University of Nottingham
UK | CHINA | MALAYSIA

Machine Learning COMP3009 Coursework 1 – ANN

Runzhuo Li, Xixuan Wang, Zike Li, Yu Zhang, Pinyuan Feng
Group YOLO, School of Computer Science, University of Nottingham
{psyrl6, scyxw3, scyzl2, scyyz4, scypf1}@nottingham.ac.uk

Abstract

This is the first coursework of Machine Learning COMP3009 on the topic of Artificial Neural Networks (ANN). We applied standard ANN methodologies on Beijing PM2.5 dataset[2] and Iris dataset[1] to solve regression and classification problems respectively. To investigate how different training settings affect the model performance, we adopted "control variables" methodology in our experiments. Then, based on the previous experiments, we chose a set of appropriate hyperparameter combinations and used cross-validation to obtain the most optimal solution. For the evaluation part, we utilised 10-fold cross validation to evaluate our model to calculate average Root Mean Square Error (RMSE) for the regression problem and average accuracy for the classification problem. In this report, dataset information is briefly described in the first section, followed by an illustration about data pre-processing. Besides, the details of our experiments, including experiment configurations, training process and evaluation, are demonstrated as well. Next, several solutions to avoid overfitting are further discussed below. After that, a summary of our works is presented.

Dataset Description

Regression

The Beijing PM2.5 dataset contains the PM2.5 data of the US Embassy in Beijing, as well as meteorological data from Beijing Capital International Airport. There are 43824 instances with 13 attributes in the raw dataset.

Attribute	Explanation	Attribute	Explanation
No.	row number	DEWP.	Dew Point (â„ƒ)
year.	year of data in this row	TEMP	Temperature (â„ƒ)
month.	month of data in this row	PRES.	Pressure (hPa)
day.	day of data in this row	cbwd.	Combined wind direction
hour.	hour of data in this row	Iws.	Cumulated wind speed (m/s)
pm2.5.	PM2.5 concentration (ug/m^3)	Is.	Cumulated hours of snow
.		Ir.	Cumulated hours of rain

Figure 1

Classification

Iris dataset contains 3 different types of Iris flowers (Iris Setosa, Iris Versicolour, Iris Virgnica) with 50 plants for each. And 4 attributes are collected to predict the iris category.

Attribute	Explanation
Class	Class of flower (Setosa, Versicolour, Virginica)
sepallength.	sepal length in cm
sepalwidth.	sepal width in cm
petallength.	petal length in cm
petalwidth.	petal width in cm

Figure 2

Data Pre-processing

In general, we divided the data pre-processing stage into 4 parts, which includes feature selection, data cleaning, data split and data normalisation.

Feature Selection & Data Cleaning

Feature selection was typically applied on Beijing PM2.5 dataset, all columns related to time were removed because it is beyond the scope to apply time attribute in a simple ANN (usually time series are considered in LSTM, RNN, etc.). Additionally, values of the attribute "cbwd"(wind directions) are text-based, so we split the attribute "cbwd" into 4 attributes(one-hot encoding), where each wind direction represents one attribute. Data cleaning is solely applied on Beijing PM2.5 dataset as well, because the PM2.5 values in the first 24 rows are NA (the target 'y' is missing), so we discarded those data. In the end, we obtained a feature matrix of 43800 * 10.

For the iris dataset, we did not go through the first two data-preprocessing steps, because the data generally satisfied our demands.

Data Split & Data Normalisation

We split the data into training set, validation set, and testing set in proportions of 80%, 10% and 10%, respectively. Then, we applied Zero-Score normalisation methods on the raw data to ensure different variables to be in the same scope and avoid unexpected biases.

Specifically, we obtained the mean and standard deviation of training dataset and applied those properties on validation set and testing set, since there is no way for us to know the unseen data in the future. The equations are as follows:

$$X_{train} = \frac{X_{train} - E(X_{train})}{\sigma_{X_{train}}} \tag{1}$$

$$X_{validation} = \frac{X_{validation} - E(X_{train})}{\sigma_{X_{train}}} \tag{2}$$

$$X_{test} = \frac{X_{test} - E(X_{train})}{\sigma_{X_{train}}} \tag{3}$$

In the equations above, $E(X_{train})$ is a mean vector of feature values and $\sigma_{X_{train}}$ is a standard deviation vector of feature values in training dataset.

