Accurate Stock Movement Prediction with Self-supervised Learning from Sparse Noisy Tweets

Yejun Soun[1,2,*], Jaemin Yoo[3,*], Minyong Cho[4], Jihyeong Jeon[1,2], U Kang[1]

[1]*Seoul National University*, [2]*DeepTrade*, [3]*Carnegie Mellon University*, [4]*Deeping Source*
sony7819@snu.ac.kr, jaeminyoo@cmu.edu, chominyong@gmail.com, {jeonjihyeong, ukang}@snu.ac.kr

***Abstract*—Given historical stock prices and sparse tweets, how can we accurately predict stock price movement? Many market analysts strive to use a large amount of information for stock price prediction, and Twitter is one of the richest sources of information presenting real-time opinions of people. However, previous works that use tweet data in stock movement prediction have suffered from two limitations. First, the number of tweets is heavily biased towards only a few popular stocks, and most stocks have insufficient evidence for accurate price prediction. Second, many tweets provide noisy information irrelevant of actual price movement, and extracting reliable information from tweets is as challenging as predicting stock prices.**

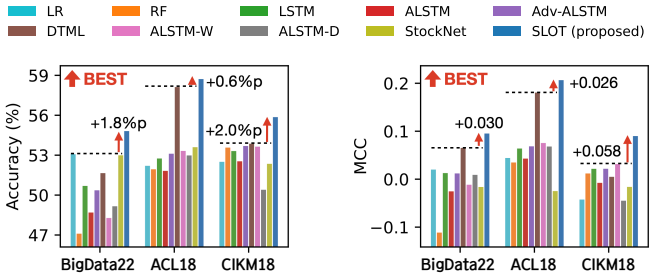
In this paper, we propose SLOT (Self-supervised Learning of Tweets for Capturing Multi-level Price Trends), an accurate method for stock movement prediction. SLOT has two main ideas to address the limitations of previous tweet-based models. First, SLOT learns embedding vectors of stocks and tweets in the same semantic space through self-supervised learning. The embeddings allow us to use all available tweets to improve the prediction for even unpopular stocks, addressing the sparsity problem. Second, SLOT learns multi-level relationships between stocks from tweets, rather than using them as direct evidence for prediction, making it robust to the unreliability of tweets. Extensive experiments on real world datasets show that SLOT provides the state-of-the-art accuracy of stock movement prediction.

***Index Terms*—stock price movement prediction, self-supervised learning, Twitter, attention LSTM, time series forecasting**

I. Introduction

Given historical stock prices and sparse tweets, how can we accurately predict the stocks' price movement? Stock price prediction is an important task that has attracted increasing attention in data mining and machine learning communities [1]–[6]. An accurate prediction is challenging due to the random and noisy nature of stock markets, but can result in enormous profit of investment. We formulate the problem as binary classification of stock price movement into a rise or a fall, rather than forecasting their exact values, since it makes the problem more tractable while maintaining the predictive power that we are interested in [1], [2], [7].

Previous works on stock movement prediction are categorized into three groups based on the type of source information. The first group [1], [2], [7] uses only price information for prediction, focusing on finding patterns from historical prices. However, they give limited performance since they do not use rich information from other sources such as news or tweets.



(a) Accuracy (b) MCC

Fig. 1: The accuracy and MCC of our proposed SLOT and baseline approaches in real datasets. SLOT consistently gives the best performance in all datasets and metrics, thanks to its consideration of multi-level price trends through self-supervised learning of tweets.

The second group [8], [9] utilizes news data for prediction. News articles provide formal and reliable information, but their information spreads more slowly than in social media, where people share their real-time opinions. The last group [4], [10] utilizes tweet data to get timely information, but there are two limitations in tweet data that prevent existing approaches from getting useful information: *sparsity* and *unreliability*.

The problem of sparsity comes from the biased distribution of the number of tweets that mention each stock. Most tweets focus only on a few popular stocks such as AAPL or GOOG, while most of the stocks have an insufficient number of tweets. The problem of unreliability comes from the characteristic of Twitter, where any user can post unconfirmed information about the market. It is not safe to rely completely on the contents of tweets to extract signals predictive of the market movement, without considering the risk of getting wrong information.

In this paper, we propose SLOT (Self-supervised Learning of Tweets for Capturing Multi-level Price Trends), an accurate method for stock movement prediction which utilizes sparse noisy tweets to extract multi-level patterns from historical prices. SLOT addresses the two limitations of tweet-based models with the following ideas. First, SLOT learns latent representations of stocks and tweets in the same vector space through self-supervised learning. This allows us to utilize any tweet for any stock based on the distance in the embedding

*Equal contribution

space, avoiding the problem of sparsity. Second, SLOT does not use tweets directly for the prediction of price movement; SLOT rather utilizes tweets to capture the global market trend and to find the local correlations between stocks. This makes SLOT robust to the unreliability of tweets while utilizing the timely information provided by a collection of tweets.

We summarize our main contributions as follows:

- **Self-supervised learning for sparse tweets.** SLOT uses a masked language model to learn tweet and stock embeddings in a self-supervised way. This allows unpopular stocks to use all available tweets based on the embedding distance, alleviating the sparsity problem of tweets.
- **Capturing multi-level tweet trends.** SLOT uses tweets to understand the multi-level correlations between stocks in global and local views, rather than as direct evidence for prediction. This allows SLOT to avoid the unreliability problem of tweets by focusing on the occurrences of stocks instead of the sentiment of tweets.
- **Experiments.** Extensive experiments show that SLOT provides the best accuracy in stock movement prediction with two types of metrics, outperforming competitors by significant margins in real datasets (in Figure 1). We also perform qualitative analysis on the learned embeddings, revealing the relationships between target stocks.

The rest of this paper is organized as follows. We present preliminaries and review related works on stock movement prediction in Section II. We perform empirical studies on the properties of public tweet data and then propose our method SLOT in Section III. We present experimental results on real-world stock datasets in Section IV and conclude at Section V. The datasets used in our experiments are available at https://github.com/deeptrade-public/slot.

II. Preliminaries and Related Works

We describe the problem definition, preliminaries, and related works on stock movement prediction.

A. Problem Definition

We formally define the problem of stock movement prediction as follows. We have a set $\mathcal{S}$ of target stocks which we aim to predict and a set $\{\mathbf{x}_{st}\}_{s\in\mathcal{S},t\in\mathcal{T}}$ of feature vectors that summarize historical prices, where $\mathcal{T}$ is the set of available training days. The historical prices consist of the opening, the highest, the lowest, and the closing prices of each stock (details in Section III-B). We also have a set $\mathcal{E}$ of tweets, each of which mentions at least one stock in $\mathcal{S}$. Then, the problem is to predict the binary movement of the price of each stock at day $T+1$, given the features and tweets until day T. This problem definition is a generalization of the problems studied in previous works for technical prediction [1], [2], which use only the historical prices for stock movement prediction.

B. Stock Movement Prediction

There are two main categories of previous works for stock movement prediction: a) methods using only historical prices for prediction and b) methods using additional text data such as news or tweets along with historical prices.

1) Methods with only price information

Many previous works assume that technical features made from historical prices provide sufficient information for stock movement prediction. They aim at finding meaningful patterns from historical prices that are often noisy but provide a useful evidence for prediction. Such *technical* models are used as the backbone of many complex models, such as text-based ones, that combine prices with other sources of information.

