A Deep Learning Framework for Pricing Financial Instruments

QIONG WU, William and Mary
ZHENG ZHANG, William and Mary
ANDREA PIZZOFERRATO, Queen Mary University of London & The Alan Turing Institute
MIHAI CUCURINGU, University of Oxford & The Alan Turing Institute
ZHENMING LIU, William and Mary

We propose an integrated deep learning architecture for the stock movement prediction. Our architecture simultaneously leverages all available alpha sources. The sources include technical signals, financial news signals, and cross-sectional signals. Our architecture possesses three main properties. First, our architecture eludes overfitting issues. Although we consume a large number of technical signals but has better generalization properties than linear models. Second, our model effectively captures the interactions between signals from different categories. Third, our architecture has low computation cost. We design a graph-based component that extracts cross-sectional interactions which circumvents usage of SVD that's needed in standard models. Experimental results on the real-world stock market show that our approach outperforms the existing baselines. Meanwhile, the results from different trading simulators demonstrate that we can effectively monetize the signals.

Additional Key Words and Phrases: Stock trend prediction, Deep learning, Financial time series

1 INTRODUCTION

This paper examines the design of deep learning algorithms to forecast equities (stocks) return. While this research question had been asked long ago and has a rich literature behind [10, 21, 28, 45, 50, 56], including works that doubted its legitimacy [9], it became clear that machine learning is a powerful tool for asset management. Such lines of work have significantly increased in popularity in recent years, mostly fueled by the fact that a large collection of high-quality financial data sets have become available. Roughly speaking, there have been two main schools of thought building machine learning (ML) models for pricing assets: (i) *financial professionals* who treat ML tools as black boxes that can learn non-linear relationships, which they *plug-in* into their existing factor research framework, and (ii) *computer scientists* who treat the task at hand as yet-another machine learning problem, and aim to generalize existing building blocks (e.g., NLP modules [26, 55] or LSTM models [2, 45]) for time series arising from stock prices data.

Limitation of existing models. Perhaps surprisingly, efforts from financial professionals and computer scientists seem to be disjoint, with little work being done to address cross-cutting constraints arising across both finance and machine learning communities. We briefly discuss three major weaknesses of existing models.
1. The source of alpha is not identified. Many machine learning models (especially those that rely only on news as the data source) do not examine the source of alpha (i.e., the reason why the model is able to make money). For example, a stock with recent robust return may result in media attention with positive commentary. Following that, an NLP-based model may pick up the news' positive sentiment and predict that the stock price will continue to rise. However, this is essentially a "momentum" strategy [4, 40], and thus the NLP-based model faces two main issues: (i) the model could be more effective if it can directly use price information, instead of news; (ii) a momentum strategy is usually considered as "low quality" alpha (i.e., low return and Sharpe).
2. Heterogeneous data sets are not used. Most models use only one type of data (i.e., either news text or "factors" [28, 38, 39, 42, 54] in numeric form). It remains open to build a model that leverages heterogeneous data and alpha sources. This model needs to reconcile (sometimes contradictory) information from different data sources.
3. Cross-sectional effects are not properly leveraged. Another salient property of the equity return forecasting problem is that the training pairs (X, y) are not independent. For example, when Google has a positive return, it is more likely that Facebook also has a positive return, and thus $y_{\text{google},t}$ and $y_{\text{facebook},t}$ are correlated. It remains open how the correlations between $y_{\text{google},t}$ and $y_{\text{facebook},t}$ may be properly leveraged.

Keeping in mind the above shortfalls, the following problem remains widely open: *Is it possible to build an end-to-end machine learning pipeline that leverages heterogeneous data sources to extract high quality alpha?*

Our approach. We propose a new global framework that integrates different sources of data which may capture different salient/latent structures underlying the same set of financial instruments. Our solution innovates on both the data and methodological sides to circumvent the above limitations in existing models.

Data source innovation. Our solution simultaneously uses multiple categories of signals (heterogeneous data) so that we can extract the "alphas" in different ways: (i) Technical signals that are primarily constructed from stocks' prices and their trading volumes [14, 48], which reflect trading activities in the market, (ii) Financial news signals that indicate the emergence of shocks/events, and (iii) Cross-sectional signals that code the co-movement between similar stocks.

Methodological innovation. Our framework needs to be computationally and statistically efficient in dealing with heterogeneous signals. Our new methodology possesses three salient properties: (i) *Low overfitting.* Although our model uses an extensive collection of technical signals, it generalizes better than linear models. (ii) *Cross-categorical interactions:* our model captures interactions between signals from different categories (e.g., how shocks of one stock and trading volumes of a second stock interact), and (iii) *Low compute cost:* a standard way to extract cross-sectional signals is to use the so-called "factor model" [48]. This often inevitably requires us to run SVD on large datasets. We design a graph-based gadget to extract cross-sectional signal without using SVD.

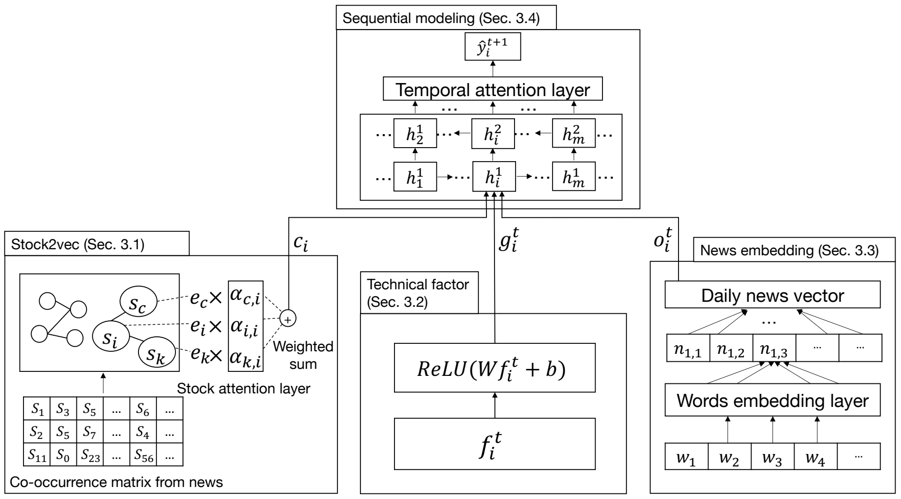


Fig. 1. The framework of our end-to-end model. Nodes in the Stock2vec model represent stocks. We learn the stock vector c_i from Stock2vec, factor embedding g_i^t from technical module and news representation o_i^t from news module. After that, we feed $[c_i, g_i^t, o_i^t]$ to a sequential model to make prediction on $\hat{y}_i^{t+1}$.

