Using a convolutional neural network to classify the colour of cars from the simulated Vehicle X dataset and explaining the results using a decision tree

Zaran Zahid
Research School of Computer Science,
The Australian National University,
Canberra ACT 2601 u6347342@anu.edu.au

Abstract: Classifying images is an important task in computer vision and deep learning. The ability to recognise and detect objects has several applications from diagnosing illnesses by interpreting medical images to implementing self-driving cars. In this paper, a convolutional neural network with two convolutional layers and one hidden layer was trained to classify images of vehicles based on their colour. The input features representing the image of a vehicle were used to classify the images. The accuracy of the trained model is found to be 83%. The results were compared with a three-layer neural network, which produced a model with an accuracy of 64%, showing that convolutional neural networks are more suited for image classification compared to a normal neural network. In addition to training these neural networks, a decision tree has been used to generate rules for both neural networks to explain the results of the model. The accuracy of the decision tree was then found to be 72% and 84% for the three-layer neural network and the convolutional neural network, respectively. The results were presented and future improvements and studies have been discussed.

Keywords: Image classification, convolutional neural network, three-layer neural network, decision tree, computer vision, vehicle identification

1 Introduction

Image classification has been a hot area of research right from the advent of computers. The ability to recognise patterns in images has been something computer have struggled to do for a long time. However, with neural networks the ability the learn from examples makes them a useful tool for image classification and for many other applications in general [1]. In particular, convolutional neural networks have been used extensively for image classification with high accuracy [12] and has also been used to improve breast cancer detection on screening mammograms [10].

In this paper, a convolutional neural network (CNN) is used to predict the colour of a vehicle in an image. By doing so, the network must be able to recognise pixels which correspond to the car and determine the features that provide information of the colour of the car, while discarding other attributes. The CNN is compared with a normal, three-layer network. Furthermore, both networks are then attempted to be explained by a set of rules which are comprehensible and show how the neural network returns an output. This is done by using decision trees, which uses several rules to distinguish between the inputs and reach a certain output. Explaining the neural network by generating a set of rules was also done in this paper [2] to predict student's grades after 40% worth of assessment has been completed.

1.1 Dataset

The dataset used in this experiment is a large-scale synthetic dataset called VehicleX, which was created in Unity by ANU and NVIDIA in 2019 [3]. The dataset contains 1362 vehicles and was originally used for vehicle re-identification. Each image feature vector was stored as a NumPy file, while the labelling of the images was stored in an xml file. The xml file contained the colour of the vehicle, along with several other labels such as the type of vehicle, the camera angle used and the lighting conditions. These labels are shown in Figure 1.This was for the purpose of multi-task learning, however for this paper only the colour of the vehicle is used. Furthermore, although there were several different cameras used, for this experiment only one camera angle was used and an image corresponding to a specific vehicle ID was used once to avoid the same vehicle being used again.

Image Attribute	Number of Classes (or Type of Attribute)
Vehicle Orientation	Float
Light Direction	Float
Light Intensity	Float
Camera Height	Float
Camera Distance	Float
Camera ID	20
Colour of Vehicle	12
Type of Vehicle	1362

Table 1: The labelled attributes of each image in the VehicleX dataset

The raw images in the VehicleX dataset were square, RGB images with sides being 256 pixel lengths long. Rather than using the pixel values as the features of the image, each image has a feature input vector with 2048 features. These features were obtained by applying a pre-trained ResNet on the images and are the output activations from the final hidden layer [3]. This makes it impossible to formulate a model just based on inspecting the features of every image and so a neural network is needed to learn from every input provided in the training phase.

There were 12 different colours in total in the VehicleX dataset. The colours were represented by integers from 0 to 11, and the mapping between the label and the colour is shown below in Table 2. There would also be 12 outputs in the neural network, corresponding to these 12 colours.

