ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) | 978-1-7281-7605-5/20/$31.00 ©2021 IEEE | DOI: 10.1109/ICASSP39728.2021.9414893

STOCK MOVEMENT PREDICTION AND PORTFOLIO MANAGEMENT VIA MULTIMODAL LEARNING WITH TRANSFORMER

Divyanshu Daiya*, *Che Lin*†

*Department of Computer Science & Engineering, The LNM Institute of Information Technology, India
† Department of Electrical Engineering & Graduate Institute of Communication Engineering,
National Taiwan University, Taiwan
*daiyadivyanshu@gmail.com, †chelin@ntu.edu.tw

ABSTRACT

This paper introduces a novel high performing multimodal deep learning architecture(Trans-DiCE) for stock movement prediction utilizing financial indicators and news data. Our multimodal architecture uses dilated causal convolutions and Transformer blocks for feature extraction from both data sources. The masked multi-head self-attention layers inside Transformers preserve causality and improve features based on contextual information. To integrate the derived multimodal model representations, we use stacked Transformer blocks. We show empirically that our model performs best compared to state-of-the-art baseline methods for S&P 500 index and individual stock prediction and provides a significant 3.45% improvement from 74.29% to 77.74%. We also demonstrate our model's utility for the Portfolio Management task. We propose a Deep Reinforcement Learning Framework utilizing Trans-DiCE for Portfolio Optimization, providing noticeable gain on Sharpe Ratio and 7.9% increase in Portfolio Value over the existing state of the art Models.

Index Terms— FinTech, Deep Learning, Multimodal Learning, Transformer, Reinforcement Learning

1. INTRODUCTION

Stock movement prediction has long intrigued investors, given the dividends paid by sound market analysis. Given the stochastic nature and volatility, it has long been a prominent and challenging subject for researchers too [1–4]. We present a model to predict stock price movement using stock data and news. Given the nonstationary nature of the stock market, its always been hard to make a satisfactory prediction, but with the onset of deep learning architectures lately, many studies have provided substantial improvements over conventional architectures. Earlier time series modeling approaches tended more to recurrent models, given the inherent nature of these models to work with the temporal data. However, the recent use of diversified deep learning architectures such as convolutional neural network variants [5], hybrid variants [6], and Transformers [7–9] has shown to provide better performance for time series modeling tasks. Some best performing stock market studies such as, [10] use dual attention augmented recurrent neural networks which performed best for the stock price regression tasks, [11] employed a hybrid model integrating the convolutional and recurrent neurons for stock movement prediction, [12] used dilated causal convolutional network and neural tensor network for stock movement prediction using news and stock data. Dilated causal convolutional network is a derivative CNN architecture developed by Google DeepMind as WaveNet [13]. It utilized dilated causal convolutions with residual connections and has provided better or comparable results to LSTMs and GRUs [14]. We build on our previous work [12] and employ a dilated causal convolutional neural network (DC-CNN), for extracting features from the financial time series.

The volatile nature of the stock market is hard to capture using only the financial indicators, i.e., the stock data as the market is dependent on a variety of social, political, and economic factors. So to better capture the market situation, data from either social media or news can is used. We are using the news data. The existing researches using heterogeneous data include event-driven approaches utilizing structured event representation from news data [3, 12, 15], use of hierarchical attention mechanisms on related news sequences for stock trend prediction [16], and feature engineering [11]. We used the extension of Ding's [3] event representations proposed in our previous work [12]. Compared to word and sentence embeddings, structured event embeddings are more useful because they inherently capture associations in the text by extracting subject-action-object pairs [17]. Most of the existing models employ recurrent networks [6], while we utilized hybrid architectures and neural tensor networks in our previous work [12]. In another notable work, Transformer-encoders are used to extract the features from tweets and stock

†Corresponding Author
Code will be made publicly available.
(*Please zoom to view the figures, tables, and graphs.*)

data and further using capsule networks for predictions [18]. Across many NLP and time-domain analysis tasks, Transformers have provided significant improvements over the existing models [19, 20]. Given the effectiveness of Transformers for natural language processing tasks and temporal modeling, we propose the use of Transformer-encoders(ETE) to extract features from the event embedding. It might seem fitting to utilize Transformers for financial time-series too, but the low input variable space is a problem since their performance falls with small embedding and requires much tuning to perform. The concatenated features from DC-CNN and event-Transformer-encoders(ETE) are passed through Transformer blocks to couple features from both the models and provide context-dependent updates, which are then used for prediction by DC-CNN blocks and feed-forward network.

