Deep Learning Using Risk-Reward Function for Stock Market Prediction

Pisut Oncharoen
Chulalongkorn University Big Data Analytics and IoT Center (CUBIC)
Department of Computer Engineering, Faculty of Engineering
Chulalongkorn University, Bangkok, Thailand
pisut.o@student.chula.ac.th

Peerapon Vateekul
Chulalongkorn University Big Data Analytics and IoT Center (CUBIC)
Department of Computer Engineering, Faculty of Engineering
Chulalongkorn University, Bangkok, Thailand
peerapon.v@chula.ac.th

ABSTRACT
Many recent studies have attempted to apply a deep learning approach to build a model for stock market prediction. Most of these studies have concentrated on using prediction accuracy as a performance metric. Some of them have also performed trading simulations to evaluate financial performance. However, financial performance was not improved significantly because the loss function used in the training process focused primarily on prediction accuracy. In this paper, we propose a new framework to train a deep neural network for stock market prediction. A new loss function was developed by adding a risk-reward function, which is derived by the trading simulation results. A new scoring metric called Sharpe-F1 score, which is a combination of Sharpe ratio and F1 score is used for model selection. We employ the best prediction model from our previous work, which consists of Convolutional Neural Network (CNN) and Long Short-Term Memory Network (LSTM) architectures and takes event embedding vectors, historical prices and a set of technical indicators as inputs. The robustness of our framework is evaluated on two datasets by varying the key parameters used in the proposed framework. The results show that financial performance can be improved by adding a risk-reward function into the loss function used in the training process.

CCS Concepts
• **Computing methodologies → Machine learning → Machine learning approaches**.

Keywords
Deep Learning; Risk-Reward Function; Convolutional Neural Network; Long Short-term Memory; Event Embedding; Stock Market Prediction

1. INTRODUCTION
Stock market prediction is a significantly challenging task because the market is highly volatile and influenced by a variety of factors. There are many studies from various areas that have attempted to address this challenge. Recently, new opportunities have been created through advances in the field of artificial intelligence and the increasing amount of information available. It is possible to employ more complex machine learning algorithms such as deep learning methods that can analyze and detect complex patterns in the tremendous amount of data.

CSAI '18, December 8–10, 2018, Shenzhen, China

ACM ISBN 978-1-4503-6606-9/18/12…$15.00
DOI: https://doi.org/10.1145/3297156.3297173

There are many studies which have attempted to build a model for stock market prediction by applying a deep learning approach. For example, Ding et al. [1] extracted text representation vectors from news headlines by using an event embedding approach, which were then fed into the deep neural network in order to predict S&P 500 index movements. Huynh et al. [2] introduced a prediction model using Bidirectional Gated Recurrent Unit (BGRU) architecture by feeding word embedding vectors from news headlines as inputs. Nelson et al. [3] attempted to forecast stock market movement by feeding historical prices into LSTM network. Sezer et al. [4] constructed a stock prediction model by using CNN with 2-D images generated from a set of technical indicators. These studies only used either numerical information, such as historical price and technical indicators, or textual information like news headlines as input into a prediction model. However, the recent research direction has concentrated on applying both types of information since investors use both types of information to make the best decisions. For instance, Akita et al. [5] attempted to predict stock market movement by feeding historical prices and news headlines as inputs into a deep neural network, while Vargas et al. [6] used a set of technical indicators with daily news headlines.

The studies mentioned above demonstrate that using a deep learning approach can yield better performance compared to traditional machine learning algorithms. Most studies focus on building a prediction model by using prediction accuracy as a performance metric. Some of them also perform trading simulations based on the results from the prediction model, which then compute a return as a performance metric. However, results based on trading simulation are not much improved because the loss functions used during trading processes are focused on prediction accuracy. A model with high prediction accuracy may lead to a worse return if the model makes accurate predictions during a small price movement period but makes incorrect predictions when prices change significantly.

