Performance of Using Neural Network with Pruning Inputs Methods on Vehicle Type Classification Task*

Chenkai Zhang

Australian National University, ACT 2601, AU
Chenkai.Zhang@anu.edu.au

Abstract. The development of autonomous vehicles technology requires training large amount of vehicle images to achieve high accuracy of vehicle identification. The features extracted from the pictures of the same vehicle type using CNN will vary greatly as influenced by many factors such as the orientation of vehicle or the positions of cameras. The problem is how to efficiently improve the accuracy on vehicle type classification given large amounts of features. Our motivation is to explore the performance of using genetic algorithm and data input analysis on classification task. This thesis describes the neural network based on the $VehicleX$ dataset, the technique of analysing input's contribution to output and the similarity between input features. Meanwhile, the genetic algorithm is implemented for comparing. We also propose a novel method that combines weight matrix analysis and genetic algorithm to prune the inputs. We find that the model can have a higher validation accuracy with the inputs pruned by genetic algorithm and weight matrix analysis compared to the baseline. Moreover, after implementing the proposed novel method, not only the accuracy is improved, but also the running time is greatly shortened.

Keywords: Vehicle type classification · Neural network · Genetic algorithm · Analysis technique.

1 Introduction

At present, machine learning is in the stage of rapid development, including neural network learning and genetic algorithm. Each of these technologies has a history of several decades, but at that time these methods did not receive enough attention because of the restriction of computer capacity. With the development of computers, there has been a convergence of these approaches. In this research, our motivation is to explore the implementation of genetic algorithm on neural network to improve the performance of model.Besides, data are now becoming more and more important in an area of deep learning, but it has always been a difficult problem to obtain a large amount of data, and the privacy security and complexity of data annotation are the focus of the problem. Synthesized data has an inherent advantage since they are synthesized without involving information of any specific person and have own annotations. So, the dataset in this research is VehicleX [7] containing the images generated by synthesized 3D vehicle models. We can manually generate the images with modifying some influencing factors like the light intensity. The picture (Figure 1) illustrates that a same vehicle type with different light intensity, different orientations, or varied background result in obvious difference in extracted features of these images.

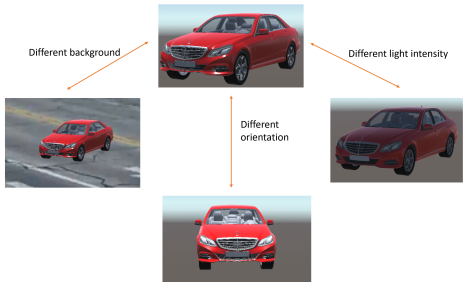

Fig. 1. Background, orientation and light intensity will influence features extracted.

VehicleX [7] provides us with 2048 extracted features of high-quality images and sufficient generation information with explicit parameters recorded in fine-grained label file. There are also 45,438 images for training, 14,936 images

* Supported by Australian National University.

for validation and 15,142 images for testing, but it is not necessary to use all training data. Empirically, the proportion to the training and test/validation set will be 80% and 20% or 70% and 30%. In this thesis, 2000 images from the training set, 500 images from the testing set and 500 images from the validation set will be randomly selected to reduce running time. Furthermore, the name of file in the dataset corresponds to the 'image Name' of fine-grained label file. So, these file names will be used to find corresponding type id in the label file and use dictionary to store 2048 features as well as type id. In the following parts, the features will be input to the neural network for classifying the vehicle type of the image. The neural network will get inputs of 2048 features to identify one of the 11 vehicle types. Moreover, the training process will be operated on GPU. According to Raina et al. (2009b) [5], when the architecture is 1024 * 4096, the speed of GPU is 16.2 times faster than the speed of Dual-core CPU. Considering the input features have 2048 dimension, the training process is supposed to put onto GPU. Consequently, the ability to accurately classify the type id with extracted features has been a problem since the big differences among features will greatly influence the performance of neural network to classify type id.

