CS-898 AB - Image Analysis and Comp Vision
Final Exam

Chakradhar Reddy Donuri
E949F496

Q) Generate two different variants of the CNN architecture, used for training CIFAR-10 dataset as a part of the lab exercise, and report the overall training and test accuracies obtained for both the variants. Please explain how the variants are generated ?

- **The variants of the CNN architecture may be generated by varying any of the following: number of kernels in a convolutional layer, varying the batch size, adding more layers, changing the learning rate, and the optimizer**

Answer)

In the below code I've considered batch size as 30

opt = keras.optimizers.RMSprop(learning_rate=0.0001, decay=1e-6)

Model: "sequential_1"

Layer (type)	Output Shape	Param #
conv2d_1 (Conv2D)	(None, 32, 32, 32)	896
activation_1 (Activation)	(None, 32, 32, 32)	0
conv2d_2 (Conv2D)	(None, 30, 30, 32)	9248
activation_2 (Activation)	(None, 30, 30, 32)	0
max_pooling2d_1 (MaxPooling2	(None, 15, 15, 32)	0
dropout_1 (Dropout)	(None, 15, 15, 32)	0
conv2d_3 (Conv2D)	(None, 15, 15, 64)	18496
activation_3 (Activation)	(None, 15, 15, 64)	0
conv2d_4 (Conv2D)	(None, 13, 13, 64)	36928
activation_4 (Activation)	(None, 13, 13, 64)	0
max_pooling2d_2 (MaxPooling2	(None, 6, 6, 64)	0

dropout_2 (Dropout) (None, 6, 6, 64) 0

_____

flatten_1 (Flatten) (None, 2304) 0

_____

dense_1 (Dense) (None, 512) 1180160

_____

activation_5 (Activation) (None, 512) 0

_____

dropout_3 (Dropout) (None, 512) 0

_____

dense_2 (Dense) (None, 10) 5130

_____

activation_6 (Activation) (None, 10) 0

==========

Total params: 1,250,858

Trainable params: 1,250,858

Non-trainable params: 0

---

opt2 = keras.optimizers.RMSprop(learning_rate=0.001, decay=1e-7)

Model: "sequential_2"

_____

Layer (type) Output Shape Param #

==========

conv2d_5 (Conv2D) (None, 32, 32, 32) 896

_____

activation_7 (Activation) (None, 32, 32, 32) 0

_____

conv2d_6 (Conv2D) (None, 30, 30, 32) 9248

_____

activation_8 (Activation) (None, 30, 30, 32) 0

_____

max_pooling2d_3 (MaxPooling2 (None, 15, 15, 32) 0

_____

dropout_4 (Dropout) (None, 15, 15, 32) 0

_____

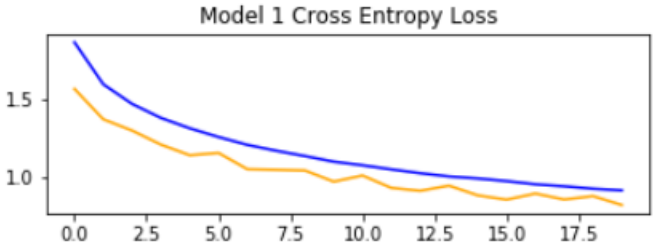
conv2d_7 (Conv2D) (None, 15, 15, 64) 18496

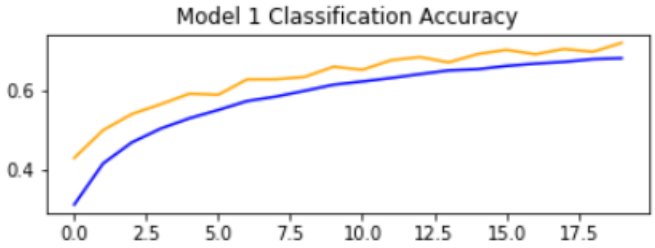
_____

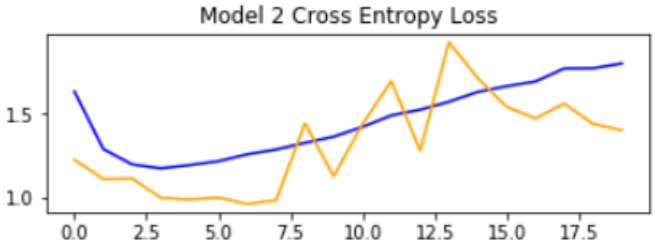
activation_9 (Activation) (None, 15, 15, 64) 0