2 RELATED WORK

Predicting equity returns (i.e., empirical asset pricing) is an extensively studied academic disciplinary that can date back to the beginning of the 20th century [5, 23, 30], so it is impossible to provide a comprehensive review here. Instead, we focus on recent work on using machine learning to forecast asset prices.

Deep learning. There are two major approaches to forecast equity returns. *Approach 1. ANN as a blackbox for standard "factors."* First, "factors" that are known to be correlated with returns are constructed. These factors can be viewed as features constructed by financial experts. Second, the factors are fed into standard ANN black boxes so that non-linear models are learned (see e.g., [17, 24, 25] and references therein). Little effort is made to optimize ANN's architecture or algorithm. *Approach 2. Forecasting the price time-series.* This approach views the price, trading volume, and other statistics representing trading activities as time series and designs specialized deep learning models to extract signals from the time series. Little feature engineering is done for these models. See e.g., [12, 16, 31, 45, 53].

Approach 1 represents the line of thought that feature engineering is critical in building machine learning models, whereas Approach 2 represents the mindset that deep learning can automatically extract features so effort on feature engineering should be avoided.

News. NLP-based techniques are developed to correlate news with the movement of stock prices. Earlier works use matrix factorization approaches (see e.g., [38, 49]) whereas more recent approaches use deep learning methods [10, 11]. These methods exclusively use news to predict equity returns, and they do not consider any other "factors" that can impact the stock prices.

Factor model (cross-sectional returns). The movement of two or more stocks usually can be explained by a small subset of factors. For example, Facebook and Google often co-move because their return can be explained by the technical factors. The so-called "factor model" (e.g., [14, 48, 54, 56]) can effectively capture the co-movement of prices but these methods usually rely on PCA/SVD techniques and are not computationally scalable.

Comparison. *1. Comparing to existing DL models.* We find that we need both careful feature engineering and optimizing DL techniques to use technical factors in the most effective manner, *2. Comparing with News/NLP-based techniques.* We do not exclusively rely on news. Instead, we explicitly model the interaction between news and other factors so that our model avoids low quality signals (e.g., news-based signals could be essentially trading momentum), and *3. Comparing with factor models.* A key innovation of our model is the introduction of Stock2vec component. This component models the interaction between stocks, and circumvents SVD computation on large matrices.

3 OUR END-TO-END NETWORK

Our problem setting. Given a universe of n stocks, we define the log return for a given stock i on day t as

$$r_i^t = \log\left(\frac{p_i^t}{p_i^{t-1}}\right),$$

where p_i^t denotes the open price of stock i on day t. Also, let f_i^t be a vector of factors associated with stock i on day t . Similarly, c_i^t is the news corpus associated with the i-th stock on day t. See below for more discussion for the selection of f_i^t and c_i^t.

We formulate our problem as a regression problem. The response is the future return r_i^t. The features stem from the information between day $t-1$ and $t-T$, where T is a hyper-parameter. In other words, the features are f_i^j and c_i^j for $t-T \leq j < t$.

Our framework. We consider two different inputs, the financial news corpus and the stock technical factors, the latter being derived from price and dollar volume in prior trading dates. As shown in

Figure.1, our framework comprises of the following components: a Stock2vec graph module, a technical factor representation module, a news embedding module, along with sequential modeling.

As shown in Figure. 2, a single piece of news can be related to one or more stocks. We build the co-occurrence matrix from news and equities by leveraging GloVe [43]. For each stock, we extract their neighbors and assign attention weights to update the stock embedding that reflects the neighborhood.

In the news module, we pre-train an unsupervised Word2vec [22] on the training data set as the word embedding layer. We use a word embedding layer to calculate the embedded vector for each word, and then average all the words' vectors to construct a news vector. For a specific stock s_i on date t, we average all the news vectors related to the s_i within date t. Next, we concatenate the stock embedding and the technical factor embedding with the constructed news factors. In the next step, these vectors are encoded by a bi-directional LSTM [20, 29]. Following this, a temporal attention layer assigns an attention value to each prior date and calculates the weighted mean of these encoded corpus vectors to represent the overall sequential information. Finally, a regressions step is performed to forecast the next day return. The details of the framework are elaborated below.

3.1 Stock2vec

This section describes the methodology for Stock2vec, shown in Figure 1. The pipeline consists of two components.

Component 1. Stock Embedding Initialization. We will (i) construct an initial embedding $e_i \in \mathbf{R}^d$ for each stock i, (ii) construct a k-nearest neighbor directed graph $G = \{\mathcal{V}, \mathcal{E}\}$, where $\mathcal{V} = \{s_1, \ldots, s_n\}$ is the set of stocks. $(s_i, s_j) \in E$ if and only if s_j is a k-nearest neighbor (kNN) of s_i. Note that kNN is not symmetric ($(s_i, s_j) \in \mathcal{E}$ does not imply $(s_j, s_i) \in \mathcal{E}$).

Co-occurrence matrix. We construct a co-occurrence matrix [37] $\mathbf{X} \in \mathbf{R}^{n \times n}$ from our news dataset. $\mathbf{X}_{i,j}$ represents the number of training news articles that mention both s_i and s_j. *Initial embedding of the stocks.* We next construct an initial embedding of the stocks. This embedding will be fine-tuned when training the full model by using backpropagation. Recall that $e_i \in \mathbf{R}^d$ is the initial embedding we aim to construct. We apply a technique called GloVe [43] to build e_i's from $\mathbf{X}$. We find e_i's that optimizes the objective function

$$J_s = \sum_{i,j=1}^{n} f_s(\mathbf{X}_{ij})(e_i^\top e_j + b_i + b_j - \log \mathbf{X}_{ij}), \tag{1}$$

where b_i and b_j are stock s_i's and s_j's bias terms, and the weighting function is given by

$$f_s(x) = \begin{cases} (x < x^{max})^\alpha & \text{if } x < x_{max}. \\ 1 & \text{otherwise} \end{cases} \tag{2}$$

Nearest neighbor graph. We construct $G = \{\mathcal{V}, \mathcal{E}\}$ from $\{e_i\}_{i \in [n]}$ in a straightforward manner: we let the distance between stock i and j be $\|e_i - e_j\|$. Then $(e_i, e_j) \in \mathcal{E}$ if and only if stock j is a k-nearest neighbor of e_i (k is a hyper-parameter to be tuned).

Component 2. Stock Attention Mechanism. We introduce an attention mechanism [52] so that we can leverage the relationship between stocks to improve our model's forecasting power.