Colour Label	Colour
0	Yellow
1	Orange
2	Green
3	Gray
4	Red
5	Blue
6	White
7	Golden
8	Brown
9	Black
10	Purple
11	Pink

Table 2: The 12 labels that were used as the output of the neural network

1.2 Explaining the results of a neural network

Artificial neural networks have one of the highest classification in many areas of application [4], but despite its success in classification, ANNs have one major drawback. They do not provide any explanation as to how a particular conclusion and output is reached [1]. A specific parameter or neuron in the model might be used to detect a certain feature of the image, but the model will not provide any indication on what feature a neuron is detecting. This lack of explanation reduces the acceptance of neural networks especially in decision systems, where the output must be presented with the evidence and reasoning used to reach it. Neural networks can be

described as a black box that simply take in an input and return an output without any explanation of how the output was reached [4].

Due to this disadvantage, researchers have proposed several rule extraction algorithms that can be used to explain the results of an ANN [4]. Rule extraction algorithms are white box models that ‘closely approximate a given network’s behaviour’. [5]. These algorithms are comprehensible and transparent in how an output is reached,

which allows the original neural network to be cross-checked by experts and can reveal the inner workings of the network [4]

1.3 Causal Index and decision trees

One major rule extraction algorithm is to use the differentiability off an activation function to derive the effect an input neuron *i* has on an output neuron *k*. This will produce the casual index [4] and the value of the causal index for a chosen *k* and *i* will indicate the relationship between these two neurons. The causal index can then be used to produce a set of rules that describe key factors and their effect on output. In addition to the Causal Index, for each output a characteristic input must be determined, which can be done by finding the arithmetic mean of every input that returns the particular output [4]. These inputs are called characteristic ON input patterns, while all other inputs are called characteristic OFF input patterns. The characteristic input is used as a representative of the set of inputs that return a specific output. Since Causal Indices are affected by the input, characteristic inputs must be used to generalise the rule extraction algorithm.

Another common rule extraction algorithm is to produce a decision tree [4]. Decision trees are widely used for classification in machine learning and data mining as a white box system that can be trained and explain easily. Training a decision tree is done by providing a labelled training dataset and running the algorithm for enough time to learn the patterns in the data.

A review of several different rule extraction algorithms are presented in the paper by Hailesilassie [4]. In this paper, the decision tree model will be used to explain the results of the three-layer network.

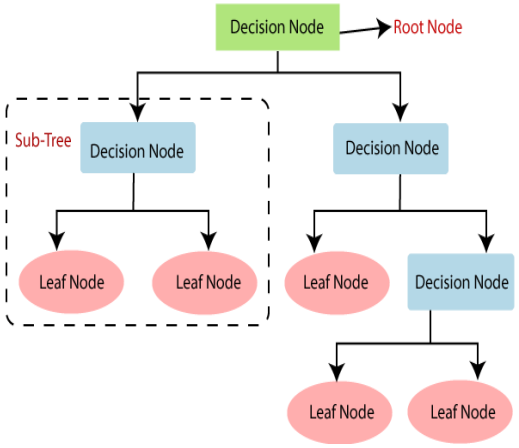

Figure 1: An example of a decision tree. The leaf nodes represent the different possible outputs. The decision nodes make choices based on the input features. Figure accessed from [9].

1.4 Convolutional Neural Networks

A Convolutional neural network (CNN) is a type of deep learning network, usually applied to analyse images. CNNs use convolutional layers that convolve the input before passing the results on the next layer. These convolutional layers extract the features and assemble patterns of increasing complexity building up from each convolutional layer. CNNs have several other layers, including the pooling layer which produces a summary statistic of the grouped outputs from the previous convolution layer, down-sampling the feature map into a condensed version

CNNs have been shown to have excellent performance in machine learning problems [12], especially for applications that deal with image data. This is due to their spatial invariance, which means that the network is robust to changes of the position of the object between images. They are also more efficient than normal neural networks, due to the convolutional layers not being fully connected as compared to neural networks where each layer has a full connection with the next layer.