In this paper, we propose a new framework to build a deep neural network by adding a risk-reward function into the loss function used in training processes. The risk-reward function is derived from the trading simulation results, so it helps to enhance the results based on trading simulation. Moreover, we also created a new scoring metric, which is a combination of Sharpe ratio and F1 score, called Sharpe-F1 score to select the final model. We used the best prediction model from our previous work [7], which consists of CNN and LSTM architectures and takes event embedding vectors, historical prices and a set of technical indicators as inputs. Our prediction target is the following day's stock market trend. We use target labels as upward trend, downward trend, and sideways trend since these are common stock market behaviors. We conducted experiments on two datasets and used F1 score, Sharpe ratio and annualized return as performance metrics. The key parameters used in our framework, such as alpha, beta and window size, are tested in order to evaluate the robustness of our approach. The results show that the annualized return and Sharpe ratio can be improved by adding a risk-reward function into the loss function used in the training process.

The remainder of this paper is organized as follows: Section 2 presents background knowledge. Section 3 describes the proposed framework. Section 4 explains the experimental setup. Section 5 presents and discusses the results while the conclusions are made in Section 6.

2. BACKGROUND KNOWLEDGE

The following section describes the performance metrics to be applied in this research and provides an explanation of the traditional trading strategies which are employed.

2.1. Performance Metrics

The performance metrics used in the previous study [1] were accuracy and Matthews Correlation Coefficient. In this work, we used the F1 score [8], which is commonly applied in multiclass classification tasks. In addition to the use of the F1 score, other metrics used to assess performance are listed as follows:

2.1.1 Annualized Return

This metric is computed based on the results from the trading simulation using the following formula:

$$Annualized\ Return = \left(\frac{Final\ Balance}{Initial\ Balance}\right)^{\frac{365}{\#SimulationDay}} - 1 \quad (1)$$

2.1.2 Sharpe Ratio

Sharpe ratio [9] is one of the most referenced risk/return measures used in the finance field. It is computed as follows:

$$Sharpe\ Ratio = \frac{Mean(Returns)}{S.D.(Returns)} \quad (2)$$

2.2. Traditional Trading Strategies

We performed a trading simulation by applying traditional trading strategies based on the technical indicators, so that we could compare the performance metrics, such as the annualized return and Sharpe ratio with the results from our proposed framework. The following trading strategies are used in this study.

2.2.1 Moving Average Crossovers

This is a trading strategy using two moving averages. The strategy will give a buy signal when the faster (shorter) moving average advances above the slower (longer) moving average. A sell signal will be given when the faster moving average crosses below the slower moving average.

2.2.2 Short-Term Bollinger Bands Reversion

Bollinger bands are calculated by (3-5). This strategy will give a buy signal when the close price is below the lower band. A sell signal will be given when the closing price is higher than the upper bands.

$$Middel\ Band\ = \mathrm{n-day\ simple\ moving\ average\ (SMA)} \quad (3)$$

$$Upper\ Band\ = \mathrm{n-day\ SMA} + 2 * (\mathrm{n-day\ standard\ deviation\ of\ price}) \quad (4)$$

$$Lower\ Band\ = \mathrm{n-day\ SMA} - 2 * (\mathrm{n-day\ standard\ deviation\ of\ price}) \quad (5)$$

2.2.3 RSI Overbought and Oversold

Relative strength index or RSI is calculated using (6) to create an oscillator that moves between 0 and 100.

$$RSI\ = 100 - \frac{100}{1+RS} \quad (6)$$

Where RS is average gain divided by average loss over the number of periods selected in the look-back period. RSI below 30 means the stock is oversold. After that, it is a buy signal when it moves above the 30 level. RSI above 70 means the stock is considered overbought. After that it is a sell signal when the RSI moves below 70.

2.2.4 Stochastic Oscillator Crossovers

Stochastic oscillator is a momentum indicator that shows the location of the closing price relative to the high-low range over a set number of periods. This indicator is calculated using the following formula.

$$\%K\ = \left(\frac{Current\ Close - LL(n)}{HH(n) - LL(n)}\right) * 100 \quad (7)$$

$$\%D\ = \mathrm{x-day\ SMA\ of\ \%K} \quad (8)$$

Where $LL(n)$ and $HH(n)$ are the lowest low price and the highest high price for the n look-back period, respectively. A buy signal will be given when a cross down (%K < %D) occurs above the 80 level. Likewise, a cross up (%K > %D) that occurs below 20 will indicate a sell signal.

