A Study Of Optimizing The Multi-Class Classifier Model For Synthetic Dataset VehicleX

Yizhi Zhao

[1] Research School of Computer Science, Australian National University
[2] u6719761@anu.edu.au

Abstract. Feature classification problems are often done by using neural networks. In this study, we will build a shallow neural network (NN) for vehicle classification problems. In general, training a reliable classifier requires a lot of real world data to support. For privacy and data security considerations, it is now more inclined to use artificially synthesized data sets. Therefore, in this study we will use artificially synthesized data sets to train and evaluate the neural network (NN). In order to get better results, the genetic algorithms will be used to optimize the hyperparameter and the structure of the neural network (NN). And the Calculation of Input Feature Distinctiveness will be used to further adjust the input features, thereby increasing the performance and the efficiency of the entire model. The performance of the model is evaluated by using k-fold cross-validation which provides solid and robust results that can enhance the reliability of the model.

Keywords: Functional Measures. Neural Network. Features Classification. Vehicle Classification. Evolutionary Algorithm. Genetic Algorithm.

1 Introduction

1.1 Motivation

Vehicle classification is widely used in modern society, such as electronic toll collection of highways, traffic enforcement and so on. Nowadays, intelligent transportation systems are being developed all over the world. Therefore, fast and reliable vehicle screening is a very important requirement and urgently needs to be satisified. The neural network (NN) is an advanced machine learning method, very suitable for solving such classification problems. As we know, the performance of a NN model can be improved by learning a large amount of data. And also, the imbalance of the training dataset will affect the performance of the model. Hence, a large training set with wide coverage is necessary to improve the accuracy of the model prediction.
However, it is not easy to obtain this kind of dataset from real-world. They may contain various problems, such as data security and privacy involving legal issues, or the typo errors which can not be avoided during the entering data manually. Therefore, using an artificially synthesized dataset is a good choice to ensure the reliability of data labeling. In this study, VehicleX [1] (an artificially synthesized dataset) will be used to train and evaluate the model. The objectives of this study are

- Constructing a shallow baseline neural network model structure as a features classifier
- Optimizing the structure and hyperparameters of the model by using Genetic Algorithm
- Validating the feasibility of distinctiveness to get the input features similarity in order to determine which features can be removed without affecting the model results.

For evaluating the model, the K-fold cross validation will be used to show the solid and robust results of the model.

1.2 Dataset

This study uses the VehicleX [1] Dataset for evaluating and training the model (including the relevant algorithm). VehicleX is the largest synthetic dataset currently available for vehicle re-ID and vehicle classification problem. This dataset was developed on the unity 3d engine. The attribute distribution of the VehicleX dataset is very close to reality. The original image of the vehicle was passed through a feature extractor (a pretrained resnet50 model) to generate a 2048-dimensional feature vector. The feature vectors are then fed into the classifier to obtain the classification result. The dataset used in this study contains 45,438 feature vectors for training set, 15,142 feature vectors for testing set and 14,936 feature vectors for validation set. There are a total of 1362 different classes in the entire dataset, which is a balanced dataset.

2 Method

2.1 Network Architecture

Based on the information provided by the dataset, the feature vectors of each image were generated by resnet50 [2]. According to the structure of resnet50, the last layer is a fully connected layer with Softmax (classification network), which is equivalent to a linear classifier. Hence, We first considered implementing a simply fully connected layer with softmax function as a multi-class classifier. Since the output layer needs to classify 1632 classes, a simple fully-connected layer may not be suitable for the complexity required by this task. Hence, the structure of this model needs to be more complicated, for example adding hidden neurons or more layers to meet the requirements of this task.

