Financial Engineering and Risk Management (2023)
Clausius Scientific Press, Canada

DOI: 10.23977/ferm.2023.060412
ISSN 2523-2576 Vol. 6 Num. 4

Research on Forecasting A-share Stock Index Using a Modified Generative Adversarial Network

Zhiming Zhou, Zhensheng Huang*

School of Mathematics and Statistics, Nanjing University of Science & Technology, Nanjing, 210094, China
**Corresponding author*

Keywords: Stock index prediction, deep learning, GAN, feature selection

Abstract: Studying stock market prediction and coming up with an effective forecasting model can help investors reduce investment risk. But it's more difficult to predict Chinese stock market (A-share) than that of developed countries because of its unique characteristics. This paper tried to make some progress on this problem and has acquired notable results. In this paper, we construct a generative adversarial network with shared pre-learning network (SPN-GAN) by introducing a pre-learning network into the architecture of GAN which is shared by generator (G) and discriminator (D) and adding a directional sub-discriminator into D out of consideration of forecasting accuracy of moving direction. SPN can preliminarily extract hidden representation from original indicators and the design of shared structure significantly reduced model complexity and training cost. The performance of proposed model is evaluated using 4 representative A-share stock indices. Results show that SPN-GAN outperforms traditional machine learning models and deep learning models in most cases.

1. Introduction

Stock market plays an important role in modern financial market and is one of the main channels of direct financing. However, due to the short history, Chinese stock market has some unique features compared to that of developed contries, such as the majority of retail investors, the lack of short selling mechanism and the greater influence of government behavior, which leads to lower market efficiency. Therefore, it is more difficult for investors to correctly grasp the trend of the market. In this context, studying the prediction of A-share stock index and developing an effective forecasting model can help investors understand the market situation better, trade more rationally and avoid unnecessary losses.

With the continuous progress of computer technology and artificial intelligence, researchers gradually shift their focus from the traditional statistical and econometric models to the frontier soft-computing technologies, such as machine learning and deep learning. For instance, Sharaf et al. [1] developed a stock price prediction framework "StockPred" by combining support vector machine (SVM), decision tree (DT), random forest (RF) and other models. Qiu et al. [2] proposed a modified sentiment index based on daily financial reviews of Eastmoney.com. SVM, Gradient Boosting Decision Tree (GBDT), K-nearest Neighbors (KNN) and several other prevailing machine learning

models were applied with the proposed index to forecast the SSE 50 index. In addition to traditional machine learning models, many scholars also used deep learning models which have more parameters and are more complex in research. Deep learning models are mainly based on neural network [3], with more powerful big data mining and nonlinear modeling capabilities, having shown better predictive performance than conventional machine learning in lots of studies [4]. Zaheer et al. [5] proposed a composed model to predict the closing and highest price of the Shanghai Composite Index on the second day. In their model, convolutional neural network (CNN), long short term memory (LSTM) and recurrent neural network (RNN) were trained in parallel and the optimal model output was selected according to R^2 value. Their hybrid model outperformed all benchmarks in experiment.

With the advancement of deep learning research, many excellent neural network architectures have been developed besides artificial neural network (ANN), CNN and RNN, Generative Adversarial Network (GAN) being one of them. GAN is a generative model based on two-agent game framework proposed by Goodfellow et al. [6] in 2014, primarily used for image, video and other data generation tasks in the beginning. Because of its outstanding data distribution learning ability, scholars have gradually tried to apply it to stock forecasting research. He and Kita [7] combined RNN models with GAN. In their model, the generator was based on LSTM and the discriminator was constructed using LSTM, RNN or gated recurrent unit (GRU) respectively. Their model achieved better performance than ARIMA, LSTM and ordinary GAN in prediction of S&P 500 index. Wu et al. [8] adapted GAN to the joint prediction of multiple time series. The model they constructed consists of three parts: time series interaction matrix generator G_i, predictive generator G_p and discriminator D. G_i is designed to learn the correlation between time series and generate a fake interaction matrix, with G_p which is built on LSTM to accept a interaction matrix then forecast the next values of time series and D to judge the authenticity of samples. Empirical results show that their model improves the accuracy of multiple time series prediction.

Given GAN's excellent performance in extensive deep learning tasks and limited application in stock market prediction so far, this paper extends it and proposes GAN with shared pre-learning network (SPN-GAN) to forecast four representative stock indices in A-share market. The extension includes two aspects. One is the introduction of pre-learning network shared by G and D and the other is embedding a directional sub-discriminator D_{dir} into whole discriminator D aimed at making the prediction of G not only have small error but also conform to the actual moving direction of the stock index as much as possible. The SPN is expected to preliminarily learn from the original stock features and extract intermediate representation of historical information. SPN's parameters will be updated during training G or D.

In terms of experiment, in order to give full play to the performance of the model, we built a large input data set containing 98 predictive features not only including usual volume and price data but also considering the influence of macro finance, the overall operational metrics of the stock market and global stock indices. Then feature selection of each stock index was performed based on elastic net (EN) and GBDT to remove unimportant variables. The experimental results show that SPN-GAN achieves better performance than the benchmark models under most conditions and the SPN structure shared with G and D significantly reduces the complexity and training cost of the model.