2. PRELIMINARIES

In the paper, we have used the Transformer architecture proposed by [7]. All the implementation and structural details are kept the same. All the references to Transformer Encoder is a reference to the Transformer's Encoder block in [7] architecture. Other ideas used in the paper have been discussed in brief below and also later as required.

Dilated Causal Convolution Convolutional neural networks (CNN) have been one of the most successful architectures used to extract features [21]. However, for time-series, CNNs fell short of providing a better receptive field without needing many convolutional layers or large filter sizes. This was attended by Wavenet architecture, which introduced CNNs with a causal, temporal bias [13] for audio signals to encode long-range temporal dependencies. Instead of the conventional convolutions, they used dilated convolutions, which utilizes the dilation rate. The dilation rate is the number of input values filter skip, thereby enabling the network to have a larger receptive field without parameter overhead. Note that dilation for traditional convolution is one. Also, skip-connection is used in the architecture to keep the features captured in the previous layers.

Event Embedding To better use the information from news, we first use the open information extraction (OpenIE) provided by Stanford University to turn a simple sentence into structural tuples, which was the subject-relation-object tuple, noted as $E = (E_1, R, E_2)$. Take "Google buys YouTube" as an example. The corresponding tuple is the Google($E1$, the subject)-buys(R, the relation or action)-YouTube($E2$, the object). We use the event embedding generating model, $InvED$, proposed in our previous work [12].

3. ARCHITECTURE

Our proposed architecture Transformer Dilated Convolution and Event Network (Trans-DiCE) (Figure 2) has three segments. Two segments are used to handle feature extraction from financial indicators and news data, respectively. The other one is used to effectively merge the extracted features from these segments to make predictions. As discussed earlier, for extracting features from financial indicators, we employ Att-DCNN, and for event embedding, i.e., the news data, we use Transformer-Encoders, i.e., Event Transformer Encoder Network (ETE). We generate Y_{DC-CNN} and Y_{ETE} as the output from DC-CNN and ETE, respectively, with $Y_{DC-CNN} \in \mathbb{R}^{a \times L}$ and $Y_{ETE} \in \mathbb{R}^{b \times L}$, where L is the number of days used to make a prediction, a number of filters in final DC-CNN layer, and b being embedding size of Transformer. Then, we concatenate the outputs from both of these segments to get $Y_{concat} \in \mathbb{R}^{(a+b) \times L}$ and use Transformer encoder blocks to extract features from the merged representation. Coupled Transformer and convolutional networks have shown to perform well as it gains from equivariance and invariance of convolutional networks while MHA in Transformer encoder blocks provides better context-driven extraction by retaining information from most relevant features [7, 22]. We stack DC-CNN blocks over Transformer encoder blocks in the final segment as previous researches have identified that the use of a single RNN layer over Transformer output provided a significant performance boost [23]. After DC-CNN blocks, the output is then flattened and passed through a feed-forward layer of dimension 100 with $ReLU$ [24] activation and L2 regularization. Finally, we apply dropout followed by a $softmax$ layer to obtain the final output probabilities by a feed-forward layer for movement prediction, whether the price is going *up* or *down*.

