bin, Yang Yang, and Tat-Seng Chua. 2024. Mm-forecast: A multimodal approach to temporal event forecasting with large language models. In *Proceedings of the 32nd ACM International Conference on Multimedia*, MM '24. ACM.

Yong Liu, Tengge Hu, Haoran Zhang, Haixu Wu, Shiyu Wang, Lintao Ma, and Mingsheng Long. 2024. itransformer: Inverted transformers are effective for time series forecasting. *Preprint*, arXiv:2310.06625.

Zixuan Liu, Gaurush Hiranandani, Kun Qian, Edward W. Huang, Yi Xu, Belinda Zeng, Karthik Subbian, and Sheng Wang. 2023. Foreseer: Product aspect forecasting using temporal graph embedding. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, CIKM '23. ACM.

Hongyuan Mei and Jason M Eisner. 2017. The neural Hawkes process: A neurally self-modulating multivariate point process. *Advances in neural information processing systems*.

Xiaoyu Mo, Yang Xing, and Chen Lv. 2021. Heterogeneous edge-enhanced graph attention network for multi-agent trajectory prediction. *arXiv preprint arXiv:2106.07161*.

Laila Rasmy, Yang Xiang, Ziqian Xie, Cui Tao, and Degui Zhi. 2021. Med-bert: pretrained contextualized embeddings on large-scale structured electronic health records for disease prediction. *NPJ digital medicine*, 4(1):86.

Ramit Sawhney, Shivam Agarwal, Arnav Wadhwa, Tyler Derr, and Rajiv Ratn Shah. 2021. Stock selection via spatiotemporal hypergraph attention network: A learning to rank approach. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Ramit Sawhney, Shivam Agarwal, Arnav Wadhwa, and Rajiv Ratn Shah. 2020. Spatiotemporal hypergraph convolution network for stock movement forecasting. In *2020 IEEE International Conference on Data Mining (ICDM)*. IEEE.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Xiang Wang, Xiangnan He, Yixin Cao, Meng Liu, and Tat-Seng Chua. 2019. Kgat: Knowledge graph attention network for recommendation. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 950–958.

Xinlei Wang, Maike Feng, Jing Qiu, Jinjin Gu, and Junhua Zhao. 2024a. From news to forecast: Integrating event analysis in llm-based time series forecasting with reflection. *Preprint*, arXiv:2409.17515.

Yuxuan Wang, Haixu Wu, Jiaxiang Dong, Guo Qin, Haoran Zhang, Yong Liu, Yunzhong Qiu, Jianmin Wang, and Mingsheng Long. 2024b. Timexer: Empowering transformers for time series forecasting with exogenous variables. *Preprint*, arXiv:2402.19072.

Hongjie Xia, Huijie Ao, Long Li, Yu Liu, Sen Liu, Guangnan Ye, and Hongfeng Chai. 2024. Ci-sthpan: Pre-trained attention network for stock selection with channel-independent spatio-temporal hypergraph. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 9187–9195.

Zhijian Xu, Yuxuan Bian, Jianyuan Zhong, Xiangyu Wen, and Qiang Xu. 2024. Beyond trend and periodicity: Guiding time series forecasting with textual cues. *Preprint*, arXiv:2405.13522.

Shun Zheng, Wei Cao, Wei Xu, and Jiang Bian. 2019. Doc2EDAG: An end-to-end document-level framework for Chinese financial event extraction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. ACL.

Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. 2021. Informer: Beyond efficient transformer for long sequence time-series forecasting. *Preprint*, arXiv:2012.07436.

Forecasting by Learning Dynamic External Influence Appendix

The dataset and relevant codes for the experiments can be accessed at https://drive.google.com/file/d/1vY9UM_RCXIy22axQPr8JKFzs6C5difNO/view?usp=sharing (anonymized).

A Temporal Knowledge Graph Construction

The construction of STOCKg involved data acquired from various sources pertaining to the NASDAQ and NSE stock markets. Temporal KGs are constructed for stocks listed in NASDAQ100, S&P500, and NIFTY500, spanning a period of 20 years, commencing from January 01, 2003, and concluding on December 30, 2022.

General News News articles were acquired by scraping data from two websites, namely "Seeking Alpha" for NASDAQ and "Money Control" for NSE, using the Selenium library in Python[1]. A total of 220,645 news articles were collected for NASDAQ and 272,808 for the NSE. These articles comprise several components, including the headline, body text, publication time, associated stock asset, related sector, and additional taglines, and contain various types of information such as corporate events, earnings reports, and analyst and investor sentiments. A rule-based pattern-matching system was used to identify events mentioned in the articles using a predefined set of events, which were then added to the knowledge graph.

Financial Statements and Ratios The financial statements of companies listed on NASDAQ and NSE were extracted by scraping data from websites "Macrotrends" and "Top Stock Research" using Selenium[2]. This data include essential components for each stock, including the balance sheet, income statement, cash flow statement, and other financial ratios. Events are derived from financial statements to identify the annual and quarterly trends of these financial parameters. These trends were then incorporated into the knowledge graph as quintuple events, where the interval was defined by the beginning and end of the corresponding financial year or a quarter.

STOCKg	NASDAQ	NSE
# Triples	7,568	2,736
# Quadruples	52,083	44,399
# Quintuples	310,916	251,551
# Entities	4,911	1,049
# Relations	370,567	298,686
# Entity types	12	14
# Relation types	56	53

Table 4: Statistics of the NASDAQ and NSE STOCKgs

Macro-Economic Indicators Macroeconomic indicators provide a snapshot of the general economic health of a country, including factors such as GDP growth, inflation rate, unemployment rate, trade balance, and fiscal deficit. This information is obtained from "Macrotrends" for both the US and India, in which the respective share markets NASDAQ and NSE are located[3]. Trends in indicators were identified as events with intervals defined by the beginning and end of the reported years.

Corporate Events Corporate events such as dividend announcements and stock splits are acquired from "Yahoo Finance" for the NASDAQ and NSE.

Inter-stock Relationship Inter-stock relationships can be classified into first- and second-order relationships. If there is a statement with company i as the subject and company j as the object, then they have a first-order relationship between them. Furthermore, companies i and j have a second-order relationship if they have statements that share the same object (Feng et al., 2019). For instance, "Apple" and "Foxconn" are in different stages in the production of the product "iPhone". This data was extracted from "Wikibase," where the knowledge from "Wikipedia" is stored in the form of triples. A preselected set of relations was identified, consisting of 6 types of first-order relations and 52 types of second-order relations, which were then filtered from Wikibase.

Sector Classification Each stock asset is classified into a four-level hierarchical system: Sector, Industry group, Industry, and Sub-Industry by Global

[1]The news articles are obtained by scraping the webpages https://seekingalpha.com/symbol/{stock_ticker}/news for NASDAQ and https://www.moneycontrol.com/company-article/{stock_name}/news/{stock_id}

[2]The financial statements are obtained from www.macrotrends.net and https://www.topstockresearch.com/ for NASDAQ and NSE, respectively.

[3]The macro-economic indicators are obtained from www.macrotrends.net.

