Stock Movement Prediction from Tweets and Historical Prices

Citation for published version:

Link:
Link to publication record in Edinburgh Research Explorer

Document Version:
Peer reviewed version

Published In:
Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)

General rights
Copyright for the publications made accessible via the Edinburgh Research Explorer is retained by the author(s) and/or other copyright owners and it is a condition of accessing these publications that users recognise and abide by the legal requirements associated with these rights.

Take down policy
The University of Edinburgh has made every reasonable effort to ensure that Edinburgh Research Explorer content complies with UK legislation. If you believe that the public display of this file breaches copyright please contact openaccess@ed.ac.uk providing details, and we will remove access to the work immediately and investigate your claim.
Stock Movement Prediction from Tweets and Historical Prices

Yumo Xu and Shay B. Cohen

School of Informatics, University of Edinburgh
10 Crichton Street, Edinburgh EH8 9AB
yumo.xu@ed.ac.uk, scohen@inf.ed.ac.uk

Abstract

Stock movement prediction is a challenging problem: the market is highly stochastic, and we make temporally-dependent predictions from chaotic data. We treat these three complexities and present a novel deep generative model jointly exploiting text and price signals for this task. Unlike the case with discriminative or topic modeling, our model introduces recurrent, continuous latent variables for a better treatment of stochasticity, and uses neural variational inference to address the intractable posterior inference. We also provide a hybrid objective with temporal auxiliary to flexibly capture predictive dependencies. We demonstrate the state-of-the-art performance of our proposed model on a new stock movement prediction dataset which we collected.\(^1\)

1 Introduction

Stock movement prediction has long attracted both investors and researchers (Frankel, 1995; Edwards et al., 2007; Bollen et al., 2011; Hu et al., 2018). We present a model to predict stock price movement from tweets and historical stock prices.

In natural language processing (NLP), public news and social media are two primary content resources for stock market prediction, and the models that use these sources are often discriminative. Among them, classic research relies heavily on feature engineering (Schumaker and Chen, 2009; Oliveira et al., 2013). With the prevalence of deep neural networks (Le and Mikolov, 2014), event-driven approaches were studied with structured event representations (Ding et al., 2014, 2015).

More recently, Hu et al. (2018) propose to mine news sequence directly from text with hierarchical attention mechanisms for stock trend prediction.

However, stock movement prediction is widely considered difficult due to the high stochasticity of the market: stock prices are largely driven by new information, resulting in a random-walk pattern (Malkiel, 1999). Instead of using only deterministic features, generative topic models were extended to jointly learn topics and sentiments for the task (Si et al., 2013; Nguyen and Shirai, 2015). Compared to discriminative models, generative models have the natural advantage in depicting the generative process from market information to stock signals and introducing randomness. However, these models underrepresent chaotic social texts with bag-of-words and employ simple discrete latent variables.

In essence, stock movement prediction is a time series problem. The significance of the temporal dependency between movement predictions is not addressed in existing NLP research. For instance, when a company suffers from a major scandal on a trading day \(d_1\), generally, its stock price will have a downtrend in the coming trading days until day \(d_2\), i.e. \([d_1, d_2]\).\(^2\) If a stock predictor can recognize this decline pattern, it is likely to benefit all the predictions of the movements during \([d_1, d_2]\). Otherwise, the accuracy in this interval might be harmed. This predictive dependency is a result of the fact that public information, e.g. a company scandal, needs time to be absorbed into movements over time (Luss and d’Aspremont, 2015), and thus is largely shared across temporally-close predictions.

Aiming to tackle the above-mentioned outstanding research gaps in terms of modeling high market stochasticity, chaotic market information and temporally-dependent prediction, we propose

\(^1\)https://github.com/yumoxu/stocknet-dataset

\(^2\)We use the notation \([a, b]\) to denote the interval of integer numbers between \(a\) and \(b\).
StockNet, a deep generative model for stock movement prediction.

To better incorporate stochastic factors, we generate stock movements from latent driven factors modeled with recurrent, continuous latent variables. Motivated by Variational Auto-Encoders (VAEs; Kingma and Welling, 2013; Rezende et al., 2014), we propose a novel decoder with a variational architecture and derive a recurrent variational lower bound for end-to-end training (Section 5.2). To the best of our knowledge, StockNet is the first deep generative model for stock movement prediction.