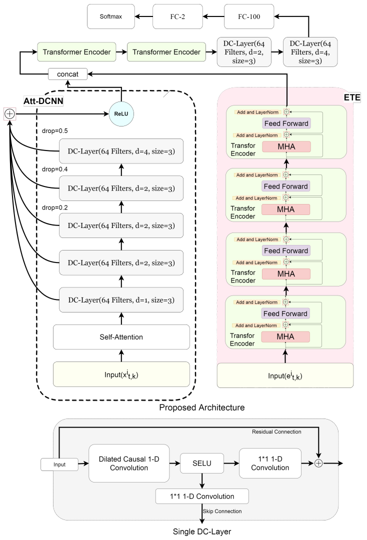

Fig. 1. Trans-DiCE[1]

3.1. Att-DCNN

We use Att-DCNN proposed in our previous work [12] for extracting features from the Financial Indicators. We use five stacked DC-CNN blocks with skip connections as described by [12]; all the settings and hyperparameters are kept the same. We do not use the fully connected layer as in our previous work [12] and pass output directly as in Figure 2. We input $x_{t,L}$, where t is the number of time variables and L being the length of the time window used for prediction, we get $Y_{DC-CNN} \in \mathbb{R}^{a \times L}$ as output with $a = 64$ as the number of filters in the last DC-CNN block.

3.2. ETE

We use four stacked Transformer encoders with each encoder block consisting specific combination of multi-head self-attention, residual connections, layer normalization and feed-forward layers, as we described earlier. We maintain the causality by using *Mask* function. We have used 5 attention heads and the output embedding size of 100 and *ReLU* activation for feed-forward layers with no dropout. For each event tuple we pass into *InvED*, we generate C and C_{inv}, and concatenate them into $C_T \in \mathbb{R}^{2d}$, where $d = 100$ is input embedding size. We aggregate events for a particular day to generate the series $E_{2d,L_{TotalDays}}$, with $L_{TotalDays} - k + 1$ training instances $e^i_{2d,k}$ each using k days, where $i \in (1, L_{TotalDays} - k + 1)$, $t = 2d$. The training instance $e^i_{2d,k}$ is passed into our model to obtain the output $Y_{ETE} \in \mathbb{R}^{b \times L}$, where $b = 100$ is the output embedding size.

3.3. Transformer DC-CNN Block

After the concatenation of the outputs from Att-DCNN and ETE Y_{concat}, we use two Transformer blocks with the same parameters as used in ETE, and stack them with three DC-CNN blocks.

4. PORTFOLIO MANAGEMENT

Portfolio management (PM) [25] is a financial planning task that maximizes forecasted profits via asset allocation. A market is made up of many assets and related information, e.g., prices and other factors that affect the market. We assume the market is amply liquid such that any transactions can be executed instantly with minimal market impact. For PM, we consider the situation that there is an ML algorithm that can gather all viable data from the market and then gradually improves its trading strategy by trial-and-error. Here the market comprises of the assets for PM, and other available knowledge is called the environment [26]. Our Trans-DiCE model, which observes the environment and then makes decisions to interact with the market and re-balance the portfolio, is the agent. We have considered the setup that the environment will provide asset prices as an internal data source and will also provide financial news articles as an external data source. The agent has access to all historical prices and news articles up to the current time step for making the asset movement prediction, which, coupled with the historical asset price, can provide better predictive information. A portfolio is a collection of multiple financial assets. A portfolio with M assets has, Portfolio vector, w_t: its i-th component represents the ratio of the total budget invested to the i-th asset, such that:

$$w_t = \begin{bmatrix} w_{1,t}, & w_{2,t}, & \ldots, & w_{M,t} \end{bmatrix}^T \in \mathbb{R}^M \tag{1}$$

where, $\sum_{i=1}^{M} w_{i,t} = 1$. The stocks in our dataset are the assets to be allocated or managed [27]. Simple return, r_t, represents the percentage change in asset price from time $(t-1)$ to time t, such that:

$$r_t \triangleq \frac{p_t - p_{t-1}}{p_{t-1}} = \frac{p_t}{p_{t-1}} - 1 = R_t - 1 \in \mathbb{R} \tag{2}$$

The linear combination of the simple returns of each constituents, weighted by the portfolio vector is called portfolio simple return [28]. Hence, at time index t, we obtain:

$$r_t \triangleq \sum_{i=1}^{M} w_{i,t} r_{i,t} = w_t^T r_t \in \mathbb{R} \tag{3}$$