Industry Classification Standard (GICS) for NASDAQ, and Macroeconomic Sector, Sector, Industry, and Basic Industry by NSE Indices for NSE. These relationships are added as triples with the asset as the head, class as the tail, and classification level as the relationship to the constructed knowledge graph.

B Computational Complexity

We analyzed the computational complexity of FLExI with respect to the major architectural components. Let N denote the number of assets, W the historical sequence length, d_s the sequential embedding dimension, d_r the relational embedding dimension, L the number of Transformer encoder layers, and H the number of attention heads. Further, let $|\mathcal{E}_T|$ denote the number of temporal KG events at time T, $|\mathcal{E}_0|$ the number of static KG edges, d_{tpp} the Hawkes-process embedding dimension, and K the number of negative samples per temporal event.

Sequential Encoder. For a standard full-attention transformer operating over a window of length W, the per-layer complexity is$\mathcal{O}(W^2 d_s + W d_s^2)$. Therefore, computing sequential representations for all N assets requires $\mathcal{O}\Big(NL(W^2 d_s + W d_s^2)\Big)$.

Temporal Hawkes Process Encoder. The HPGE module models continuous-time relational events in a temporal knowledge graph. Given $|\mathcal{E}_T|$ observed temporal events and K negative samples per event, the dominant training complexity becomes $\mathcal{O}\big(|\mathcal{E}_T| K d_{tpp}\big)$, excluding minor neighborhood aggregation overheads.

Relational Encoder. The heterogeneous relational encoder performs attention-based message passing over static and temporal graph edges. The complexity is approximately$\mathcal{O}\Big((|\mathcal{E}_0| + |\mathcal{E}_T|) d_r\Big)$, where the relation-specific projection constants are omitted for clarity.

Prediction Layer. The final ranking and prediction layer operates on the concatenated embedding representation of dimension $d = d_s + d_r + d_{tpp}$, resulting in a complexity of $\mathcal{O}(Nd)$.

Overall Complexity. Combining all components, the end-to-end training complexity for a single temporal snapshot is

$$\mathcal{O}\Big(NL(W^2 d_s + W d_s^2) + |\mathcal{E}_T| K d_{tpp} + (|\mathcal{E}_0| + |\mathcal{E}_T|) d_r + Nd\Big). \quad (3)$$

Comparison with Baselines. Table 5 summarizes the asymptotic complexity of FLExI and representative baseline methods.

Discussion. Compared with purely sequential models such as TRANSF, FLExI introduces additional relational and temporal event modeling costs to capture external dependencies between assets. Relative to hypergraph-based methods (e.g., STHGCN and STHAN-SR), the proposed framework avoids dense hyperedge construction and scales with the number of observed temporal events. Unlike CI-STHPAN, which incurs an additional $O(N^2 W)$ complexity due to pairwise DTW computations, FLExI scales with the sparse temporal event set $|\mathcal{E}_T|$, where $|\mathcal{E}_T| \ll N^2$ in real-world financial graphs.
Overall, the additional complexity introduced by FLExI grows approximately linearly with the number of temporal relational events, enabling improved scalability while jointly modeling the sequential dynamics, relational structure, and continuous-time event interactions.

C Generalizability Across Domains

Although FLExI is evaluated primarily on financial forecasting, the proposed framework is fundamentally domain-agnostic and generalizes naturally to forecasting problems involving

1. historical sequential observations, and
2. structured external relational knowledge.

The architecture is independent of domain-specific financial assumptions and learns temporal interactions between sequential dynamics and heterogeneous external influences through temporal knowledge graphs and Hawkes process-based event modeling.
Importantly, the core learning mechanism remains unchanged across the domains:

1. Transformer-based sequential representation learning,
2. Temporal event evolution modeling,
3. Heterogeneous relational attention, and

Method	Components	Time Complexity
TRANSF	Transformer (sequential)	$O\big(N(W^2d + Wd^2)\big)$
STHGCN / STHAN-SR	Transformer + Hypergraph Conv/Attention	$O\big(N(W^2d) + \lvert\mathcal{E}\rvert d\big)$
CI-STHPAN	Transformer + DTW + Hypergraph Attention	$O\big(N^2W + N(W^2d) + \lvert\mathcal{E}\rvert d\big)$
GCNKG	GCN (static graph)	$O\big(\lvert\mathcal{E}\rvert d + Nd^2\big)$
FLExI (Ours)	Transformer + TKG + Hawkes	$\mathbf{O\Big(N(W^2d + Wd^2) + \lvert\mathcal{E}_T\rvert d + \lvert\mathcal{E}_T\rvert Kd_{tpp}\Big)}$

Table 5: Asymptotic computational complexity comparison.

4. Temporal KG reasoning.

To generalize FLExI beyond financial forecasting, only lightweight task-specific modifications are required, including the following:

1. replacing domain-specific entity and relation types,
2. adapting the downstream prediction objective,
3. modifying the loss function according to the target task, and
4. replacing the original start–end timestamp interval formulation with a single-timestamp representation.

Temporal adaptation simplifies event modeling by associating each interaction with a single discrete timestamp while preserving the overall temporal reasoning framework and Hawkes process dynamics.

C.1 Sequential Recommendation (MovieLens + KG)

We adapted FLExI for sequential recommendation using MovieLens augmented with external movie knowledge graphs derived from Wikidata and DBpedia. User interaction histories form the sequential temporal component, whereas actors, directors, genres, and production relations define the heterogeneous knowledge graph.
The forecasting objective is future interaction prediction:

$$P(m_{t+1} \mid H_t, G_t), \tag{4}$$

where H_t denotes the user interaction history, and G_t denotes the temporal movie knowledge graph.
For this setting, the original ranking-based financial objective was replaced with the cross-entropy loss for next-item prediction. The evaluation metrics are correspondingly adapted to recommendation-specific measures, including NDCG@K, Recall@K, HitRate@K, and MRR.

For comparison, SASRec (Kang and McAuley, 2018), KGAT (Wang et al., 2019).

Model	NDCG@10	Recall@10	HR@10	MRR
SASRec	0.338	0.307	**0.587**	0.244
KGAT	**0.367**	0.329	0.579	0.266
FLExI	0.344	**0.347**	0.584	**0.283**

Table 6: Generalization performance on MovieLens+KG for sequential recommendation forecasting.

C.2 Healthcare Forecasting (MIMIC-IV + Medical KG)

We further generalized FLExI to healthcare forecasting using MIMIC-IV combined with medical ontologies such as UMLS and DrugBank. Patient trajectories consisting of diagnoses, medications, laboratory events, and procedures form temporal sequences, whereas medical ontologies provide structured relations between diseases, drugs, symptoms, and treatments.
The temporal Hawkes process module naturally captures the evolving clinical event dependencies and disease progression dynamics. The forecasting task is formulated as follows:

$$P(d_{t+1} \mid H_t, G_t), \tag{5}$$

where H_t represents the patient's clinical trajectory, and G_t denotes the evolving medical knowledge graph.
For healthcare forecasting, the prediction head was adapted for clinical event classification using binary cross-entropy loss. Evaluation was performed using healthcare-oriented metrics, including AUROC, AUPRC, F1-score, and Accuracy.
These experiments demonstrate that FLExI generalizes naturally across domains while preserving the same temporal relational learning mechanism, highlighting its effectiveness as a unified framework for

forecasting under dynamic, external, relational influence.