2. Model Design

2.1 Generative Adversarial Network

The model architecture of GAN is shown in Figure 1. It's able to be seen that GAN is mainly composed of two parts: generator G and discriminator D. G's task is to learn the probability distribution of X from the data set $\{x_1, x_2, ..., x_N\}$ and generate fake data. Specifically, G receives

samples from a prior distribution $p_z(z)$ and learns a function mapping the prior to the population distribution of X. The distribution of X can be represented as $p_{data}(x)$ and the distribution of data generated by G can be represented as $p_{model}(x)$. The goal of G is to make $p_{model}(x)$ as close to $p_{data}(x)$ as possible. D receives real samples from the data set or fake samples generated by G and outputs a scalar between 0 and 1 indicating the probability that D judges the received data to be true.

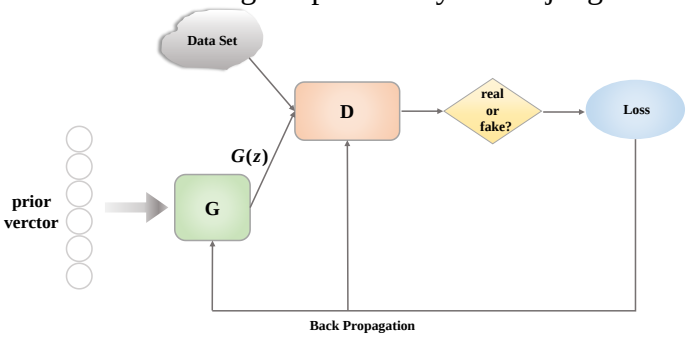

Figure 1: The architecture of GAN.

The training of GAN is adversarial. The learning objective of G is to "confuse" D, making it produce as large output as possible for data generated by G, while the goal of D is to distinguish real and fake data correctly, that is, giving real samples larger outputs and fake samples smaller outputs. Specifically, G and D play the following two-agent min-max game:

$$\min_G \max_D V(D, G) = E_{x\sim p_{data}(x)}\left[\log(D(x))\right] + E_{z\sim p_z(z)}\left[\log(1-D(G(z)))\right] \quad (1)$$

where V is value function representing D's ability to correctly distinguish between real and fake data.

To train the GAN model shown in Figure 1 according to formula (1), usually G is fixed first and the parameters of D are updated in k steps then G's parameters are updated once with D fixed. The hyperparameter k can be determined by cross validation.

2.2 SPN-GAN Model

As previously mentioned, our SPN-GAN model extends original GAN in two ways. Firstly, the pre-learning network SPN shared by G and D is added to preliminarily extract intermediate representation from inputs. The benefits of shared SPN are: (1) the magnitude of model's parameters is distinctly reduced so as to markedly save storage space and training time; (2) G and D learn the same representation of past stock movements from SPN, which can promote the joint advancement of G and D and speed model convergence. Secondly, the discriminator D consists of two sub-discriminators and a penalty factor. The first sub-discriminator, D_{adv}, which accepts real or fake stock index values as input, is used for regular adversarial training. The other, D_{dir}, whose input is real or fake stock index moving direction, is expected to prompt G to provide predictions with correct direction. The penalty factor will punish the output of D based on the prediction error of G when training G.

The model architecture of SPN-GAN is shown in Figure 2. The gray arrows indicate the data flows in forward calculation and the blue arrows indicate the data flows in back propagation. During training, the feature matrix X is firstly input to SPN for intermediate representation extraction. Whereafter, having received the intermediate representation of stock movements from SPN, G gives the next batch of forecasts for stock index and D outputs the probabilities that the samples fed to it

are real. The decision probability of D is obtained as the element-wise product of the output of G_{adv} and G_{dir} and the penalty factor:

$$D(\cdot) = e^{-\gamma_1 L(y,\tilde{y})} D_{adv}(\cdot) D_{dir}(\cdot) \tag{2}$$

where $\tilde{y}$ is actual or predicted stock index value and $L(y, \tilde{y})$ measures the forecasting error of G:

$$L(y_t, \tilde{y}_t) = \left| \frac{(1+\alpha)(y_t - \tilde{y}_t)}{y_t} \right|, \quad \alpha = \begin{cases} 0, & (y_t - y_{t-1})(\tilde{y}_t - y_{t-1}) \geq 0 \\ \gamma_0, & \text{else} \end{cases} \tag{3}$$

where γ_0 and γ_1 are penalty coefficients, being set to 0.5 and 0.8 respectively in this study.

The internal structures of G and D are both composed of a hidden network and a fully connected layer. The hidden network is responsible for further processing the intermediate features from SPN and obtain advanced representation of historical information and the fully connected layer is used for generating outputs. In this paper, the hidden networks of G and D are based on LSTM and CNN, respectively.

