Deep Learning - CS551

Course project

Image Classification

Ashutosh Mishra
1601CS06

April 30, 2020

1 Introduction

Image classification is a very common problem in current times. One of the most successful machine learning paradigms to conquer this problem is to use convolutional neural networks. This project aims to classify a variety of objects and reviews the performance of different models. The project is written in **python**, with the help of the **keras** library using **tensorflow** backend.

2 Dataset

The CIFAR-10 dataset is used. This dataset consists of 60,000 32x32 colour images in 10 classes, with 6,000 images per class. There are 50,000 training images and 10,000 test images.

3 Models

3.1 A simple model(Baseline model)

The first model consists of three convolutional layers, each followed by a max pooling layer. There are **32**, **64** and **128** **3x3** filters in each layer respectively. The max pool filter is **2x2**. There are two dense layers consisting of **256** and **128** neurons respectively. The final layer is a **10** way softmax function.

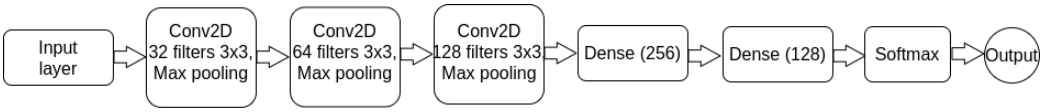


Figure 1: First model

3.2 Adding dropout

The second model adds a dropout layer after each convolutional layer.

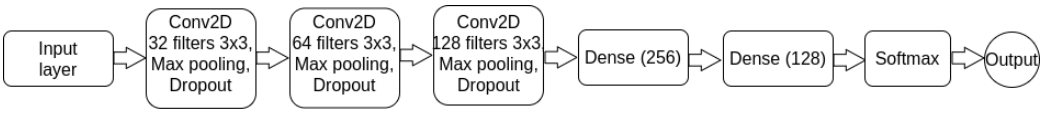


Figure 2: Second model

3.3 Adding another layer

The third model adds another convolutional, dropout and max pooling layer after the last conv layer. This layer has **256 3x3** filters. The dense layers have been updated to have 1024 and 512 neurons respectively.

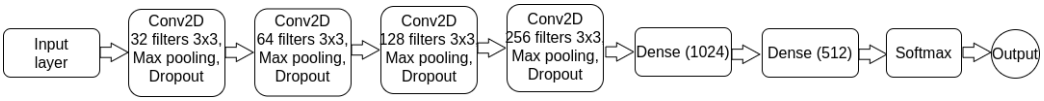


Figure 3: Third model

3.4 Adding L1 and L2 regularization

The final model consists of adding a bias regularizer to each layer. The L1 and L2 biases are both set equal to **0.01**

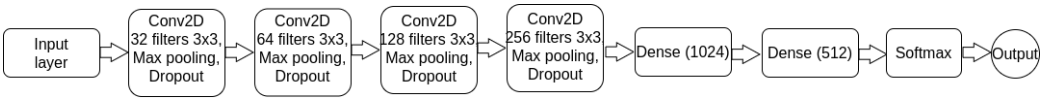


Figure 4: Final regularized model

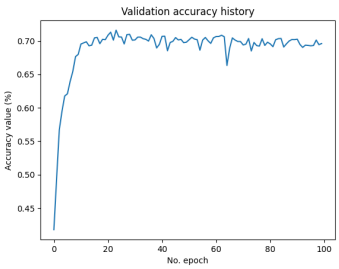
4 Experimental Results

I have developed all the models using 80% of the data as training set and rest 20% has been used as validation set. The obtained results have been reported in Table 1.

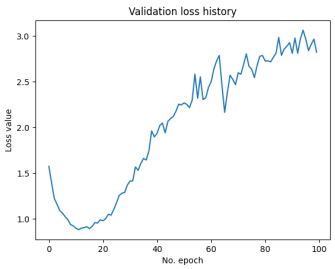
Model	Performance
Model 1	0.70
Model 2	0.74
Model 3	0.76
Model 4	0.78

Table 1: Accuracy of models on validation set

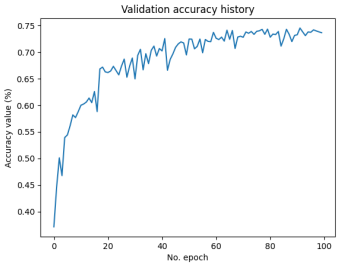
In order to show that the developed models are performing significantly, the following graphs of *validation accuracy* and *validation loss* can be viewed in Figure 5. Performance of each of the models has been measured in terms of *accuracy*.



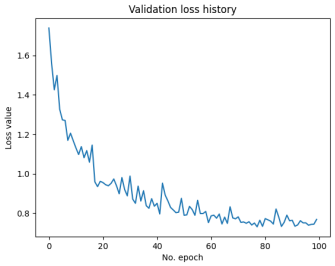
(a) Model 1 Accuracy



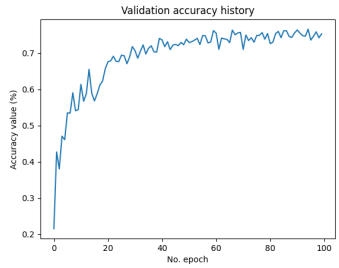
(b) Model 1 Loss



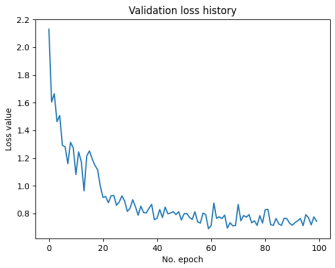
(c) Model 2 Accuracy



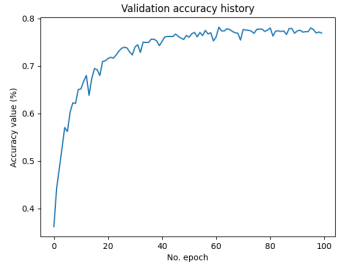
(d) Model 2 Loss



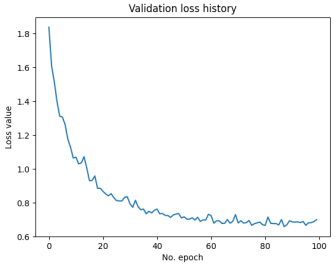
(e) Model 3 Accuracy



(f) Model 3 Loss



(g) Model 4 Accuracy



(h) Model 4 Loss

Figure 5: Model performances

4.1 Training Details

Each of the models has been trained on optimised set of hyperparameters as shown in Table 2.

Name of hyperparameter	Value
Batch size	256
No. of epoch	100
Optimizer	Adam
Loss function	Sparse categorical crossentropy

Table 2: Optimised set of Hyperparameters

5 Conclusion

The baseline model's loss history suggested that the model was overfitting. The second model corrected this by adding a dropout layer after each convolutional layer. The third model further improved the performance by adding another convolution layer. The final model used L1 and L2 regularization to achieve a smoother learning curve.