To fully exploit market information, StockNet directly learns from data without pre-extracting structured events. We build market sources by referring to both fundamental information, e.g. tweets, and technical features, e.g. historical stock prices (Section 5.1).3 To accurately depict predictive dependencies, we assume that the movement prediction for a stock can benefit from learning to predict its historical movements in a lag window. We propose trading-day alignment as the framework basis (Section 4), and further provide a novel multi-task learning objective (Section 5.3).

We evaluate StockNet on a stock movement prediction task with a new dataset that we collected. Compared with strong baselines, our experiments show that StockNet achieves state-of-the-art performance by incorporating both data from Twitter and historical stock price listings.

2 Problem Formulation

We aim at predicting the movement of a target stock s in a pre-selected stock collection S on a target trading day d. Formally, we use the market information comprising of relevant social media corpora M, i.e. tweets, and historical prices, in the lag [d − Δd, d − 1] where Δd is a fixed lag size. We estimate the binary movement where 1 denotes rise and 0 denotes fall,

\[ y = \text{1}(p^d_d > p^d_{d-1}) \]  

(1)

where \( p^d_d \) denotes the adjusted closing price adjusted for corporate actions affecting stock prices, e.g. dividends and splits.4 The adjusted closing price

3To a fundamentalist, stocks have their intrinsic values that can be derived from the behavior and performance of their company. On the contrary, technical analysis considers only the trends and patterns of the stock price.

4Technically, \( d − 1 \) may not be an eligible trading day and thus has no available price information. In the rest of this paper, the problem is solved by keeping the notational consistency with our recurrent model and using its time step \( t \) to index trading days. Details will be provided in Section 4. We use \( d \) here to make the formulation easier to follow.

5https://finance.yahoo.com/industries

6Our dataset is available at https://github.com/yumoxu/stocknet-dataset.
mode, including for tokenization and treatment of hyperlinks, hashtags and the “@” identifier. To alleviate sparsity, we further filter samples by ensuring there is at least one tweet for each corpus in the lag. We extract historical prices for the 88 selected stocks to build the historical price dataset from Yahoo Finance.\footnote{http://finance.yahoo.com}

4 Model Overview

Figure 1: Illustration of the generative process from observed market information to stock movements. We use solid lines to denote the generation process and dashed lines to denote the variational approximation to the intractable posterior.

We provide an overview of data alignment, model factorization and model components.

As explained in Section 1, we assume that predicting the movement on trading day \(d\) can benefit from predicting the movements on its former trading days. However, due to the general principle of sample independence, building connections directly across samples with temporally-close target dates is problematic for model training.

As an alternative, we notice that within a sample with a target trading day \(d\) there are likely to be other trading days than \(d\) in its lag that can simulate the prediction targets close to \(d\). Motivated by this observation and multi-task learning (Caruana, 1998), we make movement predictions not only for \(d\), but also other trading days existing in the lag. For instance, as shown in Figure 2, for a sample targeting 07/08/2012 and a 5-day lag, 03/08/2012 and 06/08/2012 are eligible trading days in the lag and we also make predictions for them using the market information in this sample. The relations between these predictions can thus be captured within the scope of a sample.

As shown in the instance above, not every single date in a lag is an eligible trading day, e.g. weekends and holidays. To better organize and use the input, we regard the trading day, instead of the calendar day used in existing research, as the basic unit for building samples. To this end, we first find all the \(T\) eligible trading days referred in a sample, in other words, existing in the time interval \([d − Δd + 1, d]\). For clarity, in the scope of one sample, we index these trading days with \(t \in [1, T]\)\footnote{It holds that \(T ≥ 1\) since \(d\) is undoubtedly a trading day.} and each of them maps to an actual (absolute) trading day \(d_t\). We then propose \textit{trading-day alignment}: we reorganize our inputs, including the tweet corpora and historical prices, by aligning them to these \(T\) trading days. Specifically, on the \(t\)th trading day, we recognize market signals from the corpus \(M_t\) in \([d_{t−1}, d_t]\) and the historical prices \(p_t\) on \(d_{t−1}\) for predicting the movement \(y_t\) on \(d_t\). We provide an aligned sample for illustration in Figure 2. As a result, every single unit in a sample is a trading day, and we can predict a sequence of movements \(y = [y_1, \ldots, y_T]\). The main target is \(y_T\) while the remainder \(y^∗ = [y_1, \ldots, y_{T−1}]\) serves as the \textit{temporal auxiliary target}. We use these in addition to the main target to improve prediction accuracy (Section 5.3).

