The Use of Neural Networks To Classify Whether An Image Contains A Car

Erik barrow

December 2014

Abstract

Computers traditionally find it very hard to identify objects in images it is presented with, to solve this problem much research into machine learning has been produced since computers were invented. The goal of this paper is to introduce the idea of neural networks and investigate their effectiveness in recognizing objects such as cars in a set of images.

Tests were performed on a number of neural networks to obtain the best classifier, and were then compared to classifiers produced by KNN and Classification Trees.

Tests showed that a neural network was 92% accurate at identifying cars in a test set of images. This is a acceptable rate of recognition for multiple applications and was better than the result of the Classification Tree which obtained only 81% accuracy. The network was however outperformed by the KNN classifier which obtained 98% accuracy. Regardless of this the neural network showed promise in computer vision, and has the capability to scale much better than the other classifiers tested here.

1 Introduction

In this paper I will introduce the idea of machine leaning and its applications within the field of computer vision and object recognition. I will then apply a well known and powerful machine learning method to be compared against some simpler methods taught in machine learning.

The problem with identifying objects in computer vision is that variance in objects, pose, lighting, noise, mean that it is very unlikely that an image of an object that needs to be identified will be the same as examples held on file. Other than trying to match pictures exactly (pixel by pixel), it is very difficult for a machine to identify what it is looking at. This is where machine learning comes in. Machine learning algorithms can help to find a way of classifying objects and features in ways that are far simpler for a computer to understand.

To demonstrate the use of machine learning on an image classification problem I will use a neural network. I will then compare this network to other classification algorithms such as KNN and a classification tree. For this demonstration I will use a simple set of images which will give me only two classes, either the image has a car in it, or it doesn't.

2 Related work

There is much work being performed with neural networks, and the use of networks to recognize objects in images is a large are of research with many applications.

2.1 Number Plate Recognition

One application of neural networks is to recognize car number plates [4]. In the paper on using neural networks for recognizing number plates presents a system that pre-processes the numberplate image to separate out the characters and numbers, those characters are then run through a neural network with multiple layers. To train the network they used a BRLS learning algorithm.

BRLS (Block Recursive LS algorithm) uses an "iterative learning procedure which solves an overdetermined linear system of equations for each layer of the network" [3], which has allowed the paper to gain high levels of convergence during training on characters from the English language.

The results of the network allowed the researchers to obtain an recognition rate of 90%. The paper speculates that the majority of rejections came from dusty number plates, which were equally as hard to be read by a human.

2.2 Recognizing Vehicle Types

Some related work to my project is the use of a neural network to recognize a type of vehicle [2]. Applications of the work produced in this paper applied to new advancements in automatic Toll systems.

Before using the images of vehicles they were pre-processed to extract the boundaries of the vehicles outline. The images were then used in a method "combining fuzzy c-means clustering and BP neural network".

The results produced by the paper showed that the network was successfully able to classify a set of test vehicles into one of 3 possible categories, "car","bus", and "Freight Car".

2.3 Recognizing Hand Written digits

Neural networks can also be used for other recognition problems. Another such problem that is being solved by neural networks is allowing computers to recognize handwritten digits [5]. The applications of this work can be used in industry for workplaces such as banks. Banks receive many cheques every day, and have a need to read all of those cheques to identify the amount of money to be transferred. If a computer can do this It would save the company much time and money.

The paper looks at three different activation functions for the networks neurons, including the sigmoid function, the sinusoidal function, and the periodic function.

The results of the paper showed that for the problem of recognizing hand written digits, that a sinusoidal function gave the best results with both the training and test data.

2.4 Conclusion

The three papers that I have looked at show that the use of neural networks for recognition in images is a very promising field of research. In all three examples the networks were successfully able to predict the result of their input. Paper 2 [2] is very applicable to me as it takes what I am attempting a step further to classify different types of cars.

