Introduction to Machine Learning

ML Semester Project Report

Group Members:

Rehan Ali (1802008)

Sharukh khan (1802029)

Slman Ali (1802010)

Course Instructor:

Dr. Malik Jahan Khan

Abstract

In this problem we have to perform classification. Our objective is to classify the healthy and bacterial peeper leaves images. We have the provided dataset at kaggle .In order to achieve this goal we trained our models on two different architectures one is AlexNet and other is our own classifier. At first, we have trained both model on given input data. The input data is imbalanced that's why we get some issues of overfitting and we didn't get accurate results on unseen data. To solve this issue we modify the data to make it balance in such a way that both classes get the equal or same data samples. As a result our accuracy get improved and now we are having good results on unseen data as well.

Table of Contents

Abstract 2

1. Introduction: 4

2. Classifiers: 4

2.1. Our Classifier 4

2.1.1 Preprocessing the Data: 4

2.1.2. Architecture of Our Classifier: 4

2.1.3. Finding of Our Classifier trained on Imbalanced Dataset: 5

2.1.4. Problem: 6

Solution to the Problem: 6

2.1.5. Finding of Our Classifier trained on Balanced Dataset: 7

2.2. AlexNet Classifier: 8

2.2.1. Preprocessing of the data: 8

2.2.2. AlexNet Architecture: 8

2.2.3. Findings on the base of Imbalanced Dataset: 9

2.2.4. Problem: 9

2.2.5. Finding of AlexNet Classifier trained on Balanced Dataset: 10

3. Conclusion: 11

4. Glossary 11

Bibliography 11

1. Introduction:

The nature of this project is classification problem. Our goal is to distinguish healthy peeper leaves and bacterial peeper leaves images. For this purpose we have used different classifier. As this is image classification problem so we are using AlexNet CNN architecture as AlexNet showed that deep convolutional neural network can be used for solving image classification. In addition to this AlexNet was first utilized in the public setting when it won the ImageNet Large Scale Visual Recognition Challenge (ILSSVRC 2012 contest) on classification 1.2 million images. Moreover we have also define our own classifier to solve this problem, our classifier was much simpler than the AlexNet. In the given dataset we have 997 bacterial images and 1478 healthy images.

2. Classifiers:

We have performed classification using two different classifiers.

- Our Classifier
- AlexNet Classifier

2.1. Our Classifier

In this section we will discussed in great detail of our own classifier that we made for classification of pepper leaves images.

2.1.1 Preprocessing the Data:

As the input dataset have pictures of size 256 x 256. We perform preprocessing on the images and resize them by 224 x 224. We input the preprocessed dataset to our model.

2.1.2. Architecture of Our Classifier:

When we design our own classifer we have made following architecture of our model.

- In our classifier we have used three convolution layers each with max pooling layers having filters 100,200,250 respectively
- We have used three dense layers. Two of these layers are fully connected having 2048 and 1024 neurons respectively. One layer is for output the bacterial and healthy peeper leaves images.
- To prevent the over fitting we set the dropout to 0.5.
  Architecture that we discussed can be seen in figure 1.

```
############  Sequential Model (CNN Architecture)  ############
model = Sequential()

# 1st CONV2D & MaxPooling Layer
model.add(Conv2D(100, (8, 8), input_shape=(224,224,3),activation='relu',strides=(3,3)))
model.add(MaxPooling2D(pool_size=(4, 4),strides=(2,2)))
#model.add(Dropout(0.3))

# 2nd Conv2D & MaxPooling Layer
model.add(Conv2D(200, (5, 5),activation='relu',strides=(2,2)))
model.add(MaxPooling2D(pool_size=(3, 3),strides=(2,2)))
#model.add(Dropout(0.3))


model.add(Conv2D(250, (3, 3),activation='relu',strides=(1,1)))
model.add(MaxPooling2D(pool_size=(2, 2),strides=(1,1)))
#model.add(Dropout(0.3))



# flatten data for feeding to feed forward NN
model.add(Flatten())

model.add(Dense(2048,activation='relu'))          # Dense layer of  neurons
model.add(Dropout(0.5))
model.add(Dense(1024,activation='relu'))


#final Layer
model.add(Dense(2))
model.add(Activation('softmax'))  #softmax activation fn for classification
```

Figure1: Model Architecture

2.1.3. Finding of Our Classifier trained on Imbalanced Dataset:

We have distributed the dataset in the following the way:

1. Training data 80%
2. Testing data 20%
3. Validating data 20% of training data.

When we have trained our model on this dataset we get the following results as we have set Epochs to 20. Result of Our Training can be seen in figure 2 and figure 3. Our accuracy in start during training start from 0.6 and then increased up to 1. We get 100% accuracy on training data and 100% accuracy on testing data as well. In figure 4 complete classification report can be seen in which precision, recall and f1-score is illustrated in table form. What is precision, recall and f1-score in classification are defined in Glossary section they have been calculated from specific formula which are also defined in Glossary section.

