Proceedings of the 4th International Conference on Computing and Data Science (CONF-CDS 2022)
DOI: 10.54254/2755-2721/2/20220567

A Comparison on Machine Learning Classifiers related to Traffic Sign Recognition

Jiajin Zhang

University of Macau, Avenida da Universidade, Taipa, Macau, China

db92812@um.edu.mo

Abstract. With the rapid development of machine learning and the automotive industry, the industry of autonomous driving continues to grow. At the same time, governments have new regulations on autonomous driving, which tells us that reliable systems have become more and more critical while developing autonomous driving. In this paper, I use three different classifiers, which are Logistic Regression (LR), Random Forest (RF), and neural network (Multilayer Perceptron), to do the traffic sign recognition tasks and set the best parameters for every classifier. I train three classifiers with the best parameters and estimate using cross-value methods. Finally, I compared the performance, which indicates Random Forest Classifier has the best result among the three classifiers.

Keywords: machine learning, logistic regression, random forest.

1. Introduction

An unavoidable aspect of the automatic driving system is the recognition of traffic signs—road traffic signs and markings guide road users to use the road orderly to promote road driving safety. The mention of traffic signs in the driving assistance system can provide corresponding help for complete vehicle control without interruption. For example, the prohibition signs can help the system make advanced hazard predictions; warning signs can help the system make advanced obstacle avoidance treatment in some cases; instruction signs can help the system make control pre-processing to ensure that the traffic follows road instructions. Therefore, the correct recognition and accurate application of traffic signs can ideally assist the driver assistance system and even automatic driving. So, it is pivotal to discover the reliability of different Machine Learning Methods in automated driving systems.

In systems years, autonomous driving has been quietly changing our lives, and more and more companies are getting involved in the self-driving car industry. For example, Tesla Autopilot is an advanced assisted driving system developed by Tesla that allows cars to drive with lane centering and traffic-aware cruise control, automatic parking, automatic lane changing, and other functions. Meanwhile, Summon provides parking assistance, even if the driver is outside the car [1] [2]. But which classifier is the most suitable when designing a system remains a mystery. I, therefore, create this comparison to identify the reliability of different classifiers.

In this paper, my goal is to train the machine using three different classifiers and classify 15 traffic signs from a dataset containing 15540 images. After training, I want to make a comparison of them and use the result as a future reference. The classifiers used in this paper are Logistic Regression (LR),

© 2025 The Authors. This is an open access article distributed under the terms of the Creative Commons Attribution License 4.0 (https://creativecommons.org/licenses/by/4.0/).

DOI: 10.54254/2755-2721/2/20220567

Random Forest (RF), and neural network (Multilayer Perceptron). This test could be an excellent example of reliability studies when choosing an appropriate classifier.

2. Literature review

2.1. Classifiers

In machine learning, supervised learning and neural network play a pivotal role. Logistic Regression (LR) and Random Forest (RF), which are supervised learning, are classic solutions in many areas of data classifications. According to J.A. Vallejos and S.D. McKinnon, when operating under the optimal decision threshold values, Logistic Regression accuracy was greater than 95 percent for the classification of seismic records. Based on a survey of current re-entry practices at 18 seismically active mines, it was determined that blasting causes 90 percent of re-entry incidents. [3]. Kurt, Ture, and Kurum used logistic Regression to predict the presence of coronary artery disease (CAD). They built models using LR, CART, and neural network algorithms (RBF, MLP, and self-organizing feature maps (SOFM)), commonly used for classification problems. [4]. Qi demonstrated how to use Random Forest to classify different samples based on gene expression data from microarrays, identify disease-associated genes from genome-wide association studies, recognize essential elements in protein sequences, and identify protein-protein interactions (PPIs). [5]. According to Zaklouta, Stanciulescu, and Hamdoun, random forests achieve state-of-the-art performance in many multi-class classification applications. Another benefit is that they are quick to build, simple to implement in a distributed computing environment and allow for online learning. [6] Multiplayer Perceptron is another essential method used to solve the problem (MLP). Diaz-Alvarez et al. created a model to analyze human lane-change execution behavior using Multiplayer Perceptron and Convolutional Neural Networks by examining how a driver has the intention to change a lane. [7]. All the related works indicate that LR RF and MLP are reliable methods of classification work. All three classifiers will be trained for traffic sign recognition, which is critical in autonomous driving.

2.2. Traffic Sign Recognition (TSR)

De La Escalera, Moreno, Salichs, and Armingol introduced a multilayer perceptron-based classification for recognizing traffic signs in the 1990s. [8]. When lighting conditions cannot be controlled or predicted, objects can be partially obscured, and their position and orientation are unknown at the outset. Escalera, Armingol, and Mata proposed a new genetic algorithm for detection that allows invariance localization to change in status, scale, rotation, weather conditions, partial occlusion, and the presence of other objects of the same color. [9]. A famous approach to the TSR is using Convolutional Neural Networks (CNN). Vennelakanti et al. applied an image processing-based traffic sign detector from road scenes and an ensemble CNN for traffic sign recognition [10]. Zhou et al. demonstrated two lightweight and efficient CNN architectures for traffic sign classification that achieved 98.2 percent accuracy on the GTSRB dataset and 72.3 percent accuracy on the 101_food dataset. The experimental results show that this module can improve network performance and control model size effectively. [11]. In addition, Li, Li, and Zeng proposed a lightweight CNN architecture for traffic sign classification that could achieve 97.4 percent accuracy on the GTSRB dataset and 98.1 percent accuracy on the BTSD dataset. [12].

3. Methodology

In this work, I proposed three classifiers which are Logistic Regression, Random Forest, and Multiplayer Perceptron, to do the classification job. The whole data set includes 15540 images in ppm format divided into 15 classes based on their types. Each class contains 1500 images except for class 0, which only contains 100 images. At the same time, I resize the images to 32*32 to control the variable.

3.1. Cross-validation: evaluating estimator performance

Cross-validation is a standard method used in machine learning to build and validate model parameters.

DOI: 10.54254/2755-2721/2/20220567

First, we introduce the Leave-one-out cross-validation (LOOCV) method. The LOOCV method also includes dividing the dataset into training and test sets. But the difference is that we now only use one data as the test set and the other data as the training set and repeat this step N times (N is the amount of data in the dataset). Figure 1 indicates a sample process of LOOCV.

