ALPHA DISCOVERY NEURAL NETWORK BASED ON PRIOR KNOWLEDGE

Abstract

In financial automatic feature construction task, genetic programming (GP) is the state-of-the-art technique. It employs reverse polish expression to represent features and then simulate the evolution process. However, with the development of deep learning, more choices to design this algorithm are available. This paper proposes Alpha Discovery Neural Network (ADNN), equipped with different kinds of feature extractors to construct diversified financial technical factors based on prior knowledge. The experiment result shows that both fully-connected network and recurrent network are good at extracting information from financial time series, but convolution network structure can't effectively extract this information. ADNN effectively enrich the current factor pool because in all cases, ADNN can construct more informative and diversified features than GP. Moreover, features constructed by ADNN can always improve original strategies' return, Sharpe ratio and max draw-down.

1 Introduction

Feature construction is a process that discovers missing information about the relationships between features and augments the space of features by inferring or creating new features. During this process, new features can be generated from a combination of existing features [Motoda and Liu, 2002]. A more straightforward description is that the algorithms use operators, hyper-parameters and existing features to construct a new feature. Feature construction is typically conducted by human experts. However, the cost of processing sophisticated information by human is substantially larger than the one by machine. Thus, automatic feature construction has received an increasing amount of attention [Liu and Motoda, 1998].

Sometimes both feature construction and feature selection occur in the same procedure. These methods consist of wrapper, filtering and embedded [Chandrashekar and Sahin, 2014]. Filtering is easy but achieves poor performance; it utilizes only some criteria to choose a feature and sometimes it can help us to monitor the feature construction process. Wrapper performs well by directly applying the model's results as an object function. Thus, it can treat an individually trained model as a new constructed feature. However, a considerable amount of computational resources and time is required. Embedded is a method that uses generalized factors and a pruning technique to select or combine features, which serves as a middle choice between filtering and wrapper. The most well-known and frequently employed automatic feature construction method is Genetic Programming (GP), which is a kind of wrapper method that reverses polish expression to represent features and then simulates the evolution process. However, different domains require different object functions, and the input data's data structure may differ [Krawiec, 2002]. Thus, it's very important to do this task within a specific domain. This method has been shown to work well in many industries, such as health care [Kwakkenbos *et al.*, 2010], object detection [Lillywhite *et al.*, 2013], education [Romero *et al.*, 2004], and finance [Thomas and Sycara, 1999]. However, the drawback of the method is that the constructed formulas are very similar and may cause collinearity. With the development of deep learning, more and more researchers begin to use neural network to extract features from raw data and then add a fully-connected layer to reshape the feature's output. Similarly, a trained model represents a newly constructed feature. In Yang Zhong's research on pattern recognition tasks, he employs a CNN model to construct facial descriptors, and this method produces features that have considerably more information than the past method [Zhong *et al.*, 2016]. K Shan conducts experiments in this task and employs a deeper and wider convolution neural network [Shan *et al.*, 2017]. Hidasi B uses a recurrent neural network to pre-locate the feature-rich region and successfully constructs more purified features [Hidasi *et al.*, 2016]. In a text classification task, Botsis T leverages recurrent neural networks to build a rule-based classifier among text data, in which each classifier represents a part of the text [Botsis *et al.*, 2011]. S Lai proposes a network structure that uses both a recurrent neural network and convolution neural network to extract text information. Their produced features contain more information than previous work [Lai *et al.*, 2015]. As shown in this development, a suitable network structure can determine the success of produced features. With the help of a neural network's strong fitting ability, we can produce highly informative features by tailoring the network structure for different industries.

In this paper, a novel network structure called ADNN is proposed, which can uses deep neural network to automatically construct financial factors. And different feature extractors and prior knowledge are equipped to further improve its performance in real-world situations.

2 ALGORITHM INTRODUCTION

2.1 Network Structure

The network structure of the ADNN is shown in Fig 1. The major contributions of this novel network structure includes: 1). ADNN uses spearman correlation to serve as loss function, which mimics common human practices of quantitative investment. And this setting also makes each batch become meaningful. 2). A meaningful derivable kennel function is proposed to replace the underivable operator rank(). 3). The network are pre-trained with many classical financial descriptors. These features serve as initial seeds, which can improve the diversity of newly constructed features.