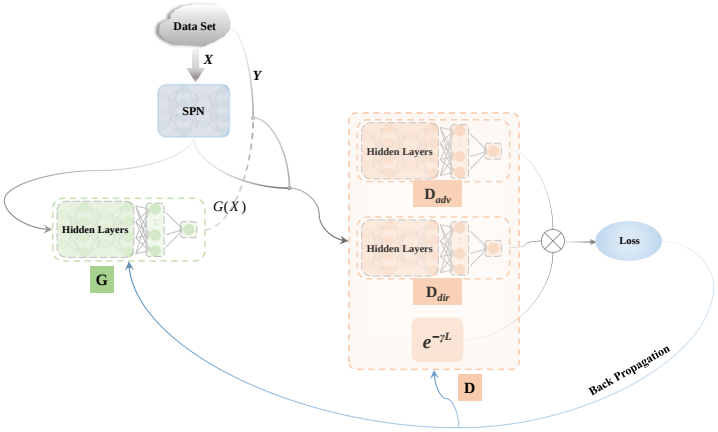

Figure 2: The architecture of SPN-GAN.

SPN is the key role in our model for representation extraction of stock index time series. We choose the encoder layer in Transformer [9], the prominent model in natural language processing (NLP) field, as main structure of SPN. Different from original Transformer, we don't add positional encodings to inputs. Instead, raw features from data set firstly pass through an RNN layer to stay chronological before be fed into encoder. The model architecture of SPN is shown in Figure 3.

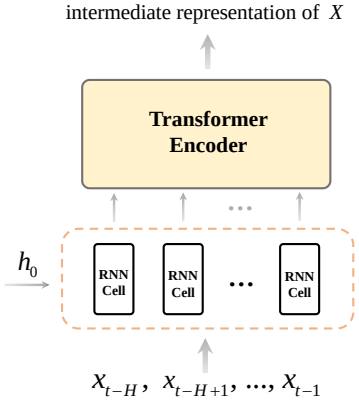

Figure 3: The architecture of SPN.

We follow the general training method of GAN to train SPN-GAN. The training algorithm of SPN-GAN is shown in Algorithm 1. It is noteworthy that the mini-batches for training G and D are determined independently. We adopt Adam [10] to update model parameters.

Algorithm 1 Train SPN-GAN

Prerequisites: Training set $T = \{(x_1, y_1), (x_2, y_2), ..., (x_N, y_N)\}$; number of Epochs K; size of mini-batch M; objectives: obj_G, obj_D; learning rates: ρ_G, ρ_D, ρ_{SPN}

for K Epochs **do**

construct mini-batch sets from T: $S_G = \{B_1, B_2, ..., B_{N//M}\}$, $S_D = \{B_1, B_2, ..., B_{N//M}\}$

for N // M **do**

train D and SPN

• get a mini-batch randomly from S_D: $B_D = \{(x_1, y_1), (x_2, y_2), ..., (x_M, y_M)\}$

• calculate gradients of D and SPN:

$$\nabla(\theta_{D_{adv}}, \theta_{D_{dir}}, \theta_{SPN}) \frac{1}{M}\sum_{i=1}^{M} \text{obj}_D\big|_{(x_i, y_i)}$$

• update D and SPN using Adam:

$$\text{Adam}(\theta_{D_{adv}}, \theta_{D_{dir}}, \theta_{SPN}; \rho_D, \rho_{SPN})$$

train G and SPN

• get a mini-batch randomly from S_G: $B_G = \{(x_1, y_1), (x_2, y_2), ..., (x_M, y_M)\}$

• calculate gradients of G and SPN:

$$\nabla(\theta_G, \theta_{SPN}) \frac{1}{M}\sum_{i=1}^{M} \text{obj}_G\big|_{(x_i, y_i)}$$

• update G and SPN using Adam:

$$\text{Adam}(\theta_G, \theta_{SPN}; \rho_G, \rho_{SPN})$$

end for

end for

Note: "//" means floor divide.

3. Experiment

3.1 Data

3.1.1 Data Source

We select Shanghai Composite Index (SHCI), Shenzhen Component Index (SZCI), CSI 300 (CSI300) and CSI 500 (CSI500) to evaluate the proposed model. Our raw data set contains following kinds of data:

(1) Trading and fundamental data, including daily open, high, low and close price, trading volume and amount, turnover rate, PE and PB.

(2) Macro financial indicators, including USD/RMB central parity rate, yield to maturity of 1-year treasury bond, London gold spot price and OPEC crude oil price.

(3) Overall operational metrics of A-share market, including total trading volume and amount, total number of trading deals and overall turnover rate.

(4) Important global stock indices, including Dow Jones Industrial Average, Nasdaq Index, S&P 500, Nikkei 225, German DAX and British FTSE 100.

Based on the release time of all data, the test period for SHCI and SZCI is determined as 2/1/2003 to 31/12/2021 while 8/4/2005 to 31/12/2021 for CSI300 and 15/1/2007 to 31/12/2021 for CSI500.

3.1.2 Feature Extraction and Selection

This paper extracts features from raw data mainly based on domain knowledge in finance and existing researches. The extracted features fall into the following types:

(1) Quotation related indicators, including technical indicators and others (Other Quote Indicators). We partially refered to Yun et al. [11] for the selection of technical indicators and also used Talib [12] library to calculate values of them. Talib is a software widely used by quantitative investors, providing calculation formulas for common technical indicators. As a supplement, we designed some additional indicators according to our understanding and domain knowledge about financial market, such as "AmountROC" and "DailyRange". Our technical indicators are divided into six categories, as demonstrated in Table.1. Other variables of this type are shown in Table.2.