Let $S(i)$ be the set of neighbors of s_i in G. Let

$$c_i = \sum_{j \in S(i)} \alpha_{ij} e_j \tag{3}$$

$$\sum_{j \in S(i)} \alpha_{ij} = 1 \text{ for } j \in S(i), \tag{4}$$

where c_i denotes the final stock representation, and $\alpha_{i,j} \in \mathbf{R}$ is the attention weight on the embedding e_j, which is given by

$$\alpha_{ij} = \frac{exp(f(e_i, e_j))}{\sum_{k \in S(i)} exp(f(e_i, e_k))}. \tag{5}$$

The weights define which neighboring stock is more significant. $f(e_i, e_j)$ measures the compatibility between embedding e_i and e_j, and is parameterized by a feed-forward network with a single hidden layer, which is jointly trained with other parts of the model. We let $f(\cdot, \cdot)$ have the following functional form:

$$f(e_i, e_j) = \mathbf{v}_a^\top \tanh(\mathbf{W_a}[e_i; e_j] + b_a), \tag{6}$$

where $\mathbf{v}_a$ and $\mathbf{W}_a$ are weight matrices and b_a is the bias vector [52].

3.2 Technical module

We next describe our module of processing technical factors. We build a large collection of technical indicators (approximately 300 factors) that are frequently used in the literature [7].

Embedding layer. We aim to learn an embedding function that maps the daily l dimensional technical factors into a vector of dimension m. Given the technical factors vector for stock s_i on day t, it is converted into a vector representation g_i^t as follows

$$g_i^t = ReLU(W_{emb} f_i^t + b_{emb}), \tag{7}$$

where weight matrix $W_{emb} \in \mathbf{R}^{m \times l}$, $b_{emb} \in \mathbf{R}^m$, and $ReLU(v) = \max(v, 0)$. See Sec 4.6.2 for more detailed explanation.

3.3 News embedding

We pre-trained Word2vec [34, 35] on the news corpus from the training data set to produce word embeddings. We average of all the word vectors in a piece of news to represent the news vector. As shown in the right plot of Figure 1, for a given date t and stock i, we compute the daily news vector o_t^i by extracting the news related to stock i and averaging the news vectors within date t.

3.4 Sequential modeling

Finally, we overlay the stock vector c_i, the technical vector g_i^t, and the news embedding o_i^t with a sequential neural net. Specifically, let h_t be the hidden state at time t

$$h_t = [c_i, g_i^t, o_i^t]. \tag{8}$$

We use Bidirectional Recurrent Neural Networks (BRNN [36, 46]) as the sequential neural net model. BRNN connects two hidden layers of opposite directions to preserve information from both past and future. We implemented BRNN by Long Short-term Memory (BiLSTM[20]). LSTM is a variant of the recurrent net, capable of learning long-term dependencies. With T denoting the number of steps for the time sequence, the final output is

$$v_{t-T}, ..., v_{t-1}, v_t = \text{BiLSTM}(h_1, h_2 ... h_T; \theta), \tag{9}$$

where θ denotes the parameters from BiLSTM.

Temporal attention layer. Since the past stock indicators contribute to the stock trend unequally, we adopt the attention mechanism on the temporal level. We consider

$$v_{t+1} = \sum_p \beta_p v_p, \\ \beta_p = \frac{exp(f(v_p, v_q))}{\sum_q exp(f(v_p, v_q))}, \tag{10}$$

where β_p is the attention weight for prior date p indicating how importance of the date. We then compute the weighted sum to incorporate the sequential data and temporal attention.

MLP and Loss Function. The final discriminative networks are a standard Multi-layer Perceptron (MLP), which takes v_{t+1} as input and produces the stock forecast.

$$\hat{y}_{t+1} = \text{softmax}(\mathbf{W}_v v_{t+1} + \mathbf{v}), \tag{11}$$

where we used *softmax* as the activation function

$$\text{softmax}(x)_i = \frac{\exp(x_i)}{\sum_j \exp(x_j)}. \tag{12}$$

We used the mean squared error as the loss function, given by

$$J = \frac{1}{N}\sum_{i=1}^{N}(y_{t+1} - \hat{y}_{t+1})^2. \tag{13}$$

Algorithm 1 describes our entire algorithmic training pipeline.

Algorithm 1 Our model

```
Input: News corpus $C$, technical factors $F$
Output: Prediction on future $\hat{y}_i^{t+1}$
1: Variables: Stock embedding matrix $\mathbf{E}$, stock attention parameters $\alpha_{i,j}$, technical embedding matrix $W_{emb}$ and $b_{emb}$, BiLSTM parameters $\theta$, temporal attention parameters $\beta_i$.
2: repeat
3:     Use the GLOVE loss function (Eq.1) to initial stock embedding matrix $\mathbf{E}$.
4: until convergence
5: repeat
6:     $s_i \leftarrow$ stock $i$ from universe
7:     for time stamp $t$ do
8:         for stock $j$ in $A_i$ do
9:             Obtain the $\alpha_{i,j}$ with Eq. 5
10:        end for
11:        Calculate the stock $i$'s vector $c_i$ and update $\mathbf{E}$
12:        Generate technical vector $g_i^t$ using Eq. 7
13:        Generate news embedding $o_i^t$ by leveraging Word2vec
14:        Obtain final representation $h_i^t$ with Eq. 8
15:        Get $v_{t-T}, ..., v_{t-1}, v_t$ by Eq. 9
16:        Forecast future return $\hat{y}_i^{t+1}$ using Eq. 10 and Eq. 11
17:    end for
18:    Calculate prediction loss $J$ by Eq.13
19:    Update parameters based on the gradient of $J$
20: until convergence
```

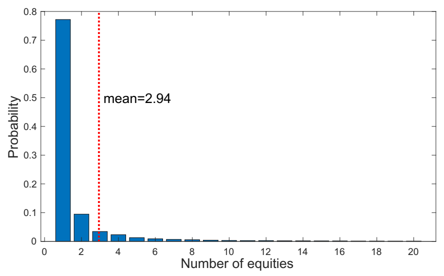


Fig. 2. Histogram for number of related stocks per news.

3.5 Design choices

Finally, we explain the major design choices in our model.

Stock graph: leveraging cross-sectional signals. Trading on cross-sectional signals (i.e., when Google goes up, Facebook is more likely to go up) is remarkably difficult because we need to examine all possible lead-lag interactions between all possible pairs of stocks (and their features). There are approximately $3000\times3000 = 1$ million interactions, so we face both compute and statistical challenges in controlling the false discoveries rate.