Paper three [5] presents several activation functions which could be useful to me, and while the sinusoidal function was very successful at it's task, I would like to use the traditional sigmoid function still, as it would require further testing on how applicable these functions are to car recognition.

3 Ethical Implications

With any research there are ethics to consider. When classifying images there are multiple possible ethicals situations to consider. The first would be the dataset, and whether the dataset has been obtained lawfully. For my dataset I have used the UIUC cars dataset [1], to collect that dataset the owners would have needed to collect permission from the owners of the cars in the images used.

The other situation to consider would be the use of the trained classifiers. For the purpose of this project I am just testing the effectiveness of neural networks for object recognition, but for other research the classifiers could be used for a multitude of purposes, including applications in security for automated systems.

Figure 1: A sample of images from the dataset.

4 Data Set

The dataset that I will be using for this experimentation is a car dataset, consisting if images containing cars and no cars [1]. The dataset consists of 1050 training images and 208 test images. The test images consist of a variety of different scales and pixel dimensions, so for the simplicity of this experiment I will be using a subset of the training data, so that the dimensions of the data stay the same.

The training data consists of 550 images containing cars, and 500 images without cars. I will split this dataset to 1000 training images, and 50 test images. The test images will be randomly selected by randomizing the order of the dataset and selecting the last 50 images.

The figure above shows a sample of 3 images with cars, and 3 images without cars which have been taken from the dataset that I will be using.

Each image is 100 pixels by 40 pixels, which gives me 4000 pixels per image. For the purpose of this experiment I will use all the pixels as features for classification, to prevent any bias from any feature selection that I could be applied to the dataset.

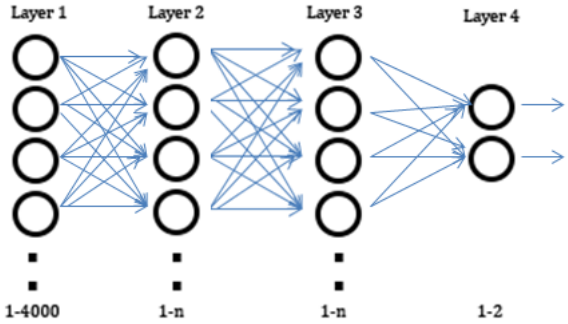
To use the dataset I have read in all the images individually into a matrix, reshaped the matrix into a vector of size 1x4000, and then combined all the images to be one large matrix. At the same time the target for all of the images are also read into an matrix of 2 columns, the first column representing a car in the image and the second representing no car. A 1 is placed in the column which is true.

5 Experiment Setup

5.1 Neural Network

The neural network that I will be using for this experiment I have created myself using object oriented programming and matrix multiplication. This has allowed

Figure 2: A representation of my network.



me to get a greater understanding of what is happening with the network when it is being trained.

The neural network is set up with a number of hidden layers and an output layer. The output layer will contain 2 output neurons, this is because we have two possible classes that we are looking for, either a car or no car. The hidden layers will contain a number of neurons. The amount of hidden layers I will use is 2 as any more would require the use of pre-training to initialize the weights, and any less means there is very few connections to be able to deal with the amount of inputs.

I will attempt several different experiments with different amounts of neurons in the hidden layers to hopefully find a reliable network for predicting with. I will attempt 5 different networks to compare the results. the table below shows the layouts of the network that I will experiment with.

Table 1: Planned network layout.

Network	Neurons	Learning Rate	Bias
1	[60,60]	0.2	0
2	[50,10]	0.5	0
3	[20,20]	1	0
4	[120,120]	1	0
5	[50,50]	0.8	1

Figure 2 shows a representation of the neural network that I will be generating. Layer 1 is the input layer which is 4000 neurons in size, layer 2 and 3 are the hidden layers, whose size is determined in Table 2. The last layer (Layer 4) is 2 neurons in size as we only have two classes.