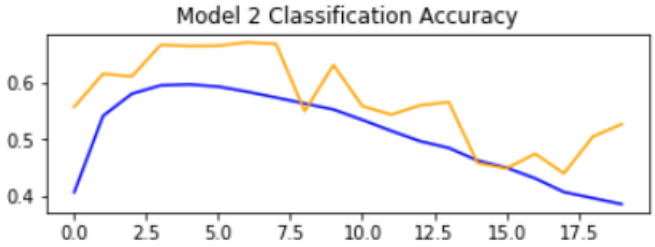
_____

```
conv2d_8 (Conv2D)           (None, 13, 13, 64)       36928
_________________________________________________________________
activation_10 (Activation)   (None, 13, 13, 64)       0
_________________________________________________________________
max_pooling2d_4 (MaxPooling2 (None, 6, 6, 64)         0
_________________________________________________________________
dropout_5 (Dropout)          (None, 6, 6, 64)         0
_________________________________________________________________
flatten_2 (Flatten)          (None, 2304)             0
_________________________________________________________________
dense_3 (Dense)              (None, 512)              1180160
_________________________________________________________________
activation_11 (Activation)   (None, 512)              0
_________________________________________________________________
dropout_6 (Dropout)          (None, 512)              0
_________________________________________________________________
dense_4 (Dense)              (None, 10)               5130
_________________________________________________________________
activation_12 (Activation)   (None, 10)               0
=================================================================
Total params: 1,250,858
Trainable params: 1,250,858
Non-trainable params: 0
```









Saved trained model at C:\Users\chakr\Desktop\Image A\saved_models\keras_cifar10_trained_model.h5
10000/10000 [==============================] - 12s 1ms/step
Test loss: 0.8178882014274598
Test accuracy: 0.7182000279426575
Saved trained model at C:\Users\chakr\Desktop\Image A\saved_models2\keras_cifar10_trained_model.h5
10000/10000 [==============================] - 12s 1ms/step
Test loss: 1.4026706340789794
Test accuracy: 0.5271999835968018

Q2. What is meant by regularization of the deep learning model? What are the ways to regularize the deep learning model?

Answer)

Regularization: Regularization is any modification we make to a learning algorithm that is intended to reduce its generalization error but not its training error.

Some of the techniques we use to regularize the deep learning model they are:

L1 & L2 regularization

1. These are the most common types of regularization.
2. These update the general cost function by adding another term known as the regularization term.

   Cost function = Loss (say, binary cross entropy) + Regularization term
3. Due to the addition of this regularization term, the values of weight matrices decrease because it assumes that a neural network with smaller weight matrices leads to simpler models. Therefore, it will also reduce overfitting to quite an extent.

However, this regularization term differs in L1 and L2

We have: L2

$$Cost\,function = Loss + \frac{\lambda}{2m} * \sum ||w||^2$$

We have: L1

$$Cost\,function = Loss + \frac{\lambda}{2m} * \sum ||w||$$

Here, lambda is the regularization parameter. It is the hyperparameter whose value is optimized for better results. L2 regularization is also known as *weight decay* as it forces the weights to decay towards zero (but not exactly zero).

In this, we penalize the absolute value of the weights. Unlike L2, the weights may be reduced to zero here. Hence, it is very useful when we are trying to compress our model. Otherwise, we usually prefer L2 over it.

Data Augmentation:

The simplest way to reduce overfitting is to increase the size of the training data. In machine learning, we were not able to increase the size of training data as the labeled data was too costly. But, now let's consider we are dealing with images. In this case, there are a few ways of increasing the size of the training data – rotating the image, flipping, scaling, shifting, etc. In the below image, some transformation has been done on the handwritten digits dataset.

This technique is known as data augmentation. This usually provides a big leap in improving the accuracy of the model. It can be considered as a mandatory trick in order to improve our predictions.

Early stopping:

Early stopping is a kind of cross-validation strategy where we keep one part of the training set as the validation set. When we see that the performance on the validation set is getting worse, we immediately stop the training on the model. This is known as early stopping.

In the above image, we will stop training at the dotted line since after that our model will start overfitting on the training data.