We circumvent the problem by building a stock graph, which enables us to identify similar stocks accurately. We then leverage news articles that mention multiple stocks in the news module to detect correlations between stocks' prices. Detecting co-movements by news appears to be much more effective than existing methods.

Technical factors: hand-built features are more effective. We note that features extracted by deep learing [12, 16, 31, 45, 53] are often less effective than features (technical factors) crafted by financial professionals [7]. Thus, we overlay an LSTM over technical factors so that we can simultaneously use expertise from financial professionals and extract serial correlations from deep learning models.

WORD2VEC: less is more. We pre-train WORD2VEC on our news dataset. No extra fine-tuning is needed.

Fundamental data: it only adds noises. Because our forecasting horizon is only a few days or a week (e.g., the response is the next 5-day return), a company's fundamental data (e.g., its earning, cashflow, book to price ratio, etc.) has only a minimum impact. We find that excluding all the fundamental data will improve our model's generalization error.

4 EXPERIMENTS

We now evaluate the methodology proposed in Sec. 3. We focus on the Chinese market. Its market value is the second largest (by value) in the world. We describe our dataset in Sec. 4.1, experimental settings in Sec. 4.2, evaluation metrics in Sec. 4.3, and benchmark in Sec. 4.4. We analyze trading simulation results in Sec. 4.5. Finally, we discuss the interpretability issue in Sec. 4.6

4.1 Data Collection

Our dataset consists of market and news data from Sina finance[1] for approximately four years.

[1]https://finance.sina.com.cn

Chinese equity market. The Chinese market consists of approximately 3500 stocks. We use standard ways to remove illiquid and excessively small (in capital) ones from the training set. Our data consists of $n = 2438$ stocks. The data contains standard minute-level bar-charts (open, high, low, close, volume) between 20150101 and 20180830. We use the open price (at 9:30 am) to compute returns at a daily frequency.

News data set. We crawled all the financial news between 20150101 and 20180830 from a major Chinese news website Sina. It has a total number of 2.6 million articles. Each article can refer to one or multiple stocks (see Figure 2). The timestamps of news published online are usually unreliable (the dates are reliable but the hour or minute information is usually inaccurate). We use news signal on the next trading day or later to avoid looking ahead issues.

4.2 Experimental settings

We implement our framework and baselines on `TensorFlow` [1]. Our response is *next 5-day returns*. Below discusses details for each component: *(i) The stock graph.* The hyper-parameters of the stock graph is $d = 32$ (dimension of embedding) and $k = 5$ (number of nearest neighbors to use). The choice of these parameters are determined from training set and our results are not sensitive to the hyper-parameters. *(ii) Technical factors.* We implement a large number of technical factors available in the literature [7]. *(iii) Word2vec* We use the standard way to process news corpus, and tokenize each news, and remove the stopwords, punctuation, and URLs. We also exclude the words that appear less than ten times. After defining the word vocabulary, we train CBOW Word2vec [22] based on news from the training data set to obtain the word level embeddings. The dimension of embedding is 400.

Training and testing data. We split our data set into a training set over the period of 20150101 and 20170820, and a testing set over the interval from 20170830 to 20180830. We consider a 10-day gap between the training and testing periods to avoid any look-ahead issues. We randomly sample 20% training data set as the validation set to tune the hyper-parameters, such as the number of epochs.

4.3 Evaluation metrics

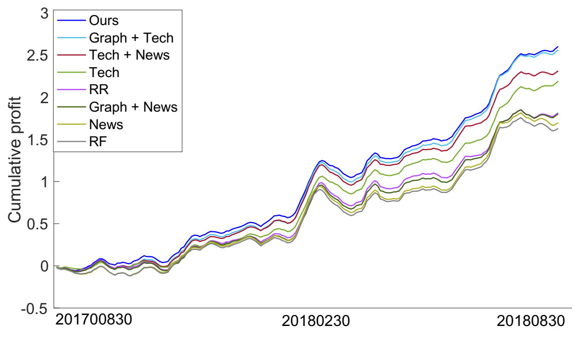
We evaluate our performance by standard metrics in the literature such R^2, Sharp ratio (SH) and P&L.

Coefficient of determination. R^2 [47] was used to evaluate the predictive performance of regression and forecasting algorithms. It compares the accuracy of the prediction regarding the simple prediction by mean of the target variable. Assuming n is the number of samples, y_i is the observed return being predicted, and $\hat{y}_i$ is the estimated value, the R^2 is given by

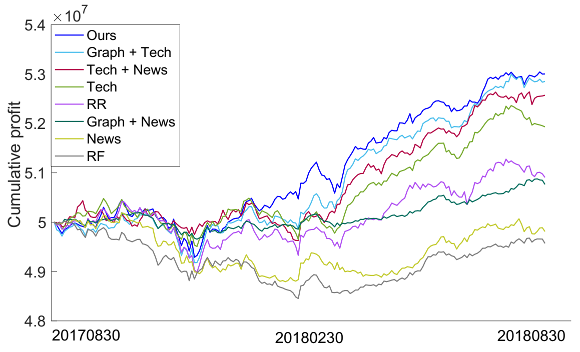
$$R^2 = 1 - \frac{\sum_i^n (y_i - \hat{y}_i)^2}{\sum_i^n (y_i - \bar{y}_i)^2}. \tag{14}$$

Sharpe ratio. In finance, the Sharpe ratio (SR) [15] is a popular measure of performance of an investment by adjusting for its risk

$$\text{Sharpe ratio} = \frac{R_p - R_f}{\sigma_p}, \tag{15}$$



(a) The profit curve from our straightforward simulation



(b) The profit curve from our realistic simulator

Fig. 3. The cumulative profit curve from different simulators with the portfolio of full universe.

where R_p is the return of portfolio, R_f is the risk-free rate, and σ_p is the standard deviation of the portfolio's excess return.

P&L. Profit & Loss (or P&L) [41] is a standard performance measure used in trading, and captures the total profit or loss of a portfolio over a specified period. The P&L of all forecasts made on day t is given by

$$\text{P\&L}_t = \sum_{i=1}^{n} \hat{y}_i^t - y_i^t, \quad t = 1, \ldots, M, \tag{16}$$

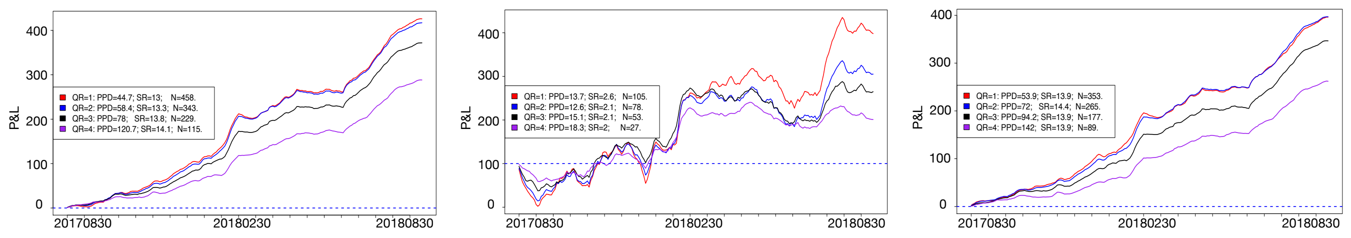
where M denotes the number of trading days.