We model the generative process shown in Figure 1. We encode observed market information as a random variable \(X = [x_1; \ldots; x_T]\), from which we generate the \textit{latent driven factor} \(Z = [z_1; \ldots; z_T]\) for our prediction task. For the aforementioned multi-task learning purpose, we aim at modeling the conditional probability distribution \(p_{θ}(y|X) = \int Z p_{θ}(y, Z|X)\) instead of \(p_{θ}(y_T|X)\). We write the following factorization for generation,

\[
p_{θ}(y, Z|X) = p_{θ}(y_T|X, Z) p_{θ}(z_T|z_{<T}, X)
\]

where for a given indexed matrix of \(T\) vectors \([v_1; \ldots; v_T]\), we denote by \(v_{<t}\) and \(v_{≤t}\) the submatrix \([v_1; \ldots; v_{t−1}]\) and the submatrix \([v_1; \ldots; v_t]\), respectively. Since \(y^∗\) is known in generation, we use the posterior \(p_{θ}(z_t|z_{<t}, x_{≤t}, y_t)\), \(t < T\) to incorporate market signals more accurately and only use the prior \(p_{θ}(z_T|z_{<T}, X)\) when generating \(z_T\). Besides, when \(t < T\), \(y_t\) is independent of \(z_{<t}\) while our main prediction target, \(y_T\) is made dependent on \(z_{<T}\) through a temporal attention mechanism (Section 5.3).

We show StockNet modeling the above generative process in Figure 2. In a nutshell, StockNet...
comprises three primary components following a bottom-up fashion,

1. Market Information Encoder (MIE) that encodes tweets and prices to $X$;
2. Variational Movement Decoder (VMD) that infers $Z$ with $X, y$ and decodes stock movements $y$ from $X, Z$;
3. Attentive Temporal Auxiliary (ATA) that integrates temporal loss through an attention mechanism for model training.

5 Model Components

We detail next the components of our model (MIE, VMD, ATA) and the way we estimate our model parameters.

5.1 Market Information Encoder

MIE encodes information from social media and stock prices to enhance market information quality, and outputs the market information input $X$ for VMD. Each temporal input is defined as

$$ x_t = [c_t, p_t] $$

where $c_t$ and $p_t$ are the corpus embedding and the historical price vector, respectively.

The basic strategy of acquiring $c_t$ is to first feed messages into the Message Embedding Layer for their low-dimensional representations, then selectively gather them according to their quality. To handle the circumstance that multiple stocks are discussed in one single message, in addition to text information, we incorporate the position information of stock symbols mentioned in messages as well. Specifically, the layer consists of a forward GRU and a backward GRU for the preceding and following contexts of a stock symbol, $s$, respectively. Formally, in the message corpus of the $t$th trading day, we denote the word sequence of the $k$th message, $k \in [1, K]$, as $W$, where $W_k = s, \ell^* \in [1, L]$, and its word embedding matrix as $E = [e_1; e_2; \ldots; e_L]$. We run the two GRUs as follows,

$$ \overrightarrow{h}_f = \overrightarrow{\text{GRU}}(e_f, \overrightarrow{h}_{f-1}) $$

$$ \overleftarrow{h}_b = \overleftarrow{\text{GRU}}(e_b, \overleftarrow{h}_{b+1}) $$

$$ m = (\overrightarrow{h}_{\ell^*} + \overleftarrow{h}_{\ell^*})/2 $$

where $f \in [1, \ldots, \ell^*]$, $b \in [\ell^*, \ldots, L]$. The stock symbol is regarded as the last unit in both the preceding and the following contexts where the hidden values, $\overrightarrow{h}_{\ell^*}, \overleftarrow{h}_{\ell^*}$, are averaged to acquire the message embedding $m$. Gathering all message embeddings for the $t$th trading day, we have a mes-
sage embedding matrix $M_t \in \mathbb{R}^{d_n \times K}$. In practice, the layer takes as inputs a five-rank tensor for a mini-batch, and yields all $M_t$ in the batch with shared parameters.