Many technical models are based on variants of recurrent neural networks (RNN), which can find complex patterns from historical prices [11]–[13]. Temporal attention has been used to improve the performance of RNN-based models by combining the information of multiple time steps [2], [14]. There are also multivariate approaches which exploit the relationships between different stocks [1], [3], [15], [16]. Lastly, there are recent works which aim to deal with the noise of stock prices with multi-frequency or multi-task learning [17]–[19].

The main limitation of such approaches is that they cannot predict price movements whose information is not given in the historical prices. The primary goal of our work in this paper is to outperform such technical approaches by utilizing tweet data in an effective way.

2) Methods with textual information

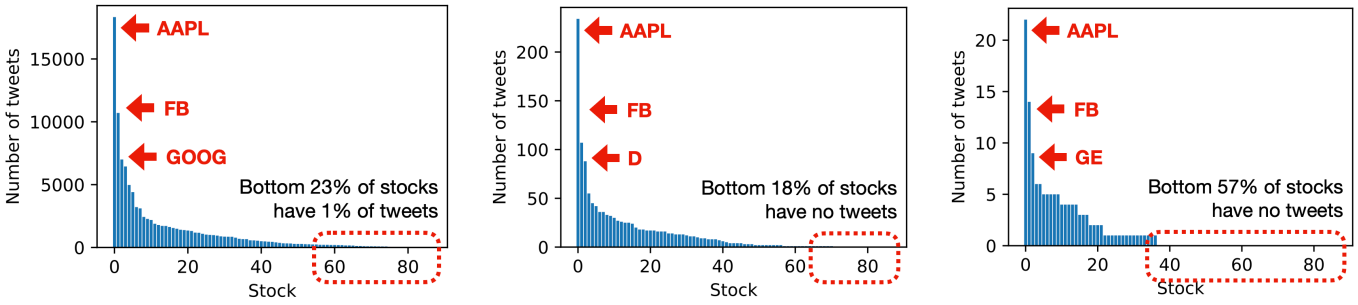
It is widely believed that information sources such as news, tweets, or financial reports give meaningful evidence for stock price prediction. Many previous works utilize textual data as additional information for stock movement prediction. Most of them focus on a few reliable sources of information such as news [20]–[27], company descriptions [28], [29], or stock reviews [30], [31]. However, such methods cannot always get timely information that precedes the movement of a market.

There are approaches focusing on online social networks such as Twitter, which provide abundant public opinions in a timely manner [4], [32]. However, previous works that use social network data have two notable limitations. First, they select only a few popular stocks that contain a sufficient amount of information as the target of prediction, ignoring unpopular stocks having few mentioned tweets. Second, they rely too much on the predictive power of such data, although many social services provide wrong information that is often irrelevant to the actual movement of stock prices.

In this work, we aim to address the limitations of previous approaches by designing a multivariate predictor that utilizes tweet data, considering their sparsity and noisiness at the same time. This results in the state-of-the-art performance of stock movement prediction by combining the strengths of technical and tweet-based models, as we present in Section IV.

C. Attention LSTM

Long short-term memory units (LSTM) is a deep neural network proposed to address the gradient vanishing problem of recurrent neural networks. LSTM has been widely used as the key component to capture temporal patterns of stock prices [4], [7], [10], [13]. LSTM takes a sequence of feature vectors



(a) All periods (Jan. 2014 to Dec. 2015) (b) A week (the first week in Oct. 2015) (c) A day (October 5, 2015)

Fig. 2: The number of tweets that mention each stock in the ACL18 dataset: (a) all periods in the dataset, (b) a week, and (c) a day. In (a), the top 6% of stocks including AAPL, FB, and GOOG have 50% of all available tweets, while the bottom 23% of stocks have only 1% of all tweets. The sparseness is worse in (b) and (c), where 18% and 57% of stocks have no tweets at all, respectively. These figures show the problem of sparsity, which is commonly observed in public tweet data.

as input and generates a hidden state vector for each time step. We represent LSTM simply as follows:

$$\mathbf{h}_1, \cdots, \mathbf{h}_{\text{last}} = \text{LSTM}(\mathbf{x}_1, \cdots, \mathbf{x}_{\text{last}}), \tag{1}$$

where $\mathbf{x}_i$ is a feature vector, and $\mathbf{h}_i$ is the hidden state for step i. LSTM typically uses the state $\mathbf{h}_{\text{last}}$ of the last time step for a classification task after consuming all input features.

Attention LSTM (ALSTM) improves LSTM by making direct connections between the output and the hidden states of all time steps using the attention mechanism [33]. Given a list $\{\mathbf{h}_1, \cdots, \mathbf{h}_{\text{last}}\}$ of hidden states generated from LSTM, we apply a single layer to each hidden state as $\tilde{\mathbf{h}}_i = \tanh(\mathbf{W}\mathbf{h}_i + \mathbf{b})$, where $\mathbf{W}$ and $\mathbf{b}$ are learnable parameters. Then, the states are combined by attention as follows:

$$\mathbf{h}_{\text{att}} = \sum_{i=1}^{\text{last}} \alpha_i \tilde{\mathbf{h}}_i \text{ where } \alpha_i = \frac{\mathbf{u}^\top \tilde{\mathbf{h}}_i}{\sum_{j=1}^{\text{last}} \mathbf{u}^\top \tilde{\mathbf{h}}_j}, \tag{2}$$

where $\mathbf{u}$ is a learnable parameter that is often called the *query* of attention. In other words, $\mathbf{u}$ selects the most relevant time steps based on the result of the dot product. The computed weight α_i shows how much step i is included in $\mathbf{h}_{\text{att}}$.

ALSTM generates the final output $\mathbf{h}_{\text{out}} = \mathbf{h}_{\text{last}} \parallel \mathbf{h}_{\text{att}}$ by concatenating the hidden state $\mathbf{h}_{\text{last}}$ of the last time step and the output $\mathbf{h}_{\text{att}}$ of the attention, where $\parallel$ is the concatenation operator between vectors. The last hidden state $\mathbf{h}_{\text{last}}$ is used in addition to $\mathbf{h}_{\text{att}}$ as the basic output of LSTM that summarizes all given features apart from the result of attention. Our SLOT utilizes ALSTM as the main module for processing historical prices due to its robust performance.

III. Proposed Method

We propose SLOT, an accurate method for stock movement prediction, which effectively combines historical prices with sparse noisy tweets. SLOT is designed to address the following challenges of stock movement prediction:

1) **Addressing the sparsity of tweets.** Despite an abundant number of available tweets, most of them mention only a few popular stocks. This leaves most stocks in a market to have an insufficient evidence for the prediction. How can we extract meaningful information from tweets for unpopular stocks?
2) **Capturing global trend.** The global trend of a market affects the movement of every individual stock, but the representative stocks leading the market keeps changing over time based on people's dynamic interests. How can we effectively capture the global market trend?
3) **Capturing local correlations.** Individual pairs of stocks make correlated movements of their prices, apart from the global trend of the market. How can we capture such correlations that consistently change over time?

We address the challenges with the following main ideas.