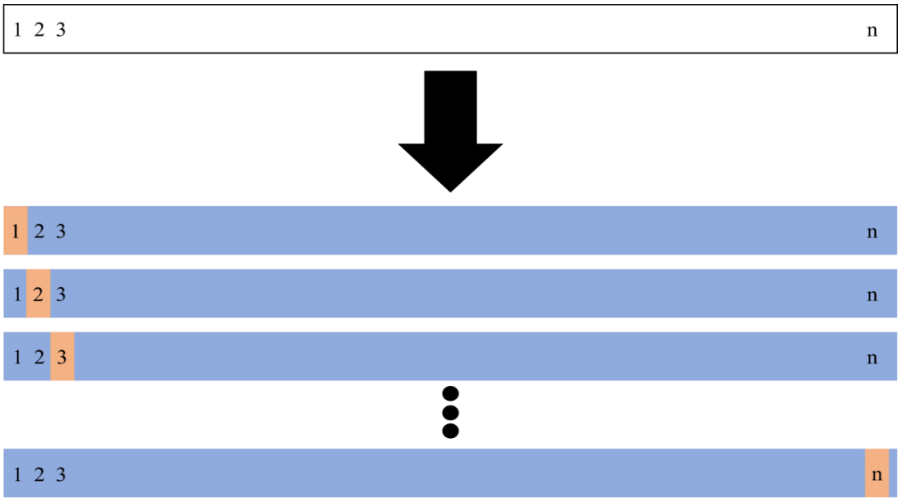

Figure 1. LOOCV.

As shown in Figure 1, if we now have a data set of n data points, the LOOCV method extracts one data point at a time as the only element of the test set. In contrast, the other n-1 data points are used as the training set for training the model and adjusting parameters. As a result, we train n models and get an MSE each time. The final test MSE is calculated by averaging these n MSEs.

$$CV_{(n)} = \frac{1}{n}\sum_{i=1}^{n} MSE_i \tag{1}$$

Compared to the test set approach, y_i, LOOCV has many advantages. First, it is not affected by dividing the training set of the test set because each data is done separately for the test set. At the same time, it uses n-1 data to train the model, and almost all information is used to ensure that the model's bias is more negligible. However, the disadvantage of LOOCV is also apparent; the amount of calculation is too large, which is n-1 times the time-consuming test set approach.

To solve the disadvantage of high computational cost, another formula is provided, which makes the computational cost of LOOCV as fast as training only one model.

$$CV_{(n)} = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{y_i - \hat{y}_i}{1 - h_i}\right)^2 \tag{2}$$

where $\hat{y}_i$ represents the ith fitted value, and h_i represents leverage.

Another compromise method is called K-fold cross-validation. The difference from LOOCV is that each test set will no longer contain only one data. Still, multiple ones and the specific number will be determined according to the selection of K. For example, if K=5, there are three steps to using five-fold cross-validation:

i. Divide all datasets into five parts randomly.

ii. Take one of them as the test set without repetition, use the other four as the training set to train the model, and then calculate theMSE_i of the model on the test set.

iii. Science LOOCV is a special K-fold Cross Validation (K=N), I can average the 5 $MSE_i$to get the final MSE.

DOI: 10.54254/2755-2721/2/20220567

$$Cv_{(k)} = \frac{1}{k}\sum_{i=1}^{k} MSE_i \tag{3}$$

3.2. Introduction to Logistic Regression, Random Forest, and Multilayer Perceptron Classifiers

3.2.1. Random Forest. Bagging (Bootstrap AGGregatING) is a well-known representative of parallelized integrated learning methods. It is directly based on bootstrap sampling, i.e., given a dataset containing m samples, we first randomly take out a sample and put it into the sampling set, and then put the sample back into the initial dataset so that the sample is still likely to be selected in the subsequent sampling. After m such random sampling operations, we can obtain a sample set with m samples, and we know that some samples in the initial training set will appear several times in the sample set, and some will never appear. [13]. It is based entirely on bootstrap sampling. Given an m-sample data set, we take a sample at random and place it in the sampling set, then return the model to the initial data set to be chosen in the following sampling. We acquire a sampling set of m samples after m rounds of random selection.

Random Forest [14] (RF for short) is an extended variant of Bagging.

RF is a Bagging integration built with a decision tree as the base learner. However, the difference is that RF's base learner decision tree introduces random attribute selection.

For a traditional decision tree, for each division of its nodes, we select an optimal attribute for division from the set of attributes of the current node (assuming there are d attributes); whereas in RF, for each node of the decision tree, we first randomly select a subset containing k attributes from the d attributes contained in the node, and then select an optimal attribute for division from this subset with the parameter k controls the degree of randomness introduced: if k=d, the base decision tree is equivalent to the traditional decision tree; if k=1, each node of the base decision tree is randomly selected an attribute for division.

In sklearn, parameters in the RandomForestClassifier are shown as follows.

```
RandomForestClassifier(
    n_estimators=10, criterion='gini',
    max_depth=None,min_samples_split=2,
    min_samples_leaf=1, min_weight_fraction_leaf=0.0,
    max_features='auto', max_leaf_nodes=None,
    min_impurity_decrease=0.0, min_impurity_split=None,
    bootstrap=True, oob_score=False, n_jobs=None,
    random_state=None, verbose=0, warm_start=False, class_weight=None)
```

3.2.2. Logistic Regression. Logistic distribution is a continuous probability distribution, and its distribution function and density function are:

$$F(x) = p(X \le x) = \frac{1}{1 + e^{-(x-\mu)/\gamma}} \tag{4}$$

$$f(x) = F'(X \le x) = \frac{e^{-(x-\mu)/\gamma}}{\gamma(1 + e^{-(x-\mu)/\gamma})^2} \tag{5}$$

Among them, μ represents the position parameter is the shape parameter.

A logistic distribution is a continuous distribution defined by its location and scale parameters. The shape of the logistic distribution is like that of the normal distribution, but the logistic distribution has longer tailed, so we can use the logistic distribution to model data distributions that have longer tails and higher peaks than the normal distribution. The sigmoid function commonly used in deep learning is a special form of the logistic distribution function when.