(2) Macro financial and market features, including all the variables of "Macro financial indicators" and "Overall operational metrics of A-share market" in 3.1.1.

(3) Global index features. In addition to daily close prices of this indices, it also includes their daily returns and the "ROC" of prices.

(4) Other indicators (OtherIndicators), including daily turnover rate, rolling P/E ratio (PE_TTM) and P/B ratio (PB_LF).

Table 1: Technical indicators used in this paper.

Category	Name	Interpretation
MomentumIndicators	ADX, ADXR, APO, AROONDOWN, AROONUP, AROONOSC, BOP, CCI, CMO, DX, MACDDIF, MACDSIGNAL, MACDHIST, MFI, MINUS_DI, MINUS_DM, MOM, PLUS_DI, PLUS_DM, PPO, RSI, STOCH_SLOWK, STOCH_SLOWD, STOCHRSI_FASTK, STOCHRSI_FASTD, TRIX, ULTOSC, WILLR	See Talib [12].
OverlapStudies	UPPERBAND, MIDDLEBAND, LOWERBAND, DEMA, MIDPOINT, SMA, T3, TEMA, TRIMA, WMA	
CycleIndicators	HT_DCPERIOD, HT_DCPHASE, HT_PHASOR_COMPONENT_INPHASE, HT_PHASOR_COMPONENT_QUADRATURE, HT_SINEWAVE_SINE, HT_SINEWAVE_LEADSINE, HT_TREND_CYCLE_MODE	
VolumeIndicators	CHAIKIN_AD_LINE, CHAIKIN_AD_OSC	
	Volume	Trading volume.
	VolumeROC	"ROC" of trading volume, just like "CloseROC" in Table 2 for close price.
	Amount	Trading amount.
	AmountROC	"ROC" of trading amount
PriceTransform	AVGPRICE, MEDIAN_PRICE, TYPICAL_PRICE, WCLPRICE	See Talib [12] .
	Amplitude	Highest price minus lowest price.
	DailyRange	Close price minus open price.
VolatilityIndicators	NATR	See Talib [12] .
	Volatility5Days, Volatility10Days, Volatility20Days	Standard deviation of daily returns over 5, 10, 20 days.

Table 2: Other quotation related indicators.

Name	Description	Formula
CloseROC	Measures the position of current close price compare to the past.	$c_t / \left(\frac{1}{H} \sum_{i=t-H+1}^{t} c_i \right)$
Change	Change of close price.	$c_t - c_{t-1}$
Return1Day Return5Days Return10Days Return20Days	Returns over 1, 5, 10, 20 trading days.	$c_t / c_{t-H} - 1, \ H = 1,\ 5,\ 10,\ 20$
KlinePattern	Describes the k-line patterns according to Lin et al. [13].	-
EightTrigram	Interday moving patterns of price according to Lin et al. [13].	-

Note: c and H represent close price and time window, respectively.

In order to reduce training cost and remove redundant information, we performed feature selection for each stock index on each category separately. Because the size of partial categories is too small or there are correlations between some categories, we merged several couples of categories as follows: merged "OverlapStudies" and "PriceTransform" into "Merged1", "GlobalIndex" and "MacroAndMarket" into "Merged2", "VolatilityIndicators" and "OtherIndicators" into "Merged3". We applied EN and GBDT for feature selection. EN was used to score the linear influence of each feature according to coefficients of features in the model. GBDT was used to score the nonlinear influence of each feature based on the total gain obtained by splitting on that feature during trees growth. The total score for each feature is obtained by adding linear and nonlinear scores after following normalization:

$$I'_j = \frac{I_j}{\|I\|}, \ j = 1,\ 2,\ ...,\ p \tag{4}$$

where $I = (I_1, I_2, ..., I_p)$ are scores and p is the number of features.

3.2 Method

In this paper, we use features of the past H trading days to predict stock index value of the next trading day by sliding window method, that is, to learn following model: $\hat{y}_t = f(X_{H \times p,\, t-1}; \theta)$. We divided the whole data set into train set, validation set and test set by chronological order in a ratio of 7:1:2. For each stock index and each model, experiments with H = 1, 10 and 20 were performed respectively and the final results were averaged over them.

We carried out the above experiments on ARIMA-GARCH, SVM, DT, GBDT, ANN, LSTM and plain GAN under the same experimental conditions and compared them with proposed SPN-GAN so as to highlight the superiority of our model.

3.3 Evaluation Measures

We evaluate the performance of each model from two aspects: forecasting error and classification accuracy for the moving direction of stock indices. Metrics measuring forecasting error are:

$$MSE = \frac{1}{N} \sum_{i \in TestSet} (y_i - \hat{y}_i)^2 \tag{5}$$

$$MAE = \frac{1}{N} \sum_{i \in TestSet} |y_i - \hat{y}_i| \tag{6}$$

$$MSE_{adj} = \frac{1}{N} \sum_{i \in TestSet} \alpha_i (y_i - \hat{y}_i)^2 \tag{7}$$

$$MAE_{adj} = \frac{1}{N} \sum_{i \in TestSet} \alpha_i |y_i - \hat{y}_i| \tag{8}$$

where α_i has the same meaning as α in formula (3). The subscript i indicates the i-th sample. Classification metrics are:

$$ACCU = \frac{TP + TN}{TP + FP + TN + FN} \tag{9}$$

$$Precision = \frac{TP}{TP + FP} \tag{10}$$

$$Recall = \frac{TP}{TP + FN} \tag{11}$$

$$F_1 = \frac{2TP}{2TP + FP + FN} \tag{12}$$

where *TP* represents the number of samples correctly predicted as up-going, *TN* represents the number of samples correctly predicted as down-falling, *FP* represents the number of samples falsely predicted as up-going and *FN* represents the number of samples falsely predicted as down-falling.