For Comparision RETAIN (Choi et al., 2016), Med-BERT (Rasmy et al., 2021)
We emphasize that the goal of these experiments is not to establish new state-of-the-art performance in recommendation or healthcare forecasting, but rather to demonstrate that the proposed temporal relational learning mechanism generalizes effectively across domains involving sequential observations and heterogeneous external knowledge.

Model	AUROC	AUPRC	F1	ACC
RETAIN	0.804	0.491	0.423	0.758
Med-BERT	0.836	**0.551**	**0.462**	0.789
FLExI	**0.841**	0.548	0.461	**0.804**

Table 7: Generalization performance of MIMIC-IV+Medical KG for healthcare forecasting.

This table corresponds to the main results discussed in the Overall Performance section.

D Loss functions

D.1 KGE loss function

Time-aware knowledge graph embeddings are learned using a translation-based model-type loss function, in which the element-wise addition of the head entity, relation embedding, and temporal embedding should be close to the tail embedding.

$$\mathcal{L}_1(h, r, t, \tau) = \|e_h^n + e^r + e^\tau - e_t^n\|. \quad (6)$$

D.2 Listwise ranking

Earlier studies (Feng et al., 2019; Sawhney et al., 2020, 2021) employed the following pairwise loss function as the objective.

$$\sum_{i=0}^{N}\sum_{j=0}^{N} \max\left\{0, -\left(\hat{Y}_i^{T+\Delta} - \hat{Y}_j^{T+\Delta}\right)\left(Y_i^{T+\Delta} - Y_j^{T+\Delta}\right)\right\}, \quad (7)$$

where $\hat{Y}_i^{T+\Delta}$ is the predicted ranking score and $Y_i^{T+\Delta}$ is the actual ranking score for stock i. This approach computes the loss for each pair of scores separately, which might be too loose as an approximation of the NDCG and MAP performance metrics. In contrast, the listwise loss function Approx-NDCG (Bruch et al., 2019) used in the listwise approach can better represent performance measures. The listwise ranking loss function is given by,

$$\mathcal{L}_2(\{\hat{Y}\}, \{Y\}) = -\frac{1}{\text{DCG}(\hat{y}, \hat{y})} \sum_i \frac{2^{\hat{y}_i} - 1}{\log_2(1 + \text{rank}_i)}, \quad (8)$$

$$\text{with } \text{rank}_i = 1 + \sum_{j \neq i} \frac{1}{1 + \exp\left(\frac{-(y_j - y_i)}{\text{temperature}}\right)}, \quad (9)$$

where y is the predicted ranking score and s is the actual score, which is the IRR for each stock from day T to $T + \Delta$.

D.3 Cross-Entropy losses

Let $D_i^{T+\Delta}$ be the binary movement direction from the closing price of stock i from day T to $T + \Delta$, in which 1 denotes rise, and 0 denotes fall, and it is expressed as $D_i^{T+\Delta} = 1(p_{T+\Delta}^i \geq p_T^i)$, where p_τ^i is the closing price of stock i on day τ. The movement ranking loss is defined as the binary cross-entropy between $\hat{Y}_i^{T+\Delta}$ and $D_i^{T+\Delta}$.

$$\mathcal{L}_3(\{\hat{Y}\}, \{D\}) = -[D \log(\hat{Y}) + (1 - D) \log(1 - \hat{Y})] \quad (10)$$

In addition, another cross-entropy loss is calculated between the predicted ranking score and stocks that appeared in the top K ranks. Such that $R_i^{T+\Delta} = 1(IRR_i \text{is in Top 5})$ is the truth label.

$$\mathcal{L}_4(\{\hat{Y}\}, \{R\}) = -[R \log(\hat{Y}) + (1 - R) \log(1 - \hat{Y})] \quad (11)$$

E Evaluation metrics

As illustrated in Figure 7, we divided the dataset into 24 smaller overlapping datasets to train and evaluate the model separately in different phases by shifting a 600-day sliding window over the entire trading period, starting with a 400-day window. Each phase is divided into a training period of 450 trading days, a validation period of 50 days, and a testing period of 100 days, with the first few phases starting with a 250-day training period. The calculated metrics are obtained by averaging the results of all test sets across the 24 phases.

E.1 Investment Return Ratio (IRR)

The Cumulative Investment Return Ratio (IRR) is defined as

$$\text{IRR} = \frac{\text{Final Value} - \text{Initial Value}}{\text{Initial Value}} \times 100,$$

Δ	Model	NASDAQ100					S&P500					NIFTY500				
		IRR	AIRR	SR	NDCG	ACC	IRR	AIRR	SR	NDCG	ACC	IRR	AIRR	SR	NDCG	ACC
1	TRANSF	0.088	25.027	0.874	0.214	1.014	0.062	16.868	0.748	0.143	0.181	0.108	33.477	0.484	0.152	0.389
	STHGCN	0.054	14.654	0.761	0.209	0.861	0.054	14.841	0.667	0.139	0.194	0.055	14.982	0.620	**0.210**	**0.820**
	STHAN-SR	0.055	14.982	0.617	0.209	0.819	0.056	15.258	0.589	0.140	0.153	0.087	24.658	-1.320	0.150	0.347
	GCNKG	0.090	**27.302**	**0.880**	0.213	1.056	**0.100**	**27.840**	**0.920**	0.142	0.292	0.128	38.140	-0.621	0.148	0.278
	CI-STHPAN	0.082	26.180	0.520	0.245	3.020	0.075	20.630	0.648	0.152	0.930	0.165	51.300	0.850	0.165	0.265
	FLExI	**0.090**	25.440	0.180	**0.291**	**5.000**	0.050	13.420	0.376	**0.163**	**1.580**	**0.200**	**64.440**	**2.320**	0.180	0.250
5	TRANSF	0.403	22.524	1.787	0.214	1.125	0.247	13.263	1.307	0.139	0.042	0.457	25.848	1.360	0.147	0.250
	STHGCN	0.455	25.744	1.926	0.215	1.306	0.323	17.640	1.542	0.142	0.306	0.498	28.956	1.149	0.149	**0.472**
	STHAN-SR	0.466	26.507	**2.090**	0.214	1.236	**0.450**	**25.600**	**2.080**	0.144	0.500	0.564	33.529	-1.340	0.145	0.292
	GCNKG	0.414	23.165	2.015	0.211	0.833	0.395	22.119	2.029	0.140	0.153	0.590	35.720	1.440	0.148	0.347
	CI-STHPAN	0.580	34.200	1.420	0.255	3.420	0.382	21.200	1.620	0.155	0.820	0.690	42.380	1.880	0.150	0.358
	FLExI	**0.740**	**45.000**	0.853	**0.296**	**6.580**	0.370	20.450	1.150	**0.166**	**1.5**	**0.800**	**49.420**	**2.290**	**0.153**	0.370
20	TRANSF	1.641	22.782	3.780	0.210	1.264	1.029	13.769	2.892	0.139	0.153	2.152	30.789	3.369	0.147	0.264
	STHGCN	1.702	23.719	4.370	0.207	1.069	1.685	23.617	3.964	0.143	0.653	2.633	38.794	4.092	0.150	0.920
	STHAN-SR	1.742	24.328	4.338	0.208	0.931	**1.790**	**25.130**	4.200	0.144	0.847	2.950	44.490	4.520	0.147	0.486
	GCNKG	1.504	21.345	3.848	0.211	1.417	1.172	16.001	3.196	0.138	0.181	2.632	38.978	4.109	0.147	0.569
	CI-STHPAN	2.620	38.140	5.020	0.255	3.120	1.340	18.420	3.820	0.152	0.640	4.280	72.640	5.420	0.149	0.980
	FLExI	**3.510**	**54.460**	**6.067**	**0.289**	**4.870**	1.500	20.630	**4.400**	**0.166**	**1.080**	**6.650**	**125.060**	**6.585**	**0.150**	**1.420**

Table 8: Top 1 Performance comparison of FLExI against baseline models, reported as averages over all phases of the corresponding datasets NASDAQ100, S&P500, and NIFTY500. Best results for each holding period are shown in **bold**.