Besides, we found that PM2.5 values in the original dataset has a high variability, so we adjusted the values according to $log_2(y+1)$, where y represents a vector of PM 2.5 values, and we added one to avoid negative outcomes.

Training Configuration Selection

We have adopted "control variables" methodologies to explore the relationship between model performance and each variable (hyperparameter). Specifically, we selected learning rate, number of epochs, loss functions, optimizers, neural network topology, etc. To ensure the reliability of our results, we iterated the training process for 50 times to obtain an average result for each "control variable" experiment. Explanations of different training configurations, as well as experiment results, are displayed in ***Appendix A*** and ***Appendix B***, respectively.

The above experiments helped us narrow the range of possible solutions and facilitate hyperparameter selection for 10-cross validation. Next, we identified appropriate hyperparameter values and discarded hyperparameter values resulting in extremely undesirable performance.

In the previous experiments, we just investigated how model performance was affected by different training configurations and obtained the single optimal hyperparameter in each "control variable" experiments. However, it did not necessarily mean the combination of them was the best. Therefore, we tested different possible combinations obtained from previous experiments and gained an optimal combination based on cross validation. The hyperparameters we have tested are as follows:

Regression (48 combinations)

- Learning rate: 0.001, 0.005
- Batch size: 512, 1024
- Loss function: MSE, MAE
- Optimizer: RMSProp, Adam
- Topology:

	Hidden Layer 1	Hidden Layer 2	Hidden Layer 3
Multi-layer Perceptron 1	20	-	-
Multi-layer Perceptron 2	20	10	-
Multi-layer Perceptron 3	20	10	5

Table 1

Classification (54 combinations)

- Learning rate: 1, 0.5, 0.1
- Loss function: softmax cross entropy, sigmoid cross entropy
- Optimizer: GradientDescent, RMSProp, Momentum
- Topology:

	Hidden Layer 1	Hidden Layer 2	Hidden Layer 3
Multi-layer Perceptron 1	12	-	-
Multi-layer Perceptron 2	12	8	-
Multi-layer Perceptron 3	12	8	4

Table 2

Evaluation (10-fold cross validation)

The entire data set was divided into Set A and Set B with ratio of 9:1. Set A is used for 10-fold cross-validation to select the optimal hyperparameters, and Set B is used for testing the optimal solutions found by 10-fold cross-validation. Generally, we integrated hyperparameter selection and model evaluation in the same code file. Each hyperparameter combination were evaluated by 10-fold cross validation. Based on the average results, the combination resulting in best performance were used to predict on Set B. The optimal combinations for regression and classification problem are presented below, as well as the 10-fold cross-validation results.

The optimal hyperparameter combination of regression:

- Learning rate: 0. 001
- Batch size: 512
- Number of epochs: 10000
- Loss function: MSE
- Optimizer: RMSProp Optimizer
- Topology:

	Hidden Layer 1	Hidden Layer 2	Hidden Layer 3
Multi-layer Perceptron 3	20	10	-

Table 2

The optimal hyperparameter combination of classification:

- Learning rate: 1
- Number of epochs: 100
- Loss function: softmax cross entropy
- Optimizer: GradientDescent Optimizer
- Topology:

	Hidden Layer 1	Hidden Layer 2	Hidden Layer 3
Multi-layer Perceptron 1	12	-	-

Table 3

Evaluation Result

Regression			Classification	
Fold	RMSE (7 s.f.)		Fold	Accuracy (3 s.f.)
1	0.7382686		1	100.000%
2	0.7353457		2	100.000%
3	0.7331458		3	91.667%
4	0.7570033		4	91.667%
5	0.7380737		5	91.667%
6	0.7373531		6	83.333%
7	0.7322881		7	100.000%
8	0.7216459		8	100.000%
9	0.7324497		9	91.667%
10	0.7361474		10	100.000%
Average	0.7378545		Average	95.000%

Table 4

Model generalization and Solutions to avoid over-fitting

The methods to avoid overfitting can improve the model performance on generalisation ability. To ensure generalisation, we chose to restrict the complexity of network topology for both of the problem. Specifically, we decreased the number of neurons in each layer, as well as the depth of hidden layers.