2.2 Activation Function, Loss Function and Optimizer

Activation Function The objective of this study is to bulid a multi-class classifier for 2048-D feature vectors, therefore three activation functions ReLu, Dropout and softmax are selected.
Rectified linear unit (ReLU) [5] is an activation function that can be inhibited on one side. Its function is to change all negative values to 0 while positive values remain unchanged. So that this allows neurons to have sparse activation. Generally, there are not all features related to the target in the classifier. Hence, the model that sparaing from the ReLU layer can better mine the relevant features, to improve the fitness of the model.
In order to avoid the overfitting as much as possible, the dropout layer [6] is also very important. During the process of the data propagating forward, this layer will randomly let some certain neurons stop working with certain probabilities. This layer can make the model become more generalizing as it can avoid the model overly relying on certain local features.
Softmax 1 is very suitable for multi-class classification, because it calculates the probability distribution of n different events. Its output probability ranges from 0 to 1, and the sum of all probabilities will be equal to 1. If the softmax function is used in a multi-class classifier, it will return the probability of each category, and the probability value of the target category will be very large.We use Logsoftmax in this study and its range is from negative infinity to 1. As softmax will perform an exponential operation, when the output of the previous layer, which is the input of softmax, is relatively large, overflow may occur. So Log calculation can suppress this situation.

$$\sigma(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \quad for\ i = 1, 2, \ldots, K \tag{1}$$

Loss Function In this study, we use NLLLoss to find the loss value. After the explanation of softmax and logsoftmax in the previous section, we know that the absolute value of log results that is closest to 0 is consistent with the target. The calculation of NLLLoss is to take the average of the absolute minimum value in each row. The best case is 0 (No prediction error). Hence, If the target value and the predicted value are more similar, the result of NLLLoss is closer to 0.

Optimizer In this study, stochastic gradient descent(SGD) is used for optimizing. Compared with the Batch Gradient Descent(BGD)algorithm, it has a fast convergence speed and will not recalculate similar data samples. But the disadvantage is that there is no guarantee to find the best overall advantage. In other words, the SGD may be trapped at the saddle point.

2.3 Optimizing by Evolution Algorithm

The performance of a neural network depends mostly on its structure and hyperparameter setting. In order to obtain a better structure, the Evolutionary Deep Networks (EDEN) [7] algorithm is implemented in this study to optimize the NN model performance. This is an algorithm that combines genetic algorithm and evolution strategy together and optimizes the NN structure by imitating the "survival of the fittest" rule. We know that the genetic algorithm proposes a general framework. To establish a Genetic Algorithm, only needs to determine the "objective function" and "fitness function". Therefore, it is very suitable for optimizing the NN structure proposed in this study. In this study, the fitness function is the accuracy of the predicted value, and the objective function is to retain the model with higher accuracy in each iteration.
Mutation setting:

- Mutation for learning rate
  - The range of learning rate changing is 0.01 to 0.02

- Mutation for network structure
  - The minimun layer num is 0, which is same as the 1-layer fc from section 2.1
  - The maximum layer num is 3, which contains 3 hidden layers with different hidden neurons.
  - The range of hidden nrurons changing is 1500 to 2000

The activation function, the loss function and the optimizer have been determined (mentioned in section 2.2), so no optimization is done here. Because the EDEN Algorithm has great randomness, the result will be different each time (for example, the learning rate will not be the same). But through multiple iterations, a relatively stable and effective model structure can be found. The Fig1 shows the network structure of EDEN results. We will disccuss it in section 3.1.

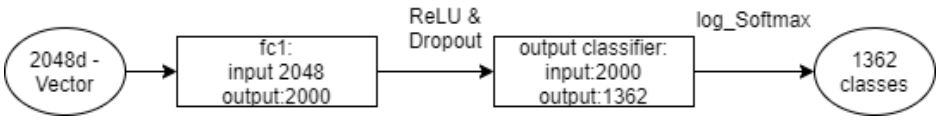


Fig. 1. The neuron network structure

2.4 Theoretical foundations for data mining of inputs

From the previous study [3], we know some of the attributes used to determine the importance of the neurons such as contribution, sensitivity, correlation and distinctiveness. These attributes can determine whether an input is important to the performance of a model or not. According to the paper [4]and [3], we know that distinctiveness is used to measure the similarity and invariance of each pair of neurons. Two input features would output identical (or highly similar) output features if their similarity to each other is very high or exactly opposite (the effects of the two input features would be cancelled out by each other). Therefore, removing one of the two highly similar inputs, or removing two completely opposite inputs together, does not affect the output of the model.