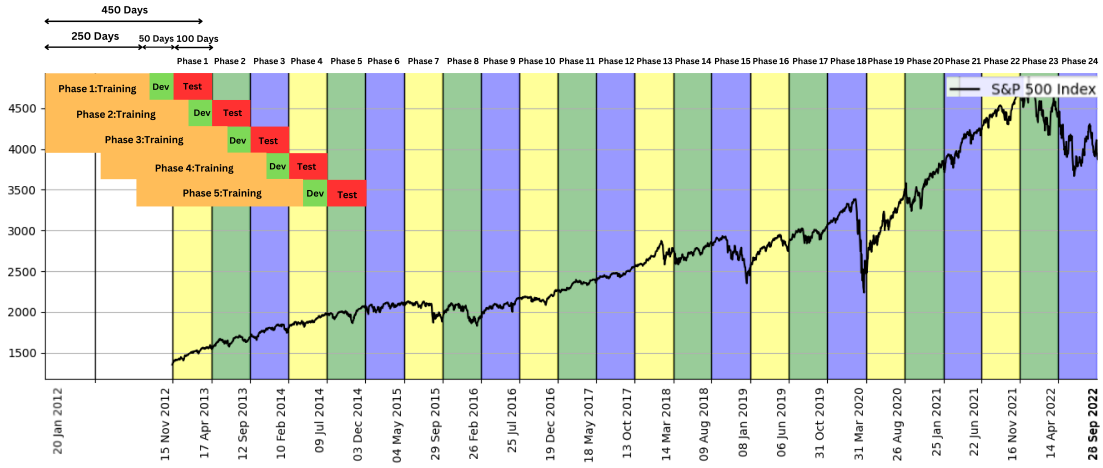

Figure 7: [Best viewed in color] The experimental setup for model training is illustrated across 24 phases. In each phase, the model undergoes a training period of 250 days, followed by a validation period of 50 days and, subsequently, a testing period of 100 days.

The percentage difference between the value of the investment at the beginning and end of the holding period.

E.2 Annualized Investment Return Ratio (AIRR)

The Annualized Investment Return Ratio (AIRR) is defined as

$$\text{AIRR} = (1 + \text{ROI})^{\frac{252}{\Delta}} - 1, \qquad (12)$$

where 252 is the average number of trading days in a year, Δ is the number of holding days, and ROI is the Return On Investment defined as IRR$/100$. The AIRR quantifies the average annual rate of return achieved on an investment, considering the compounding effect over time.

E.3 Sharpe Ratio (SR)

The Sharpe ratio measures the risk-adjusted return of an investment and can be calculated as follows:

$$\text{SR} = \frac{R_p - R_f}{\sigma_p} \qquad (13)$$

where R_p represents the average return of the investment, R_f the risk-free rate of return, and σ_p the standard deviation of the investment returns. The Sharpe ratio quantifies the excess return earned per unit of risk. It is commonly used to assess the attractiveness of investment opportunities based on risk-adjusted performance. The risk-free rates are calculated for each phase by averaging the 6-month treasury-bill yield obtained from “www.treasurydirect.gov" for the USA and

"www.investing.com" for India.

E.4 Normalized Discounted Cumulative Gain (NDCG)

Discounted Cumulative Gain (DCG) is a metric used to evaluate the quality of ranked lists. It can be calculated as

$$\text{DCG@}k = \sum_{i=1}^{k} \frac{\text{rel}_i}{\log_2(i+1)}, \tag{14}$$

where rel_i represents the relevance score of the item at position i in the ranked list. The DCG assigns higher weights to more relevant items at the top of the list by dividing the relevance score by the logarithm of their positions. The DCG accumulates these weighted scores up to position k to measure the overall quality of the ranked list. The Ideal Discounted Cumulative Gain (IDCG) is a reference value used to evaluate the quality of a ranked list, where rel_i represents the relevance score of the item at position i in the ideal ranking order. The Normalized Discounted Cumulative Gain (NDCG) is a metric used to evaluate the quality of the ranked lists. It can be calculated using the following formula:

$$\text{NDCG@}k = \frac{\text{DCG@k}}{\text{IDCG@k}} \tag{15}$$

NDCG provides a normalized measure of ranking performance ranging from 0 to 1, with higher values indicating better ranking quality.

E.5 Accuracy (ACC)

Accuracy represents the ratio of correct predictions to the total number of predictions made by a model. It provides an overall measure of the model's ability to correctly identify the top k stocks at each time point.

$$\text{ACC@}k = \frac{|\text{Top } k \text{ Predictions} \cap \text{Top } k \text{ Stocks}|}{|\text{Top } k \text{ Stocks}|} \times 100 \tag{16}$$

F Additional Experiments and Discussion

What is the best and worst case scenario in terms of return on investment? In practice, it is difficult to execute transactions optimally on any given day. To account for this uncertainty,we evaluated the best- and worst-case scenarios in terms of the investment return ratio (IRR). The best-case scenario is when the stock is bought at its lowest price on day T and sold at its highest price on day $T + \Delta$, whereas the worst-case scenario is when the stock is bought at its highest price on day T and sold at its lowest price on day $T + \Delta$. Table 9 shows the best and worst case IRR obtained across NASDAQ100, S&P500, and NIFTY500 using our model. The variance between the best and worst IRR decreases as the holding period increases. In addition, the results highlight the importance of accurately predicting closing prices is important task for real-world applications of this work.

What is the standard deviation of metrics across the 24 phases? Market conditions vary significantly over time, with different phases exhibiting bullish or bearish behavior. To assess the robustness of the proposed method under such variability, we analyze the standard deviations of the evaluation metrics across all 24 phases. Table 10 shows the standard deviation of the metrics across all 24 phases obtained using the proposed model. The standard deviation is comparatively lower for larger datasets, such as the S&P500 and NIFTY500. Furthermore, for all datasets and holding periods, selecting a larger set of stocks (Top 5) results in lower variability than selecting a single stock (Top 1). This suggests that portfolio diversification, together with the proposed temporal knowledge graph model, contributes to more consistent performance across different market conditions.