Tweet quality varies drastically. Inspired by the news-level attention (Hu et al., 2018), we weight shared parameters.

A mini-batch, and yields all

\[
\text{sage embedding matrix}
\]

where $\zeta(\cdot)$ is the softmax function and $W_{m,u} \in \mathbb{R}^{d_m \times d_m}$, $u_u \in \mathbb{R}^{d_m \times 1}$ are model parameters. Then we compose messages accordingly to acquire the corpus embedding,

\[
c_t = M_t u_t^T,
\]

Since it is the price change that determines the stock movement rather than the absolute price value, instead of directly feeding the raw price vector $\tilde{p}_t = [\tilde{p}_t, \tilde{p}_t, \tilde{p}_t]$ comprising of the adjusted closing, highest and lowest price on a trading day $t$, into the networks, we normalize it with its last adjusted closing price, $p_t = \tilde{p}_t / \tilde{p}_{t-1} - 1$. We then concatenate $c_t$ with $p_t$ to form the final market information input $x_t$ for the decoder.

### 5.2 Variational Movement Decoder

The purpose of VMD is to recurrently infer and decode the latent driven factor $Z$ and the movement $y$ from the encoded market information $X$.

**Inference**

While latent driven factors help to depict the market status leading to stock movements, the posterior inference in the generative model shown in Eq. (2) is intractable. Following the spirit of the VAE, we use deep neural networks to fit latent distributions, i.e. the prior $p_\theta (z_t | z_{<t}, x_{\leq t})$ and the posterior $p_\theta (z_t | z_{<t}, x_{\leq t}, y_t)$, and sidestep the intractability through neural approximation and reparameterization (Kingma and Welling, 2013; Rezende et al., 2014). We first employ a variational approximator $q_\phi (z_t | z_{<t}, x_{\leq t}, y_t)$ for the intractable posterior. We observe the following factorization,

\[
q_\phi (Z | X, y) = \prod_{t=1}^{T} q_\phi (z_t | z_{<t}, x_{\leq t}, y_t).
\]

Neural approximation aims at minimizing the Kullback-Leibler divergence between the $q_\phi (Z | X, y)$ and $p_\theta (Z | X, y)$. Instead of optimizing it directly, we observe that the following equation naturally holds,

\[
\log p_\theta (y | X) = \mathbb{D}_\text{KL} [q_\phi (Z | X, y) \| p_\theta (Z | X, y)] + \mathbb{E}_{q_\phi (Z | X, y)} \left[ \log p_\theta (y | X, Z) \right] - \mathbb{D}_\text{KL} [q_\phi (Z | X, y) \| p_\theta (Z | X)]
\]

where $\mathbb{D}_\text{KL} [q \| p]$ is the Kullback-Leibler divergence between the distributions $q$ and $p$. Therefore, we equivalently maximize the following variational recurrent lower bound by plugging Eq. (2, 9) into Eq. (10),

\[
\mathcal{L} (\theta, \phi; X, y) = \sum_{t=1}^{T} \mathbb{E}_{q_\phi (z_t | z_{<t}, x_{\leq t}, y_t)} \left[ \log p_\theta (y_t | x_{\leq t}, z_{<t}) - \mathbb{D}_\text{KL} [q_\phi (z_t | z_{<t}, x_{\leq t}, y_t) \| p_\theta (z_t | z_{<t}, x_{\leq t})] \right] \leq \log p_\theta (y | X)
\]

where the likelihood term

\[
p_\theta (y_t | x_{\leq t}, z_{<t}) = \begin{cases} p_\theta (y_t | x_{\leq t}, z_t), & \text{if } t < T \\ p_\theta (y_T | X, Z), & \text{if } t = T. \end{cases}
\]