The parameters for all technical indicators are optimized during a validation period. The performance of each strategy is evaluated during a testing period using the optimized parameters.

3. PROPOSED FRAMEWORK

Our objective is to improve the results of the prediction model in terms of performance metrics such as the annualized return and Sharpe ratio by modifying the loss function used in the training process. The process flow of our framework is shown in Fig. 1.

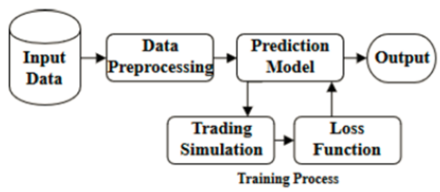

Figure 1. Process flow of the proposed framework.

3.1 Preprocessing

3.1.1 Numerical Input

In this paper, we used historical prices and a set of technical indicators as numerical inputs, as suggested by [6, 10]. The z-score method is applied to normalize the input vectors since each vector has a different range of value.

3.1.2 Textual Input

We applied the idea proposed in [11] which attempts to extract event representation from news headlines by using the Open Information Extraction (Open IE) framework. Instead of implementing our own Open IE framework, we used the free Open IE software developed by Stanford [12]. The extracted event representations were converted into word vectors by using GloVe pre-trained word vectors [13] with 100 dimensions. The output word vectors were fed into the Neural Tensor Network [1] in order to generate event embedding vectors. With this approach, news headlines describing similar events are encoded into similar event vectors, even if they do not share common words.

3.1.3 Labeling Method

We attempted to predict the following day's stock market trend as either an upward trend, downward trend or sideways trend. The trend label is marked by using historical closing prices in a sliding window with the following formula:

$$Trend_i = \begin{cases} Upward\ ;\ Close_{i+1} > Avg.\ Close_{window} + S.D.\ Close_{window} \\ Downward\ ;\ Close_{i+1} < Avg.\ Close_{window} - S.D.\ Close_{window} \\ Sideways; Otherwise \end{cases} \quad (9)$$

We used a sliding window size of 30 days as the default setting in this paper. However, we also performed an experiment based on different window sizes to test the robustness of our framework.

3.2 Proposed Model and Training Processes

3.2.1 Prediction Model

In this work, we used the best prediction model from our previous work [7]. The previous study shows that this model has outperformed other deep learning-based models. Hence, the results in this study are not compared with other prediction models which apply deep learning approach. The prediction model consists of CNN and LSTM architectures and takes textual and numerical information as inputs. For textual information, the event embedding technique [1] was employed to extract text representation vectors from news headlines. Long-term and mid-term event vectors are fed into CNN layers to extract feature maps representing the most important event during these periods. The output feature maps are concatenated with short-term event vectors and then fed into the next hidden layers. For numerical information, the historical prices and a set of technical indicators are fed into LSTM layers to analyze temporal relationships. The outcomes from textual and numerical parts are then combined and fed to the final hidden layers to make a prediction. The structure of the prediction model is shown in Fig. 2.

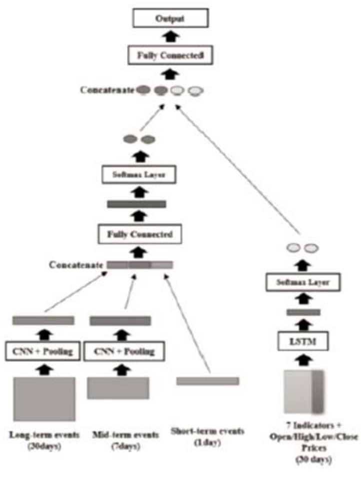

Figure 2. The prediction model structure.