1) **Self-supervised learning (Section III-C).** We learn the low-dimensional embeddings of tweets and stocks on the same semantic space with self-supervised learning, allowing unpopular stocks to utilize any tweets based on the distance in the embedding space.
2) **Global price movement attention (Section III-D).** We capture the global movement of a market by using the average of all tweet embeddings at each day as the query of attention for combining the movements of all stocks in the market with dynamic weights.
3) **Tweet-based local similarities (Section III-E).** We find the local similarities between stocks by performing two different attention steps: one for finding relevant tweets for each target stock, and the other for finding relevant pairs of stocks from the selected tweets.

In Section III-A, we present observations derived from tweet data, which motivate us to propose the main ideas of SLOT. In Section III-B, we give an overview of our SLOT, including the main predictor module which takes as input the tweet trend vectors generated from our self-supervised learning and trend aggregation modules. In Section III-C, we introduce our algorithm for learning the embeddings of stocks and tweets. In Section III-D and III-E, we introduce our ideas for aggregating tweet vectors globally and locally, respectively.

TABLE I: Confusion matrices for predicting price movement with sentiment analysis of tweets. Approaches 1 and 2 determine the sentiment of each tweet in different ways (details are in Section III-A). Meaningful correlations between sentiment and price movement are not observed from the results.

(a) Approach 1

Price	Sentiment	
	Pos.	Neg.
Rise	6,264	25,298
Fall	6,537	29,320

(b) Approach 2

Price	Sentiment	
	Pos.	Neg.
Rise	7,935	23,627
Fall	8,331	27,526

A. *Observations and Motivations*

Twitter is one of the most popular online social networks, where users publicly express their opinions by *tweets*.[1] Compared to news articles or financial reports which provide clean and reliable information, Twitter is a place where people get swift information that is unreliable and often wrong. Previous works on stock price prediction [4], [10] focused on exploiting the timeliness of tweets, based on the belief that they provide meaningful evidence of forecasting. However, there exist two essential limitations of tweet data, which have been neglected in previous works for stock price prediction.

Sparsity limitation. Most tweets mention only a few popular stocks, while other stocks have an insufficient number of tweets for prediction. Figure 2 shows the number of tweets that mention each stock in the ACL18 dataset [4]. It is clear in Figure 2a that a few popular stocks such as AAPL, FB, and GOOG have most of the tweets. Since the bottom 23% of stocks have only 1% of all tweets, it is difficult for tweet-based models to make accurate prediction for such stocks. Moreover, if we decrease the query time range, the sparseness becomes even worse due to the shortage of samples: 18% and 57% of stocks have no tweets in Figures 2b and 2c, respectively.

Random sentiment. Tweets often provide information irrelevant to the price movement. We verify this characteristic by conducting sentiment analysis with a popular language model based on BERT [34]. The model we use is publicly available,[2] and it predicts the sentiment of a text into five classes from very negative (class 1) to very positive (class 5). For each tweet e that mentions stock s, we compare predicted sentiment $\mathbf{m}_e$ with the binary price movement y_e of stock s after the tweet is released. That is, if the tweet e is from day t, the movement y_e is measured between day t and day $t+1$.

We use two approaches for classifying the sentiment of each tweet into positive: a) tweet e is positive if $\text{argmax}(\mathbf{m}_e) \geq 3$, and b) tweet e is positive if $\sum_{i=3}^{5} m_{e,i} > \sum_{i=1}^{2} m_{e,i}$, where $m_{e,i}$ is the i-th element of the sentiment vector $\mathbf{m}_e$. Then, we compare the prediction and the price movement as confusion matrices in Table I, presenting two observations. First, most tweets are predicted to have the negative sentiment while the numbers of price rises and falls are similar. Second, the amount of correlation between sentiment and price movement is negligible, implying that the sentiment of a tweet does not provide meaningful information for prediction.

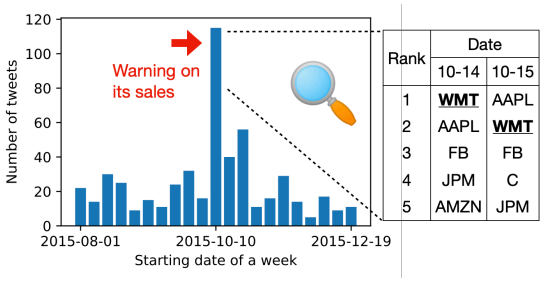


Fig. 3: The number of tweets mentioning Walmart (WMT) in the ACL18 dataset. The count increases dramatically with the release of a warning on its sales, making WMT more popular than Apple (AAPL). This shows that the popularity of a stock in tweets reflects actual important events.

On the other hand, tweets give valid information for detecting real-world events that affect the market. Figure 3 shows the number of tweets mentioning Walmart (WMT) in the ACL18 dataset [4]. The number of tweets increases dramatically with the release of a warning on its sales, making the stock become more popular than AAPL.[3] Such a steep change in the number of tweets gives us unique information on the current status of the market, which cannot be inferred from historical prices. Such descriptive nature of tweets is the key information one can extract for accurate stock movement prediction, regardless of the sentiment or the reliability of individual tweets.

Motivations. Our SLOT is designed to address these two limitations of tweet data. First, given each target stock s, we find tweets that are the most relevant to s and use them for its prediction, even though such tweets do not directly mention s. Second, we use tweets to capture the global market trend and the local similarities between stocks, rather than as the direct evidence for price prediction. These ideas allow us to use all available tweets to give information to even unpopular stocks, regardless of the unreliability of their contents.

B. *Overview of SLOT*

We first describe the overall structure of SLOT and present our main ideas in detail. Figure 4 illustrates the overview of SLOT, which takes as input the stock and tweet embeddings generated from self-supervised learning. SLOT uses attention LSTM (ALSTM) as the main predictor, which has shown good performance in previous works for stock movement prediction [2], [14]. We assume that each stock s at day t contains a price feature $\mathbf{x}_{st}$ that describes its price movement until day t. Then, ALSTM makes a binary prediction from the price features of recent days and the trend features generated in global and local ways based on the embedding vectors.

Price features. Table II shows the information of feature $\mathbf{x}_{st}$ for each stock s at day t [2], [14]. $open$, $high$, low, and $close$ represent the opening, the highest, the lowest, and the

[1] https://twitter.com/home

[2] https://huggingface.co/nlptown/bert-base-multilingual-uncased-sentiment

[3] https://www.nytimes.com/2015/10/15/business/walmart-sales-forecast-share-buyback.html

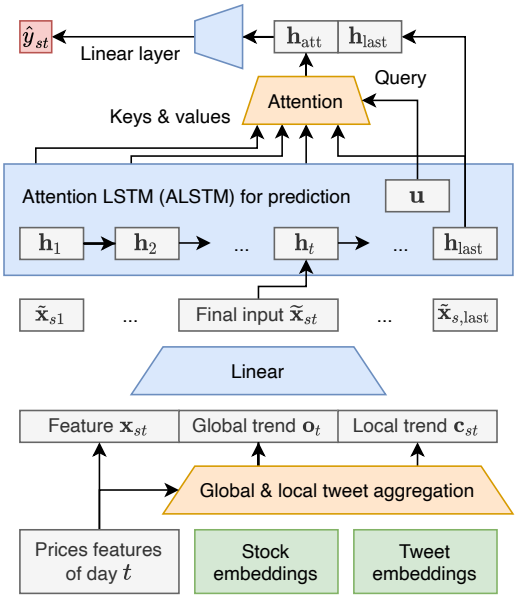


Fig. 4: The overall structure of SLOT for making a prediction $\hat{y}_{st}$ for stock s at day t. SLOT learns stock and tweet embeddings by self-supervised learning (Sec. III-C) and creates two types of trend features (Sec. III-D and III-E). The ALSTM model combines the three types of features for prediction.

closing price at each day, respectively. *adj_close* represents the closing price that is adjusted to be invariant of stock split events. The feature elements are categorized into three groups based on their properties:

- Price movements in a day: c_open, c_high, c_low
- Price movements between days: n_close, n_adj_close
- Long-term movements: $5\_day, 10\_day, \cdots, 30\_day$

Overall, daily prices on days $t-29$ to t are used to generate each feature vector $\mathbf{x}_{st}$ of day t.