**Decoding**

As per time series, VMD adopts an RNN with a GRU cell to extract features and decode stock signals recurrently,

\[
h^t_x = \text{GRU}(x_t, h^t_{x_{t-1}}).
\]

We let the approximator $q_\phi (z_t | z_{<t}, x_{\leq t}, y_t)$ subject to a standard multivariate Gaussian distribution $\mathcal{N}(\mu, \delta^2 I)$. We calculate $\mu$ and $\delta$ as

\[
\mu_t = W^\phi_{z,\mu} h^t_x + b^\phi_{\mu}
\]

\[
\log \delta^2_t = W^\phi_{z,\delta} h^t_x + b^\phi_{\delta}
\]
and the shared hidden representation \( h_t^* \) as
\[
\begin{align*}
\ h_t^* &= \tanh(W_x^\theta [z_{t-1}, x_t, h_t^g, y_t] + b_\mu^g) \\
\end{align*}
\]
where \( W_x^\theta, W_z^\theta, W_y^\theta \) are weight matrices and \( b_\mu^g, b_\delta^g, b_\epsilon^g \) are biases.

Since Gaussian distribution belongs to the “location-scale” distribution family, we can further reparameterize \( z_t \) as
\[
\begin{align*}
\ z_t &= \mu_t + \delta_t \odot \epsilon \\
\end{align*}
\]
where \( \odot \) denotes an element-wise product. The noise term \( \epsilon \sim \mathcal{N}(0, I) \) naturally involves stochastic signals in our model.

Similarly, we let the prior \( p_\theta (z_t | z_{<t}, x_{<t}) \sim \mathcal{N}(\mu_t, \delta_t I) \). Its calculation is the same as that of the posterior except the absence of \( y_t \) and independent model parameters,
\[
\begin{align*}
\mu_t' &= W_{o,\mu}^\theta h_t^* + b_{\mu}^g \\
\log \delta_t^2 &= W_{o,\delta}^\theta h_t^* + b_{\delta}^g
\end{align*}
\]
where
\[
\begin{align*}
\ h_t^* &= \tanh(W_x^\theta [z_{t-1}, x_t, h_t^g] + b_\epsilon^g). \\
\end{align*}
\]

Following Zhang et al. (2016), differently from the posterior, we set the prior \( z_t = \mu_t' \) during decoding. Finally, we integrate deterministic features and the final prediction hypothesis is given as
\[
\begin{align*}
\ g_t &= \tanh(W_g[x_t, h_t^g, z_t] + b_g) \\
\ \hat{y}_t &= \zeta(W_g g_t + b_y), t < T \\
\end{align*}
\]
where \( W_g, W_y \) are weight matrices and \( b_g, b_y \) are biases. The softmax function \( \zeta(\cdot) \) outputs the confidence distribution over up and down. As introduced in Section 4, the decoding of the main target \( \hat{y}_T \) depends on \( z_{<T} \) and thus lies at the interface between VMD and ATA. We will elaborate on it in the next section.

5.3 Attentive Temporal Auxiliary

With the acquisition of a sequence of auxiliary predictions \( \hat{y}^* = [\hat{y}_1; \ldots; \hat{y}_{T-1}] \), we incorporate two-folded auxiliary effects into the main prediction and the training objective flexibly by first introducing a shared temporal attention mechanism.

Since each hypothesis of a temporal auxiliary contributes unequally to the main prediction and model training, as shown in Figure 3, temporal attention calculates their weights in these two contributions by employing two scoring components: an

![Figure 3: The temporal attention in our model.](image)

**Information Score** and a **dependency score**. Specifically,
\[
\begin{align*}
\ v_t' &= w_t^1 \tanh(W_{g,t} G^*) \\
\ v_t^d &= \hat{y}_t \tanh(W_{g,d} G^*) \\
\ v^* &= \zeta(v_t' \odot v_t^d)
\end{align*}
\]
where \( W_{g,t}, W_{g,d} \in \mathbb{R}^{d_g \times d_g}, w_t \in \mathbb{R}^{d_g \times 1} \) are model parameters. The integrated representations \( G^* = [g_1; \ldots; g_{T-1}] \) and \( g_T \) are reused as the final representations of temporal market information. The information score \( v_t' \) evaluates historical trading days as per their own information quality, while the dependency score \( v_t^d \) captures their dependencies with our main target. We integrate the two and acquire the final normalized attention weight \( v^* \in \mathbb{R}^{1 \times (T-1)} \) by feeding their element-wise product into the softmax function.