Our Reward function, $R^T = 1/T * \sum_{t=0}^{T} (\mathcal{R}_t)$, where R^T is the reward function at time T, $\mathcal{R}^t$ is the log return of the portfolio at time t [25, 28]. We define our *policy* $\pi : S \rightarrow A$ as a mapping from state space to action space. The policy J_T is defined by the reward function as mentioned above, is a function of network parameters θ, action space $a_T = \pi_\theta(s_T)$,

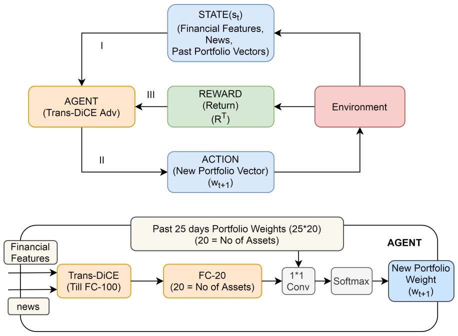

Fig. 2. Reinforcement Learning System[2]

$$J_T = R(s_1, \pi_\theta(s_1)....s_T, \pi_\theta(s_T)),\ \theta \rightarrow \theta + \lambda \bigtriangledown_\theta \left[J_{[t_0,t_f]}\right](\pi_\theta)$$

We use gradient descent to update our parameters with the given learning rate (λ) in the direction of the gradient. [28] The gradient descent process allows us to reach the optimal parameters suited for the trading environment. Complete Model Architecture is described in *Figure 2*. We have also used past asset weights for better future weight estimation and also to control sharp weight changes i.e. to reduce transaction costs [27]. Model was trained with $\lambda = 3e - 4$ for 50000 epochs and batch size of 64.

Models	Index Acc(%)	Avg Acc(%)
Ding, 2014	58.83	61.02
EB-CNN	64.21	64.23
SI-RCNN	63.09	60.89
KGEB-CNN	66.93	64.56
DA-RNN	68.05	68.81
Att-DiCE	73.89	74.29
Trans-DiCE	**77.13**	**77.74**
Trans-DiCE(w/o DC-CNN blocks)	76.22	77.14
Trans-DiCE(w/o Transformer Blocks)	75.11	75.90
Trans-DiCE(with only FF in last segment)	74.09	74.18
Att-DiCE	73.89	74.29
Att-DCNN	72.75	73.36
ETE	72.36	72.78
Att-biNTN	69.14	70.12

Table 1. Accuracy for Index Prediction and Avg. Accuracy[3]

Models	SR(1w)	SR(1m)	SR(3m)	PV(1m)
WMAMR	6.61	0.42	1.74	33.06
DPM	6.44	3.88	2.31	30.47
FILOS	7.82	3.97	2.24	37.35
SARL	7.73	3.83	2.91	37.18
Trans-DiCE	7.99	4.05	3.18	40.51

Table 2. Sharpe Ratio(SP) and Portfolio Value(PV) Comp.

5. NUMERICAL RESULTS AND DISCUSSION

In our experiment, we predict the stock movement of S&P 500 index and 20 individual company, respectively, over the period of October 2006 to November 2013. The financial data includes the original data such as open, high, close, low, volume; other derivative technical indicators commonly used in the financial industry such as moving average convergence divergence (MACD), relative strength index (RSI), stochastic oscillator (SO), rate of change (ROC), on balance value (OBV), weighted moving average (WMA). As for textual data, we obtain publicly available news information that contains various financial articles from Bloomberg and Reuters released by Ding [3]. Extraction and parsing of structured are done as followed by [12]. The train, development, and test split is 80%, 10%, and 10%, respectively. A Time window of 40 days is used, which means closing price movement prediction for a day requires data from the past 40 days. We use Adam optimizer [29] with a starting learning rate of 0.00035 that linearly decays by 0.1 for every 10000 iteration. We train the model for 100 epochs with a batch size of 32. To evaluate the models' quality, we apply the standard measure of accuracy (Acc) and Matthews correlation coefficient (MCC) for accessing performances of the S&P 500 index and individual stock prediction. For the PM task, we have already specified the training routine, and the rest is the same as for TransDiCE. For PM, we have used the top 25 stocks based on the number of news articles available for a particular stock. The data split used for PM is 70%, 15%, and 15%.