Besides, we added dropout layers between different hidden layers and initialised value of keep_prob to 1. A smaller value of keep_prob will be set if overfitting occurs in the future.

Additionally, we set the batch size when feeding the data into ANN for regression problem. Because it is a large dataset, setting an appropriate batch size can result in a convergence to a more stable model.

Summary

Result Analysis:

Generally, we successfully solve the classification problem on Iris dataset, where we achieved a high prediction accuracy of 93.34%. The graph indicates our model converges well.

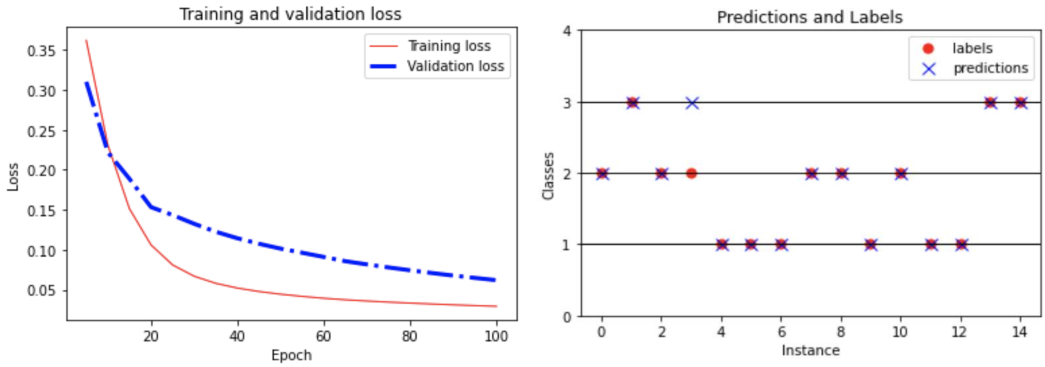


Figure 3

However, we did not get perfect results for regression task. According to graph, the data points does not fit the line $y = x$ well. The main reason is that the data we used for regression are time-series data (commonly used to evaluate LSTM), which makes a simple ANN difficult to handle.

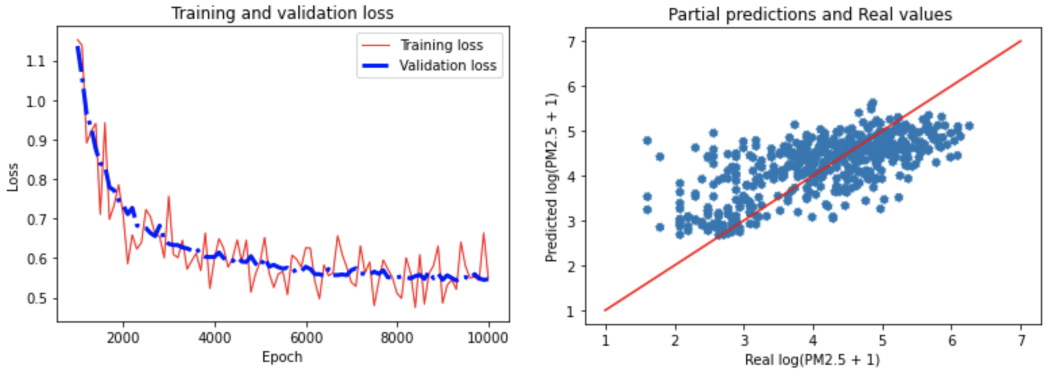


Figure 4

Challenges:

Firstly, testing different solutions required much time. Apparently, in the "control variable" experiment, it was not enough to test each training configuration for only 1 one time. To decrease the influence of some outliers, we have repeated each experiment for 50 times.