Performance on the Full Dataset In Section **??**, we have provided the results applying our method and baselines models on subsets of the whole dataset obtained as a sliding window of 24 "phases" and averaging out the results over all the phases. To study the effect of learning over a long time horizon, we provide the results of running the algorithms by dividing the entire dataset into training and test sets in Tables 11, 12, and 13.

Effect of Model Components on Performance To analyze the contribution of individual components in FLExI. We empirically compared multiple variants of the FLExI model. For the first variant, WOTPP, we removed the embeddings obtained from temporal process modeling and only considered the sequential and relational embeddings. For the second variant, LSTM*, we replace the transformer-based sequential embedding in the WOTPP with an LSTM. Next, we consider WOSEQ, in which the sequential embedding up-

Δ	Dataset	Top 1 (IRR)		Top 5 (IRR)	
		Best	Worst	Best	Worst
1	N100	4.06	-3.65	3.67	-3.26
	S&P	2.98	-2.79	2.65	-2.41
	NIFTY	5.12	-3.41	3.77	-3.43
5	N100	4.60	-3.10	4.14	-2.89
	S&P	3.62	-2.66	2.78	-2.27
	NIFTY	4.69	-2.91	4.52	-3.09
20	N100	4.26	-3.22	4.80	-2.21
	S&P	4.68	-1.58	4.33	-1.12
	NIFTY	6.13	-2.56	5.61	-2.32

Table 9: Best and worst case IRR across NASDAQ100, S&P500, and NIFTY500 for different holding periods Δ.

Δ	Dataset	Top 1				Top 5			
		NDCG	ACC	IRR	SR	NDCG	ACC	IRR	SR
1	N100	0.037	3.24	0.27	1.62	0.032	3.22	0.19	0.35
	S&P	0.029	2.58	0.23	1.76	0.023	1.6	0.15	0.41
	NIFTY	0.008	0.49	0.30	2.86	0.006	0.84	0.17	0.41
5	N100	0.05	6.40	1.42	4.23	0.032	3.94	0.84	0.69
	S&P	0.022	2.39	1.11	3.37	0.018	1.42	0.85	1.07
	NIFTY	0.023	1.79	1.36	3.42	0.02	2.24	0.96	0.86
20	N100	0.05	4.66	5.63	8.26	0.047	5.39	4.25	3.08
	S&P	0.035	3.01	3.42	13.89	0.032	3.27	2.25	1.94
	NIFTY	0.019	1.41	6.34	11.56	0.05	2.16	3.98	2.53

Table 10: Standard deviation of metrics across all 24 phases using FLExI.

date for the stock assets as input for the relational embedding layer is removed, along with the embeddings obtained from the temporal point process model. Finally, the temporal point process embeddings and heterogeneous transformation of entity features were removed to obtain WOHK. Table 14 shows that each component of our architecture plays an important role in improving the algorithm performance.

Effect of Increasing Training, Validation, and Testing Time in each phase on Model Performance As the stock price drifts in a direction over a period of time, it can negatively affect the model training. In particular, when the training and testing periods are sufficiently long, the stock price and the prevailing socio-political and economic conditions might have changed from the training to the test period. To measure the extent of this effect, two additional training setups are considered. The first (750TR) has a training period of 750 trading days (3 years), a 75-day validation period, and a 200-day testing period. In the next setup (1000TR), the training period is 1000 trading days (four years), 100-day validation, and 300-day testing period in each of the 21 phases. As shown in Table 15 the performance is affected when the training period is increased. However, the results obtained from the model still beat risk-free returns, as indicated by the Sharpe ratio.

F.1 Hyperparameters

The important hyperparameters used in the algorithm are listed in Table 16. The weight coefficients α_i for each loss component in the objective were set to 1. We performed experiments by varying the size of the Relational Embedding in $\{16, 32, 64, 128\}$ and chose the best-performing dimension. We set the size of the sequential embeddings to 20 and that of the relation embeddings to 16 for NIFTY and 128 for NASDAQ and S&P. For the ablation studies, we used embedding sizes of 20 and 16, respectively, to evaluate their effects on the model performance.

G Data Description

The dataset consists of three main components present in the `Phase-Stock-KG/data/` directory:

- `[country]_relations.json` – Contains relationships between entities in the stock market of the USA or India.

Δ	Model	Top 1					Top 5				
		IRR	AIRR	SR	NDCG	ACC	IRR	AIRR	SR	NDCG	ACC
1	TRANSF	-30	-58.5	-1.322	0.2646	1.81	-9.0	-26.6	-0.2002	0.3771	10.69
	STHAN-SR	-11.8	-26.9	-0.5448	0.2664	3.24	-14.2	-30.2	-0.2883	0.3673	10.97
	GCNKG	-12.5	-28.3	-0.4952	0.2773	5.05	**-0.16**	-30.2	**-0.2835**	0.3799	**11.19**
	CI-STHPAN	-8.90	-19.40	-0.3200	0.2801	5.82	-4.60	-25.10	-0.2550	0.3750	10.05
	FLExI	**-4.80**	**-11.6**	**-0.1845**	**0.283**	**6.49**	-8.81	**-20.6**	-0.226	**0.3712**	9.02
5	TRANSF	-0.63	-79.66	-1.206	0.2601	2.53	-0.61	-78.6	-0.4334	0.3677	7.87
	STHAN-SR	-0.82	-87.57	-1.4353	0.2865	5.77	-0.98	-90.63	-0.5805	0.3843	**11.26**
	GCNKG	-1.4	-97.11	-2.436	0.3072	**6.13**	-0.65	-69.09	-0.4497	**0.3935**	12.7
	CI-STHPAN	10.20	-30.50	-0.9000	0.2980	4.80	2.30	-20.00	-0.2200	0.3600	8.60
	FLExI	**23.14**	**11.9**	**0.6226**	**0.2899**	3.61	**6.24**	**3.2**	**0.0075**	0.3227	4.98
20	TRANSF	-2.84	-99.93	-2.4777	0.2522	2.89	-2.45	-99.9	-0.868	0.3834	**14.01**
	STHAN-SR	-3.09	-99.96	-2.823	0.2422	1.8	-3.43	-99.9	-1.112	0.3782	12.56
	GCNKG	-1.90	-90.23	**-0.9613**	0.233	2.52	-3.38	-99.9	-1.150	0.3745	12.49
	CI-STHPAN	-1.885	-55.40	-1.6000	0.2550	4.50	-2.50	-55.60	-0.9500	0.3800	11.20
	FLExI	**-1.87**	**-21.6**	**-2.5309**	**0.2796**	**6.85**	**-1.83**	**-21.0**	**-0.7427**	**0.3866**	10.1

Table 11: Comparison of FLExI with baseline methods on the NASDAQ100 dataset under full-dataset training. Results across multiple holding periods are reported, with the best values highlighted in **bold**. FLExI outperforms baseline models on the majority of evaluation metrics,.