As a result, the main prediction can benefit from temporally-close hypotheses have been made and we decode our main hypothesis \( \hat{y}_T \) as
\[
\begin{align*}
\ \hat{y}_T &= \zeta(W_T [\hat{y}^* v^*, T] + b_T)
\end{align*}
\]
where \( W_T \) is a weight matrix and \( b_T \) is a bias.

As to the model objective, we use the Monte Carlo method to approximate the expectation term in Eq. (11) and typically only one sample is used for gradient computation. To incorporate varied temporal importance at the objective level, we first break down the approximated \( L \) into a series of temporal objectives \( f_t \in \mathbb{R}^{T \times 1} \) where \( f_t \) comprises a likelihood term and a KL term for a trading day \( t \),
\[
\begin{align*}
\ f_t &= \log p_\theta (y_t | x_{<t}, z_{<t}) \\
\ &\ &\ - \log \lambda D_{KL} [q_\phi (z_t | x_{<t}, x_{<t}, y_t) \| p_\theta (z_t | x_{<t}, x_{<t})]
\end{align*}
\]
where we adopt the KL term annealing trick (Bowman et al., 2016; Semeniuta et al., 2017) and add a linearly-increasing KL term weight $\lambda \in (0, 1]$ to gradually release the KL regularization effect in the training procedure. Then we reuse $v^*$ to build the final temporal weight vector $v \in \mathbb{R}^{1 \times T}$,

$$v = [\alpha v^*, 1]$$ (28)

where $1$ is for the main prediction and we adopt the auxiliary weight $\alpha \in [0, 1]$ to control the overall auxiliary effects on the model training. $\alpha$ is tuned on the development set and its effects will be discussed at length in Section 6.5. Finally, we write the training objective $F$ by recomposition,

$$F(\theta, \phi; X, y) = \frac{1}{N} \sum_{n} v^{(n)} f^{(n)}$$ (29)

where our model can learn to generalize with the selective attendance of temporal auxiliary. We take the derivative of $F$ with respect to all the model parameters $\{\theta, \phi\}$ through backpropagation for the update.

6 Experiments

In this section, we detail our experimental setup and results.

6.1 Training Setup

We use a 5-day lag window for sample construction and 32 shuffled samples in a batch. The maximal token number contained in a message and the maximal message number on a trading day are empirically set to 30 and 40, respectively, with the excess clipped. Since all tweets in the batched samples are simultaneously fed into the model, we set the word embedding size to 50 instead of larger sizes to control memory costs and make model training feasible on one single GPU (11GB memory). We set the hidden size of Message Embedding Layer to 100 and that of VMD to 150. All weight matrices in the model are initialized with the fan-in trick and biases are initialized with zero. We train the model with an Adam optimizer (Kingma and Ba, 2014) with the initial learning rate of 0.001. Following Bowman et al. (2016), we use the input dropout rate of 0.3 to regularize latent variables. Tensorflow (Abadi et al., 2016) is used to construct the computational graph of StockNet and hyper-parameters are tweaked on the development set.

6.2 Evaluation Metrics

Following previous work for stock prediction (Xie et al., 2013; Ding et al., 2015), we adopt the standard measure of accuracy and Matthews Correlation Coefficient (MCC) as evaluation metrics. MCC avoids bias due to data skew. Given the confusion matrix

$$
\begin{pmatrix}
tp & fn \\
fp & tn
\end{pmatrix}
$$

containing the number of samples classified as true positive, false positive, true negative and false negative, MCC is calculated as

$$
MCC = \frac{tp \times tn - fp \times fn}{\sqrt{(tp + fp)(tp + fn)(tn + fp)(tn + fn)}}.
$$ (30)

6.3 Baselines and Proposed Models

We construct the following five baselines in different genres.