3.4 Experimental Settings

Experiments in this paper are implemented using Python 3.8. Machine learning models and deep learning models are programmatically based on scikit-learn library and pytorch framework respectively. The architecture and training hyperparameters of SPN-GAN are shown in Table.3.

Table 3: Experimental settings of SPN-GAN model.

	Settings
Structure of G	The number of LSTM layers: 2; the number of hidden network's units: (64, 32); the number of nodes of the fully connected layer: 32
Structure of D	The number of CNN layers: 1; CNN step size: (1, 2); kernel size: (3, 6); padding: (1, 2) with filling value 0; output channel number: 1; the fully connected layers contain two layers with 64 and 32 nodes respectively
Structure of SPN	The number of encoder layers: 3; the number of self-attention heads: 4; feature dimension: 128
Training	The epoch number: 800; the mini-batch size: 64; the learning rates of G, D and SPN are 0.001, 0.0005 and 0.0001, respectively

4. Result Analysis

4.1 Feature Selection Result

The result of normalized feature scores is demonstrated in Figure 4. The features are ranked in descending order in each category according to the average scores of the four indices.

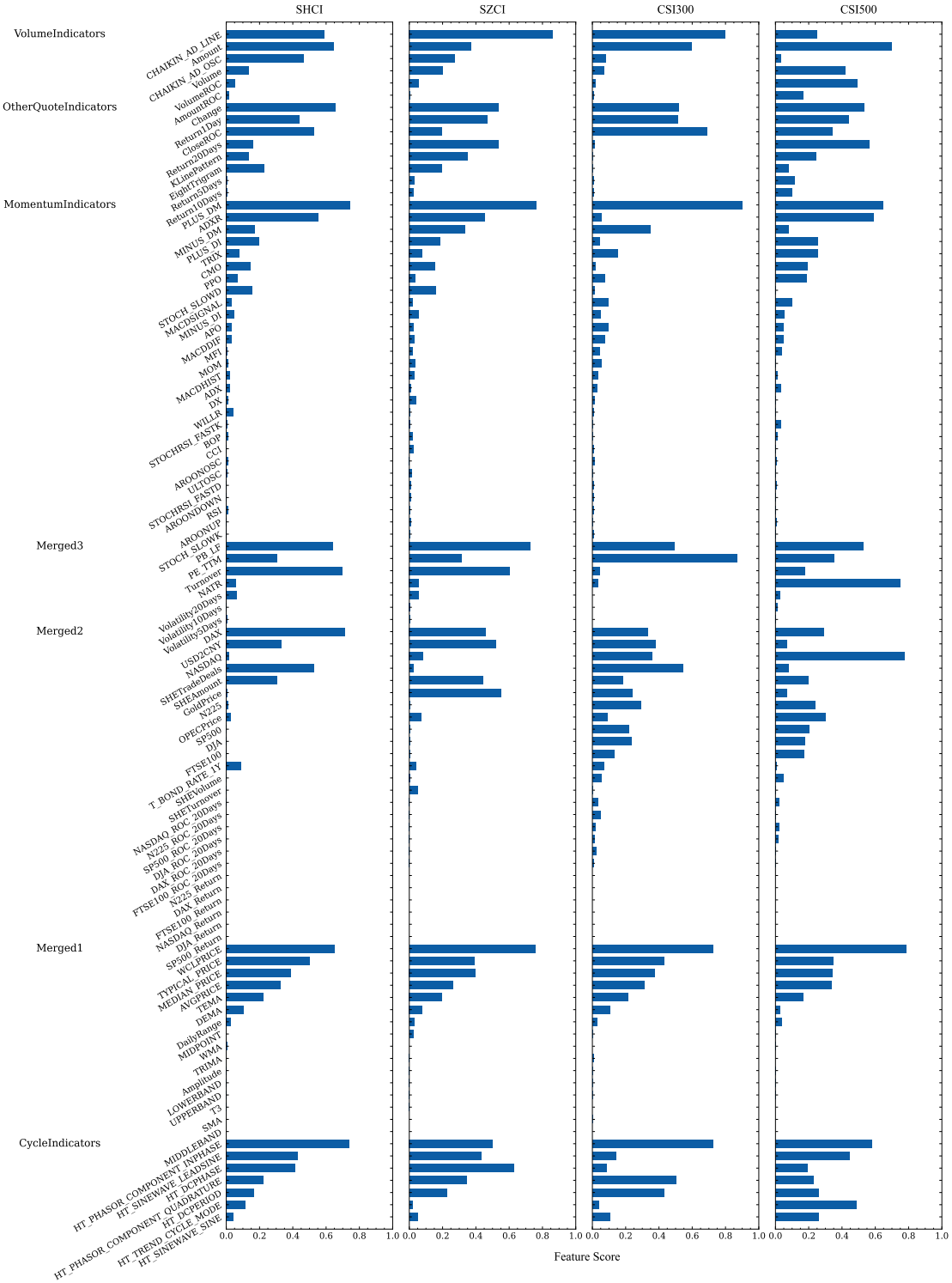

Figure 4: The result of feature selection.