3.2.2 Trading Simulation

We applied the idea suggested by [4] to simulate daily stock trading. The simulation strategy is given as follows:

$$Action(t) = \begin{cases} Buy\ k\ shares\ of\ stock\ ; Prediciton_{t-1} = Up\ trend\ and\ \#Shares = 0 \\ Sell\ k\ shares\ of\ stock\ ; Prediciton_{t-1} = Down\ trend\ \#Shares > 0 \\ Hold; otherwise \end{cases} \quad (10)$$

$$Gain/Loss(t) = \begin{cases} k * (ExitPrice - EntryPrice)\ ; Action(t) = Sell \\ k * (ClosePrice - EntryPrice)\ ; t = T\ and\ \#Shares > 0 \\ 0\ ; Otherwise \end{cases} \quad (11)$$

$$Balance(t) = Balance(t-1) + Gain/Loss(t) - TransactionCost(t) \quad (12)$$

$$Return(t) = \left(\frac{Balance(t)}{Balance(t-1)}\right)^{\frac{365}{\#HoldingPeriod}} - 1; if\ Action(t) = Sell \quad (13)$$

where k is $\frac{Balance(t-1)}{Open\ price\ (t)}$, T is the end of simulation period, $\#Shares$ is stock shares held and the transaction cost is assumed to be 0.1% of the traded volume.

The Sharpe ratio is computed by (2) for every training iteration. We used the Sharpe ratio as input for the loss function.

3.2.3 Loss Function

Cross-entropy is widely used in classification tasks. This type of loss function constructs a model focusing on accurate prediction. In this work, we attempted to improve the results of the prediction model in terms of performance metrics, such as the annualized return and Sharpe ratio. Hence, we propose a new loss function by adding a risk-reward function into cross-entropy loss as follows:

$$Loss = CrossEntropy * (1 - \alpha) + \alpha * \log\left(\frac{1}{\max(0.01, SharpeRatio)}\right) \quad (14)$$

The Sharpe ratio is used as a risk-reward function since it is one of the most referenced risk/return measures used in the finance field. The function is then scaled by natural logarithm because cross-entropy is also based on natural logarithm function. The α is used as a risk trade-off parameter for assigning weight to each component. Using higher α, the final model tends to give a higher Sharpe ratio and lower F1.

3.2.4 Training Process

As suggested by [4], we split training/validation/test periods into three different sets and trained each model separately, as shown in Fig.3. The performance of each model setting was evaluated by combining the results from three sets. The splitting periods used in this study are summarized in Table 1.

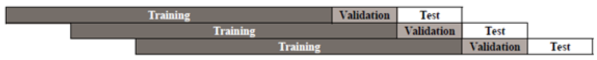

Figure 3. Training, validation and test approach.

3.2.5 Sharpe-F1 Score

The Sharpe-F1 score (SF1) is used for final model selection. The model that gives the highest SF1 score will then be selected as the final model. This score is computed by using the F1 and Sharpe ratio of models based on the validation dataset as follows:

$$SF1\ score = (1 - \beta) * F1_{norm} + \beta * SharpeRatio_{norm} \quad (15)$$

where $F1_{norm}$ and $SharpeRatio_{norm}$ are min-max normalization of the F1 and Sharpe ratio over all the models evaluated by the validation dataset. The β is used as a metric trade-off parameter for assigning weight to each component. Using higher β, the final model tends to give a higher Sharpe ratio and lower F1. We used β of 0.5 as a default setting in this study.

4. EXPERIMENTAL SETUP

4.1 Datasets

The prediction model used in this experiment takes two types of input, which are textual information and numerical information.

4.1.1 Textual Information

The textual information used in this work is news headline datasets, which are obtained from two sources as follows:

4.1.1.1 Financial News Headlines from Reuters

The financial news headlines from Reuters during a period from 20 October 2006 to 19 November 2013 are used. We used the published dataset from [11] and selected only news headlines that contain company names or abbreviations since the study in [6] suggested that specific news related to companies normally performs better than general news.

4.1.1.2 News Headlines from Reddit

The news headlines from Reddit (/r/word news) from 8 June 2008 to 1 July 2016. Top 25 news headlines ranked by Reddit users, are considered for a single day. We obtained this dataset from the Kaggle website [14].

4.1.2 Numerical Information

This is historical price data extracted from the Yahoo! Finance website. We used the following stock indexes to represent the stock market:

4.1.2.1 Standard & Poor's 500 Index (S&P500)

S&P500 is an American stock market index based on the market capitalizations of 500 large companies with common stock listed on the NYSE or NASDAQ.