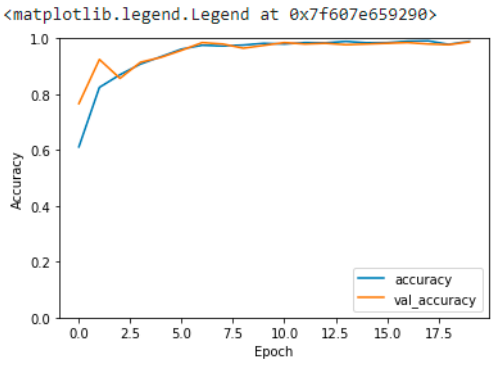

Figure 2: Accuracy during Training (Imbalanced data)

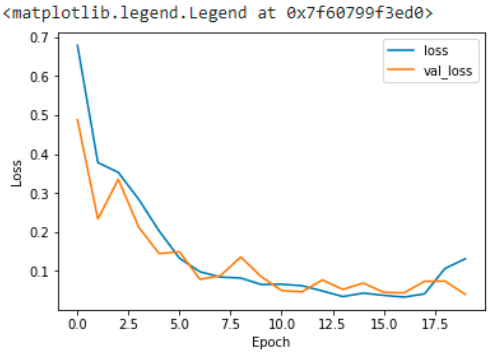

Figure 3: Loss During Training (Imbalanced Data)

2.1.4. Problem:

Our model is correctly classifying the healthy peeper leave images but unable to predict bacterial leaves which means that our model is biased. This problem is raised due to the more occurrences of the healthy peeper images about 25% more than the other class which causes the over fitting and giving the accuracy of 100% on seen data whereas less than 50% accuracy on unseen data.

Solution to the Problem:

To remove the biasness we have to modify the data to make the both classes of equal data size.

Classification Report

	precision	recall	f1-score	support
0	1.00	0.99	0.99	197
1	0.99	1.00	1.00	300
accuracy			1.00	497
macro avg	1.00	0.99	1.00	497
weighted avg	1.00	1.00	1.00	497

Figure 4: Classification Report on Testing Data (Imbalanced)

2.1.5. Finding of Our Classifier trained on Balanced Dataset:

This time we modify the data as 997 bacterial images and 1000 healthy peeper leaves images. We have distributed the data as:

1. Training data 80%
2. Testing data 20%
3. Validation data 20% of training data

When we trained the model on this dataset by the setting the epochs to 20 than we get the following results:

Accuracy up to 0.99 at training data and 0.99 on testing data as well. We have also tested the model on unseen data by taking 10 images from the internet the results were far better than previously. We try different images with different sizes and backgrounds. We get the accuracy up to 60% this time. After training we also visualize accuracy and loss as you can see in figure 5 and figure6 plot of accuracy and loss during training after each epoch.

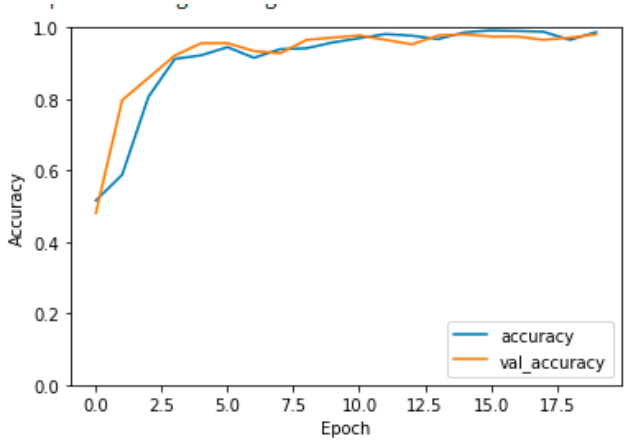

Figure 5: Accuracy during training (Balanced Data)