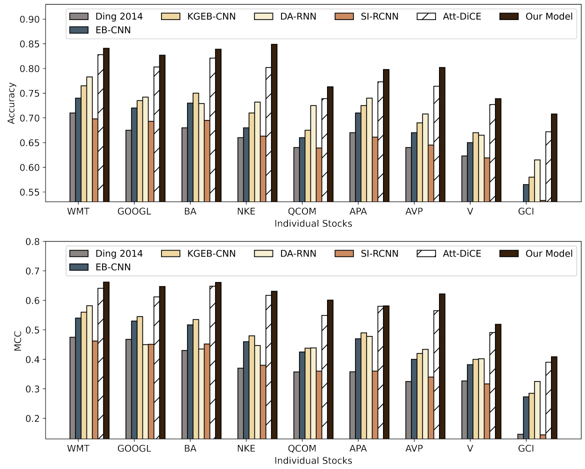

Fig. 3. Accuracy and MCC comparison

The baseline models we compared(for movement prediction) includes SI-RCNN [4], Ding (2014) [30], EB-CNN [3], DA-RNN [10], KGEB-CNN [15], Att-DiCE [12]. We can see that our model provides the best MCC and Acc for index prediction (Table 1), as well as for individual stock prediction (Figure 3). Averaging our model performance over the S&P index and Top 20 S&P 500 companies, we demonstrate that our model provided around 3.5% improvement from 74.29% to 77.74% over the best performing model Att-DiCE (our previous work) [12]. We also tested the individual performance of the Att-DCNN and ETE by removing the ETE and Att-DCNN network from Trans-DiCE, respectively; we were still able to obtain an average model performance of 73.36% and 72.78%, respectively, which is comparable with Att-DiCE, which used both financial indicators and news data. Our proposal that transformers' use might be a good fit to extract features from the concatenated Att-DCNN and ETE outputs stands well. This is evident since without using Transformer encoder blocks, we obtained a 75.90% average model performance compared to 77.74% with their use. Also, we observed that stacking DC-CNN blocks performed better over the Transformer encoder blocks as the accuracy increases from 77.14% to 77.74%. The relevance of Transformer encoder blocks and DC-CNN blocks was also verified by observing that performance dips to just 74.18% if we directly pass concatenated model outputs into a feed-forward network, which is a very significant 3% fall. Our model's utility can be further analyzed through Portfolio Management tasks (*Table 2*). We see that our model using a basic policy and Reward function can match performance and provide improvements over the current state of the art models SARL [31], WMAMR [32], FILOS [28], DPM [27]. We see a clear increase of 0.17, 0.8, 0.27 in Sharpe Ratios(SP) [28] over one week, one month, three months, respectively. We also see a gain of 7.9% in Portfolio Value [28]. The performance is remarkable given we did not use any complex RL algorithm and also used a very basic RL policy and reward function. Further research is likely to provide even better results. We were also able to ascertain the applicability of Transformers for Reinforcement Learning and provide noteworthy results, provided Transformers are hard to optimize for RL tasks. We have empirically demonstrated our proposed architecture's effectiveness for multimodal learning and its relevance for the time series forecasting tasks and Reinforcement Learning. We believe that such success can shed light on future studies for a more accurate stock price movement prediction and optimized portfolio management.

6. REFERENCES

[1] Ryo Akita and Kuniaki Yoshihara, “Deep learning for stock prediction using numerical and textual information,” *15th ICIS*, 2016.

[2] Kai Chen and Zhou, “A LSTM-based method for stock returns prediction: A case study of China stock market,” *2015 IEEE International Conference on Big Data (Big Data).*

[3] Xiao Ding, Ting Zhang, and Junwen Duan, “Deep learning for event-driven stock prediction,” *Twenty-fourth IJCAI*, 2015.