4.1.2.2 Dow Jones Industrial Average (DJIA)

This is a price-weighted average of 30 significant stocks traded on the NYSE and NASDAQ. The DJIA is one of the most watched stock indexes in the world, containing companies like General Electric, Walt Disney, Exxon Mobil, Microsoft and Coca-Cola.

We combined the new headlines from different sources and the historical prices from Yahoo! Finance, which were then split into two datasets as summarized in Table 1.

Table 1. Summary of information on datasets

Information	Dataset #1	Dataset #2
Historical Prices	S&P 500	DJIA
News Sources	Reuters	Reddit
# of Headlines	27,158	49,725
Splitting Periods 1		
- Training	20 Oct 06 - 3 May 11	8 Aug 08 - 22 Aug 13
- Validation	4 May 11 - 26 Dec 11	23 Aug 13 - 12 May 14
- Test	27 Dec 11 - 10 Aug 12	13 May 14 - 28 Jan 15
Splitting Periods 2		
- Training	20 Jun 07 - 26 Dec 11	4 May 09 - 12 May 14
- Validation	27 Dec 11 - 10 Aug 12	13 May 14 - 28 Jan 15
- Test	13 Aug 12 - 1 Apr 13	29 Jan 15 - 14 Oct 15
Splitting Periods 3		
- Training	9 Feb 08 - 10 Aug 12	20 Jan 10 - 28 Jan 15
- Validation	13 Aug 12 - 1 Apr 13	29 Jan 15 - 14 Oct 15
- Test	2 Apr 13 - 19 Nov 13	15 Oct 15 - 1 Jul 16

4.2 Hyperparameters and Training

Our deep neural network was trained by using the Adaptive Moment Estimation (Adam) algorithm [15], as many studies show that Adam works well in practice and compares favorably to other stochastic optimization methods. We also applied a modern approach, namely Batch Normalization [16], to our deep neural network in order to accelerate deep network training by reducing internal covariate shifting. The grid search technique was also used for hyperparameter turning.

5. EXPERIMENT RESULTS

The main focus of this study is the influence of parameters used in our proposed framework. Our experiment consists of two parts as follows:

5.1 Effect of Parameters in Our Framework

5.1.1 Impact of Risk Trade-off Parameter (Alpha)

Alpha is a risk trade-off parameter used in the loss function. We evaluated the impact of alpha by varying it from 0 to 0.9 and then measuring the performance metrics described in Section 2.

As shown in Table 2 for dataset #1, we can clearly see the decreasing trend of F1 score when the alpha is increased, while the Sharpe ratio tends to increase. This is because using the higher alpha means the model will focus on the risk-reward function. Hence, the F1 score will decrease. There are some fluctuations in the trend of the F1 score and Sharpe ratio in the results based on dataset #2. However, the overall trend is still the same as dataset #1; as the alpha increased, the F1 decreased, while the inverse was seen for the Sharpe ratio.

Table 2. Results based on different alpha values.

Alpha	Dataset #1			Dataset #2		
	F1	Sharpe Ratio	Annualized Return	F1	Sharpe Ratio	Annualized Return
0.0	**62.27%**	0.324	19.25%	58.38%	0.063	1.72%
0.1	61.34%	0.323	19.02%	**58.67%**	0.207	1.94%
0.2	60.45%	0.313	17.08%	58.29%	0.284	4.69%
0.3	57.79%	0.490	**21.56%**	54.75%	**0.444**	**6.26%**
0.4	56.73%	0.373	15.35%	56.68%	0.270	5.40%
0.5	56.23%	0.433	14.19%	53.37%	0.359	5.98%
0.6	55.95%	0.535	15.26%	53.56%	0.416	4.88%
0.7	55.40%	0.457	15.97%	54.43%	0.302	5.79%
0.8	55.37%	0.572	13.12%	47.69%	0.256	2.54%
0.9	54.48%	**0.611**	16.43%	51.81%	0.288	1.41%

As we can see in Table 2, including a risk-reward function into the loss function can improve the annualized return with lower risk (i.e., a higher Sharpe ratio). Based on this experiment, the optimal alpha for both datasets is 0.3 due to it providing the highest annualized return. We suggest that the alpha should not be higher than 0.5 since a higher alpha makes the model more reliant on the results based on trading strategy and trading simulation, which may lead to a divergent problem during the training process.