DOI: 10.54254/2755-2721/2/20220567

Logistic regression is mainly used for classification problems. We take binary classification as an example. For the given data set, it is assumed that there is such a straight line that the data can be linearly separable. The decision boundary can be expressed as, suppose, a sample point. Then it can be judged that its category is 1. Logistic regression also needs to add a layer. He needs to find the direct relationship between the classification probability P(Y=1) and the input vector x and then judge the category by comparing the probability values.

Consider a binary classification problem, given a dataset $D=x1, y1, x2, y2, \ldots, xn, yn, xi \subseteq Rn, yi \in 0,1, i=1, 2 \cdots n$, considering $\omega^T x + b$ is continuous, therefore it cannot fit discrete variables. Consider using it to fit conditional probabilities $P(Y = 1|x)$, because the probability values are also continuous.

But for $\omega \neq 0$ (if it is equal to the zero vector, there is no value to solve), $\omega^T x + b$ takes the value of R, and the non-conformity probability takes the value from 0 to 1, so consider using a generalized linear model.

The most ideal is the Heaviside step function

$$P(y = 1|x) = \begin{cases} 0, & z < 0 \\ 0.5, & z = 0, \\ 1, & z > 0 \end{cases} \qquad z = \omega^T x + b \tag{6}$$

But this step function is not differentiable, and the log probability function is a commonly used surrogate function:

$$y = \frac{1}{1 + \mathrm{e}^{-(\omega^T + b)}} \tag{7}$$

So, there are:

$$ln \frac{y}{1 - y} = \omega^T x + b \tag{8}$$

We consider y to be the probability that x is positive, then $1 - y$ is the probability that x is negative. The ratio of the two is called the odds, which refers to the ratio of the probability of the event occurring and not occurring if the probability of the event occurring is p. Then the log odds are:

$$ln(odds) = ln \frac{y}{1 - y} \tag{9}$$

Considering y as a class posterior probability estimate, the rewritten formula is:

$$\omega^T x + b = ln \frac{P(Y = 1|X)}{1 - P(y = 1|X)} \tag{10}$$

$$P(Y = 1|X) = \frac{1}{1 + \mathrm{e}^{-(\omega^T X + b)}} \tag{11}$$

That is, the log odds of the output is the model represented by a linear function of the input, which is the logistic regression model. The closer the value of is to positive infinity, the closer the probability value is to 1. Therefore, the idea of logistic regression is to first fit the decision boundary (not limited to linear but also polynomial), and then establish the probability connection between this boundary and the classification, to obtain the probability in the case of binary classification.

Regularization is a general algorithm and idea, so any algorithm that produces overfitting can use regularization to avoid overfitting.

Based on minimizing the empirical risk (that is, minimizing the training error), using a simple model as much as possible can effectively improve the generalization prediction accuracy. If the model is too complex and the variable values vary slightly, it will cause problems with prediction accuracy. Regularization works because it reduces the weight of features, making the model simpler.

DOI: 10.54254/2755-2721/2/20220567

3.2.3. Multilayer Perceptron. In the current field of machine learning, neural networks are widely used. Neural networks, for example, can be used for image recognition [15], speech recognition [16], and other applications that can be extended to self-driving cars. It is a highly parallel information processing system with a high self-adaptive learning capability. It is unrelated to the mathematical model of the research object. It is resistant to changes in the system parameters of the controlled object and external interference. Classification is the basic problem that neural networks must solve when dealing with complex multi-input and multi-output nonlinear systems.

Understanding the neural network mainly includes two main contents, one is the structure of the neural network, and the other is the training and learning of the neural network, which is like how our brain structure is composed, and how do we learn and identify based on this composition of different things.

A neural network is a simulation and simplification of biological neurons composed of dendrites, cell bodies, and axons. The dendrite is the input end of the cell body, which receives the surrounding nerve impulses; the axon is the output end of the cell body, which plays the role of transmitting nerve impulses to other neurons. Biological neurons have two states of excitation and inhibition. When the stimulus is higher than a certain threshold, it will enter the excited state and transmit the nerve impulse from the axon. Otherwise, there will be no nerve impulses.

Based on the biological neuron model, we can obtain the basic structure of the multi-layer perceptron MLP. The most common MLP has three layers: an input layer, a hidden layer, and an output layer. The MLP neural network's various layers are fully connected (fully connected). This means that any neuron in the previous layer is linked to all neurons in the next layer.

The neural network mainly has three basic elements: weights, biases, and activation functions.

Weight: The strength of the connection between neurons is represented by the weight, and the size of the weight represents the size of the possibility

Bias: Bias is set to correctly classify samples and is an important parameter in the model, that is, to ensure that the output value calculated from the input cannot be activated casually.

The activation function functions as a nonlinear mapping, limiting the output amplitude of neurons within a certain range, generally between $(-1\sim1)$ or $(0\sim1)$. The most used activation function is the sigmoid function, which maps numbers from $(-\infty,+\infty)$ to the range $(0\sim1)$.

4. Results

4.1. Random Forest

In Random Forest, I experimented with two parameters. The first is "criterion," which is used to assess the quality of a split. There are two options for "criterion," one of which is "geni," which stands for Gini impurity, and the other is "entropy." Another parameter I experimented with was n estimators, which represents the number of trees in the forest. According to Figure 2, the best parameters are criterion = 'gini' and n estimators = 210.

DOI: 10.54254/2755-2721/2/20220567

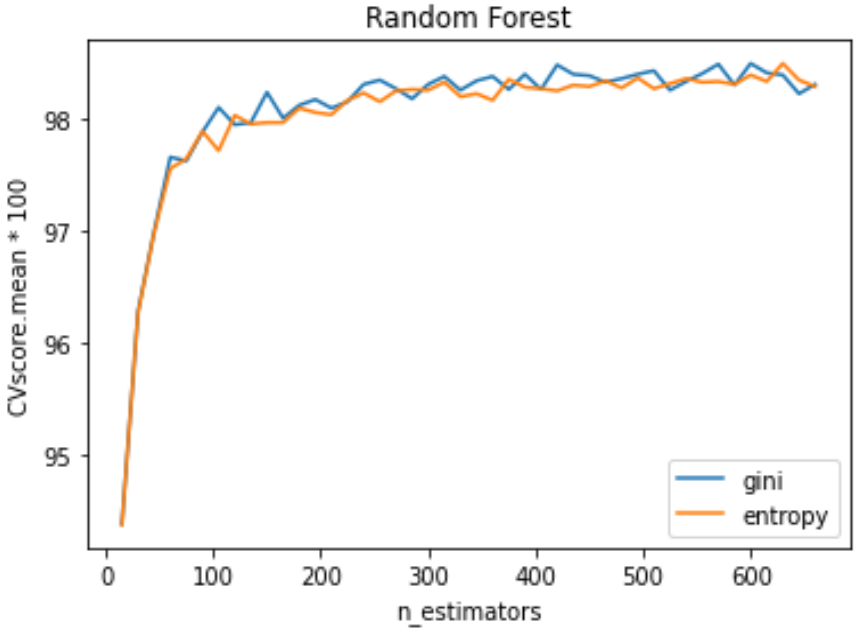

Figure 2. -RF parameters setting.