[4] Beatriz SLP Vargas and Alexandre G Evsukoff, “Deep learning for stock market prediction from financial news articles,” *2017 IEEE CIVEMSA.*

[5] Jiuxiang Gu and Wang, “Recent advances in convolutional neural networks,” *Pattern Recognition*, 2018.

[6] Hassan Ismail Fawaz, “Deep learning for time series classification: a review,” *Data Mining and Knowledge Discovery*, 2019.

[7] Ashish Vaswani and Shazeer, “Attention is all you need,” *NIPS.*

[8] Shiyang Li and Jing, “Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting,” in *NIPS*, 2019.

[9] Jeeheh Oh and Wang, “Learning to exploit invariances in clinical time-series data using sequence transformer networks,” 2018.

[10] Yao Qin and Song, “A dual-stage attention-based recurrent neural network for time series prediction,” *arXiv preprint arXiv:1704.02971*, 2017.

[11] Wen Long and Lu, “Deep learning-based feature engineering for stock price movement prediction,” *Knowledge-Based Systems*, 2019.

[12] D. Daiya, M. Wu, and C. Lin, “Stock movement prediction that integrates heterogeneous data sources using dilated causal convolution networks with attention,” in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 8359–8363.

[13] Aaron van den Oord, “WaveNet: A generative model for raw audio,” *arXiv preprint arXiv:1609.03499*, 2016.

[14] Akira Tamamori, “Speaker-dependent WaveNet vocoder.,” *Interspeech*, 2017.

[15] Xiao Ding, Ting Zhang, and Junwen Duan, “Knowledge-driven event embedding for stock prediction,” *COLING 2016*, 2016.

[16] Ziniu Hu and Liu, “Listening to chaotic whispers: A deep learning framework for news-oriented stock trend prediction,” in *11th ACM ICWDM*, 2018.

[17] Yates, “Textrunner: open information extraction on the web,” *Proceedings of Human Language Technologies*, 2007.

[18] Jintao Liu and Lin, “Transformer-based capsule network for stock movement prediction,” in *Proceedings of the First Workshop on Financial Technology and Natural Language Processing*, 2019.

[19] Cheng-Zhi Anna Huang and Vaswani, “Music transformer,” *arXiv preprint arXiv:1809.04281*, 2018.

[20] Zihang Dai and Yang, “Transformer-xl: Attentive language models beyond a fixed-length context,” *arXiv preprint arXiv:1901.02860*, 2019.

[21] Aloysius, “A review on deep convolutional neural networks,” in *2017 (ICCSP).*

[22] Daniel Povey and Hadian, “A time-restricted self-attention layer for asr,” *2018 IEEE ICASSP*, 2018.

[23] Mia Xu Chen, “The best of both worlds: Combining recent advances in neural machine translation,” in *Proceedings 56th ACL (Volume 1: Long Papers).*

[24] Andrew L Maas and Andrew Ng Hannun, “Rectifier nonlinearities improve neural network acoustic models,” 2013.

[25] Liang, “Adversarial deep reinforcement learning in portfolio management,” *arXiv preprint arXiv:1808.09940*, 2018.

[26] Zihao Zhang, “Deep reinforcement learning for trading,” *The Journal of Financial Data Science*, 2020.

[27] Jiang, “A deep reinforcement learning framework for the financial portfolio management problem,” *arXiv preprint arXiv:1706.10059.*

[28] Angelos Filos, “Reinforcement learning for portfolio management,” 2019.

[29] Diederik P Kingma and Jimmy Ba, “Adam,” .

[30] Xiao Ding and Zhang, “Using structured events to predict stock price movement: An empirical investigation,” *2014 EMNLP*, 2014.

[31] “Reinforcement-learning based portfolio management with augmented asset movement prediction states,” .

[32] Li Gao and Weiguo Zhang, “Weighted moving average passive aggressive algorithm for online portfolio selection,” IEEE, 2013.