Tang et.all (2019) [6] propose Pose-Aware Multi-Task Re-Identification framework to classify vehicle types and colour while performing Re-Identification. We continue this subject to explore better methods to classify the vehicle types.

In this thesis, we aim to prune the inputs by implementing weight matrix analysis and genetic algorithm to increase the accuracy and reduce the running time. Meanwhile, we propose a novel method that combines the weight matrix analysis and genetic algorithm to select important inputs. After using the novel method, the most significant inputs will be selected, then we can reconstruct a neural network structure to reduce its complexity and improve its accuracy correspondingly.

2 Methodology

2.1 Features Normalization

The features pre-processed by normalization improve the accuracy of the neural network. Jo and Jun-Mo (2019) [3] state that the performance of the Machine Learning can get improved while the Big Data has been pre-processed for normalization. Each data file contains 2048 features ranging from 0 to 3. We build a neural network which contains two hidden layers with, respectively, 2500 and 1500 hidden units, and train it for 5 epochs at 0.0001 learning rate. As the weights of data are stochastically initialized, this step will be repeated 5 times respectively for normalized data and unnormalized data to weaken the influence of randomness. The aim of setting up 5 epoch and 5 repeated times is to reduce running time. The result is presented below (Figure 2). The axis x is repeated times, and the axis y represents the average accuracy of each 5-epoch training. The red line represents the normalized features, and the green line represents the unnormalized features. Obviously, the performance on the same neural network with normalized data is better than the unnormalized features. To be more specific, the respective average validation accuracy is 24.5% against 19.1%.

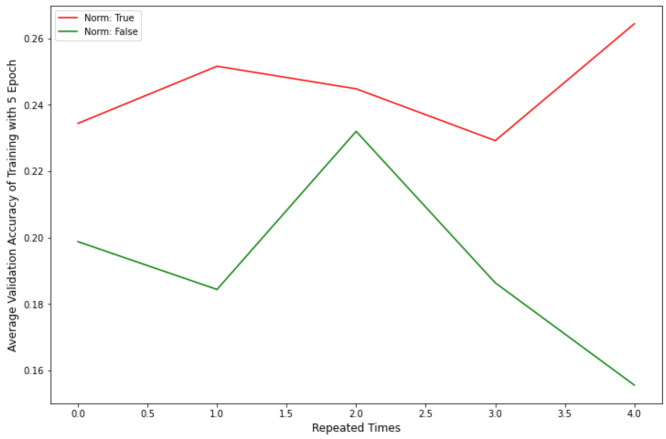

Fig. 2. Background, orientation and light intensity will influence features extracted.

Therefore, we should normalize the input features by using 'normalize' function imported from sklearn.preprocessing with 'l2' norm. We use the normalized features as input in the following parts.

2.2 Neural Network Structure

Linear activation function can result to a high accuracy on the classification task. Before comparing, it is necessary to construct a structure of neural network including hidden units and hidden layers. According to Hornik et al. (1989) [2], with as few as one hidden layer containing enough neurons, a multi-layer feedforward network can approximate a continuous function of any complexity with any precision. Based on this theorem, the neural network is empirically designed with 4096 of the first hidden units and 2048 of the second hidden units since the first hidden units is twice than the input features and the number of the second hidden units is getting smaller. Moreover, 50 is very suitable as epoch and 0.0001 is appropriate to be learning rate because the neural network will be fully trained without overfitting or underfitting over all the data from training, testing and validation set. To accelerate the process speed, each neural network with different activation will be trained with 5 epoch and this step will be repeated 5 times to weaken the influence of randomly selected data and initially stochastic weights in the network. The axis x is repeated times (Figure 3), and the axis y represents the average validation accuracy of training with 5 epochs. The red, green, blue, and orchid line, respectively, represents Linear, Sigmoid, LeakyReLU and Tanh activation function. Apparently, the green line represents the Sigmoid activation function performs worst. Although the lines of Tanh and Linear are close, the accuracy of Linear is still a little bit higher than Tanh, and the mean accuracies of both are, respectively, 22.54% and 22.42%.