Secondly, we have less computational power, especially for large dataset. To some degree, this problem was alleviated by Google Colab, but the computational resources was still limited.

Another difficulty was that the networks for both tasks were failed to converge quickly in the begining. We found that it was caused by the features with different scales. So, we used Z-Score normalisation method discussed above to adjust all data to the same scale metric.

Reference

[1] Lichman, M. (2013). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.

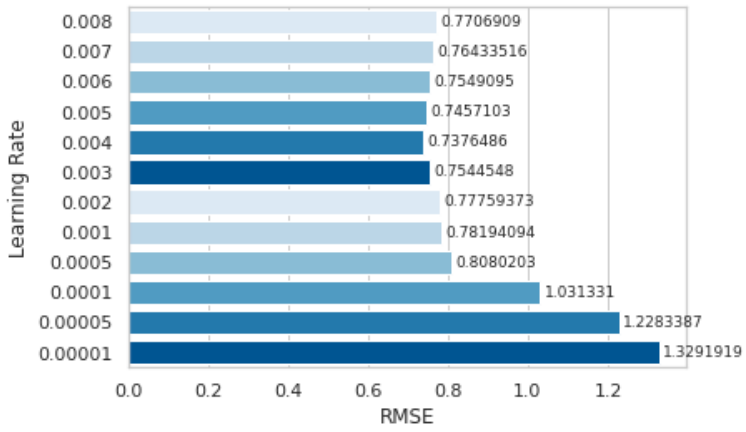
[2] Liang, X., Zou, T., Guo, B., Li, S., Zhang, H., Zhang, S., Huang, H. and Chen, S. X. (2015). Assessing Beijing's PM2.5 pollution: severity, weather impact, APEC and winter heating. Proceedings of the Royal Society A, 471, 20150257.

University of Nottingham
UK | CHINA | MALAYSIA

Appendix A : Regression results based on Beijing PM2.5 dataset

Influence of different parameters and ANN configurations

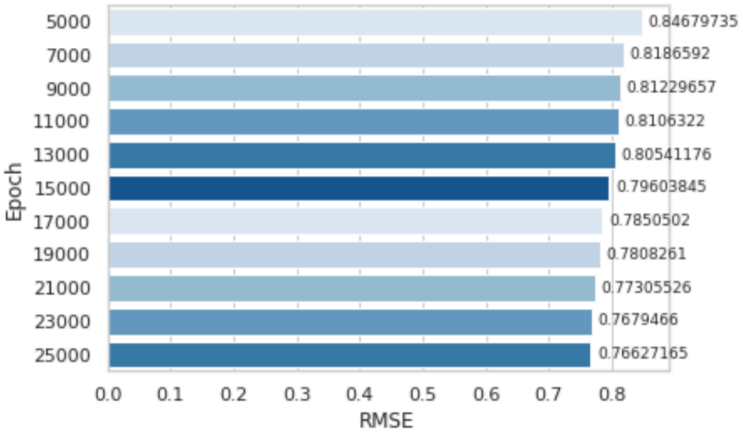
- Learning rate



The figure represents how root mean square error (RMSE) changes with different learning rates. According to the figure, RMSE decreases with the increase of learning rate in the beginning and then reaches minimum at the learning rate of 0.004, indicating that the model gradually converges with smaller moving steps. After that, RMSE gives a rise as the learning rate grows, which means that a model with a relatively lower learning rate takes more time to find the optimum within the same training iterations.

Figure 5

- Number of epochs



The figure indicates the trend of RMSE with the rise of epochs. Based on the chart, it can be obtained that RMSE goes down in the first place and then remains stable with higher iterations. This is because the model gradually converges given sufficient time period.