The below pseudo code (Algorithm 1) shows how the program will run the

network for training. The program will loop through all of the examples in the dataset. On each loop it will forward propagate the image, and then back propagate the error to "train" the network. The code will keep iterating the training set until a minimum has been found, or a hard limit on the iterations has been hit.

```
while Minimum Not Found do
    for Each image in dataset do
        FeedForward(image);
        BackPropogate(target);
    end
end
```

Algorithm 1: Neural Network basic pseudo code

During the forward propagation of the network the neurons will multiply the inputs by a list of weights stored. After this the neuron will use a sigmoid activation function on the sum of the inputs. The weights will be updated during back propagation by using the output error and feeding this back through the network to obtain the neurons individual error, which can then be use to update its list of weights.

To predict an image with the network we can simply feed forward the image into the network and use the higher of the two output classes to predict what the result of the image should be.

5.2 KNN

To set up the KNN classification I used the built in Matlab function "fitcknn" to build a classification model that I could use to make prediction. As with the Neural Network I will use 1000 training images and 50 test images to verify the accuracy of the classification model.

To train my dataset with the classification function I needed to format my targets slightly different to the way they were formatted for the neural network. Instead of two columns of 1's and 0's, the KNN function took data as a single column with the class name for the data in the dataset.

After some testing I found that the best results came from using 3 nearest neighbors for the classification.

5.3 Classification Tree

Similarly to the KNN classification, I will be using the built in classification tree function "fitctree". I will also be using the same amount of training and test data to ensure the comparison is fair.

I will use the default settings that Matlab sets for training the classification tree, and the dataset will stay in the same format that was used for the KNN training.

6 Results

6.1 Neural Networks

Out of the 5 neural networks that I ran, a number of them were able to produce some reliable results, whereas some of the other networks, which often found their minimum earlier, gave unreliable classifiers.

The more reliable networks took a long time to run as they eventually became slower and slower at reducing their validation error. Due to constraints on time, and the large number if epoch's that the networks were reaching, many of them were ended early.

Table 2 shows the results of all 5 network configurations that I ran. For each network I ran both the training set of images and the test set of images through the networks predict function to gain results.

Table 2: Neural Network Results.

Network	Training Set Correct	Test/Validation Set Correct
1	93.7%	92%
2	81.8%	81%
3	80%	76%
4	56.9%	52%
5	52.8%	50%

The results of Table 2 show that neural networks can be both good and bad classifiers based on how well they are configured. The last two networks configured had a success rate of just over 50 percent, this makes them practically random. But the top 3 networks had a better result from their training, with the first network getting 92 percent of the test set correct.

There is the possibility that the error of the top three networks could continue to go down over time, as their current state is from tests that were ended prematurely. There is also the possibility better results could be obtained with a network that has been refined to an optimal amount of connections.

Attached in Appendix B are screen shots of the graphs plotted for each neural network during training. You can see from Figure 4, 5, and 6 that the networks with the highest success in training had a nice slow curved decent in

validation error, whereas Figure 7, and 8 show the networks which were incorrectly configured and gave bad results, failed to give a smooth graph, instead zig zagging up and down irrationally.

Network 1 was the most reliable classifier out of the 5 networks, The network was ended early after 298424 epochs.

6.2 KNN

The KNN classifier was able to be trained in a fraction of the time it took to train a neural network. A classifier trained on 1000 samples from the training set was able to classify 49 our of 50 test samples correctly (98%), and 973 out of 1000 training samples correctly (97.3%).

6.3 Classification Tree

The Classification Tree was also able to be successfully trained to recognize cars in a fraction of the time of the neural networks. Using 1000 training samples the classifier identified 41 out of 50 test images correctly (81%), and identified 993 out of 1000 of the training images (99.3%).

7 Conclusions

To conclude my results I found that a neural network can be an effective tool for classifying images of objects (cars) into classes. The best network performed with 92% accuracy which is considered good for computer vision.

KNN was able to outperform the neural network by betting 98% of the test set correct, making it more effective than the network. It was also less time consuming to train than a neural network. There are many possible reasons that the KNN classifier outperformed the neural network, and provided a smaller or more complex dataset along with a better network configuration the network may still be able to outperform KNN.