4.4 Baselines for comparison

To test our proposed deep learning framework, we compare our model against several baselines as described below. For all the baselines, we use the validation dataset to configure the hyper-parameters.

Random Forest (RF). We use the Random Forest [27] regression, where the number of trees is 200. The input is given by the concatenation of the daily vectors (the news vector, the technical vector, and the corresponding stock initial embedding) from the previous 5 days.

Ridge Regression (RR). We use ridge regression [51] with the same input above as for the Random Forest. We scanned different



(a) Quantile portfolio P&L curve for the full universe. (b) Quantile portfolio P&L curve for the *long-only* universe. (c) Quantile portfolio P&L curve for the *short-only* universe.

Fig. 4. The cumulative profit curve from the quantile portfolios, against future market excess returns, for the full universe (a), long-only (b), and short-only (c). QR denotes the quantile rank threshold used corresponding to the quantile portfolios; PPD denotes the P&L in basis points per dollar traded; SR is the Sharpe Ratio, and N is the size of the corresponding quantile portfolios.

regularization strength parameters, ranging from 1e-5 to 10, and report the best one on the validation set.

News-BiLSTM (News). We use the bi-directional LSTM with the same setup as ours, but remove the Stock2vec and technical module and use the historical news vectors to exam the prediction power by from news. This is to compare our method with news based studies.

Technical-BiLSTM (Tech) We use a bi-directional LSTM on technical factors without the graph embedding and the news input. This is to compare our method with the technique that makes forecast by historical time-series technical factors.

Technical + news (Tech + news) We use the same setup from our model but remove the Stock2vec module to examine whether our Stock2vec capture additional signal to boost the performance.

Graph + Technical (Graph + Tech) We use a bi-directional LSTM on both the Stock2vec and technical factors without the input from the news embedding.

Graph + news We use a bi-directional LSTM on both the Stock2vec and the news embedding without the input from the technical factors.

Models	R^2_{in}	R^2_{out}	SR_1	$P\&L_1$	SR_2	$P\&L_2$
Ours	0.0309	0.0123	**9.7531**	**3.9223**	**3.6682**	**0.0835**
Graph + Tech	0.0311	0.0122	8.9076	3.9097	2.9732	0.077
Tech + News	0.0400	**0.0126**	8.8067	3.4027	3.013	0.0745
Tech	0.0297	0.0096	8.4359	3.4849	1.9382	0.0459
RR	**0.0488**	0.0064	6.5888	2.7693	1.4584	0.0379
Graph + News	0.0138	0.0029	5.6946	2.5304	2.4014	0.029
News	0.0182	0.0034	5.8083	2.6198	0.7224	0.0123
RF	0.0419	0.0049	6.3885	2.7788	0.0709	0.0015

Table 1. Overall performance

4.5 Performance and Discussion

4.5.1 Overall performance on evaluation metrics. Table 1 reports the forecasting power for the *out-of-sample* period from August 30, 2017 to August 30, 2018. The SH_1 and $P\&L_1$ are generated from a straightforward simulator, and SH_2 and $P\&L_2$ are computed by a more realistic simulator. We will introduce these two simulators in the next paragraph. We observe that: *(i) Small in-sample and out-of-sample gap:* our model has low overfitting, with smaller generalization error than the ridge regression. *(ii) High forecasting power:* our method yields the best performance in terms of Sharpe Ratio and P&L, reflecting that our proposed pipeline can realize appreciable profits when trading in the markets. *(iii) Effect of cross-categorical learning:* the model using Stock2vec achieves higher performance than the baselines without it, thus illustrating its ability to capture and leverage latent stock interactions. *(iv) The significance of our framework:* despite the fact RR and RF have the same input as ours, they have worse performance, indicating the use of RNN based sequence model improves the prediction.

4.5.2 Market Trading Simulation. We simulate investments on our signals using two ways. *(i) Standard long-short portfolio (i.e., straightforward simulation).* The position of each stock is proportional to the signal (i.e., dollar position of i-th stock is proportional to our forecast $\hat{r}^t_i$). The holding period is 5 days. Note that we allow short-selling, meaning that we can execute on negative forecasts. Chinese market does not allow short-selling. Thus, the purpose of the simulation is to understand the overall forecasting quality of the signals we produce. See Figure. 3(a) for the P&L. *(ii) "Realistic" simulator.* We use standard futures contracts to hedge the market and use Markowitz portfolio optimization [33] to find the positions. Then we simulate the trades by using standard methods that take into account of liquidity and market impacts [44]. Our initial portfolio value is 50 million CNY. This setup realistically reflects the challenges of trading our signals in the market. See Figure. 3(b).

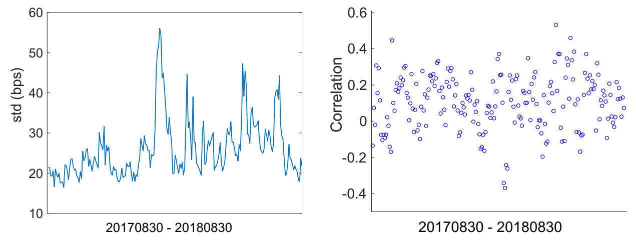
We can see that (i) The performance of all the signals (including baselines) are weaker in the realistic simulator. This is unsurprising because monetization of short signals in China is difficult, and (ii) Our signals are consistently better than other baselines in both long/short and realistic simulations, suggesting that our method generates stronger and more robust signals for trading.

4.5.3 Quantile analysis. To better highlight the interpretability and reproducibility of the results, we also compute a straightforward markup/evaluation of the forecasts against future raw returns and market excess returns, without relying on the Markowitz portfolio optimizer. To this end, we assess performance by taking into account the magnitude of the forecasts. In particular, our quantile-based analysis aims to asses the performance of a portfolio of stocks, conditioned on the signal strength falling in a certain quantile bucket.