- **RAND**: a naive predictor making random guess in up or down.
- **ARIMA**: Autoregressive Integrated Moving Average, an advanced technical analysis method using only price signals (Brown, 2004).
- **RANDFOREST**: a discriminative Random Forest classifier using Word2vec text representations (Pagolu et al., 2016).
- **TSLDA**: a generative topic model jointly learning topics and sentiments (Nguyen and Shirai, 2015).
- **HAN**: a state-of-the-art discriminative deep neural network with hierarchical attention (Hu et al., 2018).

To make a detailed analysis of all the primary components in StockNet, in addition to **HEDGEFUNDANALYST**, the fully-equipped StockNet, we also construct the following four variations.

- **TECHNICALANALYST**: the generative StockNet using only historical prices.
- **FUNDAMENTALANALYST**: the generative StockNet using only tweet information.
- **INDEPENDENTANALYST**: the generative StockNet without temporal auxiliary targets.

9Typically the lag size is set between 3 and 10. As introduced in Section 4, trading days are treated as basic units in StockNet and 3 calendar days are thus too short to guarantee the existence of more than one trading day in a lag, e.g. the prediction for the movement of Monday. We also experiment with 7 and 10 but they do not yield better results than 5.

10We do not treat event-driven models as comparable methods since our model uses no event pre-extraction tool.
- **DISCRIMINATIVE** analyse: the discriminative StockNet directly optimizing the likelihood objective. Following Zhang et al. (2016), we set $z_t = \mu'_t$ to take out the effects of the KL term.

### 6.4 Results

Since stock prediction is a challenging task and a minor improvement usually leads to large potential profits, the accuracy of 56% is generally reported as a satisfying result for binary stock movement prediction (Nguyen and Shirai, 2015). We show the performance of the baselines and our proposed models in Table 1. TLSDA is the best baseline in MCC while HAN is the best baseline in accuracy. Our model, **HEDGFUND** analyse, achieves the best performance of 58.23 in accuracy and 0.080796 in MCC, outperforming TLSDA and HAN with 4.16, 0.59 in accuracy, and 0.015414, 0.028996 in MCC, respectively.

Though slightly better than random guess, classic technical analysis, e.g. ARIMA, does not yield satisfying results. Similar in using only historical prices, **TECHNICAL** analyse shows an obvious advantage in this task compared ARIMA. We believe there are two major reasons: (1) **TECHNICAL** analyse learns from training data and incorporates more flexible non-linearity; (2) our test set contains a large number of stocks while ARIMA is more sensitive to peculiar sequence stationarity. It is worth noting that **FUNDAMENTAL** analyse, **IND**endent analyse and **DISCRIMINATIVE** analyse achieve the best performance of 58.23 in accuracy and 0.080796 in MCC, respectively.

Table 1: Performance of baselines and StockNet variations in accuracy and MCC.

<table>
<thead>
<tr>
<th>Baseline models</th>
<th>Acc.</th>
<th>MCC</th>
<th>StockNet variations</th>
<th>Acc.</th>
<th>MCC</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>RAND</strong></td>
<td>50.89</td>
<td>-0.002266</td>
<td><strong>TECHNICAL</strong> analyse</td>
<td>54.96</td>
<td>0.016456</td>
</tr>
<tr>
<td><strong>ARIMA</strong> (Brown, 2004)</td>
<td>51.39</td>
<td>-0.020588</td>
<td><strong>FUNDAMENTAL</strong> analyse</td>
<td>58.23</td>
<td>0.071704</td>
</tr>
<tr>
<td><strong>RANDFOREST</strong> (Pagolu et al., 2016)</td>
<td>53.08</td>
<td>0.012929</td>
<td><strong>INDEPENDENT</strong> analyse</td>
<td>57.54</td>
<td>0.036610</td>
</tr>
<tr>
<td><strong>TLSDA</strong> (Nguyen and Shirai, 2015)</td>
<td>54.07</td>
<td><strong>0.065382</strong></td>
<td><strong>DISCRIMINATIVE</strong> analyse</td>
<td>56.15</td>
<td>0.056493</td>
</tr>
<tr>
<td><strong>HAN</strong> (Hu et al., 2018)</td>
<td><strong>57.64</strong></td>
<td>0.051800</td>
<td><strong>HEDGFUND</strong> analyse</td>
<td><strong>58.23</strong></td>
<td><strong>0.080796</strong></td>
</tr>
</tbody>
</table>

Figure 4: (a) Performance of **HEDGFUND** analyse with varied $\alpha$, see Eq. (28). (b) Performance of **DISCRIMINATIVE** analyse with varied $\alpha$.