Figure 6

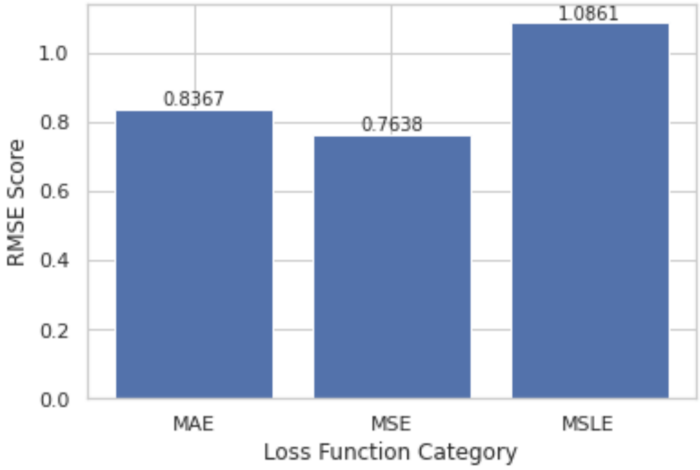
- ANN Topology

n_hidden1	n_hidden2	n_hidden3	RMSE
5	-	-	0.7678726
10	-	-	0.7724418
20	-	-	0.7680411
5	5	-	0.8329550
10	5	-	0.7769245
10	10	-	0.7775372
10	20	-	0.7858247
20	5	-	0.7994544
20	10	-	0.7753448
20	20	-	0.7602996
5	5	5	0.7815911
10	10	10	1.0152752
20	20	20	NAN

The figure on the left shows the relationship between neural network topology and RMSE based on Beijing PM2.5 dataset. Generally, the performance of complicated network architecture has higher performance. However, it is not always the case. Too complicated topology may result in overfitting.

Figure 7

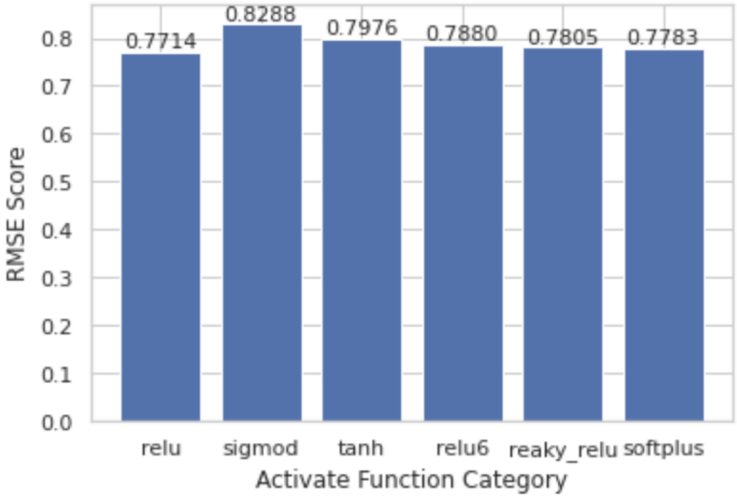
- Loss function



The chart below shows their influences on training results. From the comparison between different loss functions based on the chart, we could see that MSE leads to the highest performance on prediction of Beijing PM2.5 dataset.

Figure 8

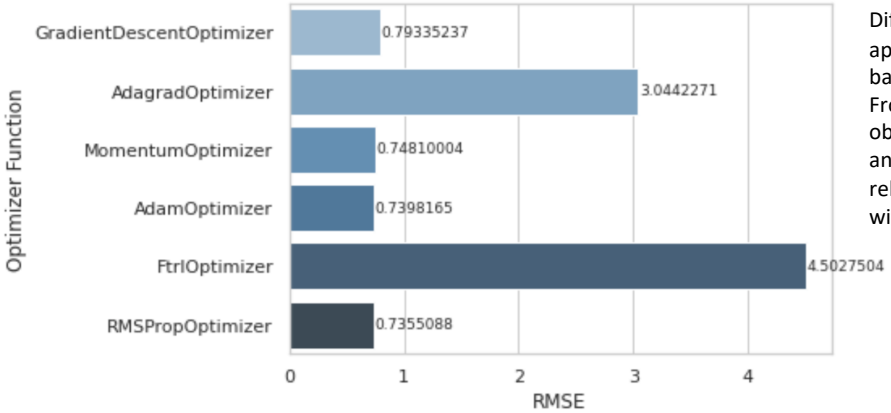
- Activate function



Six activate functions are selected to seek their influences on model performance. Based on the graph, ReLU appears to achieve the lowest RMSE, but sigmoid obtains the highest RMSE.