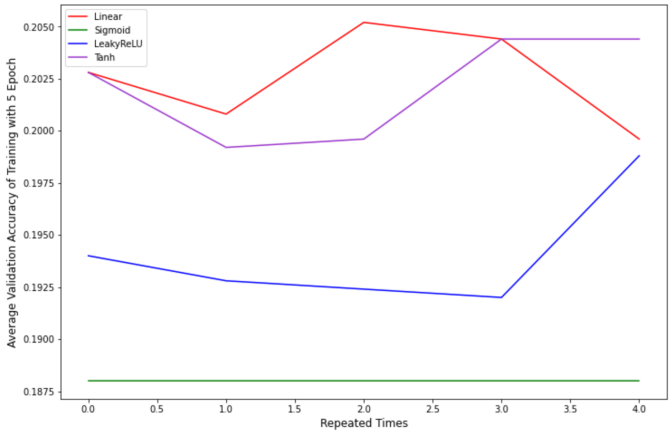

Fig. 3. The normalization on input features will performance better.

There is a note that the performance results of these activation functions may be change because of different dataset and randomly initialized weights, but the overall performance is that $Linear \approx Tanh \geq LeakyReLU > Sigmoid$ on this classification task.

2.3 Weight Matrix Analysis

This section is to analyse whether the weight matrix analysis (Gedeon, 1997) [1] can reduce the neural network complexity for accelerating the speed of classification task and can increase the accuracy of vehicle type classification.

$$P_{ij} = \frac{|w_{ij}|}{\sum_{p=1}^{ni} |w_{pj}|}$$

P_{ij} can be considered as the proportion of the weight from i^{th} to j^{th} neuron in the total weights of this layer from inputs neurons to the first hidden neurons. w_{ij} is the weight between i^{th} at the present layer and j^{th} from the next layer.

$$P_{jk} = \frac{|w_{jk}|}{\sum_{r=1}^{nh} |w_{rk}|}$$

Similarly, P_{jk} can be regarded as the proportion of the weight from j^{th} to k^{th} neuron in the total weights of this layers from the first hidden neurons to the next layer's neurons.

$$Q_{ik} = \sum_{r=1}^{nh}(p_{ir} \times p_{rk})$$

Q_{ik} is the contribution of input i^{th} to the output k^{th} throughout the weights of each layer. The fundamental principle of this technique is using the weight to calculate the proportion from input to output and then obtaining the most and the least important input features. The weight matrix analysis cannot improve the accuracy of the neural network if the activation function is not linear function since the activation function has changed the value of original input features. In other words, the output of activation function fails to reflect the characteristics of the original input. Consequently, the calculated Q cannot represent the contribution from input to output. From the table below (Table 1), if the activation function is Tanh, then the accuracy after pruning is 6% less than using Linear function. Therefore, in the following part, the activation function of neural network is Linear.

Table 1. Comparison between Linear and Tanh in weight matrix analysis

Activation Function	Accuracy
Linear	32.60%
Tanh	26.60%

2.4 Functional Measures

Gedeon (1997) [1] also proposes another method that is functional measures to determine the angle between two inputs for determining the redundant or complimentary inputs. We use functional measures to prune those unnecessary inputs, which results to higher accuracy or less complexity of neural network.

$$\text{angle}(i,j) = \tan^{-1}\left(\sqrt{\frac{\sum_{p}^{pats}\text{sact}(p,i)^2 * \sum_{p}^{pats}\text{sact}(p,j)^2}{\sum_{p}^{pats}(\text{sact}(p,i) * \text{sact}(p,j))^2}} - 1\right)$$

This angle(i,j) function is designed to calculate the similarity between i^{th} neuron and j^{th} neuron. According to Gedeon(1997) [1], for the purpose that is to determine the functional differences between each input, the $sact(p,h)$ should be equal to $norm(weight(h))$–0.5. The P should be the training data point, i^{th} and j^{th} are the neurons at the same layers. Usually, if the angle is greater than 165° or smaller than 15°, it means that these two neurons are containing sufficiently similar information and one of them should be removed. After implementing this functional measure technique, the result is that there are no redundant neurons with sufficiently similar information. From the table below (Figure 4), all the angles are round 44.5°. So, there is no neuron which should be removed in the input