Δ	Model	Top 1					Top 5				
		IRR	AIRR	SR	NDCG	ACC	IRR	AIRR	SR	NDCG	ACC
1	TRANSF	**-0.15**	-31.5	-0.6494	**0.1599**	0.72	-0.14	-24.79	-0.301	0.233	**2.53**
	STHAN-SR	-0.24	-45.42	-1.4418	0.1558	0.36	-0.05	-12.17	-0.2117	0.2186	1.73
	GCNKG	-0.42	-65.2	-2.0955	0.1586	**1.08**	-0.09	-13.95	-0.2976	0.225	2.02
	CI-STHPAN	-0.425	-30.00	-1.1000	0.1500	0.600	3.50	5.80	-0.1200	0.2170	1.20
	FLExI	-0.43	**-1.08**	**-0.0618**	0.1429	0	**7.20**	**20.2**	**0.0443**	**0.2087**	0.72
5	TRANSF	-0.41	-64.49	-1.3532	0.1501	0	-0.49	-57.03	-0.5134	0.2196	1.52
	STHAN-SR	-0.61	-78.6	-1.4651	0.1518	0	-0.04	-9.71	-0.0767	**0.2234**	**2.45**
	GCNKG	-0.51	-72.43	-1.356	0.141	0	-0.32	-41.61	-0.3208	0.2159	1.66
	CI-STHPAN	3.20	-34.10	-0.5400	0.1470	0	12.40	23.80	0.0100	0.2120	1.30
	FLExI	**7.51**	**3.9**	**0.2717**	**0.1529**	**0**	**34.2**	**89.3**	**0.3642**	0.2087	1.01
20	TRANSF	-1.10	-94.15	-1.8781	0.1457	0	-1.34	-95.41	-0.6883	0.2133	1.37
	STHAN-SR	-1.5	-98.05	-2.666	0.147	0.36	-0.51	-65.65	-0.2844	0.2095	**1.81**
	GCNKG	-0.09	-20.3	-1.1363	0.1553	1.44	**-0.24**	-45.42	**-0.1537**	**0.2119**	2.09
	CI-STHPAN	0.210	-12.80	-1.0800	0.1560	0.720	-0.980	-12.60	-0.7200	0.2060	1.05
	FLExI	**0.496**	**-6.1**	**-1.0253**	**0.1572**	**0**	-1.71	**19.1**	-1.2973	0.2006	0

Table 12: Comparison of FLExI with baseline methods on the S&P500 dataset under full-dataset training. Results across multiple holding periods are reported, with the best values highlighted in **bold**. FLExI outperforms baseline models on the majority of evaluation metrics,

Δ	Model	Top 1					Top 5				
		IRR	AIRR	SR	NDCG	ACC	IRR	AIRR	SR	NDCG	ACC
1	TRANSF	-0.18	-36.49	-1.3665	0.1544	0	-0.02	-4.86	-1.3643	0.2292	1.52
	STHAN-SR	-0.01	-34.8	-0.3998	0.1621	0	0.16	48.92	-1.0408	**0.2345**	**2.45**
	GCNKG	0.01	2.60	-0.1551	0.1511	0	0.06	-1.61	-1.515	0.2237	1.29
	CI-STHPAN	3.40	11.20	0.1200	0.1515	0	2.10	6.40	-1.4700	0.2237	1.26
	FLExI	**7.0**	**19.5**	**0.3843**	**0.1518**	**0**	**4.89**	**13.5**	**-1.4227**	0.2237	1.23
5	TRANSF	-0.62	-79.14	-1.7195	0.1493	0	-0.29	-72.21	-0.9045	**0.2232**	1.52
	STHAN-SR	-0.91	-91.42	-2.761	0.1527	0	**0.06**	17.12	**-0.5584**	0.2239	**2.23**
	GCNKG	**0.04**	**-11.04**	**0.0318**	0.1418	0	-0.13	-39.85	-0.7612	0.2127	0.86
	CI-STHPAN	-13.50	-28.40	-0.6890	0.1465	0	-10.15	-39.82	-0.9750	0.2170	0.54
	FLExI	-27.01	-45.7	-1.4099	**0.1513**	**0**	-20.17	**-39.8**	-1.1897	0.2210	0.21
20	TRANSF	-2.78	-99.9	-2.9371	0.1271	0	-1.10	-94.14	-1.2607	0.1981	**0.87**
	STHAN-SR	-3.66	-99.9	-5.645	0.1439	0	-0.86	-90.23	-0.729	0.2126	0.79
	GCNKG	-0.55	-74.92	-1.124	0.1298	0	-1.05	-93.66	-1.131	0.1984	0.43
	CI-STHPAN	-0.525	-66.85	-0.9870	0.1406	0	-0.91	-82.63	-1.0220	0.2100	0.25
	FLExI	**-0.5**	**-58.8**	**-0.8499**	**0.1514**	**0**	**-0.77**	**-71.6**	**-0.914**	**0.2216**	0.07

Table 13: Comparison of FLExI with baseline methods on the NIFTY500 dataset under full-dataset training. Results across multiple holding periods are reported, with the best values highlighted in **bold**. FLExI outperforms baseline models on the majority of evaluation metrics,

Δ	Model	Dataset	Top 1					Top 5				
			IRR	AIRR	SR	NDCG	ACC	IRR	AIRR	SR	NDCG	ACC
1	WOSEQ	N100	0.13	38.73	0.60	0.27	3.87	0.08	22.32	0.04	0.37	10.82
		S&P	0.103	19.61	0.77	0.15	0.49	0.08	22.32	0.05	0.221	1.95
		NIFTY	0.15	45.89	0.85	0.154	0.29	0.14	42.26	0.244	0.225	1.67
	WOHK	N100	0.13	38.73	0.61	0.29	5.24	0.13	38.73	0.14	0.40	14.4
		S&P	0.59	340.36	0.23	0.17	1.37	0.10	28.64	0.078	0.24	3.44
		NIFTY	0.06	16.31	0.425	0.157	0.33	0.13	38.73	0.22	0.22	2.08
	LSTM*	N100	0.16	49.61	0.77	0.28	5.12	0.13	38.73	0.14	0.39	12.54
		S&P	0.016	4.11	0.279	0.16	0.62	0.06	16.31	0.031	0.23	2.57
		NIFTY	0.17	53.42	1.00	0.15	0.58	0.12	35.28	0.17	0.22	1.76
	WOTPP	N100	0.09	25.44	0.48	0.27	3.58	0.11	31.92	0.11	0.37	10.77
		S&P	0.11	31.92	0.90	0.16	0.79	0.09	25.44	0.99	0.22	2.23
		NIFTY	0.18	57.33	1.14	0.156	0.29	0.13	38.73	0.19	0.23	1.65
5	WOSEQ	N100	0.55	31.84	1.38	0.27	3.87	0.58	33.84	0.54	0.38	13.39
		S&P	0.06	3.06	0.56	0.16	1.20	0.236	12.61	0.34	0.22	2.36
		NIFTY	0.55	31.84	0.86	0.151	0.24	0.39	21.67	0.53	0.227	1.56
	WOHK	N100	0.50	28.57	0.23	0.31	6.02	0.68	40.71	0.57	0.41	15.8
		S&P	0.624	36.82	1.90	0.17	1.66	0.417	23.33	0.49	0.24	4.14
		NIFTY	0.745	45.36	1.86	0.16	0.79	0.63	37.23	0.646	0.23	2.65
	LSTM*	N100	0.68	40.71	1.51	0.28	4.83	0.5	28.57	0.49	0.38	12.6
		S&P	0.68	40.71	1.64	0.17	2.08	0.36	19.85	0.36	0.24	3.74
		NIFTY	0.72	43.55	1.45	0.15	0.99	0.72	43.55	0.76	0.22	2.10
	WOTPP	N100	0.71	42.84	2.18	0.28	4.24	0.6	35.18	0.586	0.38	12.7
		S&P	0.77	47.19	3.07	0.16	0.95	0.55	31.84	0.68	0.22	2.51
		NIFTY	0.98	63.47	2.32	0.15	0.54	0.62	36.54	0.56	0.22	1.61
20	WOSEQ	N100	3.35	51.46	5.22	0.29	7.83	1.92	27.07	0.86	0.39	13.36
		S&P	1.68	23.35	0.45	0.177	2.37	1.36	18.55	0.89	0.24	4.10
		NIFTY	3.53	54.82	2.57	0.153	0.66	2.48	36.16	1.37	0.229	2.07
	WOHK	N100	0.84	36.16	0.617	0.28	3.33	1.43	19.59	0.616	0.39	13.98
		S&P	0.14	1.77	-3.11	0.179	1.87	0.92	12.23	0.16	0.24	4.77
		NIFTY	2.48	36.16	2.96	0.158	0.24	2.89	43.186	1.84	0.23	3.52
	LSTM*	N100	3.66	57.29	3.13	0.30	5.74	2.18	31.22	0.38	0.40	15.13
		S&P	0.38	4.89	-0.84	0.18	1.16	1.05	14.06	0.255	0.24	4.59
		NIFTY	2.08	29.61	1.81	0.15	0.62	2.64	38.86	1.24	0.23	2.18
	WOTPP	N100	2.43	35.32	3.48	0.28	4.58	1.42	19.44	0.68	0.379	11.34
		S&P	1.95	27.54	3.20	0.182	2.20	1.40	19.14	0.738	0.24	4.24
		NIFTY	6.09	110.61	6.96	0.16	1.66	2.22	31.87	1.54	0.24	2.89