Figure 9

- Optimizer



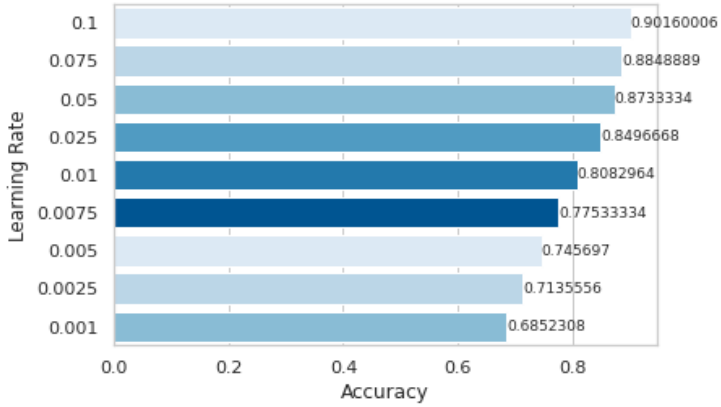
Different learning rules are applied to make comparisons based on the prediction results. From the figure below, it can be obtained that Adam optimizer and RMSprop optimizer achieve a relatively lower RMSE compared with other optimizers.

Figure 10

Appendix B : Classification results based on Iris dataset

Influence of different parameters and ANN configurations

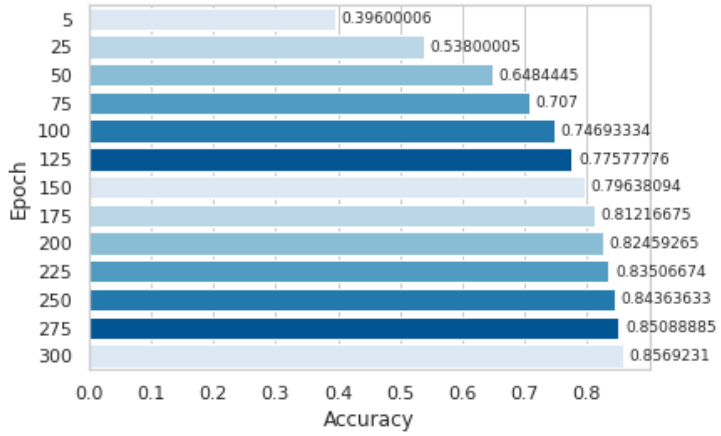
- Learning rate



The chart shows the change in accuracy with different learning rates. As can be seen, accuracy decreases with smaller learning rate and 0.1 appears to be an optimal learning rate for iris dataset. With the decrease of learning rate, the prediction accuracy becomes lower due to relatively slower convergence.

Figure 11

- Number of epochs



Iris dataset is a small dataset, in general, 100 epochs is enough. From the graph, the accuracy increases reapidly as the number of epochs increases in beginning. Then, the trend becomes relatively stable after the number of epochs is more than 100, which indicates that the model gradually converges with larger epochs.

Figure 12

- ANN Topology

n_hidden1	n_hidden2	n_hidden3	Accuracy
4	-	-	0.8466667
8	-	-	0.8800000
12	-	-	0.9000000
16	-	-	0.9786670
4	4	-	0.7413334
8	8	-	0.8760000
12	12	-	0.9066666
16	16	-	0.9133333
4	4	4	0.4613334
8	8	8	0.7893333
12	12	12	0.8906667
16	16	16	0.9493334

The figure on the left shows the relationship between neural network topology and RMSE based on Iris dataset. Because the original dataset has only 4 features. Complicated topology leads to low accuracy compared to accuracy of simple topology. For this dataset, one hidden layer is enough to achieve great results.