In other words, we ask the question: *What are the portfolios performance statistics, if we only trade the top q% strongest in magnitude signals?* In the plots of Figure 4, we plot performance statistics, for different values of quantile rank (QR) thresholds used to define quantile portfolios. The colors denote the fraction of stocks traded, based on their magnitude (e.g., the $QR = 4$ purple curves denote that we only consider the top 75% largest in magnitude forecasts, and ignore the bottom 25%), and are usually referred to as *quantile* portfolios in the financial literature [14].

We clarify that the quantile buckets are not disjoint, but cumulative, i.e., the red curves (with $QR = 1$) correspond to trading the full universe of n stocks, while the remaining blue, black, respectively, purple, P&L curves comprise of only 0.75%, 0.50%, respectively, 0.25% of the universe of stocks. Furthermore, we assume the notional amount allocated to each individual instrument to be fixed to $1, independent of the magnitude of its associated signal. We denote by PPD the average, P&L per dollar traded, and show the resulting numerical values in the legend of Figure 4, for each quantile portfolio. The results showcase a strong association between the magnitude of the signal and future returns, with the PPD values for QR thresholds $1-4$ being approximately 45, 58, 78, and 120 bps, respectively.

4.5.4 Characteristics of the portfolio. We next describe the characteristics of the portfolio constructed using our model's forecast. Figure. 5(a) shows the distribution of the standard deviation of our returns at daily granularity from the forecast of the full universe. This measures the "aggressiveness" of our forecasts, meaning that whenever the forecasts have high standard deviation, they incentivize the optimizer to take on larger positions. We note that while the standard deviations are time-varying, our forecasts rarely bet more than $60bps$. We construe this as a piece of evidence that the forecast picks up the market structure from training. Figure 5(b) shows plots of the daily correlation from the testing period. One can see that a large fraction of our daily forecasts have positive correlations with the stock movement.



(a) Standard deviation for our prediction (b) The out-of-sample daily correlation

Fig. 5. Statistic from our predicted returns. The x-axis is the testing duration. 1 basis point (bps) is one hundredth of one percent.

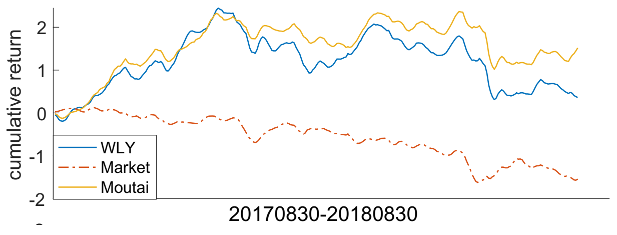
4.6 Interpretation analysis

In this section, we assess the interpretability for stock relationships, temporal weights, and news as follows.

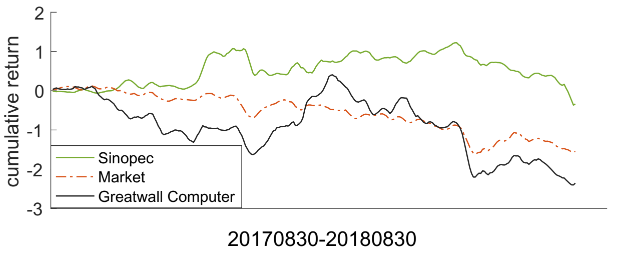
4.6.1 Stock2vec interpretation.

Insight similarity distance From the Stock2vec, we obtain the 32-dimensional vector for each stock. To reveal the insights from similarity distance, we sort the pair-wise distance between the stocks in ascending order. We sample two pairs of stocks from the first 1% percentile and last 1% percentile and show their cumulative return in the testing duration. In Figure 6(a), WLY is Wuliangye Yibin Company and Moutai is Kweichow Moutai company. Both companies are famous in liquid supplement. We can see that two stocks exhibit quite similar trends for cumulative return. In Figure 6(b), Sinopec is one of the biggest petrol company, and Greatwall computer is an IT company. It shows that two stocks with big distance are in opposite trends.

Visualizing learned stock embedding. As depicted in Figure 7(a), we use a two-dimensional t-SNE embedding [32] from our final stock representation to assess the interpretation of the universe of stocks. Each dot represents a stock, and the color denotes the largest sector from Barra [13] associated with the given stock, while the text annotations represent the detailed stock categories. For comparison, Figure 7(b) shows the t-SNE scatterplot from GloVe, with the same t-SNE parameters. It confirms our intuition that an interpretable representation cannot simply be learned only by running GloVe on the co-occurrence matrix, while our end-to-end model learns the interpretable stock representations that are well aligned with the Barra sectors in the Chinese market.



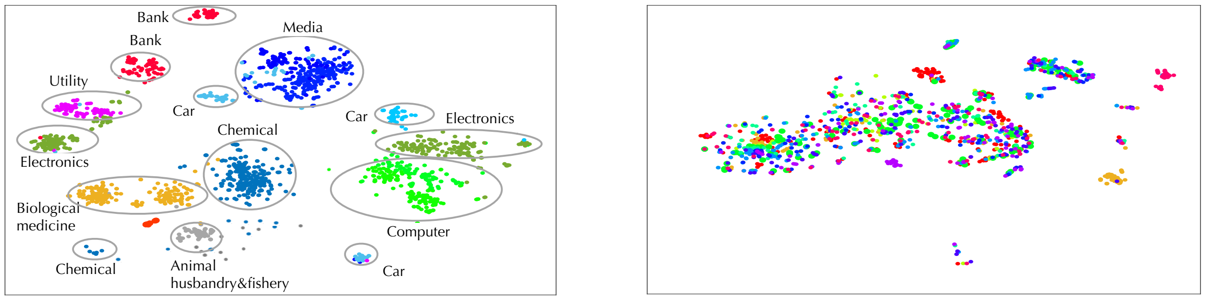
(a) Stocks in close proximity



(b) Two far away stocks

Fig. 6. Cumulative returns from testing duration. The red line is the market return.

4.6.2 Interpretation of the stock feature importance. The dimension of technical factor f_i^t is 320. As shown in Eq. 7, we embed f_i^t into a lower-dimensional (m) space. In our experiment, we $l = 320$ and $m = 200$. Note that the i-th row from W_{emb} will be the representation of the i-th factor among total l factors. However, directly interpret W_{emb} will be challenging since W_{emb} has both positive



(a) Scatterplot of final stock representations

(b) Scatterplot of stock vectors learned by GloVe

Fig. 7. t-SNE scatterplots of stock by our method and Glove

Factors	Description
ROE_TTM	Return on Equity [8]
HAlpha_12m	Regression on last one year's daily return [7]
Amihuds_1y	Last year's exponential weighted Momentum [3]
Vol_regress_4m	Volume regression on the last 4 month [19]
SRMI	Modified Momentum [18]

Table 2. The top five most frequent technical factors.

