Deep Learning

Assignment 1

Msc in Data Science

2017-2018

Vasiliki Konstantopoulou

Introduction

In this assignments we try perform image classification using Feed Forward Neural Networks.

The dataset we use is cifar 10 containing 60000 color images which could be classified in ten categories, airplane, automobile, bird, cat, deer, dog, frog, horse, ship, and truck depending on what they picture.

To perform the classification with the higher accuracy we try many models and parameter tuning described below.

Preprocessing

In order to experiment with dataset, we perform some normalisation methods on it so we can discover which works better with our network.

The methods examined are MinMax normalisation, normalisation with the maximum value of the dataset (in this dataset 255, the white color) and normalisation with l2 norm.

The table below shows the results delivered with a simple neural network with 1 input , 2 hidden with 30 and 20 neurons respectively working with Relu and Sigmoid activation functions and 1 output layers running 10 epochs, with Adam optimizer, default parameters using categorical cross entropy loss function. The evaluation accuracy score is selected as the measurement medium.

Normalisation Method	Accuracy
Original	0.1
MinMax	0.3964
Max value	0.3941
z-score	0.4657

If we perform the same test with the exact same parameters but choosing SGD optimiser with the default parameters we are going to end up with the table above.

Normalisation Method	Accuracy
Original	0.1
MinMax	0.4049
Max value	0.3937
z-score	0.4709

As we can assume from the results the appropriate normalisation for the image data is z-score.

Lastly, augmentation has been performed as a preprocessing method but the results where too low with the accuracy value lie in 0.4378 with z-score normalised data.

Model Selection

In order to select the best neural network model for the classification, it was very useful to experiment with some parameters. Some of them are the scale of the network, the activation and loss functions and the optimiser.

To begin with we are going to experiment with various optimizers and their default parameters to discover which one fits best with our dataset. We use the same neural network as in the normalisation experiments.

Optimiser	Accuracy
SGD	0.4709
RMSprop	0.4559
Adam	0.4607
Adadelta	0.4555
Adagrad	0.4465
Nadam	0.4251

Consequently, the most appropriate optimisers for our image classification task would be SGD and Adam. To further examine this assumption we would select Adam from the adaptive optimisers family and SGD.

The next step is to perform some trials experimenting with hyper parameters.

Optimiser	Batch size	Epochs	Learning Rate	Accuracy
SGD	10	10	0.01	0.4726
SGD	100	10	0.01	0.4328
SGD	10000	10	0.01	0.1749
SGD	10	50	0.01	0.4743
SGD	10	100	0.01	0.4626
SGD	10	50	0.1	0.3821
SGD	10	50	0.001	0.4772
SGD	10	50	1	0.1362

Let's revise the experiments for Adam.

Optimiser	Batch size	Epochs	Learning Rate	Accuracy
Adam	10	10	0.001	0.4301
Adam	100	10	0.001	0.4638
Adam	10000	10	0.001	0.3543
Adam	100	50	0.001	0.4714
Adam	100	100	0.01	0.3254
Adam	100	50	0.1	0.1581
Adam	100	50	0.001	0.4726
Adam	100	10	1	0.128

In our experiment both models are trained better with 50 epochs. Generally, Adam finds the best accuracy value during less epochs than SGD but SGD gives better results in the end. In addition the learning rate which both models work better with is the default learning rate. We expected higher values as learning rate to result in lower accuracy which is happening in our trials. Small difference appears in accuracy from the

batch size changes in Adam model in contrast with SGD model which works better with smaller batches.

We now select the most accurate models and experiment with neural networks' parameters as shown below.

Optimiser	Width	Depth	Epochs	Activation	Accuracy
SGD	512 - 128	4	50	Relu	0.5353
SGD	512 - 256 - 128 - 64 - 32	7	50	Relu	0.5146
SGD	128 - 64 - 32	5	50	Relu	0.4996
SGD	128 - 64	4	50	Sigmoid	0.4435
Adam	500 - 400	4	50	Relu	0.5125
Adam	512 - 128	4	50	Relu	0.4975
SGD	512 - 128	5 (with Dropout layer)	10	Relu	0.5313

The results show that SGD with 2 hidden layers (512 and 128 neurons) performs better than the others. In addition the same model including the Dropout layer has many potentials to hit the score after more epochs.