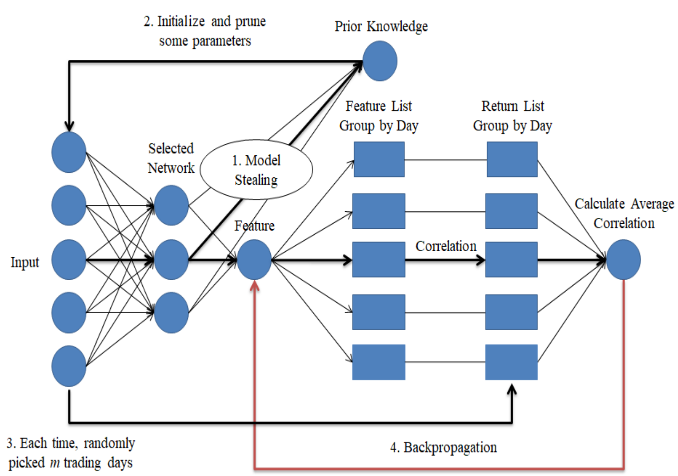

Figure 1: Alpha discovery neural network's structure

As shown in Fig 1, grouping each iteration's output by trading day is an idea based on the multi-factor strategy principle, because quantitative investors care about only the relative strength of each stock on the same trading day. We apply Spearman correlation to calculate the correlation between a factor value and a factor return. This setting can help us obtain powerful features that are suitable for forecasting the future. And this setting also makes our batch size and sampling rules become meaningful. Only the data belongs to the same trading day, should be involved in the same batch.

However, spearman correlation uses operator rank() to gid rid of some anomalies in financial time series. Rank() is not derivable, which is not acceptable for neural network. Thus, we uses a derivable kernel function g(x) to replace rank().

$$g(x) = \frac{1}{1 + exp(-p * \frac{x-\bar{x}}{2*std(x)})} \tag{1}$$

As shown in formula 1, at first, it projects x into a normal distribution which is zero-centralized. Next, it uses a hyper-parameter p to make sure that the 2.5%-97.5% of data should lay in the range between $[mean - 2 * std, mean + 2 * std]$. Thus, p equals to 1.83. we can get p=1.83. For example, one out-lier $x_i = \bar{x} + 2 * std(x)$, and $\frac{g(x_i)-g(\bar{x})}{g(\bar{x})} \le \frac{x_i - \bar{x}}{\bar{x}}$, so the result is $std \le 0.362 * \bar{x}$. It means, if one distribution's standard deviation is large, and it is larger than $0.362 * \bar{x}$, the g(x) can shorten the distance between outliers and the central point. If the distribution's standard deviation is very small, g(x) will make it worse. However, even in this case, we can make sure that 95% of the points are between $[mean - 2 * std, mean + 2 * std]$, which is acceptable. With kernal function g(x), we get ADNN's loss function, is defined in formula 2.

$$IC(x, y) = \frac{cov(g(x) - g(\bar{x}), g(y) - g(\bar{y}))}{var(g(x) - g(\bar{x})) * var(g(y) - g(\bar{y})))} \tag{2}$$

In formula (2), x refers to the feature value, y refers to the sample's return in the next 5 trading days, cov is an operator for calculating the co-variance, and var is an operator to calculate the variance.

2.2 Put prior knowledge into network

Combining with model stealing [Juuti *et al.*, 2019] and pruning on input data can improve signal's diversity. Model stealing means that if the input x and output y are known, our network can obtain the suitable parameter w to fit this function. However, its technique is not always helpful to learn a distribution without tailoring the network structure. If we have a fixed network structure, and we have no idea about the target distribution, the techniques such as removing the outliers (for continuous prior knowledge) and using high temperature T (for discrete prior knowledge) can be helpful to stealing the knowledge.

Pre-train uses $f(x) = a(w^T x + b)$ to embed the input data (the data is embedded by MLP, w means kenel weights, b means bias, a means activation function) and then use this embedded layer to mimic the prior knowledge. In this part, we use the mean squared error as the object function.

$$\arg\min_{W} \frac{1}{n} \sum_{i=1}^{N} (y_i - f(x_i))^2 \tag{3}$$

For deep neural network, almost all technical factors can be easily learned. Here, MSE or MAE can't represent the real pre-training performance, because all factor values are really small, which makes all MSE value really small. In order to have a better measure of the performance, $\frac{1}{n}\sum_{i=1}^{N} abs(\frac{y_i - f(x_i)}{y_i})$ is used to measure its error rate. The error rate of SMA, ROC, RSI, BOLL, FORCE, and other typical financial descriptors are selected as prior knowledge for pre-train. Some descriptors with different parameters such as RSI(6,12) and RSI(7,21) will be regarded as different prior knowledge, because they have given enough diversity to ADNN. The error rate of 50 financial descriptors selected in this paper is 0.055 ± 0.031.

