A Self-Attention Deep Residual Network Approach to Stock Price Prediction

Zhiwei Lin
Department of Applied Mathematics and Statistics
Stony Brook University

Abstract

Drawing inspiration from the success of deep learning in various fields and its increasing applications in finance, this paper introduces a deep residual network (ResNet) model for predicting stock prices. ResNet is widely known for its exceptional accuracy in capturing intricate patterns in image data. This paper investigates its potential of capturing patterns in financial time series data. Additionally, a self-attention layer is integrated with traditional ResNet to see if accuracy is improved.

1 Introduction

In the field of financial time series, conventional statistical approaches often perform under several assumptions, which might not always hold in the real world stock price. For example, ARIMA, introduced by Box et al. [1], is a widely used time series forecasting model. However, stock data is non-linear and non-stationary, presenting challenges for ARIMA model. In recent years, sophisticated deep learning models such as RNNs and LSTMs become prominent in predicting stock prices, demonstrating notable performance compared to traditional methods. This paper would explore the performance of a Residual Neural Network (ResNet) in the context of financial data.

ResNet, introduced by He et al. [2], is widely known for its outstanding performance in image recognition. A key innovation of ResNet is the use of residual blocks, $X + F(X)$. This novel approach effectively tackles the challenge of gradient vanishing, making it easier to train very deep neural networks and achieve high accuracy.

However, ResNet is built upon convolutional neural networks (CNN), which focus on capturing local features of the input, but capturing long-range dependencies like stock data may require more complex architectures or additional mechanisms. Therefore, this paper explores the potential of incorporating an attention mechanism into the Residual Neural Network (ResNet) framework for predicting financial data. It is expected the integration of an attention mechanism will improve accuracy.

2 Relate Works

Applying the ResNet architecture to finance stock prices was inspired by Liu's paper – *Stock Price Trend Prediction Model Based on Deep Residual Network and Stock Price Graph* [3]. Liu introduced a deep residual network (ResNet) model for prediction, utilizing the stock price graph as input. This lets us see the potential of ResNet architecture in the financial field.

In 2019, Ramachandran and his team at Google proposed a novel self-attention layer for CNN type of neural network in the paper *Stand-Alone Self-Attention in Vision Models*[4]. Unlike traditional approaches where self-attention mechanisms were typically combined with convolutional layers, this new approach uses a stand-alone self-attention layer as a replacement for the traditional convolutional

Submitted to 37th Conference on Neural Information Processing Systems (NeurIPS 2023). Do not distribute.

layer. This novel approach offers a flexible and powerful mechanism for the model to focus on important information in the input.

3 Method

The daily returns of individual stocks tend to be highly volatile and often unpredictable; it is often not subject to the company's fundamentals, but rather market volatility and investor sentiment. It's challenging for a model to accurately capture the dynamic daily patterns for individual stocks. However, when focusing on forecasting the average return of a portfolio for more than a day, it tends to be less volatile and reflects real movement in stocks

3.1 Data Collection

We use 32 stock data from the U.S. stock market across four different sectors: communication services, consumer cyclical, consumer defensive, and technology. Eight stocks would be chosen from each of the four sectors, forming a total of 32 stocks. For each stock, the *open, close, high, low, volume,* and *daily return* data are recorded from 12/01/2008 to 12/01/2023.

Open: The price of a stock at the beginning of a trading period.

Close: The price of a stock at the end of a trading period.

High: The highest price of a stock during a trading day.

Low: The lowest price of a stock during a trading day.

Volume: The total number of shares or contracts traded for a stocks during a trading day

Daily Return: The percentage change in the price of a stock from the beginning to the end of a trading day.