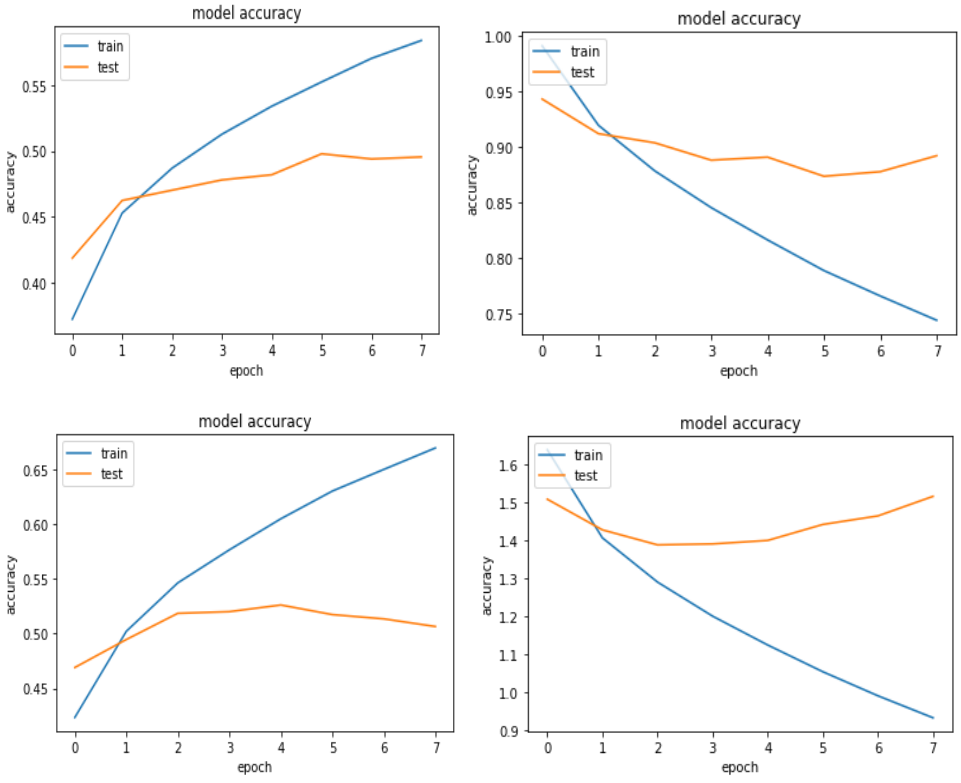
The final step is to check different loss functions with the best of the above model which includes SGD optimiser working for 40 epochs with 4 layers, 1 input, 2 dense layers with 512 and 128 neurons respectively and Relu activation function.

Optimiser	Loss	Epochs	Accuracy
SGD	MSE (*with callback ReduceLROnPlateau() with patience at 5 epochs)	40	0.5166
SGD	categorical_crossentropy (*with callback ReduceLROnPlateau())	40	0.555
SGD	categorical_hinge (*with callback ReduceLROnPlateau())	40	0.5378

It is obvious that SGD model running 40 epochs with 2 hidden layers including 512 and 128 neurons respectively with callback function ReduceLROnPlateau(patience = 5) scores the best accuracy result. This shows that a very deep network is not suitable for our dataset and the width must be properly selected. SGD works better with more epochs than other optimisers and if the hardware was not limited the results could be better. The ReduceLROnPlateau function reduces the learning rate value after five epochs that validation loss is not improving and allows the model to find the gradient.

Finally the below diagrams show the accuracy for the SGD model with categorical_cross entropy and SGD with categorical hinge.

In the first line we can see the accuracy and loss for the categorical hinge function respectively and the cross entropy model in the second line.



As we see the categorical hinge function offers low difference between train and test accuracy in contrast with the cross entropy model where the difference is obviously bigger.