5.1.2 Impact of Metric Trade-off Parameter (Beta)

Beta is a metric trade-off parameter used in SF1 score calculation. The SF1 score was used to select the final model. We have designed the beta as an adjustable parameter so that users can adjust this parameter based on their own risk appetite. In this experiment, we used the optimal alpha of 0.3 from the previous experiment and varied the beta value to observe their impact.

Table 3. Results based on different beta values.

Beta	Dataset #1			Dataset #2		
	F1	Sharpe Ratio	Annualized Return	F1	Sharpe Ratio	Annualized Return
0.00	**65.08%**	0.312	21.88%	62.42%	0.211	0.56%
0.25	59.57%	0.354	**23.55%**	**62.43%**	0.125	-0.08%
0.50	57.79%	0.490	21.56%	54.75%	0.444	6.26%
0.75	57.79%	0.490	21.56%	47.92%	0.543	**7.98%**
1.00	63.31%	**0.548**	19.80%	42.63%	**0.553**	4.60%

As shown in Table 3, the F1 score and Sharpe ratio trends are consistent with our intention, which was to use more beta in SF1 score calculation to focus on the Sharpe ratio, and vice-versa for the F1 score. Based on this experiment, applying greater weight to the Sharpe ratio (i.e., beta = 0.75) gives the optimal annualized return in both datasets.

5.1.3 Impact of Window Size

To test the robustness of our framework, we performed an experiment based on the revised label, which was generated using a different window size.

Table 4 shows that the F1 score decreased when the window size was small. This is because using a small window size makes the label more volatile, meaning it is more difficult to make an accurate prediction. Hence, the F1-score is lower.

Table 4. Results based on different window size and loss function.

Window Size	With risk-reward function (alpha = 0.3)			Without risk-reward function (alpha = 0)		
	F1	Sharpe Ratio	Annualized Return	F1	Sharpe Ratio	Annualized Return
Dataset #1						
30	57.79%	0.490	21.56%	62.27%	0.324	19.25%
15	56.48%	0.436	20.97%	60.86%	0.362	10.02%
10	53.85%	0.569	20.83%	54.07%	0.358	16.42%
5	52.62%	0.516	18.67%	50.78%	0.277	3.88%
Dataset #2						
30	54.75%	0.444	6.26%	58.38%	0.063	1.72%
15	51.49%	0.272	4.48%	55.23%	-0.598	0.17%
10	50.40%	0.193	1.92%	52.46%	-0.226	0.96%
5	44.45%	0.264	1.33%	45.29%	-0.189	-3.28%

Normally, the model with a risk-reward function would give a lower F1 score due to less concentration on the cross-entropy function. However, adding a risk-reward function to the loss function would help the model to improve the annualized return and Sharpe ratio.

Our experiment shows that the model using a risk-reward function in the loss function gives a better annualized return and Sharpe ratio for both datasets.

5.2 Traditional Trading Strategy

The prediction model used in this study is the best model from our previous work [7]. The previous study demonstrates that this prediction model outperforms other models that apply a deep learning approach. Hence, we did not perform an experiment against other deep learning-based models. In this experiment, we performed trading simulations using traditional trading strategies as described in Section 2. This is compared with the results based on our proposed model.

Table 5. Results using traditional trading strategy.

Model	Dataset #1		Dataset #2	
	Annualized Return	Sharpe Ratio	Annualized Return	Sharpe Ratio
Traditional trading strategy				
Moving average crossovers	7.44%	0.397	-6.59%	-0.740
Short-term Bollinger reversion	3.52%	0.546	5.42%	0.563
RSI overbought and oversold	11.76%	**0.709**	4.12%	**0.620**
Stochastic oscillator crossovers	11.45%	0.328	-1.95%	0.148
Proposed model				
Window 30, alpha 0.3, beta 0.5	**21.56%**	0.490	**6.26%**	0.444
Window 15, alpha 0.3, beta 0.5	20.97%	0.436	4.48%	0.272
Window 10, alpha 0.3, beta 0.5	20.83%	0.569	1.92%	0.193
Window 5, alpha 0.3, beta 0.5	18.67%	0.516	1.33%	0.264

Table 5 shows that on average, our proposed model performs better than using traditional trading strategies based on technical indicators for both datasets.