4.2. Logistic Regression

Regularization generally adopts L1 norm or L2 norm. But here I decided not to use regularization or penalty. Max_iter, which sets the maximum number of iterations taken for the solvers to converge, is pivotal to the result. The last parameter is solver, which is an algorithm to use in the optimization problem. From my testing result from as Figure 3 shows, I finally set the parameters as max_iter = 14, penalty = 'none' and solver = 'newton-cg'.

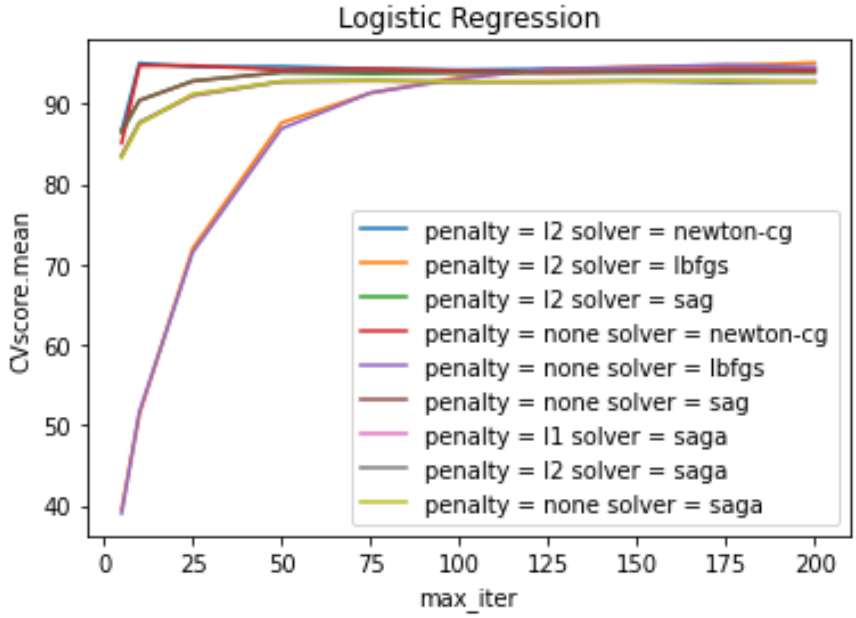

Figure 3. -LR parameters setting.

DOI: 10.54254/2755-2721/2/20220567

4.3. Multilayer Perceptron

The MLP classifier has a plethora of parameters. We run tests on the five most important parameters we can think of: solver, alpha, learning rate init, hidden layer sizes, and batch size. The sizes of the hidden layers show that the ith element represents the number of neurons in the ith hidden layer. Weight optimization is used differently by different solvers. The penalty for the second layer is alpha. The batch size of stochastic optimizers is the size of mini-batches. The step-size in updating the weights is determined by the initial learning rate. Figure 4 depicts the accuracy using the "sgd" solver with various lr and alpha values, Figure 5 depicts the corresponding validation accuracy, and Figure 6 depicts the time cost. Figure 7, 8 and 9 present the accuracy, validation accuracy and time cost for “adam” slover with different lr and alpha values. Correspondingly, Figure 10, 11 and 12 are test result for “lbfgs” solvers with different lr and alpha value.

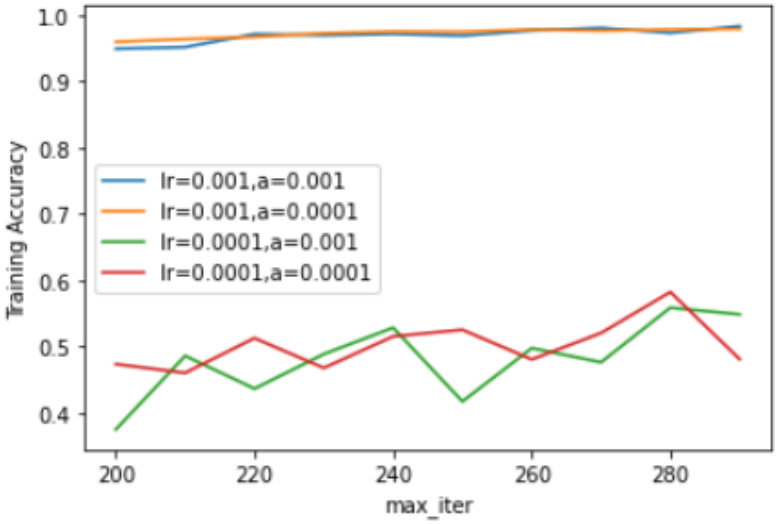

Figure 4. -MLP parameters setting with ‘sgd’ solver, different lr and alpha in training accuracy.

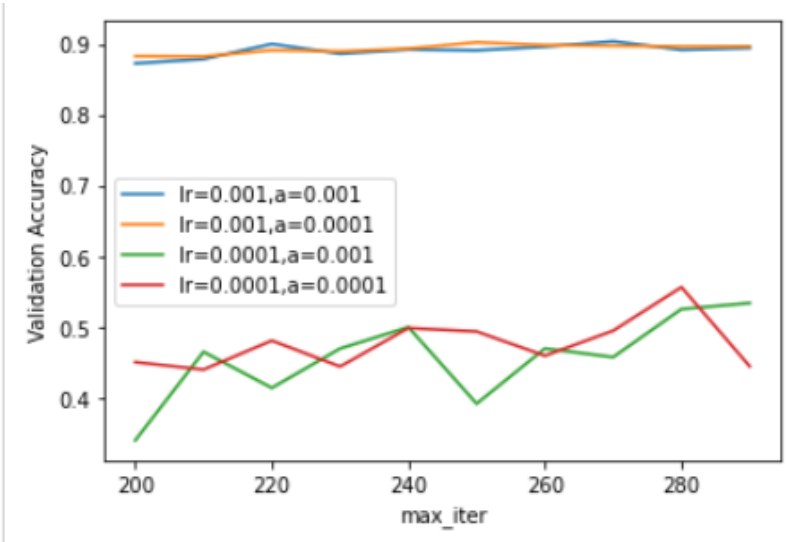

Figure 5. -MLP parameters setting with ‘sgd’ solver, different lr and alpha in validation accuracy.