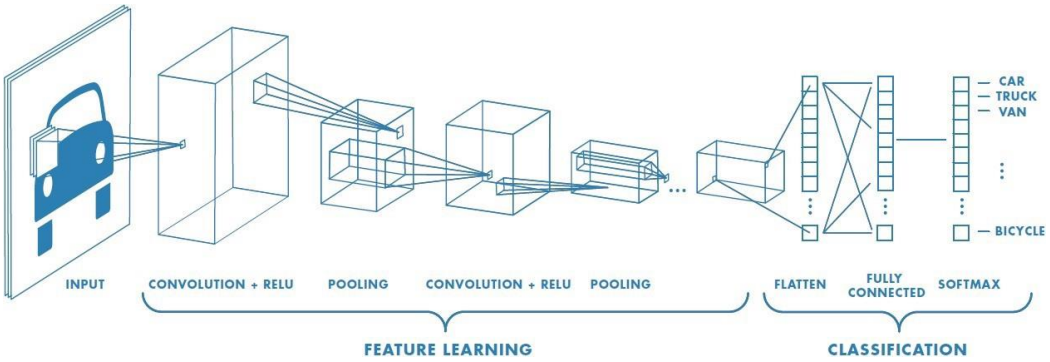

Figure 2: Convolutional Neural networks use several convolutional layers coupled with Relu and pooling to extract features from an image. The classification section is a fully-connected hidden layer using the extracted features as input [11].

2 Method

The hardware used to train the three-layer network is: CPU: Intel® Core™ i7-10700k @ 3.8GHz; Memory: 16.00GB; Software: python3

2.1 Processing raw data

To process the images, first the dataset was downloaded. Python was used to access the image features as they were encoded in a NumPy file. Only images which used the camera ID 1 were used and one image was taken for each vehicle. This was to limit the amount of data being used so that the data can be processed and the model trained in a shorter time duration.

By accessing the xml files, the colour of each vehicle was associated for each image vector. The result was an array with 1362 rows (corresponding to each image) and 2049 columns (2048 features along with the label of the vehicle). This array was then converted into a csv file so that the raw data does not be processed every time the file is run.

This csv file is then accessed, and every column apart from the last column (which contains the label) is normalised. The normalising procedure was done by looping through the 2048 columns and for each, the mean of the column was subtracted. The results was then divided by the standard deviation of the column. This is essentially normalising the data to have mean 0 and standard deviation 1. The data is then split randomly into a training set and testing set, with the training set being 80% of the original data.

2.2 The three-layer neural network

After the data was pre-processed, the training data was fed into a neural network. The hyperparameters are shown below:

Hyperparameter	Value/Assignment
Epoch	500
Hidden layer 1 (HL1)	16 neurons
Activation Function for HL1	Sigmoid Function
Output Layer	12 neurons (for 12 colours)
Output Layer Activation Function	Sigmoid Function
Batch Size	1
Optimiser	Adam Optimiser
Learning Rate	0.005
Loss Function	Cross Entropy Loss

Figure 3: Hyperparameters used for the three-layered neural network. There is only one hidden layer. These parameters were modified and updated based on the results they returned.

The number of epochs was decided by checking when the model had reached a stable state and was not changing. The hidden layer was simply decided by trial and error. Using 16 layers resulted in the best results in the training set that did not start overfitting the test set. The activation layers were both chosen to be the sigmoid function, which is a common function that returns a value between 0 and 1. The Cross Entropy Loss was used for the loss function, which is a common classification loss function.

The hyperparameters were decided by running multiple tests with different hyperparameters and recording the set which gave the best results.

Once the hyperparameters were decided, the neural network was trained and in the python file, the accuracy of the model was calculated after every 50 epochs. Once the neural network had trained for the specified number of epochs, the confusion matrix showing the distribution of what the neural network was classifying was calculated for both the training set and the test set.