$$\frac{\text{Close Price} - \text{Open Price}}{\text{Open Price}}$$

Ticker	**Stock Name**	**Sector**
GOOGL	Alphabet Inc.	Communication Services
NTES	NetEase Inc.	Communication Services
NFLX	Netflix Inc.	Communication Services
DIS	The Walt Disney Company	Communication Services
T	AT&T Inc.	Communication Services
VZ	Verizon Communications Inc.	Communication Services
TMUS	T-Mobile US, Inc.	Communication Services
CMCSA	Comcast Corporation	Communication Services
AMZN	Amazon.com Inc.	Consumer Cyclical
BKNG	Booking Holdings Inc.	Consumer Cyclical
HD	The Home Depot, Inc.	Consumer Cyclical
TM	Toyota Motor Corporation	Consumer Cyclical
MCD	McDonald's Corporation	Consumer Cyclical
SBUX	Starbucks Corporation	Consumer Cyclical
NKE	NIKE, Inc.	Consumer Cyclical
LOW	Lowe's Companies, Inc.	Consumer Cyclical
WMT	Walmart Inc.	Consumer Defensive
PG	Procter & Gamble Co.	Consumer Defensive
COST	Costco Wholesale Corporation	Consumer Defensive
KO	The Coca-Cola Company	Consumer Defensive
PEP	PepsiCo, Inc.	Consumer Defensive
PM	Philip Morris International Inc.	Consumer Defensive
UL	Unilever PLC	Consumer Defensive
MDLZ	Mondelez International, Inc.	Consumer Defensive
AAPL	Apple Inc.	Technology
MSFT	Microsoft Corporation	Technology
NVDA	NVIDIA Corporation	Technology
TSM	Taiwan Semiconductor Manufacturing Company Limited	Technology
CRM	Salesforce.com Inc.	Technology
ORCL	Oracle Corporation	Technology
ADBE	Adobe Inc.	Technology
ASML	ASML Holding N.V.	Technology

Table 1: Stock Information

3.2 Rolling Window Technique:

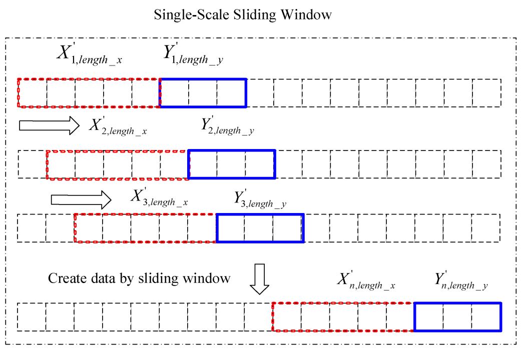


Figure 1: Rolling Window. Source of the image: https://www.mdpi.com/2072-4292/13/16/3328

The training data is constructed by moving a selected time window of 32 days for the *open, close, high, low, volume,* and *daily return* resulting in a 32x32x6 matrix.

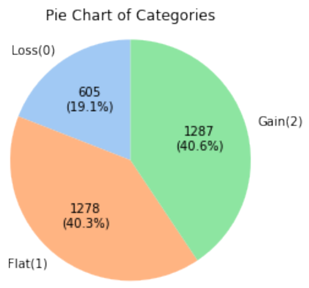
The test data is the classification of the average stock returns for the next 5 days. Let $\bar{R}$ represent the average return of the 32 stocks for the next 5 days.

$$Y = \begin{cases} 0 & \text{if } \bar{R} < h_1 \\ 1 & \text{if } h_1 \leq \bar{R} \leq h_2 \\ 2 & \text{if } \bar{R} > h_2 \end{cases}$$

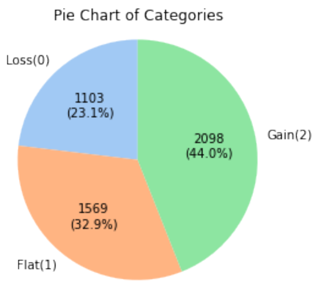
To avoid class imbalance, the cutoff points h_1 and h_2 need to be chosen carefully. The objective is to predict whether Y is 0 (loss), 1 (flat), or 2 (gain) for the next 5 days.

3.3 Data Augmentation and Normalization

There are approximately 252 trading days a year, and 15 years would generate about 3200 training samples. To augment the dataset and extract more information, each matrix will be vertically flipped at a 50% chance. This operation involves exchanging the second half of the stocks with the first half while preserving the information within the array. By using this vertical flipping technique, the dataset is expanded by 50%.



(a) Before data augmentation



(b) After data augmentation

Figure 2: Comparison before and after data augmentation

The cutoff points h_1 and h_2 were chosen to be -0.002 and 0.002, respectively, for this dataset. So, $-h_1 = h_2$, and this choice of cutoff points ensures that the distribution of Y is not highly imbalanced.