Main predictor. We use ALSTM as our main predictor for stock movement prediction. Compared to Transformer [35] or convolutional neural networks [36] which have been also used for time series forecasting, ALSTM can effectively capture complex evolving patterns of stock prices by a) updating the hidden state in the chronological order and b) applying the attention function to select the most informative time steps in the history.

The main ideas of SLOT are designed to provide additional information to ALSTM from tweet data, which cannot be captured from historical prices, by making a *multi-level* feature $\tilde{x}_{st}$ for each stock s at day t. Assume that we have generated a global trend vector $\mathbf{o}_t$ and a local trend vector $\mathbf{c}_{st}$ for stock s and day t from the price features of all stocks and tweets at day t, based on the result of self-supervised learning. That is, the tweets at day t enable us to understand the global and local relationships between stocks shown at day t.

Then, we use a linear layer to generate a multi-level feature:

$$\tilde{x}_{st} = \mathbf{W}(\mathbf{x}_{st} \parallel \mathbf{o}_t \parallel \mathbf{c}_{st}) + \mathbf{b}, \tag{3}$$

TABLE II: Price features that compose a feature vector $\mathbf{x}_{st}$ for each stock s and day t. They summarize the price movements of stock s until day t (details are in Section III-B).

Features	Calculation
c_open, c_high, c_low	e.g., $c\_open = open_t / close_t - 1$
n_close, n_adj_close	e.g., $n\_close = close_t / close_{t-1} - 1$
5_day, 10_day, 15_day	e.g., $5\_day = \frac{\sum_{i=0}^{4} adj\_close_{t-i/5}}{adj\_close_t} - 1$
20_day, 25_day, 30_day	

where $\mathbf{W} \in \mathbb{R}^{d \times 3d}$ and $\mathbf{b}^d$ are the learnable weight and bias terms, respectively, $\parallel$ is the concatenation operator between vectors, and d is the size of features $\mathbf{x}_{st}$, $\mathbf{o}_t$ and $\mathbf{c}_{st}$.

The resulting features are fed into ALSTM as follows:

$$\hat{y}_{st} = \text{ALSTM}(\tilde{\mathbf{x}}_{s,t-w+1}, \tilde{\mathbf{x}}_{s,t-w+2}, \cdots, \tilde{\mathbf{x}}_{st}), \tag{4}$$

where w is the window size of ALSTM, which is chosen as a hyperparameter between 10 and 15 in our experiments.

Optimization. We train ALSTM to minimize the following objective function:

$$\mathcal{L}(\theta) = \sum_{t \in \mathcal{T}} \sum_{s \in \mathcal{S}} \max(0, 1 - y_{st}\hat{y}_{st}) + \lambda \|\theta\|_2^2, \tag{5}$$

where θ is the set of learnable parameters of ALSTM, $\mathcal{T}$ is the set of available days in training data, $\mathcal{S}$ is the set of target stocks, $y_{st} \in \{-1, +1\}$ is the binary label of stock s at day t, and λ is a hyperparameter for regularization. We use the hinge loss [37] as in previous work [2] for stock movement prediction. Note that the real-valued score $\hat{y}_{st}$ is directly used in Equation (5), and $\hat{y}_{st} > 0$ represents that we predict it as a rise. The supervised training of ALSTM is done separately from the self-supervised learning of embedding vectors.

C. Self-supervised Learning of Embeddings

The embeddings of stocks and tweets play an essential role in the overall framework of our SLOT. We describe how to learn tweet and stock embeddings in the same semantic space in a self-supervised approach. Let $\mathcal{S}$ be a set of target stocks and $\mathcal{E}_s$ be a set of tweets that mention stock $s \in \mathcal{S}$. Then, for each stock s and tweet $e \in \mathcal{E}_s$, we learn a stock embedding $\mathbf{h}_s$ and a tweet embedding $\mathbf{h}_e$ together so that one embedding vector can be used to query for the other. The goal of training is to solve *stock identification*, which is to predict stock s that a tweet $e \in \mathcal{E}_s$ mentions from the content of the tweet, when the stock symbol is masked. Figure 5 illustrates the process of self-supervised learning of the embedding vectors.

Tokenization and masking. We first tokenize the words in every tweet e with SentencePiece [38], which is a popular word tokenizer that requires no preprocessing of texts. We use a learnable embedding vector for each token as done in many language models [34], [39]. Then, we replace all tokens in $\mathcal{E}_s$ that correspond to the name of stock s with a special token MASK. This is because its name is the answer to the stock identification problem. For example, assume that we have a tweet "Thank you Apple for the new iPhone." Then, we mask the target symbol Apple and predict Apple from "Thank you [MASK] for the new iPhone."

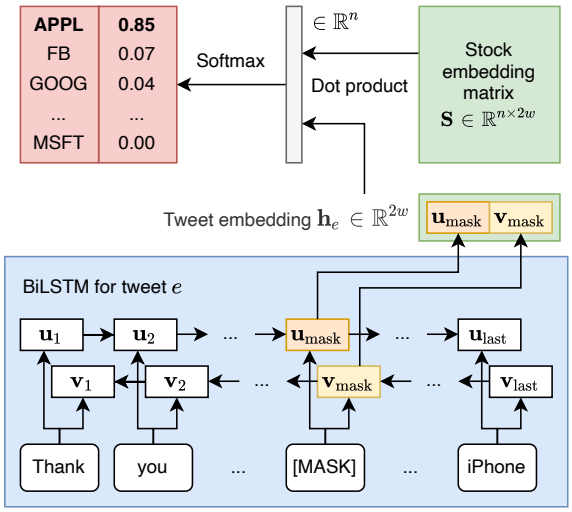


Fig. 5: The self-supervised learning step of SLOT for learning the stock and tweet embeddings. BiLSTM makes an embedding vector $\mathbf{h}_e$ for the given tweet e, and $\mathbf{h}_e$ is used to predict the masked stock by the dot product with stock embeddings. The stock embedding matrix $\mathbf{S}$ is learned as a parameter.

Stock predictor model. We use different approaches for modeling stock and tweet embeddings. For stocks, we learn the embedding $\mathbf{h}_s$ of each stock s as a free parameter, which is updated directly from backpropagation. Each embedding $\mathbf{h}_s$ is the s-th row of the stock embedding matrix $\mathbf{S}$. On the other hand, for tweets, we introduce a predictor network f and use its output $f(e)$ given the tokenized contents of each tweet e as the embedding $\mathbf{h}_e$. This is based on the observation that each stock conveys unique information which is not decomposed, while each tweet is a sequence of word tokens.