Table 14: Ablation analysis: comparison of FLExI without various components of the architecture as described in Section 6.

Δ	Model	Dataset	Top 1					Top 5				
			IRR	AIRR	SR	NDCG	ACC	IRR	AIRR	SR	NDCG	ACC
1	750TR	N100	0.112	32.47	0.652	0.268	2.5	0.121	35.67	0.428	0.375	12.12
		S&P	0.004	0.92	0.179	0.154	0.25	0.066	18.02	0.306	0.225	2.62
		NIFTY	0.139	41.83	-499.2	0.163	0.57	0.094	26.7	0.470	0.233	1.97
	1000TR	N100	0.059	16.18	0.266	0.271	0.377	0.123	36.22	0.448	0.382	11.58
		S&P	0.158	48.78	0.463	0.162	1.16	0.078	21.81	0.329	0.233	2.15
		NIFTY	0.087	24.38	0.515	0.155	0.17	0.111	32.4	0.485	0.227	1.83
5	750TR	N100	0.38	21.58	1.043	0.275	2.49	0.617	36.34	1.035	0.384	12.46
		S&P	0.718	43.42	1.923	0.166	1.05	0.457	25.85	0.856	0.236	3.39
		NIFTY	0.447	25.2	1.008	0.159	0.5	0.332	18.21	0.665	0.231	1.93
	1000TR	N100	0.411	22.97	0.984	0.279	3.27	0.645	38.25	1.006	0.390	13.81
		S&P	0.142	7.4	0.597	0.169	1.03	0.621	36.65	1.433	0.244	4.14
		NIFTY	0.901	57.1	2.102	0.157	0.71	0.638	37.76	1.181	0.23	2.16
20	750TR	N100	0.378	4.87	1.536	0.258	2.17	1.456	19.98	1.698	0.3723	11.58
		S&P	0.588	7.66	1.254	0.187	1.35	1.256	17.03	1.420	0.258	4.34
		NIFTY	2.637	38.8	3.404	0.162	0.1	2.332	33.7	2.107	0.235	2.51
	1000TR	N100	1.334	18.18	1.433	0.265	4.01	2.082	29.64	2.440	0.373	11.35
		S&P	0.907	12.05	1.767	0.162	0.11	1.853	26.02	1.945	0.237	3.6
		NIFTY	0.613	8.0	0.565	0.147	0	1.679	23.34	1.724	0.219	1.33

Table 15: Performance of the FLExI model with a 750-day training period (750TR) and a 1000-day training period (1000TR).

PARAMETER	VALUE
LEARNING RATE	0.00001
NUMBER OF EPOCHS	10
TEMPORAL POINT PROCESS EMBEDDINGS SIZE	128
LEARNING RATE FOR HPGE MODEL	0.0001

Table 16: Hyperparameters

- `[country]_nodes.json` – Represents the nodes (companies, sectors) in the knowledge graph.
- `[company_ticker]-[name].csv` – Contains historical stock price movements, including features Close, Open, Low, and High prices, and trading volume for each trading day.

G.1 `[country]_relations.json` & `[country]_nodes.json`

These files contain the relationships between different entities in the stock market and nodes in the knowledge graph. The information required to create these files was derived from multiple sources, including:

- **Financial Reports:** Balance sheets, cash flow statements, and financial statements of individual companies.
- **News Articles:** Relevant financial news that may influence stock prices and company relationships.
- **Events:** Events in India and the USA that could affect market trends.

G.2 `[company_ticker]-[name].csv`

The directory `data/[nifty500/nasdaq100/sp500]` contains CSV files with time-series data of stock price indicators, such as Open, High, Low, Close, and Volume for each company.

G.3 Knowledge Graph Construction

The `temporal_kg_nifty.pkl` and `temporal_kg.pkl` files, for the Indian and US markets, respectively, are generated by processing relationships and nodes extracted from stock market data. Initially, JSON files such as `[country]_relations.json` and `[country]_nodes.json` are loaded to extract the relationships between different entities and the corresponding nodes in the knowledge graph. Each node is mapped to its corresponding attributes.

Once the nodes are mapped, relationships (consisting of a head entity, tail entity, relation type, and timestamps) are iterated and linked to the respective entities, forming a structured knowledge graph. The temporal aspect was introduced by incorporating time-based stock market events from CSV files containing financial statements, news articles, and event data from India and the USA. These time-sensitive events are mapped to the corresponding entities in the graph to ensure a dynamic representation of market fluctuations.

Handling timestamps:

- If an expiry timestamp is available, it is used.
- If only a timestamp is present, expiry is assumed as one day later.
- If no timestamp exists, it defaults to 1970-01-01.

The processed relations are stored as

$$[\, head,\ tail,\ relation_id,\ timestamp,\ expiry_timestamp \,]. \tag{17}$$

The data were sorted by timestamp for chronological ordering. Finally, the processed temporal knowledge graph was serialized and stored as `temporal_kg_nifty.pkl` and `temporal_kg.pkl` using the pickle.

G.4 Batch Dataset Creation from Historical CSV and PKL Files

The batch dataset was constructed using historical stock data from CSV files and a temporal knowledge graph from PKL files.