Why is pre-train with prior knowledge needed? According to the concept of Muti-task Learning, pre-training can permanently keep some part of the domain knowledge in the network. The knowledge is the source of diversity, we should keep it. Pruning can help us to keep it, and pruning can filter out noisy signals from prior knowledge. And the pruning rate should be controlled, a big pruning rate will bring too much difficulty to the final optimization direction, a small pruning rate will lose the diversity. Based on Frankle and Carbin, 2018, the ideal pruning rate should be about 0.2-0.5, and the mask matrix is only made up by 0 and 1. All the setting is same as [Frankle and Carbin, 2018], and here are more explanations. After embedding the data as f(x), we get its parameter matrix w. Then we create a mask matrix to prune the parameter. For example, x_{ij} in parameter matrix is relatively small, which means some of the input data is useless. Then m_{ij}=0 is set to permanent mask this value. If the x_{ij} is not useless, then we set m_{ij}=1. This method can help us keep the diversity in the network. What's more, it can help us focus on improving current situation, but not to heading the unknown local minimal. The pruning process is shown in formula (4):

$$f(x) = (w \cdot m)^{\mathrm{T}} x + b \tag{4}$$

After pre-training and pruning the network, we use the object function shown in Fig 1 to train ADNN. We simply reshape the input data into a picture. And then we uses Saliency Map to look how the raw data contribute to the final constructed factor. The training process is shown in Fig 2, the y-axis is [open price, high price, low price, close price, volume], the x-axis is time step.

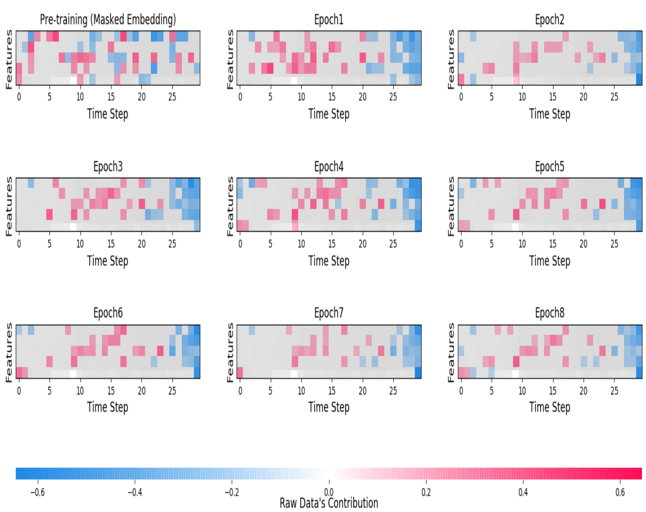

Figure 2: How ADNN leads the prior knowledge to become a better technical factor, ADNN can help us to adjust the raw data's contribution, and make it better according to its initial state

Even if the network structure without prior knowledge can beat the GP on automatic financial feature construction task, noisy financial data may cause troubles for training processes. Without guidance of prior knowledge, the training of FCN can be easily trapped in local optimum. Thus, initializing ADNN by existing features from prior knowledge can be a potential route to seek new features. How to effectively put prior knowledge into ADNN shall be the key. Thus, in this paper, prior knowledge are the seeds to serve the source of diversity.

2.3 Different feature extractors

We conduct experiments on different feature extractors, for having better understanding of the financial time series. A lot of experiments show that different feature extractors have its own strong comings and short comings on this task, these performances varies on information coefficient, diversity, training time, strategy return, etc. There are two motivations to conduct experiments on different feature extractors. First, different feature extractors require different input data's data structure. After performing a literature review and consulting professional experts in the market, we discover many different ways to organize the input data. However, none of them can prove that their structure is the best. Thus, experiments on these structures should be performed. We conduct experiments on many basic networks and SOTA works, including Fully-connected neural network), Le-net, Resnet-50, LSTM, Transformer (Transformer is too large, we use only its self-attention encoder) and TCN. FCN has 3-5 hidden layers, with 64-128 neurals in each layer. We recommend the use of tanh and relu to serve as the activation function, and the output layer should not be activated. For Le-net, Resnet-50, Transformer and TCN, we did not change its structure but employ only a smaller stride because the input data has low dimension compared with a real picture. The second motivation is that different extractors have their own strong comings and short comings. Some of them aim at extracting temporal information but the others aims at spatial information. Some of them designed for a long term series, but some of them are designed for quick training. We think they can make our factor pool more diversified.