粤传媒: 2017年度资产盈利预测 (Annual Earning projection/ forecast)
合力泰: 增持公司股份公告 (Increase company holding)
雅致股份: 净收入与净利润增长分析 (Profit express and analysis)
天马股份:重大事项停盘 (Important event and trade suspension)
东方明珠:董事即将回购部分股权 (Buyback equity /share repurchase)
[High accurate: important events]

经济日报: 18年我国经济结构持续优化 (Good domestic economic structure)
中投研究: 2018全球宏观前景怎么看 (Global Economic Analysis)
航发科技: 公司举行迎新年升旗仪式 (New Year flag-raising ceremony)
[Low accurate: not predictive news]

Fig. 8. The demonstrative news from high accurate and low accurate performance.

and negative values. Instead of trian W_{emb}, we followed [6] to train $W'_{emb} = ReLU(W_{emb})$ to avoid the negative weights. We can find the top k factors with the largest values for the i^{th} coordinate of the embedding space as follows

$$\text{argsort}(W'_{emb}[i,:])[1 : k_{emb}], \tag{17}$$

where we sort value in descending order and argsort returns the indices of a vector. We set $k_{emb} = 50$, extract the top five most frequent technical factors, and show the factors in Table 2.

4.6.3 News Interpretation. To understand the news predictive ability, we track back the news from two groups in the test dataset. We focus the samples within the smallest 5% and the samples within the largest 5% errors based on Eq. 13. We extract the corresponding news and show the detailed results in Figure 8. We focus on the demonstrative news from these two groups of samples. One can see that the news in the high accurate group contains significant events with predictive ability while the news in the low accurate group mainly has no obvious influence on the stock forecast.

4.6.4 Temporal attention explanation. Next, we show the overall attention weights from the testing data set in Table 3. The recent days have larger weights indicating the recent days play more significant roles in the prediction.

	-5day	-4day	-3day	-2day	-1day
Weights	0.0055	0.0265	0.1662	0.3064	0.4954

Table 3. The temporal overall attention weights.

5 CONCLUSION AND FUTURE WORK

This paper presents an end-to-end deep learning model to answer two research questions. *(i)* How can we build an end-to-end machine learning pipeline that leverages heterogeneous data sources to extract high-quality alpha? *(ii)* How can we interpret the relationship between stocks? Through extensive evaluation against seven baselines, we confirm that our method achieves superior performance with low overfitting. Meanwhile, the results from different trading simulators demonstrate that we can effectively monetize the signals. In addition, we interpret the stock relationships highlighting they align well with the sectors defined by commercial risk models, extract important technical factors, and explain what kind of news has more predictive power.

We identify several potential future directions. First, it is worth to explore more effective features from social media such as financial discussion forums. As individual investors often engage in insightful discussions on finance topics and stock movements, the large volume of such discussions could indicate potential upcoming major events. Second, deep learning models seem to have severe overfitting problems when we supply fundamental factors. It remains open what new ML techniques are needed to extract fundamental signals.

6 ACKNOWLEDGEMENT

MC and AP acknowledge support from The Alan Turing Institute EPSRC grant EP/N510129/1. AP also acknowledges support from the National Group of Mathematical Physics (GNFM-INdAM), and by the EPSRC grant EP/P002625/1.

REFERENCES

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, et al. 2016. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467* (2016).

[2] Ryo Akita, Akira Yoshihara, Takashi Matsubara, and Kuniaki Uehara. 2016. Deep learning for stock prediction using numerical and textual information. In *2016 IEEE/ACIS 15th International Conference on Computer and Information Science (ICIS)*. IEEE, 1–6.

[3] Yakov Amihud. 2002. Illiquidity and stock returns: cross-section and time-series effects. *Journal of financial markets* 5, 1 (2002), 31–56.

[4] Johan Bollen, Huina Mao, and Xiaojun Zeng. 2011. Twitter mood predicts the stock market. *Journal of computational science* 2, 1 (2011), 1–8.

[5] John Y Campbell, John J Champbell, John W Campbell, Andrew W Lo, Andrew W Lo, and A Craig MacKinlay. 1997. *The econometrics of financial markets*. princeton University press.

[6] Edward Choi, Mohammad Taha Bahadori, Elizabeth Searles, Catherine Coffey, Michael Thompson, James Bost, Javier Tejedor-Sojo, and Jimeng Sun. 2016. Multi-layer representation learning for medical concepts. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 1495–1504.

[7] Robert W Colby and Thomas A Meyers. 1988. *The encyclopedia of technical market indicators*. Dow Jones-Irwin Homewood, IL.

[8] Aswath Damodaran. 2007. Return on capital (ROC), return on invested capital (ROIC) and return on equity (ROE): Measurement and implications. *Return on Invested Capital (ROIC) and Return on Equity (ROE): Measurement and Implications (July 2007)* (2007).

[9] Marcos Lopez De Prado. 2018. *Advances in financial machine learning*. John Wiley & Sons.

[10] Xiao Ding, Yue Zhang, Ting Liu, and Junwen Duan. 2015. Deep learning for event-driven stock prediction. In *Twenty-fourth international joint conference on artificial intelligence*.

[11] Yi Ding, Weiqing Liu, Jiang Bian, Daoqiang Zhang, and Tie-Yan Liu. 2018. Investor-imitator: A framework for trading knowledge extraction. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 1310–1319.

[12] Jonathan Doering, Michael Fairbank, and Sheri Markose. 2017. Convolutional neural networks applied to high-frequency market microstructure forecasting. In *2017 9th Computer Science and Electronic Engineering (CEEC)*. IEEE, 31–36.

[13] Frank J Fabozzi and Harry M Markowitz. 2011. *The theory and practice of investment management: Asset Allocation, Valuation, Portfolio Construction, and Strategies*. Vol. 198. John Wiley & Sons.

[14] Eugene F Fama and Kenneth R French. 1992. The Cross-Section of Expected Stock Returns. *Journal of Finance* 47, 2 (June 1992), 427–65. https://ideas.repec.org/a/bla/jfinan/v47y1992i2p427-65.html

[15] Bruce J Feibel. 2003. *Investment performance measurement*. Vol. 116. John Wiley & Sons.

[16] Fuli Feng, Huimin Chen, Xiangnan He, Ji Ding, Maosong Sun, and Tat-Seng Chua. [n.d.]. Enhancing Stock Movement Prediction with Adversarial Training. ([n. d.]).