Look at the results from different feature categories. It can be seen that the most important features of category "VolumeIndicators" are "CHAIKIN_AD_LINE" and "Amount", having high scores on each index. Except "Return5Days" and "Return10Days", all of other features in "OtherQuoteIndicators" have high scores. In "MomentumIndicators", most features have weak impact. In category "Merged3", three fundamental indicators, "PB_LF", "PE_TTM" and "Turnover", had the best performance on average. In category "Merged2", the distribution of feature scores is flatter under CSI300 and CSI500 than other indices. However, features in "Merged1" have almost the same distribution of scores across 4 indices. Finally, in "CycleIndicators", most features show considerable influence on all of 4 indices.

We eliminated the features whose scores are 0 or close to 0 in each category and got the remaining features for modeling, as shown in Table.4.

Table 4: The selected features for each stock index.

Index	Feature Name	Count
SHCI	CHAIKIN_AD_LINE, Amount, CHAIKIN_AD_OSC, Volume, VolumeROC, Change, Return1Day, CloseROC, Return20Days, KLinePattern, EightTrigram, PLUS_DM, ADXR, MINUS_DM, PLUS_DI, TRIX, CMO, PPO, STOCH_SLOWD, MINUS_DI, MACDSIGNAL, APO, MACDDIF, WILLR, PB_LF, PE_TTM, Turnover, NATR, Volatility20Days, DAX, USD2CNY, SHETradeDeals, SHEAmount, T_BOND_RATE_1Y, WCLPRICE, TYPICAL_PRICE, MEDIAN_PRICE, AVGPRICE, TEMA, DEMA, DailyRange, HT_PHASOR_COMPONENT_INPHASE, HT_SINEWAVE_LEADSINE, HT_DCPHASE, HT_PHASOR_COMPONENT_QUADRATURE, HT_DCPERIOD, HT_TREND_CYCLE_MODE, HT_SINEWAVE_SINE	48
SZCI	CHAIKIN_AD_LINE, Amount, CHAIKIN_AD_OSC, Volume, VolumeROC, Change, Return1Day, CloseROC, Return20Days, KLinePattern, EightTrigram, PLUS_DM, ADXR, MINUS_DM, PLUS_DI, TRIX, CMO, PPO, STOCH_SLOWD, MINUS_DI, DX, PB_LF, PE_TTM, Turnover, NATR, Volatility20Days, DAX, USD2CNY, NASDAQ, SHEAmount, GoldPrice, OPECPrice, WCLPRICE, TYPICAL_PRICE, MEDIAN_PRICE, AVGPRICE, TEMA, DEMA, HT_PHASOR_COMPONENT_INPHASE, HT_SINEWAVE_LEADSINE, HT_DCPHASE, HT_PHASOR_COMPONENT_QUADRATURE, HT_DCPERIOD, HT_SINEWAVE_SINE	44
CSI300	CHAIKIN_AD_LINE, Amount, CHAIKIN_AD_OSC, Volume, Change, Return1Day, CloseROC, PLUS_DM, ADXR, MINUS_DM, PLUS_DI, TRIX, PPO, MACDSIGNAL, MINUS_DI, APO, MACDDIF, MFI, MOM, PB_LF, PE_TTM, Turnover, NATR, DAX, USD2CNY, NASDAQ, SHETradeDeals, SHEAmount, GoldPrice, N225, OPECPrice, SP500, DJA, FTSE100, T_BOND_RATE_1Y, SHEVolume, WCLPRICE, TYPICAL_PRICE, MEDIAN_PRICE, AVGPRICE, TEMA, DEMA, DailyRange, HT_PHASOR_COMPONENT_INPHASE, HT_SINEWAVE_LEADSINE, HT_DCPHASE, HT_PHASOR_COMPONENT_QUADRATURE, HT_DCPERIOD, HT_TREND_CYCLE_MODE, HT_SINEWAVE_SINE	50
CSI500	CHAIKIN_AD_LINE, Amount, Volume, VolumeROC, AmountROC, Change, Return1Day, CloseROC, Return20Days, KLinePattern, EightTrigram, Return5Days, Return10Days, PLUS_DM, ADXR, MINUS_DM, PLUS_DI, TRIX, CMO, PPO, MINUS_DI, MACDSIGNAL, APO, MACDDIF, MFI, PB_LF, PE_TTM, Turnover, NATR, DAX, USD2CNY, NASDAQ, SHETradeDeals, SHEAmount, GoldPrice, N225, OPECPrice, SP500, DJA, FTSE100, SHEVolume, WCLPRICE, TYPICAL_PRICE, MEDIAN_PRICE, AVGPRICE, TEMA, DailyRange, HT_PHASOR_COMPONENT_INPHASE, HT_SINEWAVE_LEADSINE, HT_DCPHASE, HT_PHASOR_COMPONENT_QUADRATURE, HT_DCPERIOD, HT_TREND_CYCLE_MODE, HT_SINEWAVE_SINE	54

4.2 Result of Stock Index Prediction

The forecasting results of SPN-GAN on 4 stock indices are shown in Table.5. The bold values indicate the best results under corresponding metrics and the underlined values indicate the second best performance.