3 Experiments

3.1 Experiment setting

We use daily trading data in the Chinese A-share stock market (in the following part, we call it A-share market data), including the daily open price, high price, low price, close price and trading volume, in the past 30 trading days. The raw data is standardized by using its time-series mean and standard deviation in training set. Both the mean and standard deviation are calculated from the training set. We attempt to use these inputs to predict the stock return in the next 5 trading days (3-15 trading days also acceptable). Moreover, we should obey the market policy when we form a trading strategy. In A-share market, we can not short single stocks, but we can short some important index.

We have done a lot of experiment to select the reasonable hyper-parameters. During the model training process, we calculate the average value of Spearman Correlation in 20 randomly selected trading days. For each experiment, 250 trading days serve as the training set (no technical factor can work well for a long period of time), the following 30 trading days serve as the validation set, and the following 90 trading days serve as the testing set. The constructed factors can keep high IC during the next 90 trading days. A longer period of time will make it perform badly. Thus, in real application, investors should construct factors during each period of time. To make a fair comparison, the same setting are deployed for the GP algorithm. This algorithm's logic references relative work [Thomas and Sycara, 1999] and [Allen and Karjalainen, 1999]. Besides, the input data's period and type should be the same.

In this paper, we analyze the construed features' performance from different perspectives. Normally, institutional investors uses information coefficient (IC) , shown in formula 2, to measure how much information carried by a feature. For diversity, the cross-entropy is used to measure the distance between two different features' distributions on the same trading day.

$$Distance(f_1, f_2) = \sum softmax(f_1) log \frac{1}{softmax(f_2)} \tag{5}$$

In formula (5), f_1 and f_2 refers to different features' distribution in the same trading day. The softmax function can help us get rid of the effect from scale without losing its rank information. And k-means is used to cluster the distance matrix of this relative distance between two features. The average distance between each cluster center refers to the diversity of this algorithm in this trading day. Besides measurements of IC and diversity, the performance of a trading strategy based on the constructed features are also measured, such as absolute return, max-drawdown and sharp-ratio. Basically, all these indicators are really important to assess a feature's performance.

3.2 Beat the state-of-the-art technique

The network structure shown in Fig 1 can equip ADNN with different neural networks. In this test case, ADNN equipped with 4 layer fully-connected neural network. The experiment shows that ADNN can beat the genetic programming algorithm. We put forward three schemes to help illustrate ADNN's contribution and to show how it beat the genetic programming. Only GP means only using genetic programming, Only ADNN means only use ADNN to construct factors, GP&ADNN means uses GP's value to initialize ADNN, and then construct factors. All the experiments are conducted out of sample, we summarized it in Table 1.

Table 1: The performance of different schemes

Object	Information Coefficient	Diversity
Only GP	0.094	17.21
GP&ADNN	0.122	25.44
Only ADNN	0.107	21.65

As shown in Table 1, Only ADNN is better than Only GP, which means ADNN outperform GP on this task. And we also find that GP&ADNN is the best, it means that our method can even improve the performance of GP. However, in real practice, we should leverage the constructed factors to form a multi-factor strategy and compare its performance with genetic programming. The specific strategy setting is same as section 3.4, and we have repeated this experiment on different period of time, the long-term backtest result is shown in table2.

Table 2: Strategy's absolute return for each scheme.

Time	Only GP	GP&ADNN	Only ADNN	ZZ500
Train:2015.01-2015.12 Test: 2016.02-2016.03	+2.59%	+5.74%	+4.52%	+1.67%
Train:2016.01-2016.12 Test: 2017.02-2017.03	+5.40%	+10.26%	+8.33%	+2.53%
Train:2017.01-2017.12 Test: 2018.02-2018.03	-5.27%	-4.95%	-4.16%	-6.98%
Train:2018.01-2018.12 Test: 2019.02-2019.03	+13.00%	+15.62%	+15.41%	+13.75%

As shown in Table2, Only ADNN always have better performance than Only GP during long-term backtest. It shows that our method have also beaten the SOTA in real practice. However, will there be more powerful feature extractors to discovery knowledge from financial time series? And what shall be the suiatble input data structure for financial time series?