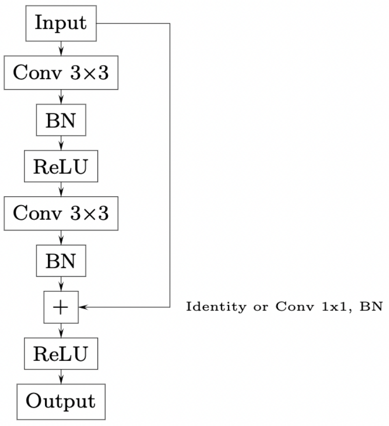
In Liu's paper[3], h_1 is set as 33% of the sequence and h_2 as 66% of the sequence. This results in an equal number of samples in the three categories (loss, flat, and gain). However, I disagree with this choice. First, it may result in $-h_1 \neq h_2$. If $h_1 = -0.002$ and $h_2 = 0.004$, it makes no sense why classify stocks below -0.002 as loss but require them to be above 0.004 to be classified as gain. Second, S&P 500, a well-known stock market index, has reported positive returns on 52.4% of days and negative returns on 47.6% of days during the past decade, according to Crestmont Research[5]. The stocks I picked are all very well-known, and most, if not all, performed better than S&P 500 in the past decade. Therefore, it's common for the number of positive days to be more than the number of negative days.

The dataset would then be normalized. There are six features:the *open, close, high, low, volume,* and *daily return*. Each feature will be normalized with its respective mean and standard deviation.

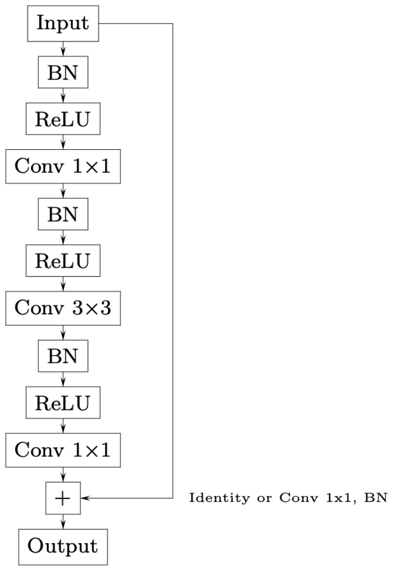
$$x'_\mathrm{i} = \frac{x_\mathrm{i} - \mu_\mathrm{i}}{\sigma_\mathrm{i}} \quad \text{where } i = \textit{open, close, high, low, volume, and daily return}$$

3.4 Residual Network(ResNet)

We use a 32-layer ResNet network(5 residual blocks). Two versions of ResNet would be implemented: Standard ResNet and Pre-activation Bottleneck.



(a) Standard ResNet Block



(b) Pre-activation Bottleneck Block

Figure 3: Standard ResNet Block vs. Pre-activation Bottleneck Block

Input Layer: Input matrices are prepared by concatenating the information from the stocks, resulting in a shape of (B, C, H, W) matrix, where B is the number of batch size, C is the number of features, H is the number of stocks, W is the number of lookback days.
Hidden Layers: A stack of residual blocks is shown in Figure 2. There is a bias term in layers. It's due to:

$$R_i = R_{f_i} + \frac{1}{N}\sum_{j=1}^{N} R_j \quad \text{, where } i \text{ is the } i\text{th day and } j \text{ is the } j\text{th stock}$$

R_{f_i} is the daily risk-free return, and R_j is the excess rate of return. All assets should have the same risk-free return. R_{f_i} is often according to the 3-month U.S. Treasury bill or London bank. Hence, R_{f_i} should be a bias term in the layers.
Output Layer: Classification of average returns, Y, for the selected time window. $Y = 0, 1, \ldots, n$, n represents the number of classifications.

3.5 Stand-Alone Self-Attention

The convolution layer is the fundamental building block in ResNet, but it has certain locality limitations, which makes it perform not very well in long-range dependencies datasets. Incorporating an attention mechanism into ResNet might enhance the model's ability to focus on the important part of the data set. For example, information from the last trading day has the most impact on the stock's movement on the next day and features like previous *returns* might be more important than *open, close, high*, and *low*. Therefore, adding an attention mechanism might help the model focus on more important features and the latest day's information.

Ramachandran et al. [4] paper proposed a self-attention convolution layer to replace the traditional convolution layer, which provides a flexible and powerful way to integrate attention mechanism to ResNet architecture.