2.3 The convolutional neural network

The pre-processed data was also fed into a convolutional neural network. The hyperparameters are shown below:

Hyperparameter	Value/Assignment
Epoch	500
Convolutional Layer 1 (CL1)	10 outputs, kernel size = 5
Activation Function for CL1	Relu
Pooling for CL1	Max Pooling (kernel size = 2)
Convolutional Layer 2 (CL2)	12 outputs, kernel size = 5
Activation Function for CL2	Relu
Pooling for CL2	Max Pooling (kernel size = 5)
Batch Size	64
Learning Rate	0.01
Loss Function	Negative Log Likelihood Loss
Hidden layer 1 after convolutions (HL1)	50 neurons
Activation Function for HL1	Relu
Output Layer	12 neurons (for 12 colours)
Output Layer Activation Function	Log softmax function
Batch Size	1
Optimiser	Stochastic Gradient Descent (SGD)(momentum = 0.5)

Figure 4: Hyperparameters used for the three-layered neural network. There is only one hidden layer. These parameters were modified and updated based on the results they returned.

These parameters were chosen as they resulted in the best CNN performance with the same amount of epochs as the three-layer neural network. Note that the convolutions applied here are 1-dimensional convolutions, because the features provided had already been extracted from another CNN and each image was represented by a 1-dimensional vector of length 2048. The same data was used in the Convolutional Neural network as the three-layer neural network. The accuracy was then displayed using a confusion matrix in the terminal output

2.4 Decision Trees

The decision tree was then implemented using the built-in function in the sklearn.tree library. The class DecisionTreeClassifier was imported and this decision tree was trained using the training set, however the target output were those given by the neural network rather than the actual target output. For example, if the neural networks predicted the vehicle colour to be red even though the actual colour is blue, then the target output for the decision tree would be red rather than blue. This is because the decision tree is being used to explain the results of the neural network.

Another confusion matrix was calculated to check how many times the decision tree agreed with the neural network on the test set. The accuracy of the decision tree to predict the same output for a given input was calculated using the confusion matrix.

A decision tree was implemented for both the three-layer neural network and the convolutional neural network, to check whether convolutional neural networks are more suited than normal neural networks to have their results explained by a decision tree.

3 Results and Discussion

3.1 Results of the Three-layer Neural Network

Figure 4 shows the loss of the neural work against the number of epochs that the neural network has been trained during the training phase of the neural network. Clearly, as the number of epochs increases, the network has more time to recognise the learn the patterns in the data, which then decreases the loss function after every iteration.

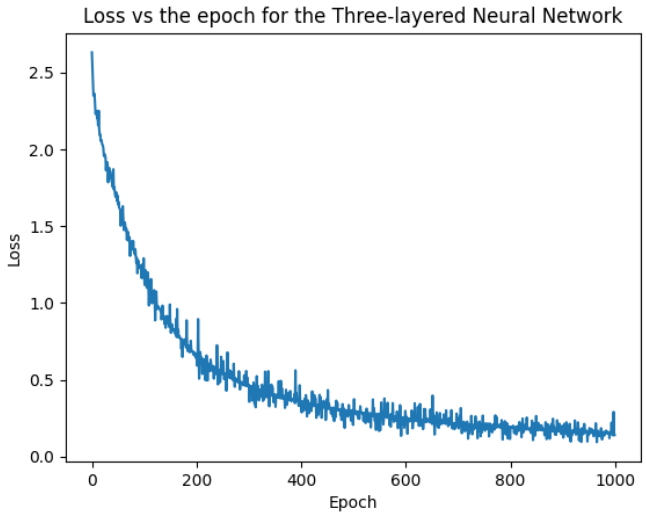

Figure 5: Loss vs Epoch Iteration for the Three-layered Neural Network

After the training is completed, the accuracy on the training set was found to be 96%, which shows that there are still some training datapoints that are incorrectly classified. On the test set, the neural network did not perform as well, and achieved an accuracy of 64%. The confusion matrix is shown below for the test dataset.