We use bi-directional LSTM (BiLSTM) as the embedding model f to generate tweet embeddings. This is because unidirectional LSTM is inappropriate for considering the context of both sides of the masked token, which is the target of our stock identification problem. Other language models such as Transformer [34] can also be used, but we have found in experiments that BiLSTM is more effective in our scenario, as training data are insufficient to learn many parameters of Transformer. Specifically, we run BiLSTM and use the state vector generated at the masked token as $\mathbf{h}_e$, instead of the end of the tweet, to focus on its local context.

Optimization. We update three kinds of parameters in this step: the embeddings of word tokens, the parameters of f, and the embeddings of stocks. Given a tweet embedding $\mathbf{h}_e$ made by f, our prediction for stock s is defined as follows:

$$\hat{y}_{es} = \frac{\exp(\mathbf{h}_s^\top \mathbf{h}_e)}{\sum_{s' \in \mathcal{S}} \exp(\mathbf{h}_{s'}^\top \mathbf{h}_e)}. \tag{6}$$

Then, we update all parameters to minimize the following objective function, which is based on the cross entropy function:

$$l(\theta) = -\sum_{s \in \mathcal{S}} \sum_{e \in \mathcal{E}_s} \log \hat{y}_{es}. \tag{7}$$

Advantages of stock identification. We expect the following two advantages of stock and tweet embeddings. First, they can be used interchangeably in the main model for stock movement prediction, because we make the tweet embeddings become similar to the embeddings of their target stocks. This is done by maximizing the dot product between embeddings during training. Second, stock embeddings convey the information of stock similarities, as the stocks frequently mentioned together are likely to have similar embedding vectors. This is a great advantage of our proposed approach for self-supervised learning, since the preservation of similarities (and dissimilarities) of entities is one of the most important requirement for good embeddings that preserve their original properties.

D. Global Price Movement Attention

Since the price movement of each stock is affected by the global trend of the market, it is essential to design an effective summary of the market for predictions. However, traditional market indices such as Nasdaq 100 cannot capture the dynamic influence of each stock to the global trend due to the following reasons. First, a traditional index computes a weighted average of stock prices based on the market capitalization of stocks, but the influence of each stock is not only determined by its market capitalization. Second, the influence of each stock on the market changes dynamically over time, since the interests of people keep changing with real-world events.

We make a comprehensive market index by combining the price information of multiple stocks based on their popularity in tweet data. The generated global trend vector $\mathbf{o}_t$ shown in Figure 4 gives the comprehensive market information to each prediction. Assume that we have embedding vectors $\mathbf{h}_s$ and $\mathbf{h}_e$ for each stock s and tweet e, respectively. We compute the average $\mathbf{g}_t$ of all tweet embeddings at day t. Then, we apply the attention using $\mathbf{g}_t$ as the query vector for combining the price features of all stocks based on their stock embeddings. The attention process is defined as follows:

$$\mathbf{o}_t = \sum_{s \in \mathcal{S}} \alpha_{st} \mathbf{x}_{st} \quad \text{where} \quad \alpha_{st} = \frac{\exp(\mathbf{g}_t^\top \mathbf{h}_s)}{\sum_{s' \in \mathcal{S}} \exp(\mathbf{g}_t^\top \mathbf{h}_{s'})}. \tag{8}$$

$\mathbf{x}_{st}$ is the price feature of stock s at day t, $\mathcal{S}$ is the set of all stocks, and α_{st} is the attention weight for aggregation.

Figure 6 illustrates the process of global aggregation, which uses the average $\mathbf{g}_t$ of all tweet embeddings at day t and the matrix $\mathbf{S}$ of stock embeddings as the query and the key of attention, respectively. The price features in $\mathbf{X}_t$ are aggregated as a result of attention, generating the global trend vector $\mathbf{o}_t$. As shown in Figure 4, the generated vector $\mathbf{o}_t$ is concatenated with the original feature vector $\mathbf{x}_{st}$ and then fed into ALSTM, providing comprehensive information of the market.

Recall that we learn the embedding vectors of stocks and tweets to make the embedding $\mathbf{h}_e$ of tweet e become similar to the embedding $\mathbf{h}_s$ of stock s that the tweet e mentions. Thus,

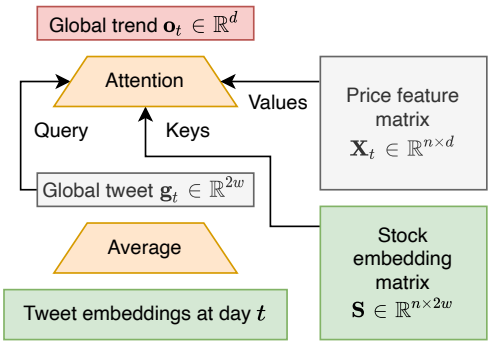


Fig. 6: The overview of global tweet aggregation for generating the global trend vector $\mathbf{o}_t$ at day t. It uses the average of all tweets at day t as the query vector of attention to combine the price features based on their embeddings.

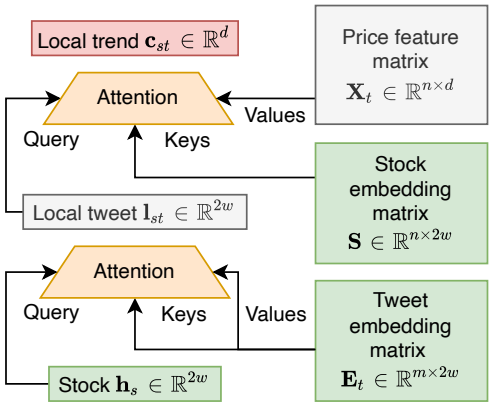


Fig. 7: The overview of local tweet aggregation for generating the local trend vector $\mathbf{c}_{st}$ for stock s at day t. It uses the stock embedding $\mathbf{h}_s$ as the query of the first attention and then uses the result as the query of the second attention.

the attention of Equation (8) effectively finds the stocks that are mentioned most actively at each day, and we expect that such stocks represent the global trend of the market effectively in terms of people's interest. As a result, the global trend vector works as a dynamic summary of stock movements, based on people's collective opinions represented in tweets.

E. Local Similarities between Stocks

Stocks in a market make local correlations based on their unique properties, and such correlations often make their price movements different from the global market trend. Accurate estimation of such correlations between stocks is essential for multivariate movement prediction. We propose to make a local trend vector $\mathbf{c}_{st}$ for each stock s at day t, which combines the price movements of stocks relevant to s. The local trend $\mathbf{c}_{st}$ provides personalized market information for each stock s, which is distinct from the global trend vector $\mathbf{o}_t$ shared for all target stocks. The vector $\mathbf{c}_{st}$ is concatenated with the feature $\mathbf{x}_{st}$ and the global trend $\mathbf{o}_t$ as shown in Figure 4, providing comprehensive information to each prediction.

Figure 7 illustrates our proposed approach to aggregate the price movements of multiple stocks in a local stock-wise view. The module runs two steps of attention functions for a) finding tweets that are relevant to each target stock and b) finding the correlated stocks, respectively. The stock embedding matrix $\mathbf{S}$ and the tweet embedding matrix $\mathbf{E}_t$ are the main inputs of the attention functions, where $\mathbf{S}$ contains the embedding $\mathbf{h}_s$ of the target stock s as its s-th row. The two attention functions work similarly but with different objectives and inputs.