DOI: 10.54254/2755-2721/2/20220567

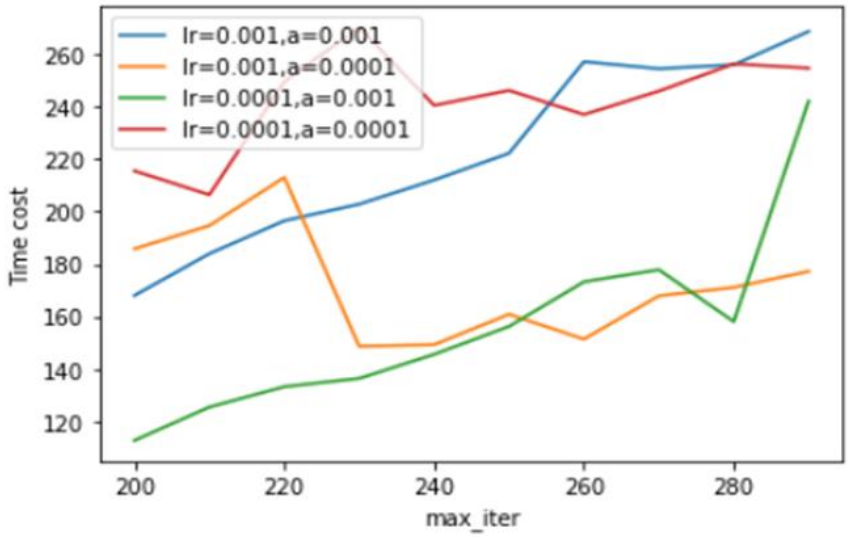

Figure 6. -MLP parameters setting with 'sgd' solver, different lr and alpha in time cost.

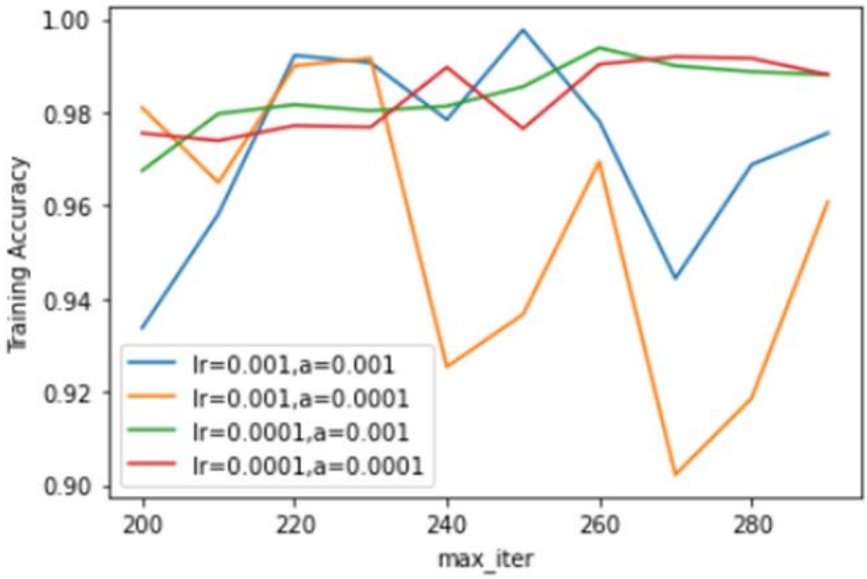

Figure 7. -MLP parameters setting with 'adam' solver, different lr and alpha in training accuracy.

DOI: 10.54254/2755-2721/2/20220567

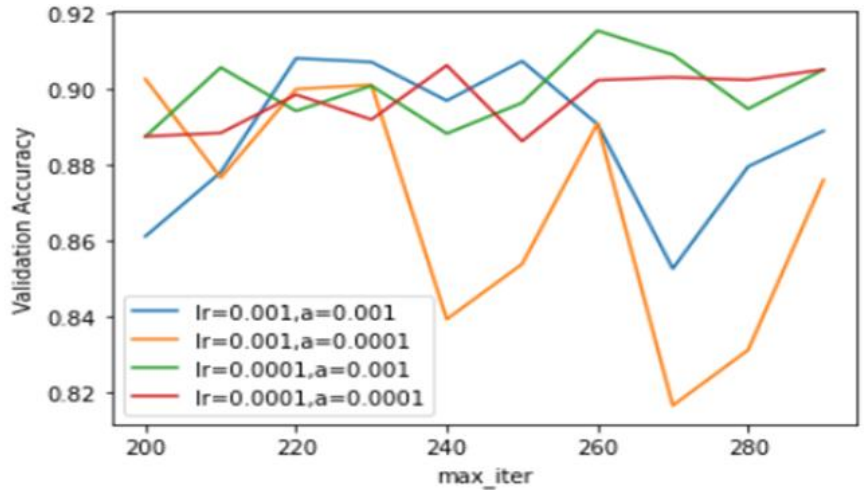

Figure 8. -MLP parameters setting with 'adam' solver, different lr and alpha in validation accuracy.