3.3 Comparing different feature extractors

All experiments are conducted in the same setting mentioned in section 3.1, and the results are summarized after generating 50 features. For the hardware equipment, we use 20 g GPU (NVIDIA 1080Ti) and 786 g CPU (Intel Xeon E5-2680 v2, 10 cores). Based on this setting, we show the amount of time that we need to train 50 neural networks. Moreover, the time to restore 50 trained networks and obtain their feature values will be substantially faster than traditional features. Because most traditional features are constructed with complicated explicit formulas, these formulas are not suitable for matrix computing. Using neural networks to represent features is matrix computing, which can have a faster testing speed.

Table 3: The higher are the information coefficient (IC) and diversity, the better is their performance. Normally, a good feature's long-term IC should be higher than 0.05, but it cannot be higher than 0.2 in an A-share market.

Type	Network	IC	Diversity	Time
Baseline	GP	0.072	17.532	0.215 hours
Vanilla	FCN	0.124	22.151	0.785 hours
Spatial	Le-net	0.123	20.194	1.365 hours
	Resnet-50	0.108	21.403	3.450 hours
Temporal	LSTM	0.170	24.469	1.300 hours
	TCN	0.105	21.139	2.725 hours
	Transformer	0.111	25.257	4.151 hours

Shown in Table3, basically, all neural networks can produced more diversified features than using GP. But temporal extractors are especially better at producing diversified features, such as LSTM [Hochreiter and Schmidhuber, 1997] and Transformer [Vaswani *et al.*, 2017]. As for TCN [Lea *et al.*, 2017], the author who put forward this network structure proves its ability to capture the temporal rules buried in data.

However, there is a huge difference. TCN relies on convolution neural network, but LSTM and Transformer still contains recurrent neural network (Normally, transformer uses recurrent neural network to embedded the input data). The existence of recurrent neural network structure may contribute to the difference of diversity. For Le-net [LeCun *et al.*, 1998] and Resnet [He *et al.*, 2016], they don't provide us with more informative features. It looks like that the convolution network structure is not suitable to extract information from financial time series.

To conclude all neural networks mentioned above can produce more informative and diversified features than GP. For the same type of networks, the sample network performs relatively better than the sophisticated networks. The result suggests that most investors' decisions are linear and use of a simple neural network is suitable. Because an increasing number of people use a neural network to do trading, the nonlinear part will be increased. In this situation, we believe that a sophisticated network will be more suitable to fit and capture trading signals.

3.4 Real-world use case

The real-world use case here combines human experts' features and a constructed feature to maximize the performance of the quantitative investment strategy. This experiment attempts to create trading strategies using both features constructed by human experts and our ADNN to see if ADNN has value added for human traders.

Table 4: Back testing starts from Jan 2019 to June 2019. The investment target is all A-share, except for the stock can't be traded during this period of time. Strategy's commission fee is 0.5%.

Type	Target	Group	Strategy Return	Max Drawdown	Sharpe Ratio
Baseline	ZZ500	Stock Index	19.60%	13,50%	1.982
	HS300	Stock Index	18.60%	20.30%	1.606
	PK	PK 50	24.70%	18.90%	2.314
	GP	GP 50	17.60%	25.30%	1.435
		GP-PK 50	25.40%	14.80%	2.672
Vanilla	FCN	New 50	20.60%	15.80%	2.189
		Combined 50	29.60%	15.70%	3.167
Spatial	Le-net	New 50	18.00%	16.90%	1.800
		Combined 50	27.50%	16.40%	2.921
	Resnet-50	New 50	19.90%	15.40%	1.962
		Combined 50	29.30%	17.20%	2.787
Temporal	LSTM	New 50	19.50%	13.00%	2.205
		Combined 50	29.90%	15.00%	3.289
	TCN	New 50	22.40%	14.70%	2.440
		Combined 50	26.90%	16.80%	2.729
	Transformer	New 50	21.10%	15.90%	2.203
		Combined 50	27.20%	15.10%	2.806

In the training set, the sample whose return ranked in the top 30% in each trading day is labeled as 1 and the sample whose return ranked in the last 30% of each trading day is labeled as 0. We abandon the remaining samples in the training set [Fama and French, 1993]. After training these features with XGBoost [Chen *et al.*, 2015] using binary logistics mode, the prediction result reflects the odds that this stock has outstanding performance in the following 5 trading days. It defines the 50 features constructed by human experts as PK 50, the features constructed by ADNN as New 50, and the features constructed by both GP and PK as GP-PK 50. In separate experiments, we use XGBoost to pre-train both PK 50 and New 50 in training set, and then using the weight score from XGBoost to choose the 50 most important features as Combined 50.