Attention to find relevant tweets. The first attention is to find tweets relevant to each target stock s. Previous work [4] found the relevant tweets based on whether they mention s or not. However, this causes a severe imbalance between stocks, following the sparsity problem that we described in Section III-A. Thus, we find relevant tweets by using the embedding $\mathbf{h}_s$ of stock s as the query of attention as follows:

$$\mathbf{l}_{st} = \sum_{e \in \mathcal{E}_t} \alpha_e \mathbf{h}_e \quad \text{where} \quad \alpha_e = \frac{\exp(\mathbf{h}_s^\top \mathbf{h}_e)}{\sum_{e' \in \mathcal{E}_t} \exp(\mathbf{h}_s^\top \mathbf{h}_{e'})}. \tag{9}$$

$\mathcal{E}_t$ is the set of tweets at day t, $\mathbf{h}_e$ is the embedding of tweet e, and α_e is the attention weight. The resulting vector $\mathbf{l}_{st}$ is the weighted average of all tweets at day t based on their relevance to the target stock s, computed from their embeddings.

Attention to find similar stocks. As a result of the attention function of Equation (9), every stock has a representation $\mathbf{l}_{st}$ that summarizes all tweets at day t considering the relevance to stock s. Our second attention uses the resulting tweet vector to find the most relevant stocks at the moment. This is done by using $\mathbf{l}_{st}$ as the query vector of attention to aggregate the price features at day t based on the stock embeddings:

$$\mathbf{c}_{st} = \sum_{s \in \mathcal{S}} \alpha_{st} \mathbf{x}_{st} \quad \text{where} \quad \alpha_{st} = \frac{\exp(\mathbf{l}_{st}^\top \mathbf{h}_s)}{\sum_{s' \in \mathcal{S}} \exp(\mathbf{l}_{st}^\top \mathbf{h}_{s'})}. \tag{10}$$

The resulting local trend vector $\mathbf{c}_{st}$ is concatenated to the original feature vector $\mathbf{x}_{st}$ and the global trend vector $\mathbf{o}_t$, making the input of ALSTM for predicting the stock movement.

IV. Experiments

We conduct experiments to answer the following questions about the performance of SLOT:

- Q1. **Accuracy (Section IV-B).** Does SLOT outperform previous approaches in stock movement prediction?
- Q2. **Embedding quality (Section IV-C).** Do the embeddings of stocks learned by SLOT preserve their actual relationships in the vector space? Do the tweet embeddings find the stocks that are relevant to their contents?
- Q3. **Ablation study (Section IV-D).** How do the global and local tweet trends of SLOT affect its performance?

A. Experimental Setup

We present our experimental setup including datasets, baseline approaches, hyperparameters, and evaluation metrics.

Datasets. We use 3 benchmark datasets of stock movement prediction: BIGDATA22, ACL18 [4], and CIKM18 [10] shown in Table III. All datasets consist of high-trade-volume stocks

TABLE III: Summary of datasets. The Days column represents the number of available days in each dataset.

Data	Stocks	Tweets	Days	Dates
BigData22[1]	50	272,762	362	2019-07-05 to 2020-06-30
ACL18[2]	87	106,271	696	2014-01-02 to 2015-12-30
CIKM18[3]	38	955,788	352	2017-01-03 to 2017-12-28

[1] https://github.com/stocktweet/stock-tweet
[2] https://github.com/yumoxu/stocknet-dataset
[3] https://github.com/wuhuizhe/CHRNN

in US stock markets. BIGDATA22 is a new dataset that we collect and publicly release, while the other two datasets were used in previous works for stock movement prediction. We label the instances according to the increase rate of adjusted closing prices. The increase rate is given as $r_d^s = p_d^s / p_{d-1}^s - 1$, where p_d^s is the adjusted closing price of stock s at day d. Instances with the $r_d^s \geq 0.55\%$ and $r_d^s \leq -0.5\%$ are labeled as 1 and -1, respectively, for binary classification. We split each dataset into training, validation, and test data chronologically, as in recent works for stock movement prediction [1], [2].

Competitors. We compare SLOT with technical and tweet-based competitors for stock movement prediction. Technical approaches focus on capturing patterns of historical prices for stock movement prediction. We concatenate the price features of the last w days for non-sequential approaches, which take only a single feature vector for each prediction.

- **Logistic regression (LR)** is the simplest baseline, which draws a linear decision boundary between classes.
- **Random forest (RF)** is a strong feature-based approach, which shows better performance than that of LR in many cases by combining randomized decision trees.
- **LSTM** is a representative model for sequential data. We take the model used in [11] for stock prediction.
- **Attention LSTM (ALSTM)** combines the hidden states of multiple LSTM cells with the attention technique [7], preserving the information of distant inputs.
- **Adv-ALSTM** [2] applies adversarial training to ALSTM to improve its generalization performance. Adv-ALSTM generates artificial features that the current model fails to predict during training, making the model more robust.
- **DTML** [1] is the previous state-of-the-art model for stock movement prediction, which combines the contexts of all target stocks based on Transformer. DTML uses the same price features as in Adv-ALSTM and SLOT.

Tweet-based competitors utilize tweet data to improve the performance of stock prediction, and are given as follows:

- **ALSTM-W** makes tweet embeddings by Word2Vec [40] and uses them in prediction. We find tweets that directly mention each target stock, run Word2Vec to generate an embedding for each tweet, compute the average of them, and concatenate it with the price feature of ALSTM.
- **ALSTM-D** is similar to ALSTM-W, except that ALSTM-D uses Doc2Vec [41] for generating tweet embeddings. Doc2Vec makes more representative embeddings of texts, but often shows poor generalization performance.

TABLE IV: Classification performance of SLOT and competitors, measured with the accuracy (ACC) and the Matthews correlation coefficient (MCC). The best is in bold, and the second best is underlined. Our SLOT shows the best performance in all datasets with both evaluation metrics.

Method	BIGDATA22		ACL18		CIKM18	
	ACC	MCC	ACC	MCC	ACC	MCC
LR	53.07	0.0200	52.20	0.0442	52.50	-0.0425
RF	47.10	-0.1114	51.94	0.0348	53.57	0.0119
LSTM	50.69	0.0127	52.75	0.0639	53.31	0.0216
ALSTM	48.69	-0.0254	51.82	0.0429	52.54	-0.0077
Adv-ALSTM	50.36	0.0120	53.11	0.0685	53.69	0.0217
DTML	51.65	<u>0.0651</u>	<u>58.12</u>	<u>0.1806</u>	<u>53.86</u>	0.0049
ALSTM-W	48.28	-0.0116	53.32	0.0754	53.64	<u>0.0315</u>
ALSTM-D	49.16	0.0090	52.98	0.0681	50.40	-0.0449
StockNet	<u>52.99</u>	-0.0163	53.60	-0.0248	52.35	-0.0161
SLOT (proposed)	**54.81**	**0.0952**	**58.72**	**0.2065**	**55.86**	**0.0899**

- **StockNet** [4] is a recent approach that uses tweets as the main evidence of stock movement prediction. StockNet uses variational autoencoders to represent tweets as low-dimensional vectors. We use their official implementation and take the average of five runs with different seeds.[4]