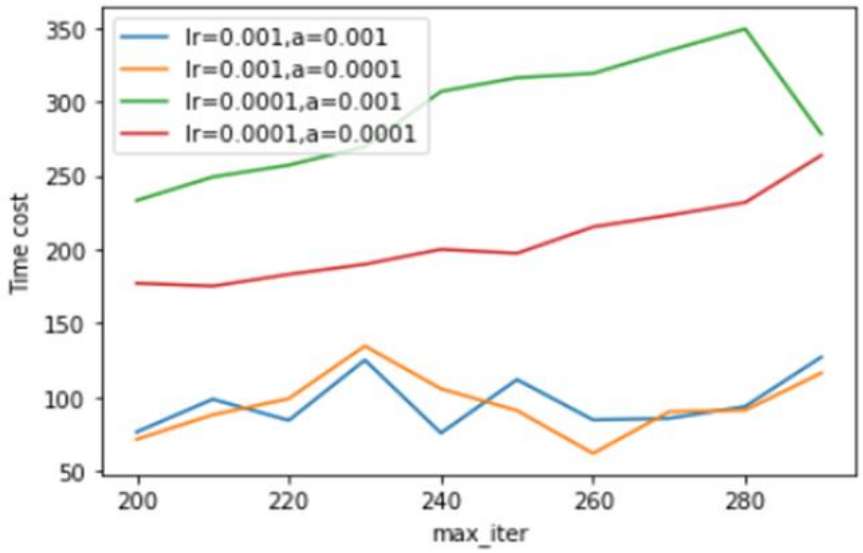

Figure 9. -MLP parameters setting with 'adam' solver, different lr and alpha in time cost.

DOI: 10.54254/2755-2721/2/20220567

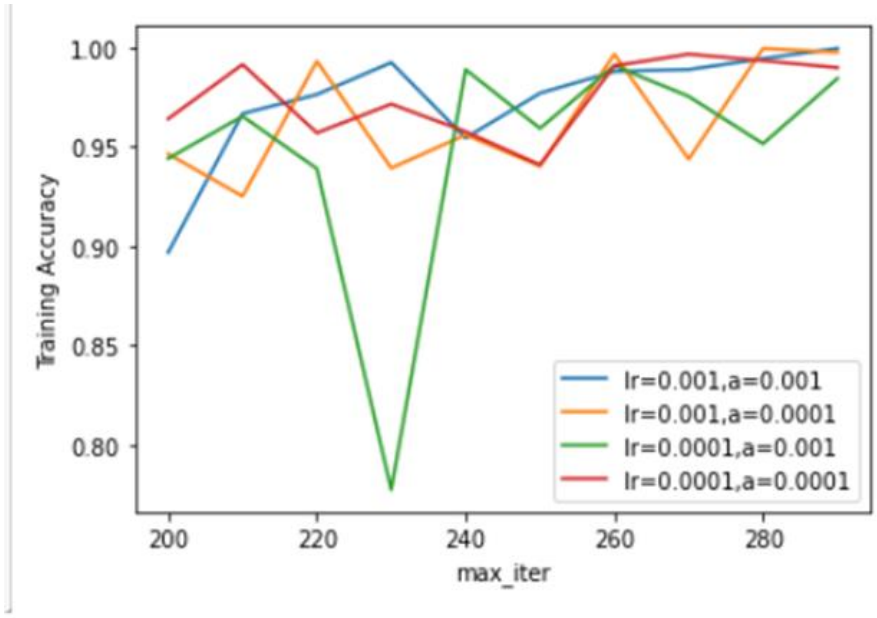

Figure 10. -MLP parameters setting with 'lbfgs' solver, different lr and alpha in training accuracy.

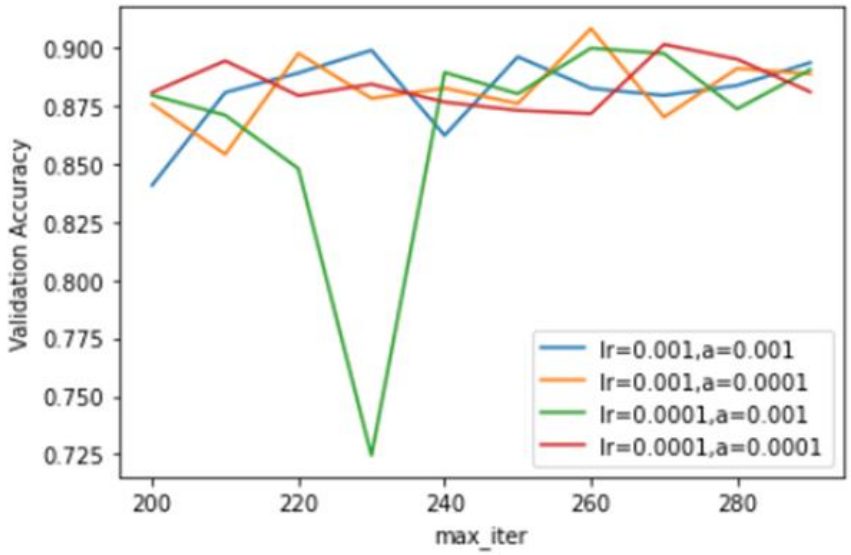

Figure 11. -MLP parameters setting with 'lbfgs' solver, different lr and alpha in validation accuracy.

DOI: 10.54254/2755-2721/2/20220567

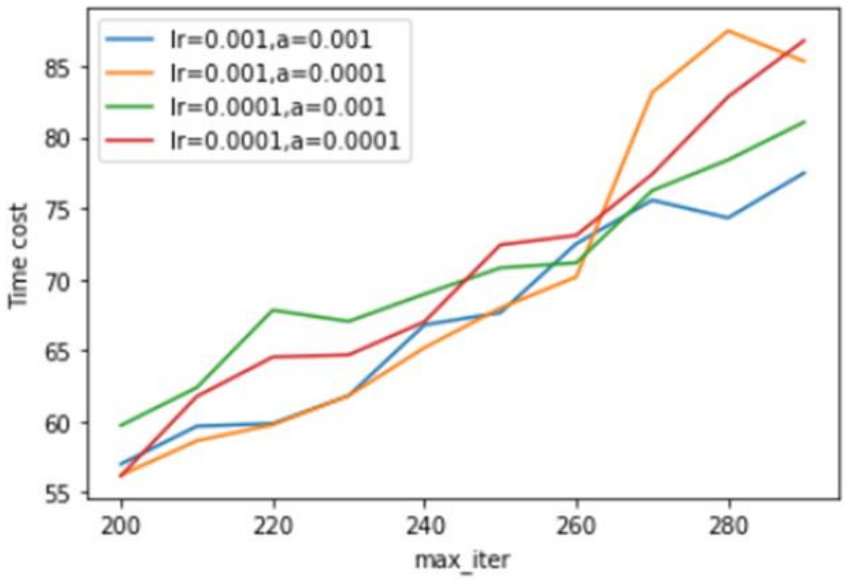

Figure 12. -MLP parameters setting with 'lbfgs' solver, different lr and alpha in time cost.