2.5 Input Features Distinctiveness Calculation

Based on Functional measures in paper [4], I have converted each input into a weighted vector from the first layer of hidden neurons and then calculated the angle between them to determine the distinctiveness(as formula 3). The input feature to the model is a 2048-dimensional vector and it would lead to the first layer of the network that contains 1800 hidden neurons, so a weight matrix of 2048x1800d was generated, with each column representing a weighted vector of each input feature. [4] refers to removing the vector bias by subtracting 0.5 (2). However, in the model of this study it was not necessary to subtract 0.5 from the vector. As the activation function used in [4] is sigmoid which can be used to do binary classification, it can map any real number to the interval (0,1) . Hence, the angle between the two vectors will always be in the range of 0-90 degree. It requires the vector to be minus 0.5 to ensure that the angle between the two vectors is in the 0-180 degree range. The activation function used in the model in this study is Relu, so there is no need to subtract 0.5 from the vector. This study would calculate the pinch angles of all input features, sort them from smallest to largest, and then filter and delete some of them. We would train the model with the same structure on the new input features and compare the before and after results.

$$sact(p,h) = norm(weight(h)) - 0.5 \tag{2}$$

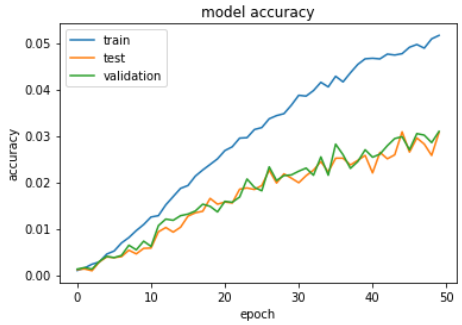
$$angle(i,j) = \arccos\left(\frac{i \cdot j}{\|i\| \cdot \|j\|}\right), where\ \|i\| = norm(weight(i)) \tag{3}$$

3 Results and Discussion

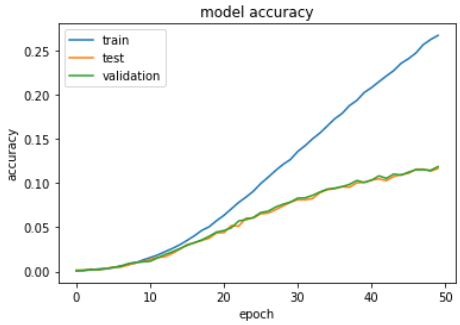
3.1 Performance of the baseline models with different network structures

Fig2 shows the performance of the 3 different networks.
From the comparison of Fig2(a) and Fig2(b) or Fig2(c), the increase of hidden neurons can greatly improve the accuracy of the model. This proves the inference of section 2.1 that a single-layer fc is not suitable for multi-class classifiers with large class numbers, the model needs more neurons to fit the data. For the EDEN algorithm mentioned

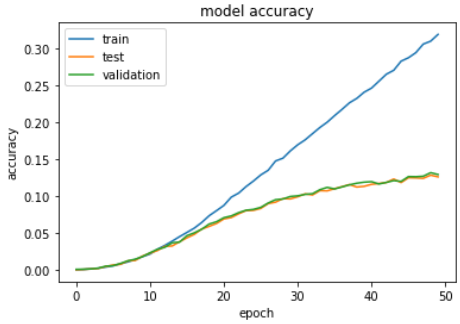
in section 2.3, due to the limitation of computing resources, we have set the 20 generation and 5 populations for iterating. Fig2(b) is the model that was eliminated in the last round and Fig2(c) is the best model. It can be seen that the accuracy of Fig2(c) is slightly higher than that of Fig2(b). Hence, the EDEN algorithm is suitable for optimizing this classifier.
From the iteration results of EDEN, the tenth-generation optimized models are all 2 layers fc with a different number of hidden neurons and learning rate. So for the classifier, adding more than 2 fully connected layers does not improve the prediction accuracy.



(a) Model 1: 1-layer fc.



(b) Model 2: 2-layer fc with 1800 hidden neurons.



(c) Model 3: 2-layer fc with 2000 hidden neurons.