Evaluation metrics. We use two metrics to evaluate the performance of stock movement prediction in different perspectives: accuracy (ACC) and the Matthews correlation coefficient (MCC) [42]. ACC is a popular metric, which has been used widely in various classification problems. MCC improves the fairness of evaluation especially when the numbers of positive and negative samples are different. Let TP, TN, FP, and FN denote true positives, true negatives, false positives, and false negatives, respectively. Then, the MCC is defined as follows:

$$\mathrm{MCC} = \frac{\mathrm{TP} \times \mathrm{TN} - \mathrm{FP} \times \mathrm{FN}}{\sqrt{(\mathrm{TP}+\mathrm{FP})(\mathrm{TP}+\mathrm{FN})(\mathrm{TN}+\mathrm{FP})(\mathrm{TN}+\mathrm{FN})}}.$$

Hyperparameters. We train our SLOT and all competitors based on deep neural networks with the Adam optimizer [43]. The number of epochs is 150, and the batch size is set to 1024. The space of hyperparameter search is given as follows:

- Regularization strength λ: $\{0.1, 0.5, 1.0\}$
- Learning rate: $\{0.001, 0.005, 0.01\}$
- Hidden dimension size of ALSTM: $\{8, 16, 32\}$
- Lag window size: $\{10, 15\}$

We run each model five times with different random seeds and report the average performance in all cases.

B. Accuracy (Q1)

We compare the accuracy of SLOT and baseline methods with the two evaluation metrics. The results are shown in Table IV and Figure 1. Our SLOT consistently makes the best performance on all datasets and metrics; SLOT shows up to 2.0%p higher accuracy and 0.0584 higher MCC, compared to second-best methods.

[4] https://github.com/yumoxu/stocknet-code

TABLE V: The result of queries for finding the stocks whose embeddings are the closest in the ACL18 dataset. Stocks in similar sectors are clustered in the embedding space.

Query	Top 1	Top 2	Top 3
AAPL	AMZN	PCLN	BABA
FB	GOOG	PCLN	BABA
BA	CAT	PM	HD
BAC	JPM	HON	C

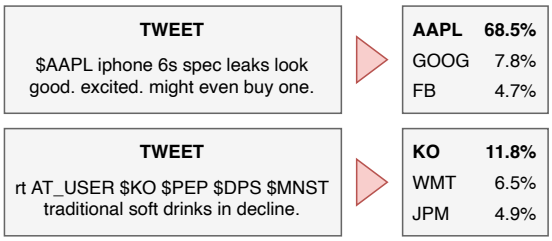


Fig. 8: Tweets and the top 3 stocks whose embeddings are the most matched with the embeddings of the tweets. The tweet embeddings effectively capture the information of target stocks from the contents of tweets.

In the ACL18 dataset, the performances of SLOT and DTML, the best competitor, are significantly better than the other baselines. This is because SLOT and DTML are multivariate methods that correlate the predictions for multiple stocks, while the other baselines are univariate methods that focus on individual ones. The global trend of a market affects every stock in the market, making strong correlations between stocks whether they are in the same financial sector or not. It is essential to combine the information of multiple stocks to maximize the performance of stock prediction.

The main difference between SLOT and DTML is how to combine the price information from multiple stocks. DTML uses only historical prices for conducting multi-head attention between target stocks with a Transformer encoder. This is effective to some extent, but does not give additional information to the predictor. On the other hand, SLOT utilizes tweet data to learn inherent properties of target stocks that cannot be found from the prices. This gives SLOT large flexibility for learning stock correlations, resulting in higher accuracy.

C. Embedding Quality (Q2)

We analyze the stock and tweet embeddings learned by self-supervised learning (Section III-C). Table V shows the result of querying for the stocks whose embedding vectors are the closest in the vector space. We measure the distance between vectors based on the cosine distance, since our self-supervised algorithm uses the dot product to select relevant stocks for each tweet. The result shows that stocks in similar sectors are close in the embedding space, even though we do not provide any knowledge of such stocks. Note that popular IT companies such as Apple (AAPL), Amazon (AMZN), and Alibaba (BABA) are clustered in the embedding space. We also observe that 1) manufacturing companies such as Boeing (BA) and Caterpillar (CAT) are clustered, and 2) banks such as Bank of America (BAC) and JP Morgan (JPM) are clustered in the embedding space.

TABLE VI: An ablation study of SLOT, where ALSTM is the baseline without tweet data, SLOT-G is without the global trend, and SLOT-L is without the local trend. Both global and local trends improve the performance of SLOT.

Method	BIGDATA22		ACL18		CIKM18	
	ACC	MCC	ACC	MCC	ACC	MCC
ALSTM	48.69	-0.0254	51.82	0.0429	52.54	-0.0077
SLOT-G	52.21	0.0272	56.99	0.1718	51.29	0.0186
SLOT-L	51.17	0.0283	57.48	0.1829	55.10	0.0745
SLOT (proposed)	**54.81**	**0.0952**	**58.72**	**0.2065**	**55.86**	**0.0899**

Figure 8 shows examples of tweets and the top three stocks whose embeddings are the most matched with the embeddings of the given tweets. KO represents the Coca-Cola company. The predictions reflect the properties of stocks in the market. For instance, GOOG and FB are predicted as the second and the third options in the first tweet, respectively, based on the tech-related content of the tweet. Walmart (WMT) appears as the second-most related stock in the second tweet, due to its relationship with drink companies. Such results demonstrate the effectiveness of our self-supervised algorithm for learning stock and tweet embeddings, which preserve the properties of their original entities in a single semantic space.

D. Ablation Study (Q3)

We conduct an ablation study for SLOT and summarize the result in Table VI. SLOT-G is a variant of SLOT without the global tweet trend (Section III-D), and SLOT-L is the one without the local tweet trend (Section III-E). ALSTM uses the same price features as in SLOT but does not use tweet data at all. Thus, ALSTM works as the baseline of SLOT and its two variants, and its performance is the same as in Table IV which compares SLOT with other competitors.

We have two observations from Table VI. First, SLOT outperforms all its variants, showing the effectiveness of our two main ideas for utilizing tweets to learn the relationships between multiple stocks. Second, the two ideas of SLOT have different effectiveness depending on the datasets. The global trend is more effective in ACL18 and CIKM18, while the local trend is more effective in BIGDATA22. This implies that ACL18 and CIKM18 have strong market movements that affect the movements of individual stocks, while BigData22 does not have such a strong market movement and it is more important to focus on the relationships between stock pairs. Our two ideas for utilizing sparse noisy tweets work effectively in different ways, having their strengths in different datasets.

V. CONCLUSION

In this paper, we propose SLOT, an accurate method for stock movement prediction, which exploits historical stock prices and sparse tweets to maximize accuracy of prediction.

SLOT effectively addresses the problems of sparsity and unreliability of using tweets for stock movement prediction. SLOT represents tweets and stocks as embedding vectors on the same semantic space via self-supervised pre-training, allowing us to use any tweet for any stock based on the distance in the vector space which addresses the sparsity problem. SLOT also captures global and local tweet trends by aggregating multiple stocks based on the occurrences of stocks in tweets. This allows us to focus on the popularity and the co-occurrence information of stocks, rather than the sentiment of individual tweets, avoiding the unreliability problem of tweet contents. Extensive experiments on real world datasets show that SLOT provides the state-of-the-art performance. We also demonstrate the effectiveness of our proposed ideas through ablation and case studies. Future works include combining multiple sources of texts such as news articles or financial reports to further improve accuracy.