It can be found that, overall, SPN-GAN has achieved the best performance on all of 4 indices. From the perspective of forecasting error, except MAEs under SHCI and SZCI which rank third and second respectively, SPN-GAN is consistently superior to all other models. However, the MAE_{adj}

which is adjusted by taking the accuracy of predicted moving direction into account of SPN-GAN is still the lowest. As for classification metrics, the ACCU of SPN-GAN is always the optimal under 4 stock indices while the Recall, Precision and F_1 of it are also better than those of other models in most cases.

In order to explore the influence of SPN shared by G and D on our model, the structure of SPN was separately embedded into G and D, constructing non-SPN GAN model. The comparative results of SPN-GAN and non-SPN GAN are shown in Table.6. We find that the forecasting errors of non-SPN GAN are slightly lower than those of SPN-GAN, generally. But SPN-GAN has better performance on the changing direction prediction. On the whole, the two models have little difference in forecasting performance. In addition, by comparing these results with those of plain GAN in Table.6, it can be evidently observed that SPN helped GAN produce superior predictions, suggesting that meaningful intermediate representation of historical information has been effectively learned.

Table 5: The performance of different models on 4 stock indices.

Index	**Model**	**Forecasting Error**				**Classification Metric (%)**			
		MSE	MAE	MSE_{adj}	MAE_{adj}	ACCU	Recall	Precision	F_1
SHCI	ARIMA-GARCH	3944.76	46.11	5347.63	60.52	52.92	51.08	54.00	52.50
	SVM	3021.82	41.22	4203.14	54.98	53.69	53.81	54.64	54.19
	DT	3054.03	42.14	4189.66	55.95	52.58	54.61	53.45	53.97
	GBDT	2372.75	36.39	3260.76	48.14	54.31	52.52	55.49	53.95
	ANN	1530.07	28.66	2022.08	37.24	54.27	54.89	55.19	55.03
	LSTM	2327.83	33.63	3103.82	43.84	52.66	55.40	53.47	54.39
	GAN	1496.30	**28.21**	1897.56	35.32	53.14	**61.45**	53.50	57.14
	SPN-GAN	**1386.72**	29.83	**1645.82**	**34.85**	**56.77**	56.92	**57.72**	**57.25**
SZCI	ARIMA-GARCH	81304.75	215.54	110695.52	282.85	52.81	50.00	53.97	51.91
	SVM	76839.14	200.03	106548.34	265.87	53.21	55.37	53.97	54.63
	DT	83925.31	227.43	116366.12	302.63	53.28	53.92	54.18	54.05
	GBDT	63316.73	185.00	87438.30	245.62	54.49	56.38	55.20	55.77
	ANN	37004.54	**142.91**	49243.18	185.84	53.21	53.06	54.18	53.60
	LSTM	50030.77	156.38	65651.31	200.98	51.85	55.44	52.59	53.98
	GAN	49112.24	157.48	64780.77	201.33	52.70	56.52	53.44	54.83
	SPN-GAN	**28773.24**	145.82	**36796.96**	**157.82**	**56.21**	**57.09**	**57.02**	**57.05**
CSI300	ARIMA-GARCH	10818.98	73.75	14677.78	96.99	51.63	48.22	54.72	51.26
	SVM	7017.16	62.03	9582.65	81.67	53.04	53.08	55.78	54.39
	DT	10524.54	81.54	14514.42	108.25	52.54	52.84	55.24	54.01
	GBDT	5351.46	54.77	7389.65	72.87	53.96	54.18	56.68	55.39
	ANN	6439.86	57.66	8702.48	75.75	53.13	54.11	55.73	54.90
	LSTM	6261.85	55.42	8245.28	70.72	52.42	**59.56**	54.50	**56.87**
	GAN	4911.71	50.38	6558.32	65.46	54.67	56.08	57.19	56.63
	SPN-GAN	**3249.37**	**41.34**	**4144.30**	**52.24**	**54.71**	55.21	**57.35**	56.26
CSI500	ARIMA-GARCH	20687.43	107.48	28304.11	142.93	49.72	42.71	55.48	48.27
	SVM	14954.75	87.03	20469.43	115.20	52.94	50.34	58.28	54.01
	DT	19789.76	111.11	27622.40	148.81	51.96	53.32	56.66	54.89
	GBDT	13883.49	87.49	19231.57	116.54	52.94	53.27	56.41	54.64
	ANN	11438.81	78.46	15498.84	103.62	52.29	51.45	57.29	54.21
	LSTM	13748.35	82.12	18062.13	105.60	52.24	55.36	56.72	55.98
	GAN	10877.18	76.29	14495.59	99.26	53.45	**55.95**	57.91	56.89
	SPN-GAN	**6812.75**	**60.75**	**8949.39**	**77.85**	**55.00**	55.70	**59.65**	**57.60**

Table 6: Predictive results of SPN-GAN and non-SPN GAN.