Fig. 2. The model performance

3.2 Input eliminating performance comparison with the baseline model

According to the calculation method introduced in section 2.5, the trained NN model can be used to calculate the angle (as the distinctiveness) between 2048 inputs. Based on the algorithm, there are a total of 2096128 included angles for 2048 input vectors. In order to find the most similar vectors, we sorted the 2096128 angles in increasing order.
** All training epoch is 50*

Table 1. Eliminating the input features

Index	Angle threshold	No.of input features eliminating	Accuracy (%)	Prediction	Loss
None	None	None	13	1918/14936	4.6695
Top 30	50.235367	20	12	1826/14936	4.6917
Top 15	46.10372	13	13	1894/14936	4.6701
Top 10	**42.7529**	**4**	**13**	**1935/14936**	**4.6464**

The first row of Table 1 is the result of the baseline model without input eliminating operation. All Accuracy/prediction/loss are evaluated based on the validation set.

It can be found from the Table 1 that the number of selected angles and the number of deleted features do not match. Because some features are repeated several times (which means it is similar to many other vectors), we keep these repeated features as it can represent other vectors. The remaining selected features will be deleted.
From the table, we can find that the best result is the last row, which is to delete 4 similar input features. The steps to calculate similar input features are redundant for the model, which may aggravate overfitting. So deleting these similar input features may improve the accuracy of the model.
From these computational results, these similar angles are actually not very close to each other (the angle is not very small). As mentioned before, for a classifier, it is not necessarily sensitive to every input feature. If these features are not the target features of the classifier, deleting them will not have an excessive impact on the accuracy of the model.

The evaluation results with K-Fold Cross-Validation Because the process of training NN is very random, there will always be subtle differences in the results each time, and it is difficult to prove the stability and robustness of the model by evaluating the validation and test dataset only.
Therefore, this study uses K-fold cross validation to compare the results of the models. The value of K is selected as 5. Due to limited computing resources, it takes too long when the value of K is 10, and the dataset used here is the validation set. According to the above in this section, we only compare the NN model with 4 inputs deleted and the baseline NN model.
** All training epoch is 50*

Table 2. The performence of two models

Model	Average Accuracy (%)	Average Loss
Baseline Model	13	4.6683
Baseline Model + feature selection	**13**	**4.6432**

As shown in the table 2, although the accuracy of the two models is the same, the loss of the Baseline Model + feature selection is closer to 0 than Baseline Model, so it performs better.

4 Conclusion and Future Work

In this study, we constructed a neural network model as a multi-class classifier and used it to classify the samples in the VehicleX dataset. The result of Evolution Algorithm Optimization is a structure of 2 layer fc with 2000 hidden neurons (Figure1). In addition, we have added the input features selection for the model, the features are filtered according to their similarity, and the performance of the model is improved. For further study, we could consider the hidden neurons pruning technique [3] to simplify model structure for avoiding the overfitting. We can also use multi-task learning (Soft parameter sharing) [8] to further analyze the features, so that the network structure can be more sensitive to certain features.

References

1. Yao, Yue, Liang Zheng, Xiaodong Yang, Milind Naphade, and Tom Gedeon. "Simulating content consistent vehicle datasets with attribute descent." arXiv preprint arXiv:1912.08855 (2019).
2. He, Kaiming, 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, pp. 770-778. 2016.
3. T.D Gedeon, and Harris, D "Network Reduction Techniques," Proceedings International Conference on Neural Networks Methodologies and Applications, AMSE, vol. 1, pp. 119-126. 1991.
4. T.D Gedeon Data mining of inputs: analysing magnitude and functional measures Int. J. Neural Sys., 8 (2), pp. 209-218. 1997.
5. Glorot, Xavier, Antoine Bordes, and Yoshua Bengio. "Deep sparse rectifier neural networks." In Proceedings of the fourteenth international conference on artificial intelligence and statistics, pp. 315-323. JMLR Workshop and Conference Proceedings, 2011.
6. Hinton, Geoffrey E., Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R. Salakhutdinov. "Improving neural networks by preventing co-adaptation of feature detectors." arXiv preprint arXiv:1207.0580 (2012).
7. Dufourq, Emmanuel, and Bruce A. Bassett. "Eden: Evolutionary deep networks for efficient machine learning." In 2017 Pattern Recognition Association of South Africa and Robotics and Mechatronics (PRASA-RobMech), pp. 110-115. IEEE, 2017.

8. Misra, Ishan, Abhinav Shrivastava, Abhinav Gupta, and Martial Hebert. "Cross-stitch networks for multi-task learning." In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 3994-4003. 2016.