As shown in Table 4, HS300 and ZZ500 is an important stock index in the A-share market. A comparison with these indexes can show our strategy's performance regardless of the macro environment. The strategy return indicates how much we can win by using this strategy. Max-drawdown indicates how much we will lose in the worst case that if we continue to use this strategy during this period of time. Sharp-ratio means how much we can win after considering the risk. This indicator can show the strategy's performance considering both revenue and risk.

For the new 50, although they have a higher IC than the PK 50, this strategy does not achieve a higher strategy return. One of potential explanation is that the diversity of features constructed by human experts, we discover that it is remarkably higher than the automatic constructed features. Low diversity will cause collinearity and reduce the contribution of constructed features to the strategy. Thus, combining the features from both the new and existing human experts' features is reasonable. This approach is more reasonable and suitable for practical application. In all cases, our combine 50 is better than PK 50 and GP-PK 50, which means that the ADNN can construct more useful features than GP with a reasonable feature selection process. Fig 3 plots the exceed return curve of these strategies over HS300 Index.

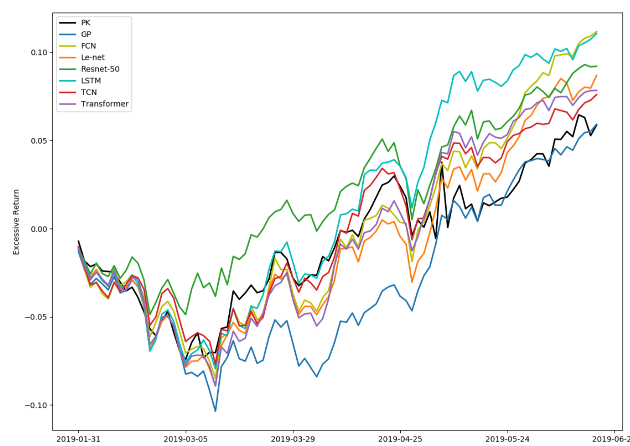

Figure 3: Different feature extractors' exceed return in testing set (hedge on HS300 Index)

Shown in Fig 3, all these curves are similar, due to the fact that they all shared some factors from PK50. GP's curve is always in the lowest position, which means GP provide us with fewest effective trading signals. However, all these schemes have positive exceed return, which means all these schemes and all these features are useful, and they successfully beat the market. Among the schemes powered by ADNN, FCN and LSTM perform best. During this period of time, they have beaten the market more than 10 percent. It is a reasonable performance because all the features are only constructed from price and volume data. They don't contain any fundamental data or even sentiment data. What's more, we will get a lot of extra information during feature construction process. This information is helpful to feature selection process. That's the main reason why some wrapper methods will do feature selection and construction at the same time. For further research, the current structure can be improved to conduct both the feature construction and feature selection process at the same time. This paper directly use this reasonable feature selection method, because it only focuses on feature construction task.

3.5 Comprehend the result

In section 3.3, it shows that the LSTM extracts more information than the FCN. We suspect that only the recurrent structure and fully connected structure are helpful for extracting information from financial time series.

To verify this idea, 150 features are constructed by using FCN, the network focused on spatial information and the network focused on temporal information. Then the diversity is clustered into three groups using k-means; this method has been mentioned in section 3.1. In order to be shown more clearly, they are clustered into three groups. To visualize this distance matrix, this matrix should be transformed into a 2D graph. We initialize one of the cluster centers is set as $(0, 0)$ and then determine the other two cluster centers according to their relative distance and a given direction. (This direction will influence only the outlook of this graph and will not influence the shared space between two different clusters.) For the other samples that belong to the same cluster, their location are determined according to their relative distance between a cluster center and a randomly generated direction. All the subplots share the same x-axis and y-axis. Thus, the different feature extractors' sparsity are compared according to the size of circle.