Figure 13

- Loss function

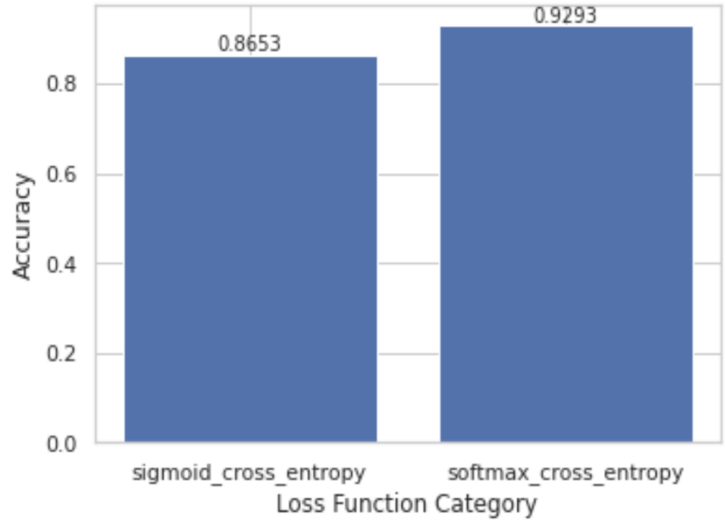


Figure 14

For the classification problem, two loss functions are applied, respectively. Based on comparison, softmax cross entropy turns out to be better than sigmoid cross entropy for iris dataset.

- Activate function

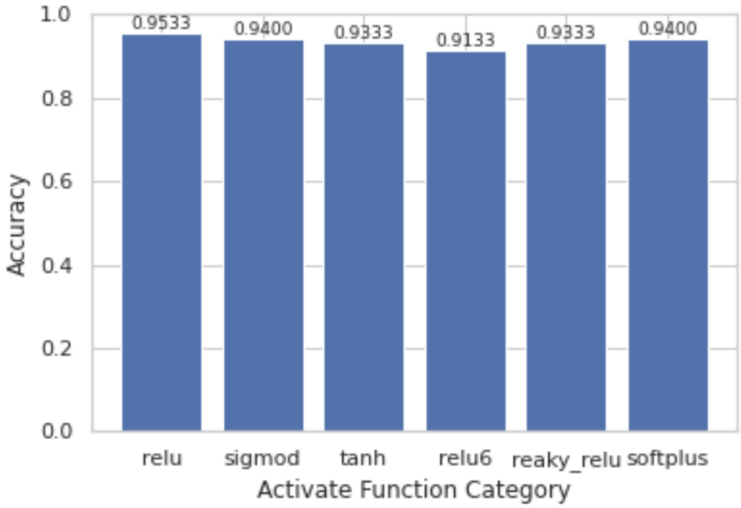


Figure 15

Six activate functions are chosen to compare their influences on model prediction. According to the figure, all activate functions achieve the accuracy of over 90%, among which ReLU appears to achieve the highest accuracy.

- Optimizer

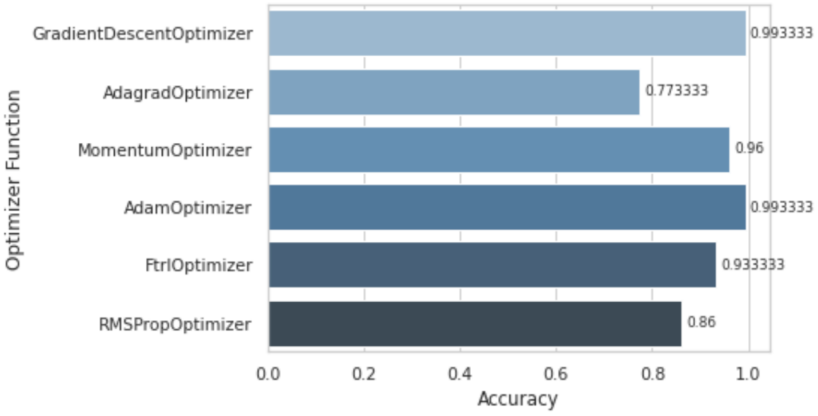


Figure 16

Six optimizers are chosen to find out the highest accuracy. It's obvious that the gradient descent optimizer and Adam optimizer perform best in our dataset, but the rest optimizers also achieve good results.