Table 1. -MLP parameters setting with hidden layer size and batch size.

hidden_layer_size	batch_size	max_iter	solver	alpha	learning_rate	accuracy
100,50	200	260	adam	0.001	0.0001	0.8819
100,50	500	260	adam	0.001	0.0001	0.8514
150,75	200	260	adam	0.001	0.0001	0.9066
300,150,75	200	260	adam	0.001	0.0001	0.9003
300,150,15	2000	260	adam	0.001	0.0001	0.7886
300,150,15	500	260	adam	0.001	0.0001	0.8732
300,150,15	50	260	adam	0.001	0.0001	0.88
300,75,15	50	260	adam	0.001	0.0001	0.9
300,75,15	30	260	adam	0.001	0.0001	0.904
300,75,15	500	260	adam	0.001	0.0001	0.8321
300,75,15	20	260	adam	0.001	0.0001	0.8923
300,150,15	30	260	adam	0.001	0.0001	0.9095
300,150,15	35	260	adam	0.001	0.0001	0.9047
300,150,15	40	260	adam	0.001	0.0001	0.8957
300,150,15	45	260	adam	0.001	0.0001	0.87
300,150,15	25	260	adam	0.001	0.0001	0.9028
75,150,15	25	260	adam	0.001	0.0001	0.9203
75,150,15	30	260	adam	0.001	0.0001	0.9375
150,300,15	30	260	adam	0.001	0.0001	0.927
50,100,15	30	260	adam	0.001	0.0001	0.7754
30,60,15	30	260	adam	0.001	0.0001	0.8123

DOI: 10.54254/2755-2721/2/20220567

To find the better parameters of solver, alpha, initial learning rate and max_iter, the machine split the dataset to 4/5 for training and 1/5 for testing. I set different values of these parameters and the output figure shows the result of the training accuracy, testing accuracy and time cost with different parameters. Compared the results from Table 1 with each other, I finally choose solver =’adam’, lr = 0.0001, alpha = 0.001 and max_iter = 260.

For batch size and hidden layer size, I also test lots of values with the whole dataset. After the experiments, I found (75,150,15) as the hidden layer (15 neutrons for the last layer because we have 15 different class images) and batch size equal to 30.

4.4. Performance of three classifiers

After setting the suitable parameters for each classifier, we used the whole dataset to train the machine and saved the models by using joblib. The performance of three classifiers is shown with Figure 13, 14 and 15. By using the cross-value training method, the mean score is 0.953 for LG, 0.979 for MLP, and 0.982 for RFC.

We have also analyzed the matching accuracy for different 15 classes in RF and LR. Both results indicate that class 0 has the worst accuracy. Test results can be shown with Figure 16, 17 and 18.

```
Logistic Regression: Trained  Training accuracy: 0.9981981981981982
Time cost: 50.68 seconds
Model saved in 'Trained_LR.model'.
Logistic Regression(penalty = 'none', solver = 'newton-cg', max_iter = 14)
[CVscore.mean = 0.9539253539253539] [CVscore.var = 7.188655837304496e-06]
```

Figure 13. -Training result for Logistic Regression Classifier.

```
Multilayer Perceptron (MLP) Classifier: Trained  Training accuracy: 0.9797940797940798
Time cost: 1844.96 seconds
```

Figure 14. -Training result for Multilayer Perceptron Classifier.

```
Random Forest Classifier: Trained  Training accuracy: 1.0
Time cost: 17.08 seconds
Model saved in 'Trained_RF.model'.
Random Forest(criterion = 'gini', n_estimators = 210)
[CVscore.mean = 0.9824324324324325] [CVscore.var = 1.5694127663239894e-06]
```

Figure 15. -Training result for Random Forest Classifier.

DOI: 10.54254/2755-2721/2/20220567

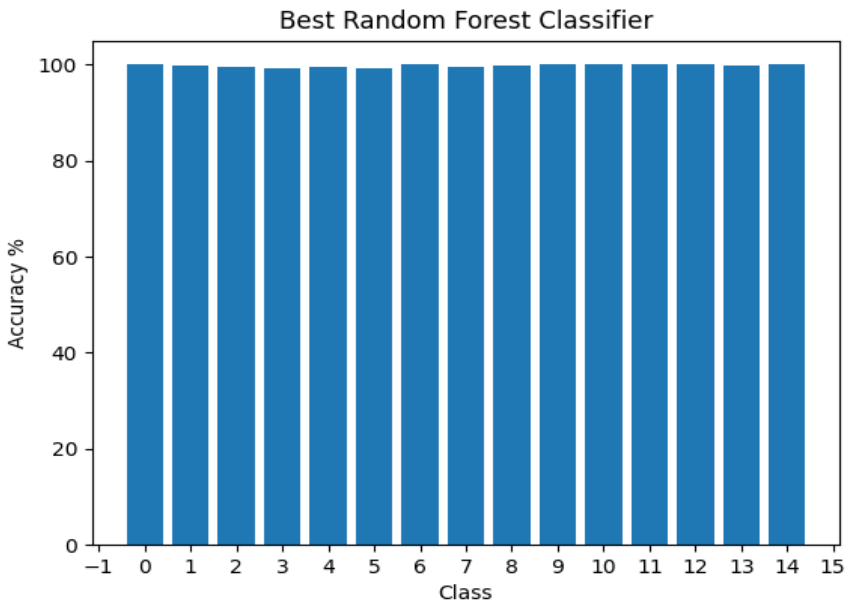

Figure 16. -Test result for Random Forest Classifier.



Figure 17. -Test result for Logistic Regression Classifier.