The Classification Tree classifier was less effective at identifying the Test set than the neural network, with only 81% effectiveness compared to 92% with the neural network. The Classification Tree was able to recognize the training set better than any of the other classifiers, this could suggest that the classifier started to over-fit the training data.

In conclusion I would still use a neural network for solving object recognition problems, as theoretically they can deal with lager and more complex sets than KNN. Although for just recognizing cars from the side on view, KNN proved to be the best method for this.

Table 3: Classifier Comparison.

Classifier	Correct Test Set	Correct Training Set
Neural Network	92%	93.7 %
KNN	98%	97.3%
Classification Tree	81%	99.3%

For an extra test on the training algorithms I took a photo of my own car from the side, after formatting the image to the correct size and gray scale, I tested it against my best neural network, the KNN classifier, and the Classification Tree. Figure 3 shows the picture of my car that I will run through the classifiers, unlike the datasets used in training and classification, this car is more modern and as such a slightly different shape.

Figure 3: Modern car not from dataset

Both KNN, and the Classification Tree were able to successfully identify my car as an car. Unfortunately the neural network incorrectly classifier my image as having no car in it. This could be because my car looks slightly different to the ones in the dataset, and shows that neural networks can be more sensitive to new unseen data, another possibility for the neural network not classifying the car could be because a different camera was used, the car could have a different level of light and gray scale.

7.1 Further Work

To further the work on this problem I would pre-process the images being used to extract features that would reduce the amount of input neurons. This would not only make the network train faster, it could also make it more reliable.

I would also like to look further into methods of finding the optimal amount of neurons, layers, and weights to provide an optimal solution on the network that is also capable of training quicker.

the use of unsupervised pre-training could also provide a good improvement to the network by initializing the weights more accurately, rather than initializing them at random.

References

[1] S. Agarwal and D. Roth, "Learning a sparse representation for object detection," in *Computer Vision ECCV 2002*, ser. Lecture Notes in Computer Science, A. Heyden, G. Sparr, M. Nielsen, and P. Johansen, Eds. Springer Berlin Heidelberg, 2002, vol. 2353, pp. 113–127. [Online]. Available: http://dx.doi.org/10.1007/3-540-47979-1_8

[2] X. bo Zhang and L. Jiang, "Vehicle types recognition based on neural network," in *Computational Intelligence and Natural Computing, 2009. CINC '09. International Conference on*, vol. 1, June 2009, pp. 3–6.

[3] E. Di Claudio, R. Parisi, and G. Orlandi, "Application of the block recursive least squares algorithm to adaptive neural beamforming," in *Neural Networks for Signal Processing [1997] VII. Proceedings of the 1997 IEEE Workshop*, Sep 1997, pp. 560–567.

[4] R. Parisi, E. Di Claudio, G. Lucarelli, and G. Orlandi, "Car plate recognition by neural networks and image processing," in *Circuits and Systems, 1998. ISCAS '98. Proceedings of the 1998 IEEE International Symposium on*, vol. 3, May 1998, pp. 195–198 vol.3.

[5] K.-W. Wong, C.-S. Leung, and S.-J. Chang, "Handwritten digit recognition using multilayer feedforward neural networks with periodic and monotonic activation functions," in *Pattern Recognition, 2002. Proceedings. 16th International Conference on*, vol. 3, 2002, pp. 106–109 vol.3.