[17] Guanhao Feng, Jingyu He, and Nicholas G Polson. 2018. Deep learning for predicting asset returns. *arXiv preprint arXiv:1804.09314* (2018).

[18] Tak-Chung Fu, Chi-Pang Chung, and Fu-Lai Chung. 2013. Adopting genetic algorithms for technical analysis and portfolio management. *Computers & Mathematics with Applications* 66, 10 (2013), 1743–1757.

[19] Ramazan Gencay and Thanasis Stengos. 1998. Moving average rules, volume and the predictability of security returns with feedforward networks. *Journal of Forecasting* 17, 5-6 (1998), 401–414.

[20] Felix A Gers, Jürgen Schmidhuber, and Fred Cummins. 1999. Learning to forget: Continual prediction with LSTM. (1999).

[21] Mustafa Göçken, Mehmet Özçalıcı, Aslı Boru, and Ayşe Tuğba Dosdoğru. 2016. Integrating metaheuristics and artificial neural networks for improved stock price prediction. *Expert Systems with Applications* 44 (2016), 320–331.

[22] Yoav Goldberg and Omer Levy. 2014. word2vec Explained: deriving Mikolov et al.'s negative-sampling word-embedding method. *arXiv preprint arXiv:1402.3722* (2014).

[23] Richard C Grinold and Ronald N Kahn. 2000. Active portfolio management. (2000).

[24] Shihao Gu, Bryan Kelly, and Dacheng Xiu. 2018. *Empirical asset pricing via machine learning*. Technical Report. National Bureau of Economic Research.

[25] JB Heaton, NG Polson, and Jan Hendrik Witte. 2017. Deep learning for finance: deep portfolios. *Applied Stochastic Models in Business and Industry* 33, 1 (2017), 3–12.

[26] Frederick SM Herz, Lyle H Ungar, Jason M Eisner, and Walter Paul Labys. 2012. Stock market prediction using natural language processing. US Patent 8,285,619.

[27] Tin Kam Ho. 1995. Random decision forests. In *Proceedings of 3rd international conference on document analysis and recognition*, Vol. 1. IEEE, 278–282.

[28] Ziniu Hu, Weiqing Liu, Jiang Bian, Xuanzhe Liu, and Tie-Yan Liu. 2018. Listening to chaotic whispers: A deep learning framework for news-oriented stock trend prediction. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*. ACM, 261–269.

[29] Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional LSTM-CRF models for sequence tagging. *arXiv preprint arXiv:1508.01991* (2015).

[30] Antti Ilmanen. 2011. *Expected returns: An investor's guide to harvesting market rewards*. John Wiley & Sons.

[31] Tao Lin, Tian Guo, and Karl Aberer. 2017. Hybrid neural networks for learning the trend in time series. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence*. 2273–2279.

[32] Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of machine learning research* 9, Nov (2008), 2579–2605.

[33] Harry Markowitz. 1952. Portfolio selection. *The journal of finance* 7, 1 (1952), 77–91.

[34] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* (2013).

[35] Tomas Mikolov, Kai Chen, Gregory S Corrado, and Jeffrey A Dean. 2015. Computing numeric representations of words in a high-dimensional space. US Patent 9,037,464.

[36] Tomáš Mikolov, Martin Karafiát, Lukáš Burget, Jan Černockỳ, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *Eleventh annual conference of the international speech communication association*.

[37] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*. 3111–3119.

[38] Felix Ming, Fai Wong, Zhenming Liu, and Mung Chiang. 2014. Stock market prediction from WSJ: text mining via sparse matrix factorization. In *2014 IEEE International Conference on Data Mining*. IEEE, 430–439.

[39] Marc-Andre Mittermayer and Gerhard F Knolmayer. 2006. Newscats: A news categorization and trading system. In *Sixth International Conference on Data Mining (ICDM'06)*. Ieee, 1002–1007.

[40] Venkata Sasank Pagolu, Kamal Nayan Reddy, Ganapati Panda, and Babita Majhi. 2016. Sentiment analysis of Twitter data for predicting stock market movements. In *2016 international conference on signal processing, communication, power and embedded system (SCOPES)*. IEEE, 1345–1350.

[41] Julien Pantz. 2013. PnL prediction under extreme scenarios. *Available at SSRN 2281873* (2013).

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

[43] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 1532–1543.

[44] pyfolio. [n.d.]. Slippage Analysis. http://quantopian.github.io/pyfolio/notebooks/slippage_example/.

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

[46] Mike Schuster and Kuldip K Paliwal. 1997. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing* 45, 11 (1997), 2673–2681.

[47] Robert George Douglas Steel, James Hiram Torrie, et al. 1960. Principles and procedures of statistics. *Principles and procedures of statistics.* (1960).

[48] James H Stock and Mark Watson. 2011. Dynamic factor models. *Oxford Handbooks Online* (2011).

[49] Andrew Sun, Michael Lachanski, and Frank J Fabozzi. 2016. Trade the tweet: Social media text mining and sparse matrix factorization for stock market prediction. *International Review of Financial Analysis* 48 (2016), 272–281.

[50] Jonathan L Ticknor. 2013. A Bayesian regularized artificial neural network for stock market forecasting. *Expert Systems with Applications* 40, 14 (2013), 5501–5506.

[51] S Twomey. 1963. On the numerical solution of Fredholm integral equations of the first kind by the inversion of the linear system produced by quadrature. *Journal of the ACM (JACM)* 10, 1 (1963), 97–101.

[52] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*. 5998–6008.

[53] Jia Wang, Tong Sun, Benyuan Liu, Yu Cao, and Hongwei Zhu. 2019. CLVSA: A Convolutional LSTM Based Variational Sequence-to-Sequence Model with Attention for Predicting Trends of Financial Markets. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*. International Joint Conferences on Artificial Intelligence Organization, 3705–3711. https://doi.org/10.24963/ijcai.2019/514

[54] Qiong Wu, Felix Ming Fai Wong, Zhenming Liu, Yanhua Li, and Varun Kanade. 2019. Adaptive Reduced Rank Regression. *arXiv preprint arXiv:1905.11566* (2019).

[55] Frank Z Xing, Erik Cambria, and Roy E Welsch. 2018. Natural language based financial forecasting: a survey. *Artificial Intelligence Review* 50, 1 (2018), 49–73.
[56] Liheng Zhang, Charu Aggarwal, and Guo-Jun Qi. 2017. Stock price prediction via discovering multi-frequency trading patterns. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*. ACM, 2141–2149.