ACKNOWLEDGMENT

This work was supported by DeepTrade Inc. U Kang is the corresponding author.

REFERENCES

[1] J. Yoo, Y. Soun, Y. chan Park, and U. Kang, "Accurate multivariate stock movement prediction via data-axis transformer with multi-level contexts," in *KDD*, 2021.

[2] F. Feng, H. Chen, X. He, J. Ding, M. Sun, and T. Chua, "Enhancing stock movement prediction with adversarial training," in *IJCAI*, 2019.

[3] Y. Chen, Z. Wei, and X. Huang, "Incorporating corporation relationship via graph convolutional neural networks for stock price prediction," in *CIKM*, 2018.

[4] Y. Xu and S. B. Cohen, "Stock movement prediction from tweets and historical prices," in *ACL*, 2018.

[5] Y. Kara, M. A. Boyacioglu, and Ö. K. Baykan, "Predicting direction of stock price index movement using artificial neural networks and support vector machines: The sample of the istanbul stock exchange," *Expert Syst. Appl.*, vol. 38, no. 5, pp. 5311–5319, 2011.

[6] C. Tsai and Y. Hsiao, "Combining multiple feature selection methods for stock prediction: Union, intersection, and multi-intersection approaches," *Decis. Support Syst.*, vol. 50, no. 1, pp. 258–269, 2010.

[7] Y. Qin, D. Song, H. Chen, W. Cheng, G. Jiang, and G. W. Cottrell, "A dual-stage attention-based recurrent neural network for time series prediction," in *IJCAI*, 2017.

[8] Q. Liu, X. Cheng, S. Su, and S. Zhu, "Hierarchical complementary attention network for predicting stock price movements with news," in *CIKM*, 2018.

[9] X. Ding, Y. Zhang, T. Liu, and J. Duan, "Deep learning for event-driven stock prediction," in *IJCAI*, 2015.

[10] H. Wu, W. Zhang, W. Shen, and J. Wang, "Hybrid deep sequential modeling for social text-driven stock prediction," in *CIKM*, 2018.

[11] D. M. Q. Nelson, A. C. M. Pereira, and R. A. de Oliveira, "Stock market's price movement prediction with LSTM neural networks," in *IJCNN*, 2017.

[12] C. Chen, L. Zhao, J. Bian, C. Xing, and T.-Y. Liu, "Investment behaviors can tell what inside: Exploring stock intrinsic properties for stock trend prediction," in *KDD*, 2019.

[13] L. Zhang, C. Aggarwal, and G.-J. Qi, "Stock price prediction via discovering multi-frequency trading patterns," in *KDD*, 2017.

[14] J. Yoo and U. Kang, "Attention-based autoregression for accurate and efficient multivariate time series forecasting," in *SDM*, 2021.

[15] H. Wang, S. Li, T. Wang, and J. Zheng, "Hierarchical adaptive temporal-relational modeling for stock trend prediction," in *IJCAI*, 2021.

[16] Q. Ding, S. Wu, H. Sun, J. Guo, and J. Guo, "Hierarchical multi-scale gaussian transformer for stock movement prediction." in *IJCAI*, 2020.

[17] M. Hou, C. Xu, Y. Liu, W. Liu, J. Bian, L. Wu, Z. Li, E. Chen, and T.-Y. Liu, "Stock trend prediction with multi-granularity data: A contrastive learning approach with adaptive fusion," in *CIKM*, 2021.

[18] J. Liu, H. Lin, L. Yang, B. Xu, and D. Wen, "Multi-element attention capsule network for stock prediction," *IEEE Access*, vol. 8, pp. 143 114–143 123, 2020.

[19] C. Li, D. Song, and D. Tao, "Multi-task recurrent neural networks and higher-order markov random fields for stock price movement prediction," in *KDD*, 2019.

[20] R. Akita, A. Yoshihara, T. Matsubara, and K. Uehara, "Deep learning for stock prediction using numerical and textual information," in *ICIS*, 2016.

[21] X. Ding, Y. Zhang, T. Liu, and J. Duan, "Knowledge-driven event embedding for stock prediction," in *COLING*, 2016.

[22] ——, "Using structured events to predict stock price movement: An empirical investigation," in *EMNLP*, 2014.

[23] T. H. Nguyen and K. Shirai, "Topic modeling based sentiment analysis on social media for stock market prediction," in *ACL*, 2015.

[24] W. Li, R. Bao, K. Harimoto, D. Chen, J. Xu, and Q. Su, "Modeling the stock relation with graph network for overnight stock movement prediction," in *IJCAI*, 2021.

[25] Z. Hu, W. Liu, J. Bian, X. Liu, and T.-Y. Liu, "Listening to chaotic whispers: A deep learning framework for news-oriented stock trend prediction," in *WSDM*, 2018.

[26] Q. Chen, "Stock movement prediction with financial news using contextualized embedding from bert," *arXiv preprint arXiv:2107.08721*, 2021.

[27] X. Du and K. Tanaka-Ishii, "Stock embeddings acquired from news articles and price history, and an application to portfolio optimization," in *ACL*, 2020.

[28] W. Xu, W. Liu, C. Xu, J. Bian, J. Yin, and T.-Y. Liu, "Rest: Relational event-driven stock trend forecasting," in *WWW*, 2021.

[29] X. Ying, C. Xu, J. Gao, J. Wang, and Z. Li, "Time-aware graph relational attention network for stock recommendation," in *CIKM*, 2020.

[30] H. Wang, T. Wang, and Y. Li, "Incorporating expert-based investment opinion signals in stock prediction: A deep learning framework," in *AAAI*, 2020.

[31] C. Zhang, Y. Wang, C. Chen, C. Du, H. Yin, and H. Wang, "Stockassistant: A stock AI assistant for reliability modeling of stock comments," in *KDD*, 2018.

[32] J. Bollen, H. Mao, and X. Zeng, "Twitter mood predicts the stock market," *J. Comput. Science*, vol. 2, no. 1, pp. 1–8, 2011.

[33] K. Cho, B. van Merrienboer, Ç. Gülçehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," in *EMNLP*, 2014.

[34] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[35] S. Li, X. Jin, Y. Xuan, X. Zhou, W. Chen, Y. Wang, and X. Yan, "Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting," in *NeurIPS*, 2019.

[36] I. Koprinska, D. Wu, and Z. Wang, "Convolutional neural networks for energy time series forecasting," in *IJCNN*, 2018.

[37] L. Rosasco, E. D. Vito, A. Caponnetto, M. Piana, and A. Verri, "Are loss functions all the same?" *Neural Comput.*, vol. 16, no. 5, pp. 1063–107, 2004.

[38] T. Kudo and J. Richardson, "Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing," in *EMNLP*, 2018.

[39] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *NIPS*, 2017.

[40] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," in *ICLR Workshop*, 2013.

[41] Q. V. Le and T. Mikolov, "Distributed representations of sentences and documents," in *ICML*, 2014.

[42] B. W. Matthews, "Comparison of the predicted and observed secondary structure of t4 phage lysozyme," *Biochimica et Biophysica Acta (BBA)-Protein Structure*, vol. 405, no. 2, pp. 442–451, 1975.

[43] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *ICLR*, 2015.