3.2 Results of the Convolutional Neural Network

After the training is completed, the accuracy on the training set for the convolutional neural network was found to be 98%, which shows that there are still some training datapoints that are incorrectly classified. On the test set, the CNN still performed significantly well, with an accuracy of 83%. This indicates that the CNN is more accurate in classifying the images based on their colour.

The high accuracy results for the CNN is expected due to their prevalence in image classification [12]. The convolutional layers can extract the image features and then these features can be used for the fully-connected neural network that is found at the end of CNNs. Considering that the original dataset obtained the feature images using another CNN [3], the high accuracy results show that CNN are significantly better than normal three-layer neural networks for image classification.

The neural network is making many incorrect classifications in nearly every colour. There are many reasons why the neural network is performing like this. Firstly, the inputs that were given were not from the raw image. Rather, the pixel values of the images had already been fed into another neural network. Since this other neural network was used to re-identify the vehicles, the features take into account several other variables which might be more indicative of a vehicle than simply its colour. The dataset was also synthetically created, and so there is a difference between the real and simulated data [3]. Neural networks have been largely successful in image classification problems related to identifying cars [6], and has even been used for plate recognition with accurate results [7]. Using the raw pixel values is a valid improvement that can be made for this investigation rather than inputs which have already been passed through a neural network. However, due to the large image size (256 pixels by 256 pixels) in the dataset, with each pixel holding 3 channels, the number of inputs for the first layer will be 65,536 3-channel features. Using a fully-connected neural network will make the number of the parameters in the system

to increase significantly, compared to using a convolutional neural network which is able to group similar pixel values together.

The neural network could be overfitting the training dataset. This is because the model has a very high accuracy on the training dataset, but fails to perform as well in the test dataset, which indicates overfitting. This could be fixed by implementing an early stopping feature that tells the network to stop training once a specified accuracy is reached [8]. Alternatively, the number of neurons in the output layer can be decreased and modified until there is a more balanced network which performs similar in both the training and testing set. As the convolutional neural network performs well on both the training and test set, overfitting does not seem to be happening in the CNN.

Another aspect that could be improved was having an equal number of datapoints for each label. In the dataset, there were many more vehicles with colour "0" compared to "11" (Pink). Because of unbalanced class sizes, the model is more inclined to predict the more heavily weighted class for every input. If instead there is a balanced number for the class size, there would not be this bias towards the bigger class sizes. Furthermore, some colour were very similar to each other and could have been combined to form one large class. This would have resulted in fewer labels and make it easier for the neural network to accurately predict the labels because similar colours would be grouped together.

3.3 Decision Tree Results

The decision tree correctly predicted the same output as the three-layer neural network 72% of the time. The decision tree was 100% for the training set.

The results indicate that using a decision tree to explain the outputs of a neural network does not return accurate results. The decision tree is making several incorrect labels on the test set, but achieves 100% on the training dataset. This shows that the decision tree has been overfitted for the testing set and cannot be expanded to the testing set. Due to the thousands of input features that have no semantic meaning, the rules that the decision tree has generated cannot be cross-checked. However, the rules can be viewed and it can be seen how the decision tree returns the output that it thinks is correct, which is the purpose of using the decision tree.

For the convolutional neural network, the decision tree correctly predicted the same output 84% percent of the time for the test set and 95% for the training set. The decision tree for the convolutional neural network seems to as overfitted compared to the three-layer neural network. A reason why the CNN might have less overfitting compared to the normal neural network is that the CNN has higher accuracy on the test set, and so the model is more accurate and not overfitting. In contrast, since the neural network performed badly on the test set, this model is most likely overfitting the test set and so proper rules have not been found that can classify the image.

4 Conclusion and Future Studies

The results for this paper have been summarised in the table below.