DOI: 10.54254/2755-2721/2/20220567

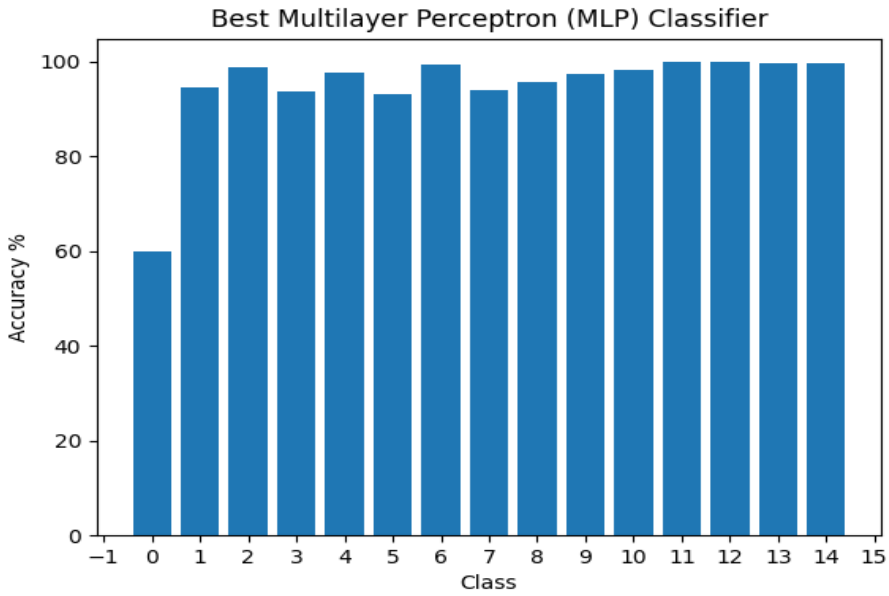

Figure 18. -Test result for Multilayer Perceptron Classifier.

5. Discussion

From the result, we can conclude that all three classifiers have a high score in the cross-value training method, which is higher than 0.95. Although the mean value of the cross-value training method is satisfactory, we still find Logistic Regression Classifier and Multilayer Perceptron Classifier do not have high accuracy when dealing with dataset No.0, which is confusing. But if we check the training set, we can find out easily that there are only 150 images in Class No.0, which is only one-tenth of the other data set. Therefore, we can conclude that a compared large data set could help the classifier improve the accuracy.

6. Conclusion

In this project, I designed three classifiers to classify the dataset, which is traffic sign images. The main purpose of the project is to establish a comparison reference for future researchers no matter in traffic sign recognition or in Machine Learning methods fields. I can quantify the image processing capability of the classifier using the cross-value method in RF and LR classifiers. As a result, I can find out Random Forest Tree performances best in the test section when comparing with the other two classifiers. Logistic Regression and Multilayer Perceptron do not have a good response for Class 0, which may lead to potential risk when real implemented.

References

[1] Dikmen, M., & Burns, C. M. (2016). Autonomous driving in the real world. Proceedings of the 8th International Conference on Automotive User Interfaces and Interactive Vehicular Applications. https://doi.org/10.1145/3003715.3005465

[2] Model S owners manual touchscreen 7.1 das ap north ... - tesla. (n.d.). Retrieved March 20, 2022, from https://www.tesla.com/sites/default/files/model_s_owners_manual_touchscreen_7.1_das_ap_north_america_r20160112_en_us.pdf

[3] Vallejos, J. A., & McKinnon, S. D. (2013). Logistic regression and neural network classification of Seismic Records. International Journal of Rock Mechanics and Mining Sciences, 62, 86–95. https://doi.org/10.1016/j.ijrmms.2013.04.005

[4] Kurt, I., Ture, M., & Kurum, A. T. (2008). Comparing performances of logistic regression, classification and regression tree, and neural networks for predicting coronary artery disease. Expert Systems with Applications, 34(1), 366–374. https://doi.org/10.1016/j.eswa.2006.09.004

[5] Qi, Y. (2012). Random Forest for Bioinformatics. Ensemble Machine Learning, 307–323. https://doi.org/10.1007/978-1-4419-9326-7_11

[6] Zaklouta, F., Stanciulescu, B., & Hamdoun, O. (2011). Traffic sign classification using K-D trees and random forests. The 2011 International Joint Conference on Neural Networks. https://doi.org/10.1109/ijcnn.2011.6033494

[7] Díaz-Álvarez, A., Clavijo, M., Jiménez, F., Talavera, E., & Serradilla, F. (2018). Modelling the human lane-change execution behaviour through multilayer perceptrons and Convolutional Neural Networks. Transportation Research Part F: Traffic Psychology and Behaviour, 56, 134–148. https://doi.org/10.1016/j.trf.2018.04.004

[8] de la Escalera, A., Moreno, L. E., Salichs, M. A., & Armingol, J. M. (1997). Road traffic sign detection and classification. IEEE Transactions on Industrial Electronics, 44(6), 848–859. https://doi.org/10.1109/41.649946

[9] de la Escalera, A., Armingol, J. M., & Mata, M. (2003). Traffic sign recognition and analysis for intelligent vehicles. Image and Vision Computing, 21(3), 247–258. https://doi.org/10.1016/s0262-8856(02)00156-7

[10] Vennelakanti, A., Shreya, S., Rajendran, R., Sarkar, D., Muddegowda, D., & Hanagal, P. (2019). Traffic sign detection and recognition using a CNN ensemble. 2019 IEEE International Conference on Consumer Electronics (ICCE). https://doi.org/10.1109/icce.2019.8662019

[11] Zhou, Y., Feng, Y., Zeng, S., & Pan, B. (2019). Design of lightweight convolutional neural network based on Dimensionality Reduction Module. IOP Conference Series: Materials Science and Engineering, 533(1), 012045. https://doi.org/10.1088/1757-899x/533/1/012045

[12] Li, W., Li, D., & Zeng, S. (2019). Traffic sign recognition with a small convolutional neural network. IOP Conference Series: Materials Science and Engineering, 688(4), 044034. https://doi.org/10.1088/1757-899x/688/4/044034

[13] Breiman, L. (1996). Bagging predictors. Machine Learning, 24(2), 123–140. https://doi.org/10.1007/bf00058655

[14] Tin Kam Ho, "Random decision forests," Proceedings of 3rd International Conference on Document Analysis and Recognition, 1995, pp. 278-282 vol.1, doi: 10.1109/ICDAR.1995.598994.

[15] Xie, S., Kirillov, A., Girshick, R., & He, K. (2019). Exploring randomly wired neural networks for image recognition. 2019 IEEE/CVF International Conference on Computer Vision (ICCV). https://doi.org/10.1109/iccv.2019.00137

[16] Abdel-Hamid, A. Mohamed, H. Jiang, L. Deng, G. Penn, and D. Yu, "Convolutional Neural Networks for Speech Recognition," in IEEE/ACM Transactions on Audio, Speech, and Language Processing, vol. 22, no. 10, pp. 1533-1545, Oct. 2014, doi: 10.1109/TASLP.2014.2339736.