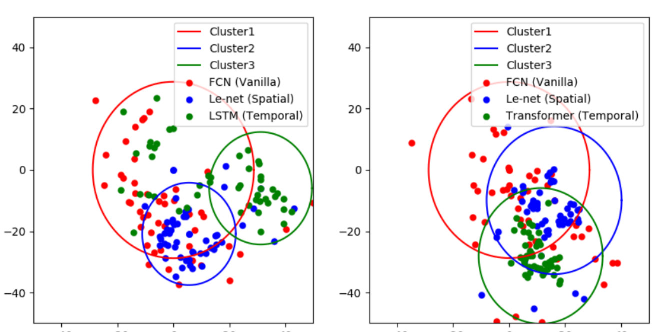

Figure 4: Cluster different networks (spatial against temporal)

As shown in Fig 4(left), the features constructed by the LSTM have the sparsest distribution, which means that the network structure that focuses on temporal information is excellent at extracting information from financial time series. However, a large space is shared by FCN and Le-net. We can regard Le-net's information as a subset of FCN. Combined with the convolution neural network's poor performance in sections 3.2 and 3.3, it looks like that the convolution neural network structure does not have a substantial contribution to extracting information from the financial time series. Fig 4(right) is an extra experiment, whose result supports this conclusion as well.

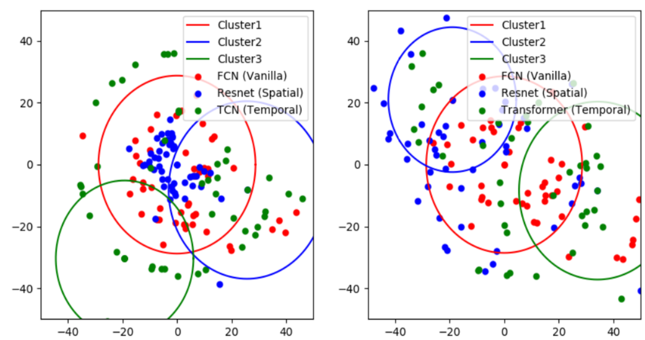

Figure 5: Cluster different networks (simple against complex)

Both in Fig 5(left) and Fig 5(right), the shared space between Vanilla and Spatial is still larger than the space shared between Vanilla and Temporal. Thus, it looks like the reason why LSTM's information coefficient can outperform other networks is that recurrent structure and fully-connected structure are truly helpful to extract information from financial time series. Meanwhile, these two structure focus on different part of information, which makes their joint effort more valuable. The experiments come out some conclusions. First, shaping the financial time series as a picture is not reasonable. A financial time series has considerably fewer points than a picture's pixels, which means that the series does not contain high-dimension data compared with a picture. However, a convolution and pooling structure is designed to extract information from high-dimension data, which explains why this network and input data's structure is not as competitive as other structures in this task. Second, the recurrent structure's performance is outstanding. We suggest that the time decay principle in this structure renders it more suitable for extracting information from a financial time series.

4 Conclusion

In this paper, we put forward alpha discovery neural network, which is equipped with different SOTA networks and different prior knowledge from human experts. In both the numerical experiment and the real-world use case, all networks can produce more informative and diversified features than the state-of-the-art technique in this task.Although different structures perform differently in this task, they have contributed to the diversity of the constructed features, and their performance is better than that of genetic programming in both the numerical experiment and the real-world case. Thus, they are also highly valuable for the trading strategy's factor pool. For further research, this framework may also be suitable for a company's fundamental and sentiment data.

References

[Allen and Karjalainen, 1999] Franklin Allen and Risto Karjalainen. Using genetic algorithms to find technical trading rules. *Journal of financial Economics*, 51(2):245–271, 1999.

[Botsis *et al.*, 2011] Taxiarchis Botsis, Michael D Nguyen, Emily Jane Woo, Marianthi Markatou, and Robert Ball. Text mining for the vaccine adverse event reporting system: medical text classification using informative feature selection. *Journal of the American Medical Informatics Association*, 18(5):631–638, 2011.

[Chandrashekar and Sahin, 2014] Girish Chandrashekar and Ferat Sahin. A survey on feature selection methods. *Computers & Electrical Engineering*, 40(1):16–28, 2014.