Index	Model	Forecasting Error				Classification Metric (%)			
		MSE	MAE	MSE_{adj}	MAE_{adj}	ACCU	Recall	Precision	F_1
SHCI	non-SPN GAN	**1250.42**	**25.51**	**1577.52**	**32.07**	55.12	**58.10**	55.69	56.87
	SPN-GAN	1396.05	29.87	1639.83	34.80	**57.32**	56.37	**58.39**	**57.36**
SZCI	non-SPN GAN	**28607.84**	**124.59**	**36181.31**	**155.82**	55.45	56.80	56.20	56.50
	SPN-GAN	29173.19	145.68	38262.21	159.32	**56.66**	**57.02**	**57.52**	**57.27**
CSI300	non-SPN GAN	**3199.84**	**40.66**	**3990.41**	**50.66**	**56.00**	**56.87**	**58.54**	**57.69**
	SPN-GAN	3213.74	41.18	4036.96	51.84	55.13	54.98	57.86	56.38
CSI500	non-SPN GAN	**6559.69**	59.91	**8271.57**	**74.94**	54.62	**57.65**	58.85	**58.25**
	SPN-GAN	6583.20	**59.11**	8522.75	75.33	**55.60**	55.87	**60.33**	58.01

Table.7 shows model complexity and training cost of SPN-GAN and non-SPN GAN, where training time is for one epoch and averaged over 4 stock indices. We can learn that non-SPN GAN needs to be trained at a finer learning rate so that more epochs are spent for convergence and SPN-GAN has much less parameters than non-SPN GAN so that its training time is shorter. Therefore, training SPN-GAN takes much less time and memory space and it's the preferred one given their similar predictive performance.

Table 7: Model complexity and training overhead of SPN-GAN and non-SPN GAN.

	Number of Parameters	Epochs to Convergence	Training Time	Learning Rate of G and D
non-SPN GAN	1,131,369	1500	27.6s	0.0002, 0.0001
SPN-GAN	710,377	800	18.2s	0.001, 0.0005

5. Conclusion

In this paper, we propose SPN-GAN model by introducing SPN network shared by G and D into GAN framework and taking into account the direction of stock index changes in model's optimization strategy. The empirical results on 4 representative A-share stock indices show that SPN-GAN has better forecasting performance than the benchmarks. The comparison between SPN-GAN and non-SPN GAN indicates that SPN improves the performance of GAN and the sharing design markedly reduces the number of model parameters and training overhead. In the future, we will extend this research by covering more types of data sources such as financial news and social media, applying different network structures for SPN and designing model based investment strategies.

References

[1] Sharaf M., Hemdan E.E., El-Sayed A. and El-Bahnasawy N.A. (2021) StockPred: a framework for stock Price prediction. Multimedia Tools And Applications, 12, 17923-17954.

[2] Qiu Y., Song Z.W. and Chen Z.S. (2022) Short-term stock trends prediction based on sentiment analysis and machine learning. Soft Computing, 5, 2209-2224.

[3] Goodfellow I., Bengio Y. and Courville A. Deep learning. MIT Press, Cambridge, MA, 2016.

[4] Jiang W. (2021) Applications of deep learning in stock market prediction: Recent progress. Expert Systems with Applications, 184.

[5] Zaheer S., Anjum N., Hussain S., Algarni A.D., Iqbal J., Bourouis S. and Ullah S.S. (2023) A Multi Parameter Forecasting for Stock Time Series Data Using LSTM and Deep Learning Model. Mathematics, 3.

[6] Goodfellow I.J., Pouget-Abadie J., Mirza M., Xu B., Warde-Farley D., Ozair S., Courville A. and Bengio Y. (2014) Generative Adversarial Nets. Proceedings of Advances In Neural Information Processing Systems, 2672-2680.

[7] He B. and Kita E. (2020) Stock Price Prediction by Using Hybrid Sequential Generative Adversarial Networks. proceedings of the 20th IEEE International Conference on Data Mining (ICDM).

[8] Wu W., Huang F., Kao Y., Chen Z. and Wu Q. (2021) Prediction Method of Multiple Related Time Series Based on Generative Adversarial Networks. Information, 2.
[9] Vaswani A., Shazeer N., Parmar N., Uszkoreit J., Jones L., Gomez A.N., Kaiser Ł. and Polosukhin I. (2017) Attention is all you need. Advances in neural information processing systems.
[10] Kingma D.P. and Ba J. (2014) Adam: A method for stochastic optimization. arXiv preprint arXiv:14126980.
[11] Yun K.K., Yoon S.W. and Won D. (2021) Prediction of stock price direction using a hybrid GA-XGBoost algorithm with a three-stage feature engineering process. Expert Systems with Applications, 115716.
[12] Benediktsson J. Python wrapper for TA-Lib. https://ta-lib.github.io/ta-lib-python.
[13] Lin Y.H., Liu S.C., Yang H. and Wu H. (2021) Stock Trend Prediction Using Candlestick Charting and Ensemble Machine Learning Techniques With a Novelty Feature Engineering Scheme. IEEE ACCESS, 101433-101446.