	1	2	3	…	2047	2048
1	44.54836	44.54837	44.54839	…	44.56971	44.43826
2	44.54843	44.54859	44.56736	…	44.52736	44.62736
…	…	…	…	…	…	…

Fig. 4. The normalization on input features will performance better.

layer and we don't use functional measures to prune the inputs. Then the output should be the original output of the neural network. Meanwhile, the original model which is based on the functional measures was trained with 100 epoch and was compared to the weight matrix analysis technique, the performance using functional is better than the model using the weight matrix technique.

2.5 Genetic Algorithm

Maulik and Bandyopadhyay (2000) [4] mention that genetic algorithm (GA) is a stochastic search and optimization techniques instructed by the fundamental principles of evolution, and it provides near-optimal solutions for the complex and large problems. In this research, genetic algorithm is used to prune 2048 inputs for reducing the running time and increasing accuracy of the neural network. There are several steps to implement this algorithm as well as Elitism method.

1. (Initialization) We randomly select N initial points from $[0, 2047]$ as an individual of population, and all values of the individual are integer to represent the index of features. Here, the N is set to 1024 since we can adjust for this value. For example, we can configure a smaller value of N while the performance of genetic algorithm is better than the baseline. The size of population is 50 and generation is 20. Because of the large dataset size and the limitation of computer capacity, these values help us find the optimal solution. Then, the following steps will be iterated for each generation.

2. (Fitness) A trained neural network is used to calculate the accuracy of input (each individual) as the fitness value as we can evaluate the performance of inputs by accuracy in the classification task. If we use loss as fitness, its accuracy after several generations is lower than using accuracy as fitness value (Table 2). So, we use accuracy as fitness value to choose better features with higher accuracy of neural network.

Table 2. Comparison between accuracy and loss in fitness

Activation Function	Accuracy
Accuracy	32.60%
Loss	29.40%

3. (Select) The individual with the best fitness value is elitism which should be selected. Other individuals are randomly chosen by proportional selection to compose a population for the following step.

4. (Crossover) The integer crossover points representing the index of inputs are randomly chosen from $[0, 2047]$ by exponential distribution $\lambda e^{-\lambda x}$ which enables us to control the rate of crossover. To be more specific, if $\lambda > 1$ and $x \in [0, 1]$, the probability of crossover reduces correspondingly, and then less points are exchanged, which results to lower rate of crossover; otherwise, more points get swapped between two parents. In the early stage, we can set $0 < \lambda < 1$ to accelerate to find the global minimum, and then gradually increase to converge to a local minimum. For every individual in population, we randomly choose a parent which is also from population to implement crossover function.

5. (Mutation) The integer mutation points are also randomly chosen by exponential distribution $\lambda e^{-\lambda x}$ from $[0, N]$ where N represents the size of an individual. As the mutation operation is to prevent the algorithm from converging to a local minimum, we set the λ to 1 and the mutation is operated while $e^{-x} < 0.4, x \in [0, 1]$.

6. We put the elitism back in the population. If the accuracy is higher than the accuracy of baseline or the generation is reached to the configured value, then the termination condition is satisfied and genetic algorithm should be stopped; otherwise, update the population and elite, go to step 2.

2.6 Novel Method

We propose a novel method that combines weight matrix analysis and elitism selected by genetic algorithm. 1024 features are selected from 2048 inputs with using both methods, respectively. Then, we keep all the same value and remove the different value to generate a new list containing the index of inputs. Based on the size of the new list, the neural network is reconstructed, and the number of neurons is greatly reduced to 512 (the original hidden neurons are 4096) since the number of inputs is pruned by this novel method to around 500 and this structure results to better performance of model.

3 Results and Discussion

The following results are based on 20 epoch, 0.0001 learning rate, Linear activation function, Adam optimizer, and all the accuracies are from validation dataset. However, the structure of neural network changes according to the specific task. Besides, the baseline is the validation accuracy of using 2048 features as inputs to train the neural network.