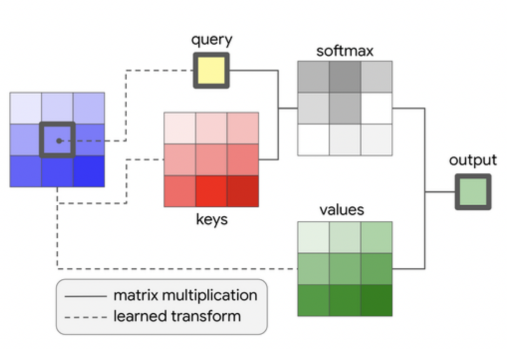


Figure 4: 3X3 self-attention Convolutions. Source: from Stand-Alone Self-Attention in Vision Models paper.

Single-Headed Self-Attention(group=1):

let's denote x as input(given pixel) and y as output of that pixel shown in Figure 3.

$$y = \sum_{a,b \in N_k(i,j)} \text{softmax}_{ab}(q_{ij}^T k_{ab}) v_{ab} \tag{1}$$

where the queries are $q_{ij} = W_Q x_{ij}$, key are $k_{ab} = W_K x_{ab}$ and values are $v_{ab} = W_V x_{ab}$ $W_Q, W_K, W_V \in R^{d_{out} \times d_{in}}$ are learned transforms. The equation (1) is repeated for every input pixel.

Multi-Headed Self-Attention(group>1):

Transferring this concept to multi-headed, the input is partitioned into N groups, and each group is processed separately using single-headed attention. Let H_i represent the output from the i-th attention head. The final output is obtained by concatenating the outputs from all attention heads.

$$y = \sum_{a,b \in N_k(i,j)} \text{softmax}_{ab} \left(q_{ij}^T k_{ab} + q_{ij}^T r_{a-i,b-j} \right) v_{ab} \tag{2}$$

where $ab \in N_k(i,j)$, and ij represents the position of a pixel in an image. The relative distance of ij to each $ab \in N_k(i,j)$ is given by $(a-i, b-j)$.

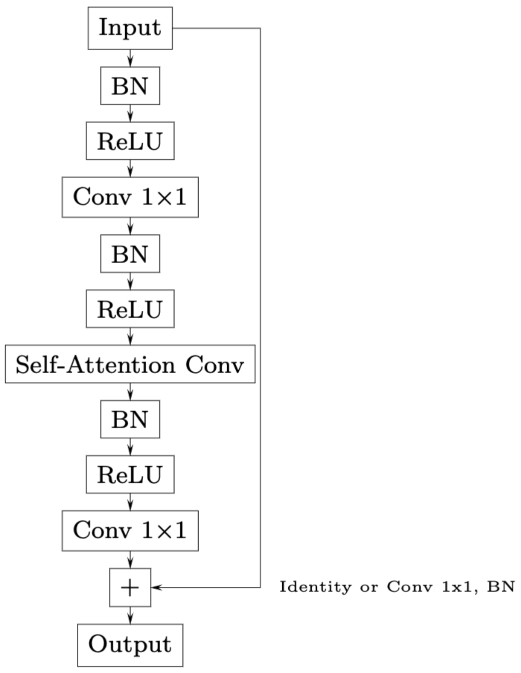


Figure 5: Stand Along Self-Attention ResNet

3x3 spatial convolution is replaced by a self-attention layer. All other structure of the bottleneck remains unchanged.

Ramachandran et al. [4] also proposed an attention stem to replace the convolution stem layer; however, the performance after adding the attention stem does not show any improvement in the original paper. Hence, this technique would not be used here.

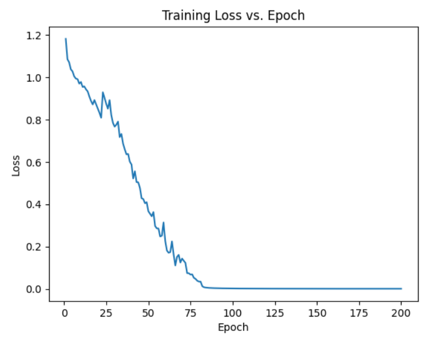
4 Experiment

Parameters I used for this experiment:

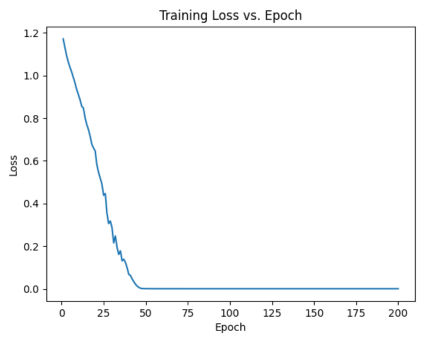
Parameter	**Value**
window_x	32
window_y	5
resnet_size	5
batch_size	16
num_classes	3
first_num_filters	16
weight_decay	2×10^{-4}
learning rate	0.01
epoch	200

Table 2: List of Parameters

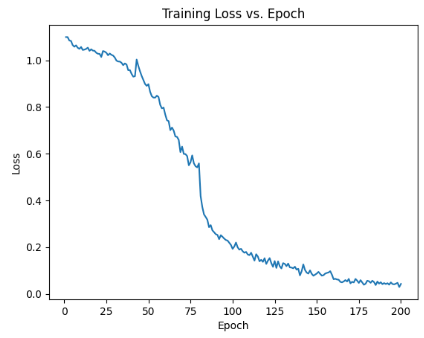
Stochastic Gradient Descent was used with a learning rate decay of 0.1 every 80 epochs.
The results of standard ResNet, Bottleneck, A single-headed (group = 1) Bottleneck, and multi-headed (group = 4) Bottleneck will be shown. The self-attention layer is used to capture global dependencies. A multi-headed self-attention layer can capture more diverse and complex patterns.



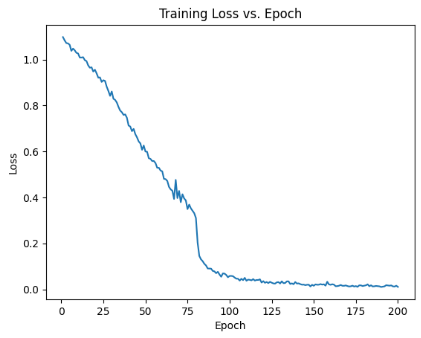
(a) Standard ResNet Training Loss



(b) Pre-Activation Bottleneck Training Loss



(c) Single-Headed Self-Attention Layer Bottleneck Training Loss



(d) Multi-Headed Self-Attention Layer Bottleneck(group=4) Training Loss

Figure 6: Training Loss Figures

Table 3: Performance of the Models

Models	**Test Accuracy**
Standard ResNet	0.3977
Pre-Activation Bottleneck	0.4181
Single-Headed Self-Attention Bottleneck	0.3830
Multi-Headed Self-Attention Bottleneck	0.3887

I implemented standard ResNet and pre-activation bottleneck. The performance of the pre-activation bottleneck is better than standard ResNet; however, we see worse test accuracy after adding a self-attention mechanism to ResNet. It might be due to this mechanism working well in image/text data, as shown in Ramachandran et al. [4] paper, but unable to capture the stochastic and dynamic nature of financial data.

In Liu's paper [3], the average accuracy he achieved was 0.4. However, since we used a completely different dataset and different data processing techniques, the results are not comparable, but it serves as a point of reference.

5 Conclusion

In conclusion, applying CNN to financial data is still under research. Personally, I believe CNN has the potential to predict stock prices. It is known that stock prices depend on various factors, and these factors can be treated as features in a ResNet. By adding more features, the model can capture what drives the stock movement. However, CNN focuses on local features. If a suitable temporal architecture or attention mechanism is integrated into the CNN, it might be able to predict stock movements accurately.

References

[1] George E.P. Box and Gwilym M. Jenkins. *Time Series Analysis: Forecasting and Control*. Holden-Day, San Francisco, 1970.

[2] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015.

[3] Heng Liu and Bowen Song. Stock price trend prediction model based on deep residual network and stock price graph. In *2018 11th International Symposium on Computational Intelligence and Design (ISCID)*, volume 02, pages 328–331, 2018.

[4] Prajit Ramachandran, Niki Parmar, Ashish Vaswani, Irwan Bello, Anselm Levskaya, and Jonathon Shlens. Stand-alone self-attention in vision models, 2019.

[5] Crestmont Research. Stock market yo-yo, 2023.