G.4.1 Stock Data from CSV Files

- Iterate over stock price CSV files stored in the directory (e.g., `data/nasdaq100/`).
- Each file corresponds to a company and contains columns: Date, Open, High, Low, Close, Volume.
- Ignore files with less than 2800 rows to ensure sufficient training samples.

G.4.2 Time-Series Windowing

For each company:

- Apply a sliding window approach with window size W and prediction horizon Δ.
- Normalize input features by dividing by the first value in the window.
- Predict returns at different horizons: $T = \{1, 5, 20\}$.
- Extract low and high prices for multiple horizons.

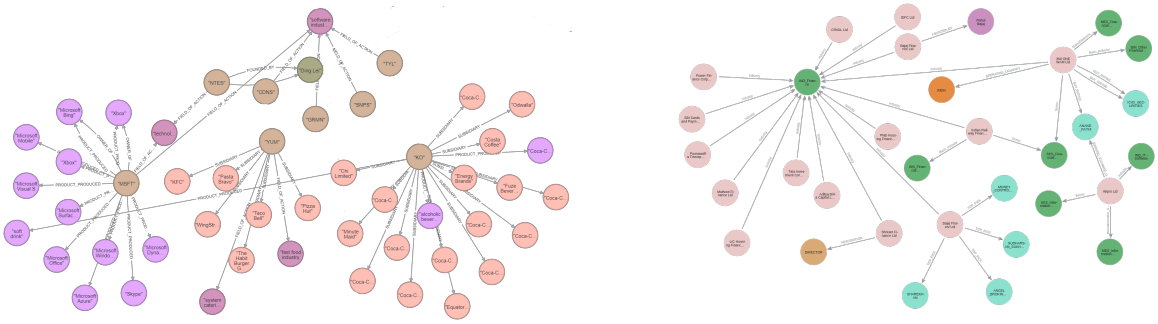

Figure 8: Visualization of a subgraph from the constructed STOCKgs for (left) NASDAQ and (right) NSE share markets. Entities and relations are highlighted by type.

G.4.3 Loading the temporal knowledge graph

- Load the preprocessed knowledge graph from `temporal_kg.pkl`.
- Extract relevant temporal facts corresponding to each time window.
- Constructs a sector-based graph using adjacency relations.

G.4.4 Final Dataset Construction

For each time window:

- Collect stock price window features and targets.
- Extract corresponding temporal KG snapshot.
- Form a dataset tuple:

$$(\text{Stock Price Window Features}, \text{Temporal KG}) \tag{18}$$

The final output is a file `data/pickle/[dataset]/full_graph_data.pkl` containing a dictionary with keys `train` and `company`, which consist of a batch dataset containing stock price sequences and the temporal knowledge graph and a dictionary of company IDs, respectively.

G.5 Node Data Format

The nodes JSON file consists of structured data that represent entities in the knowledge graph. Each node entry contains the following:

- **identity**: A unique identifier for the node.
- **labels**: A list of labels describing the type of the entity (e.g., "Person", "Company", etc.).
- **properties**: Key-value pairs defining attributes of the node, such as `name` and `id`
- **elementId**: A system-generated unique identifier for the node.

Example JSON node entry:

```
{
  "n": {
    "identity": 0,
    "labels": ["Person"],
    "properties": {
      "name": "Tommy Millner",
      "id": 114689399
    },
    "elementId": "0"
  }
}
```

In this example, a node labeled Person represents Tommy Millner with an assigned ID of 114689399.

G.6 Relation Data Format

In this example, there is a `SUBSIDIARY` relationship between the nodes with identities 1007 and 2591, with a relationship ID of "P355".

Example JSON relation entry:

```
{
  "r": {
    "identity": 0,
    "start": 1007,
    "end": 2591,
    "type": "SUBSIDIARY",
    "properties": {
      "id": "P355"
    },
    "elementId": "0",
    "startNodeElementId": "1007",
    "endNodeElementId": "2591"
  }
}
```

The relations JSON file captures the relationships between nodes and describes how entities are linked. Each relationship entry consists of the following:

- `identity`: A unique identifier for the relationship.

- `start`: The identity of the starting node in the relationship.
- `end`: The identity of the ending node in the relationship.
- `type`: A string indicating the relationship type (e.g., `"SUBSIDIARY"`, `"OWNS"`, `"EMPLOYS"`, etc.).
- `properties`: Additional metadata, such as a unique `id` for the relationship.
- `elementId`: A system-generated identifier for the relationship.
- `startNodeElementId` / `endNodeElementId`: Unique element IDs of the connected nodes.

This structured data enables efficient querying and analysis within the knowledge graph, forming the basis for graph-based learning and reasoning. The relationships between entities allow the construction of a knowledge-driven system for applications such as recommendation systems, risk analysis, and trend forecasting, among others.

H Web Scraping Details

H.1 Entity Recognition and Event Extraction for News Articles

Entity Recognition Entities such as companies, dates, and financial metrics are recognized using predefined patterns and keywords in the news content. The functions `earning_pattern` and `guidance_pattern` use regular expressions and keyword matching to identify the relevant entities and events.

Event Extraction

- ***Earnings Events***: The `earning_pattern` function looks for keywords like 'beat', 'miss', and 'in-line' within the news content to determine if a company has met, exceeded, or fallen short of earnings expectations. It extracts the relevant information and categorizes the event as `BEATS_EXPECTATION`, `MISSES_EXPECTATION`, or `INLINE_EXPECTATION`.
- ***Guidance Events***: The `guidance_pattern` function searches for keywords indicating changes in company guidance, such as 'raise', 'cut', or 'in-line'. It categorizes the event as `RAISES_GUIDANCE`, `CUTS_GUIDANCE`, or `INLINE_GUIDANCE`.

Timestamp Extraction The `get_timestamp` function extracts timestamps from the news content. It identifies the location of the timestamp within the content structure and converts it to a datetime object using the `Pandas` library.

Head and Tail Recognition

- ***Head***: The head of a triple typically represents the subject of the event, such as the company name. It was extracted from the news content using keyword matching and context analysis.
- ***Tail***: The tail of a triple represents the object or outcome of the event, such as the financial metric or result. It was identified using similar keyword matching and context analysis techniques.

This process ensures that financial events are accurately extracted and structured for further analysis or integrated into a graph database.

H.2 Profile and Relationship

The notebook `wikidata_create.ipynb` was designed to create a knowledge graph (KG) from Wikidata for companies listed on NASDAQ and NYSE. It involves several steps, including data loading, processing, mapping, and storing the relationships between the companies and their attributes.

CSV files containing mappings between ticker symbols and Wikidata codes for NASDAQ and NYSE companies were loaded.

H.2.1 Ticker List Creation

Iterates through directories containing company data files to create a list of tickers with sufficient data points (more than 2600 rows).

H.2.2 Wikidata Querying

Using `pywikibot`, the notebook queries Wikidata to fetch the relationships and attributes of the companies. Relationships such as `"OWNED_BY"`, `"SUBSIDIARY"`, `"PARENT_ORGANISATION"`, `"CEO"`, etc., are considered.

Hardware Configuration. All experiments were performed on computing server equipped with eight NVIDIA A100-SXM4-40GB GPUs. Model training and evaluation were implemented using PyTorch with CUDA support.