3.1 Inputs Pruning

After implementing the weight matrix analysis technique on the trained network, the result of significance is shown in the table below. There are 1024 most significant features (Figure 5) are selected as inputs. The reason for selecting 1024 inputs is that we can adjust it based on the performance. If the accuracy is low, it indicates that there are not enough features for model to make the right classification, then we should select more features as input, and vice versa.

	Least Significance (Feature Index)					Most Significance (Feature Index)		
Q_{ik}	1817	2022	1985	.....	.....	1032	262	538

Fig. 5. 1024 features are selected by weight matrix analysis.

We implement the genetic algorithm with a population of 50 individuals lasting 20 generations. Once the termination condition is satisfied, we reuse the selection function to pick out the elite which contains the index of inputs presented below (Figure 6).

Elite	1294	1949	232	.....	246	607	917

Fig. 6. 1024 features are selected by genetic algorithm.

Here, when we mention Same NN, it means that the selected features are remained, and other features are set to 0. We use the trained model to evaluate these inputs, and the results presented below (Figure 7). Apparently, the neural network fits the inputs pruned by GA better since these inputs are the Elitism that is to keep the best fitness (accuracy) of individual in each generation. Whereas the inputs pruned by weight matrix are calculated by the weights from inputs to the outputs. So, accuracy of using weight matrix analysis is 20% lower than using GA.

Same NN	Accuracy
Baseline	29.80%
Weight Matrix	18.20%
GA	38.20%

Fig. 7. Same NN means that they run under the same neural network.

Then, we re-import these inputs to continue to train the neural network that has already been trained for another 20 epochs, and results are shown below (Figure 8). We can find that the validation accuracies of weight matrix and GA have improved compared to the baseline, but the training accuracies get greatly increased and result to overfitting since the accuracies are over 13% higher than the validation accuracies. So, it is not appropriate to use these pruned inputs to continue train the fully trained network.

Same NN	Train Accuracy	Validation Accuracy
Baseline	37.20%	29.80%
Weight Matrix	47.60%	32.00%
GA	45.30%	32.60%

Fig. 8. Training on the same trained neural network.

Next, the original neural network that is initialized network without being trained is trained with the inputs pruned by weight matrix and GA presented below (Figure 9). We can see that, the inputs pruned by GA can bring

higher accuracy than weight matrix and the baseline at respectively, 30.40%, 29.80% and 29.80%. Meanwhile, the training accuracy (36.8%) of using GA is also close to the validation accuracy (30.4%), which indicates that the model using GA to prune the inputs has a better generalization.

Same NN	Train Accuracy	Validation Accuracy
Baseline	37.20%	29.80%
Weight Matrix	39.60%	29.80%
GA	36.80%	30.40%

Fig. 9. Training on the same initialized neural network.

In the following part, we will prune the inputs by remaining the 1024 selected features and removing the rest rather than setting to 0. After that, the input neurons are reduced to 1024 correspondingly, and then the structure of the neural network should be adjusted for this change.

Based on Hornik et al. (1989) [2] theorem, the neural network can be empirically reconstructed with 2048 of the first hidden and 1024 of the second hidden units. The reconstructed neural network is trained with the inputs pruned by weight matrix and GA, and results presented below (Figure 10). We can see that the running time has been reduced almost 50% while pruning unnecessary features and reconstructing the neural network. Meanwhile the validation accuracies of both methods do not decrease with the shortening of running time but get increased. The validation accuracy of using weight matrix to prune is 0.8% higher than the baseline, and GA's accuracy has improved 1.4% compared to the baseline.

Original NN	Time	Train Accuracy	Validation Accuracy
Baseline	2min 42s	37.20%	29.80%
Reconstructed NN	**Time**	**Train Accuracy**	**Validation Accuracy**
Weight Matrix	1min 14s	39.65%	30.60%
GA	1min 10s	36.65%	31.20%

Fig. 10. Training on the reconstructed neural network.