Model	Accuracy on test set	Accuracy of corresponding decision tree
Three-layer Neural Network	64%	72%
Convolutional Neural Network	83%	84%

Figure 6: Summary of the results obtained in this paper.

The purpose of this paper was to train a convolutional neural network and a three-layered neural network on the VehicleX dataset to detect the colour of different vehicles, comparing the accuracy of both of these models and verifying if convolutional neural networks perform with a higher accuracy as shown in existing literature [13] After setting the hyperparameters, the models were successfully trained and it the three-layer neural network was found to be 72% accurate on the test set. The lower percentage can be attributed to the way the inputs have been stored, since the inputs are the outputs of another neural network on which the original images of the vehicles were performed. Furthermore, there was evidence of overfitting since the neural network performed significantly better (96%) on the training set.

The convolutional neural network obtained an accuracy of 83% on the test set, which is nearly 20% more accurate than the three-layer neural network.

The next aim for this paper was to find a white box system that can explain the results of the convolutional neural network and the three-layer neural network. This was done by implementing a decision tree, as it has been shown that decision trees are robust models which perform well with datasets that have many features. The decision tree was found to be 100% accurate on the training dataset where the labelled outputs were the neural network's outputs instead. However, in the testing set the decision tree was unable to perform as well and had an accuracy of 72%.

In future studies, the raw input features of a vehicle can be used instead of the attributes taken from another neural network, as it is likely that the other neural network could discard the colour of the cars and focus on other attributes instead. This will also allow the rules of the decision tree to be reasoned upon. Further studies can also be used to see which rule extraction algorithms performs better on a three-layered neural network. Furthermore, the effect of keeping the same number of hidden layers in the convolutional neural network and the normal three-layer network can be recorded. Since in this experiment, the hyperparameters were tuned until the best accuracy for both of the networks were achieved, the results are not as good a comparison if the activation functions and the layers were all kept the same.

5 References

[1] Turner, H., and Tamás D. Gedeon. "Extracting Meaning from Neural Networks." Proceedings 13th International Conference on AI. Vol. 1. 1993.

[2] Gedeon T. D. and Turner H. S.:Explaining student grades predicted by a neural network. In Proceedings of 1993 International Joint Conference on Neural Networks(1993)

[3] Yao, Yue, et al. "Simulating content consistent vehicle datasets with attribute descent." arXiv preprint arXiv:1912.08855 (2019).

[4] Hailesilassie, Tameru. "Rule extraction algorithm for deep neural networks: A review." arXiv preprint arXiv:1610.05267 (2016).

[5] M. W. Craven, "Extracting Comprehensible Models from Trained Neural Networks", Ph.D. dissertation, Department of Computer Sciences, University of Wisconsin-Madison, 1996

[6] Huttunen, Heikki, Fatemeh Shokrollahi Yancheshmeh, and Ke Chen. "Car type recognition with deep neural networks." 2016 IEEE intelligent vehicles symposium (IV). IEEE, 2016.

[7] Parisi, Raffaele, et al. "Car plate recognition by neural networks and image processing." 1998 IEEE International Symposium on Circuits and Systems (ISCAS). Vol. 3. IEEE, 1998.

[8] Prechelt, Lutz. "Early stopping-but when?." Neural Networks: Tricks of the trade. Springer, Berlin, Heidelberg, 1998. 55-69.

[9] Decision tree figure, accessed from https://www.theinsaneapp.com/2021/02/types-of-machine-learning-algorithms.html

[10] Shen, L., Margolies, L.R., Rothstein, J.H., Fluder, E., McBride, R. and Sieh, W., 2019. Deep learning to improve breast cancer detection on screening mammography. Scientific reports, 9(1), pp.1-12.

[11] Convolutional Neural Network figure, accessed from https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53

[12] Albawi, S., Mohammed, T.A. and Al-Zawi, S., 2017, August. Understanding of a convolutional neural network. In 2017 International Conference on Engineering and Technology (ICET) (pp. 1-6). Ieee.