**HEDGFUND** analyse gains exceptionally competitive results with only 0.009092 less in MCC than **HEDGFUND** analyse. The performance of **FUNDAMENTAL** analyse and **TECHNICAL** analyse confirm the positive effects from tweets and historical prices in stock movement prediction, respectively. As an effective ensemble of the two market information, **HEDGFUND** analyse gains even better performance.

Compared with **DISCRIMINATIVE** analyse, the performance improvements of **HEDGFUND** analyse are not from enlarging the networks, demonstrating that modeling underlying market status explicitly with latent driven factors indeed benefits stock movement prediction. The comparison with **INDEPENDENT** analyse also shows the effectiveness of capturing temporal dependencies between predictions with the temporal auxiliary. However, the effects of the temporal auxiliary are more complex and will be analyzed further in the next section.

### 6.5 Effects of Temporal Auxiliary

We provide a detailed discuss of how the temporal auxiliary affects model performance. As introduced in Eq. (28), the temporal auxiliary weight $\alpha$ controls the overall effects of the objective-level temporal auxiliary to our model. Figure 4 presents how the performance of **HEDGFUND** analyse and **DISCRIMINATIVE** analyse fluctuates with $\alpha$.

As shown in Figure 4, enhanced by the temporal auxiliary, **HEDGFUND** analyse approaches the best performance at 0.5, and **DISCRIMINATIVE** analyse
achieves its maximum at 0.7. In fact, objective-level auxiliary can be regarded as a denoising regularizer: for a sample with a specific movement as the main target, the market source in the lag can be heterogeneous, e.g. affected by bad news, tweets on earlier days are negative but turn to positive due to timely crises management. Without temporal auxiliary tasks, the model tries to identify positive signals on earlier days only for the main target of rise movement, which is likely to result in pure noise. In such cases, temporal auxiliary tasks help to filter market sources in the lag as per their respective aligned auxiliary movements. Besides, from the perspective of training variational models, the temporal auxiliary helps \textsc{HedgeFundAnalyst} to encode more useful information into the latent driven factor $Z$, which is consistent with recent research in VAEs (Semeniuta et al., 2017). Compared with \textsc{HedgeFundAnalyst} that contains a KL term performing dynamic regularization, \textsc{DiscriminativeAnalyst} requires stronger regularization effects coming with a bigger $\alpha$ to achieve its best performance.

Since $y^*$ also involves in generating $y_T$ through the temporal attention, tweaking $\alpha$ acts as a trade-off between focusing on the main target and generalizing by denoising. Therefore, as shown in Figure 4, our models do not linearly benefit from incorporating temporal auxiliary. In fact, the two models follow a similar pattern in terms of performance change: the curves first drop down with the increase of $\alpha$, except the MCC curve for \textsc{DiscriminativeAnalyst} rising up temporarily at 0.3. After that, the curves ascend abruptly to their maximums, then keep descending till $\alpha = 1$. Though the start phase of increasing $\alpha$ even leads to worse performance, when auxiliary effects are properly introduced, the two models finally gain better results than those with no involvement of auxiliary effects, e.g. \textsc{IndependentAnalyst}.

7 Conclusion

We demonstrated the effectiveness of deep generative approaches for stock movement prediction from social media data by introducing StockNet, a neural network architecture for this task. We tested our model on a new comprehensive dataset and showed it performs better than strong baselines, including implementation of previous work. Our comprehensive dataset is publicly available at https://github.com/yumoxu/stocknet-dataset.

Acknowledgments

The authors would like to thank the three anonymous reviewers and Miles Osborne for their helpful comments. This research was supported by a grant from Bloomberg and by the H2020 project SUMMA, under grant agreement 688139.

References


Steven Bird, Ewan Klein, and Edward Loper. 2009. \textit{Natural language processing with Python: analyzing text with the natural language toolkit}. O’Reilly Media, Inc.