Appendix A Code

You can obtain a copy of the code used for my neural networks, KNN, and Classification Tree, from GitHub at the following location : https://github.com/ejbarrow/COV-MachineLearningNeuralNetwork

Appendix B Network Graphs

Figure 4: Neural Network 1

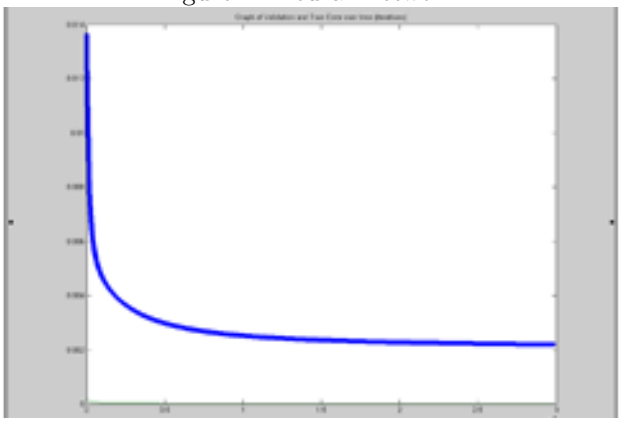


Figure 5: Neural Network 2

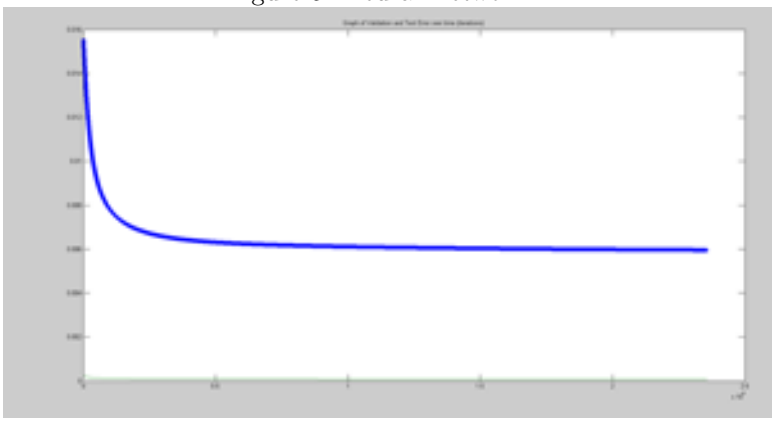


Figure 6: Neural Network 3

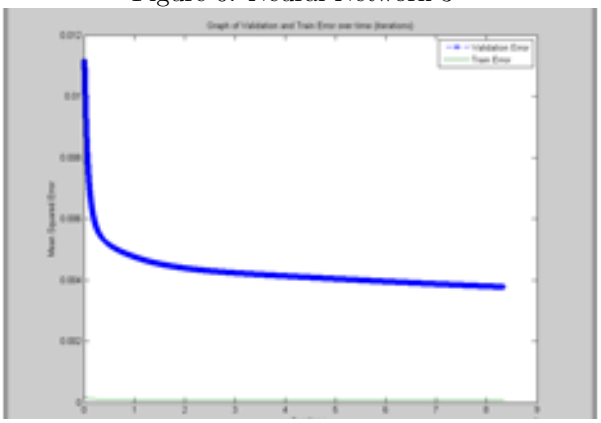


Figure 7: Neural Network 4

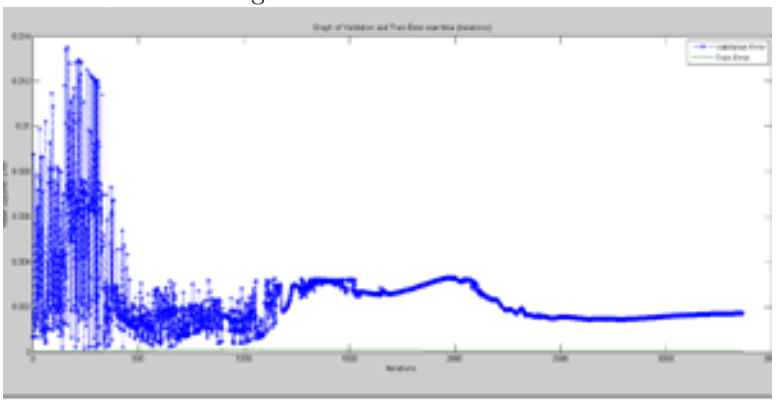


Figure 8: Neural Network 5