Therefore, we can draw a conclusion that the model of using weight matrix and GA to prune the inputs get a better accuracy compared with the baseline that trains on the original features. The performance of using GA is also better than using weight matrix. Moreover, the efficiency can get greatly improved while reconstructing the neural network based on the size of the pruned inputs without causing the decrease of validation accuracy.

3.2 Novel Method

We find that there are many same features in the pruned inputs by weight matrix and GA. Inspired by this finding, we pick out the same features, and then reconstruct the neural network based on the new input size which is 507. Then, the model with the new input is trained for 20 epochs with 0.0001 learning rate, linear activation function and Adam optimizer. The results presented below (Figure 11). We find that the novel method can result in a better validation accuracy with much shorter running time, meanwhile, the validation accuracy gets greatly improved 3.8% compared to the baseline and is also better than the accuracies of simply using weight matrix analysis or GA to prune the inputs. Furthermore, we repeatedly this experiment to validate the novel method is useful, and all results show that it can greatly increase the accuracy and reduce the running time. So, the novel method is an effective way to prune the inputs.

Original NN	**Time**	**Train Accuracy**	**Validation Accuracy**
Baseline	2min 42s	37.20%	29.80%
Reconstructed NN	**Time**	**Train Accuracy**	**Validation Accuracy**
Novel method	42.7s	34.45%	33.60%

Fig. 11. Novel method with training on the reconstructed neural network.

4 Conclusion and Future Work

4.1 Conclusion

The normalization of input features can improve the accuracy on classification task and the linear activation function is very appropriate to output. Moreover, if the network contains activation functions, the weight matrix analysis technique does not have very good performance on pruning the input based on the contribution of each input to the output. However, this technique can indicate what the input contributions are important in the neural network, then these significant neurons can be kept, and the least important neurons can be removed. This technique can increase the accuracy compared to the baseline. The model of implementing GA to prune the inputs results to better performance than using weight matrix. Furthermore, the proposed novel method that combines the weight matrix and GA have a better performance that leads to much shorter running time and higher validation accuracy compared to the baseline, weight matrix analysis and GA.

4.2 Future Work

In the future work, we want to figure out the relationship between the fitness and the λ of $\lambda e^{-\lambda x}$ used in crossover and mutation. In other words, we aim to implement a self-adaptive genetic algorithm which can automatically the adjust the λ to find and converge to the global minimum faster.

References

1. Gedeon T. D. (1997). Data mining of inputs: analysing magnitude and functional measures. International journal of neural systems, 8(2), 209–218. https://doi.org/10.1142/s0129065797000227
2. Hornik, K., Stinchcombe, M., White, H. (1989). Multilayer feedforward networks are universal approximators. Neural Networks, 2(5), 359-366. https://doi.org/10.1016/0893-6080(89)90020-8
3. Jo, J.-M. (2019). Effectiveness of Normalization Pre-Processing of Big Data to the Machine Learning Performance, 14(3), 547-552. https://doi.org/10.13067/JKEICS.2019.14.3.547
4. Maulik, U., Bandyopadhyay, S. (2000). Genetic algorithm-based clustering technique. Pattern Recognition, 33(9), 1455-1465. https://doi.org/10.1016/S0031-3203(99)00137-5
5. Raina, R., Madhavan, A., Ng, A. (2009). Large-scale deep unsupervised learning using graphics processors. Paper presented at the 873-880. https://doi.org/10.1145/1553374.1553486
6. Tang, Z., Naphade, M., Birchfield, S., Tremblay, J., Hodge, W., Kumar, R., Wang, S., Yang, X. (2019). PAMTRI: Pose-aware multi-task learning for vehicle re-identification using highly randomized synthetic data. Paper presented at the 211-220. https://doi.org/10.1109/ICCV.2019.00030
7. Yao, Y., Zheng, L., Yang, X., Naphade, M., Gedeon, T. (2019). Simulating content consistent vehicle datasets with attribute descent. arXiv preprint arXiv:1912.08855.