6. CONCLUSION

In this paper, we propose a new framework to train a deep neural network by adding a risk-reward function into the loss function and to create a new scoring metric called Sharpe-F1 score, which is a combination of the Sharpe ratio and F1 score. This score was used for final model selection. In this work, we used the best prediction model from our previous work, which consists of CNN and LSTM architectures and takes event embedding vectors, historical prices and technical indicators as inputs. The robustness of our framework was evaluated by varying the key parameters, such as alpha, beta and window size. We conducted the experiment on two datasets and employed the F1 score, Sharpe ratio and annualized return as performance metrics. The results show that adding a risk-reward function into the loss function can improve the annualized return and Sharpe ratio, but reduces the F1 score. Moreover, the results based on the trading simulation show that on average our framework performs better than traditional trading strategies using technical indicators.

7. REFERENCES

[1] X. Ding, Y. Zhang, T. Liu, and J. Duan, "Deep learning for event-driven stock prediction," presented at the IJCAI International Joint Conference on Artificial Intelligence, 2015.

[2] H. D. Huynh, L. M. Dang, and D. Duong, "A New Model for Stock Price Movements Prediction Using Deep Neural Network," in Proceedings of the Eighth International Symposium on Information and Communication Technology, 2017, pp. 57-62: ACM.

[3] D. M. Nelson, A. C. Pereira, and R. A. de Oliveira, "Stock market's price movement prediction with LSTM neural networks," in Neural Networks (IJCNN), 2017 International Joint Conference on, 2017, pp. 1419-1426: IEEE.

[4] O. B. Sezer and A. M. Ozbayoglu, "Algorithmic Financial Trading with Deep Convolutional Neural Networks: Time Series to Image Conversion Approach," Applied Soft Computing, 2018.

[5] R. Akita, A. Yoshihara, T. Matsubara, and K. Uehara, "Deep learning for stock prediction using numerical and textual information," in Proceedings of the 15th International Conference on Computer and Information Science (ICIS), 2016, pp. 1-6.

[6] M. R. Vargas, B. S. L. P. De Lima, and A. G. Evsukoff, "Deep learning for stock market prediction from financial news articles," in Proceedings of the 2017 IEEE International Conference on Computational Intelligence and Virtual Environments for Measurement Systems and Applications (CIVEMSA), 2017, pp. 60-65.

[7] P. Oncharoen and P. Vateekul, "Deep Learning for Stock Market Prediction Using Event Embedding and Technical Indicators," in Advanced Informatics, Concepts, Theory, and Applications (ICAICTA), 2018 International Conference on, 2018: IEEE.

[8] M. Sokolova and G. Lapalme, "A systematic analysis of performance measures for classification tasks," Information Processing & Management, vol. 45, no. 4, pp. 427-437, 2009.

[9] W. F. Sharpe, "The sharpe ratio," Journal of portfolio management, vol. 21, no. 1, pp. 49-58, 1994.

[10] Y. Zhai, A. Hsu, and S. K. Halgamuge, "Combining news and technical indicators in daily stock price trends

prediction," in International symposium on neural networks, 2007, pp. 1087-1096.

[11] X. Ding, Y. Zhang, T. Liu, and J. Duan, "Using structured events to predict stock price movement: An empirical investigation," in Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, 2014, pp. 1415-1425.

[12] Stanford Open Information Extraction. Available: https://nlp.stanford.edu/software/openie.html

[13] J. Pennington, R. Socher, and C. D. Manning, "GloVe: Global vectors for word representation," in EMNLP 2014 - 2014 Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference, 2014, pp. 1532-1543.

[14] Daily News for Stock Market Prediction. Available: https://www.kaggle.com/aaron7sun/stocknews/

[15] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," arXiv preprint arXiv:1412.6980, 2014.

[16] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," arXiv preprint arXiv:1502.03167, 2015.