[Chen *et al.*, 2015] Tianqi Chen, Tong He, Michael Benesty, Vadim Khotilovich, and Yuan Tang. Xgboost: extreme gradient boosting. *R package version 0.4-2*, pages 1–4, 2015.

[Fama and French, 1993] Eugene F. Fama and Kenneth R. French. Common risk factors in the returns on stocks and bonds. *Journal of Financial Economics*, 33(1):3 – 56, 1993.

[Frankle and Carbin, 2018] Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. *arXiv preprint arXiv:1803.03635*, 2018.

[He *et al.*, 2016] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[Hidasi *et al.*, 2016] Balázs Hidasi, Massimo Quadrana, Alexandros Karatzoglou, and Domonkos Tikk. Parallel recurrent neural network architectures for feature-rich session-based recommendations. In *Proceedings of the 10th ACM conference on recommender systems*, pages 241–248. ACM, 2016.

[Hochreiter and Schmidhuber, 1997] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[Juuti *et al.*, 2019] Mika Juuti, Sebastian Szyller, Samuel Marchal, and N Asokan. Prada: protecting against dnn model stealing attacks. In *2019 IEEE European Symposium on Security and Privacy (EuroS&P)*, pages 512–527. IEEE, 2019.

[Krawiec, 2002] Krzysztof Krawiec. Genetic programming-based construction of features for machine learning and knowledge discovery tasks. *Genetic Programming and Evolvable Machines*, 3(4):329–343, 2002.

[Kwakkenbos *et al.*, 2010] Mark J Kwakkenbos, Sean A Diehl, Etsuko Yasuda, Arjen Q Bakker, Caroline MM Van Geelen, Michaël V Lukens, Grada M Van Bleek, Myra N Widjojoatmodjo, Willy MJM Bogers, Henrik Mei, et al. Generation of stable monoclonal antibody–producing b cell receptor–positive human memory b cells by genetic programming. *Nature medicine*, 16(1):123, 2010.

[Lai *et al.*, 2015] Siwei Lai, Liheng Xu, Kang Liu, and Jun Zhao. Recurrent convolutional neural networks for text classification. In *Twenty-ninth AAAI conference on artificial intelligence*, 2015.

[Lea *et al.*, 2017] Colin Lea, Michael D Flynn, Rene Vidal, Austin Reiter, and Gregory D Hager. Temporal convolutional networks for action segmentation and detection. In *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 156–165, 2017.

[LeCun *et al.*, 1998] Yann LeCun, Léon Bottou, Yoshua Bengio, Patrick Haffner, et al. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[Lillywhite *et al.*, 2013] Kirt Lillywhite, Dah-Jye Lee, Beau Tippetts, and James Archibald. A feature construction method for general object recognition. *Pattern Recognition*, 46(12):3300–3314, 2013.

[Liu and Motoda, 1998] Huan Liu and Hiroshi Motoda. *Feature extraction, construction and selection: A data mining perspective*, volume 453. Springer Science & Business Media, 1998.

[Motoda and Liu, 2002] Hiroshi Motoda and Huan Liu. Feature selection, extraction and construction. *Communication of IICM (Institute of Information and Computing Machinery, Taiwan) Vol*, 5(67-72):2, 2002.

[Romero *et al.*, 2004] Cristóbal Romero, Sebastián Ventura, and Paul De Bra. Knowledge discovery with genetic programming for providing feedback to courseware authors. *User Modeling and User-Adapted Interaction*, 14(5):425–464, 2004.

[Shan *et al.*, 2017] Ke Shan, Junqi Guo, Wenwan You, Di Lu, and Rongfang Bie. Automatic facial expression recognition based on a deep convolutional-neural-network structure. In *2017 IEEE 15th International Conference on Software Engineering Research, Management and Applications (SERA)*, pages 123–128. IEEE, 2017.

[Thomas and Sycara, 1999] James D Thomas and Katia Sycara. The importance of simplicity and validation in genetic programming for data mining in financial data. In *Proceedings of the joint AAAI-1999 and GECCO-1999 Workshop on Data Mining with Evolutionary Algorithms*, 1999.

[Vaswani *et al.*, 2017] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

[Zhong *et al.*, 2016] Yang Zhong, Josephine Sullivan, and Haibo Li. Face attribute prediction using off-the-shelf cnn features. In *2016 International Conference on Biometrics (ICB)*, pages 1–7. IEEE, 2016.