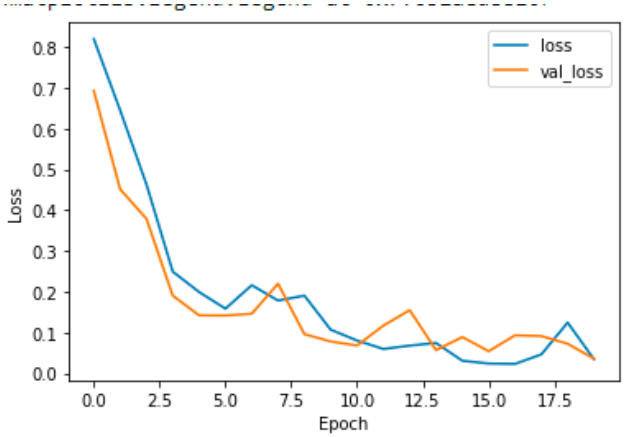

Figure 6: Loss during training (Balanced Data)

After training our model on balanced data, We made complete classification report on training and testing data. When we test our model on 400 test images following report was build shown in figure 7. Our total test images were 400 in which 210 are of bacterial class and 190 of healthy. Percision of bacterial class was 1.00 and of healthy class was 0.99 and recall is 0.99 for both classes.

Classification Report

	precision	recall	f1-score	support
0	1.00	0.99	0.99	210
1	0.99	0.99	0.99	190
accuracy			0.99	400
macro avg	0.99	0.99	0.99	400
weighted avg	0.99	0.99	0.99	400

Figure 7: Classification Report on Testing Data (Balanced)

2.2. AlexNet Classifier:

In this section we will discussed in great detail of Alexnet classifier which is the most famous deep neural network used for classification of images.

2.2.1. Preprocessing of the data:

As the input dataset have pictures of size 256 x 256. We perform preprocessing on the images and resize them by 227 x 227. We input the preprocessed dataset to our model.

2.2.2. AlexNet Architecture:

Alexnet architecture was designed in 2010 for ImageNet classification. It has some specific number of layers removal of any layer may affect the model. Alexnet model have how many layers and how many filters in each convolution layer is show below in figure 8.

In AlexNet we have used 5 convolution layers each with max pooling layers having filters 96,256,384,385,256 respectively. Three dense layers are used, two of them are fully connected having 4096 neurons by each. To prevent the over fitting we have set dropout to 0.5. The last one is output layer giving the output of bacterial and healthy peeper leaves images.

```
model = keras.models.Sequential([
    keras.layers.Conv2D(filters=96, kernel_size=(11,11), strides=(4,4), activation='relu', input_shape=(227,227,3)),
    keras.layers.BatchNormalization(),
    keras.layers.MaxPool2D(pool_size=(3,3), strides=(2,2)),
    keras.layers.Conv2D(filters=256, kernel_size=(5,5), strides=(1,1), activation='relu', padding="same"),
    keras.layers.BatchNormalization(),
    keras.layers.MaxPool2D(pool_size=(3,3), strides=(2,2)),
    keras.layers.Conv2D(filters=384, kernel_size=(3,3), strides=(1,1), activation='relu', padding="same"),
    keras.layers.BatchNormalization(),
    keras.layers.Conv2D(filters=384, kernel_size=(3,3), strides=(1,1), activation='relu', padding="same"),
    keras.layers.BatchNormalization(),
    keras.layers.Conv2D(filters=256, kernel_size=(3,3), strides=(1,1), activation='relu', padding="same"),
    keras.layers.BatchNormalization(),
    keras.layers.MaxPool2D(pool_size=(3,3), strides=(2,2)),
    keras.layers.Flatten(),
    keras.layers.Dense(4096, activation='relu'),
    keras.layers.Dropout(0.5),
    keras.layers.Dense(4096, activation='relu'),
    keras.layers.Dropout(0.5),
    keras.layers.Dense(2, activation='softmax')
])
```

Figure 8: AlexNet Architecture

2.2.3. Findings on the base of Imbalanced Dataset:

We have distributed the data as following:

- Total images: 2475
- Training data: 2000
- Testing data: 300
- Validating data: 175

When we trained our model on this dataset using above distribution by setting the epochs to 6. We get the following results:

- The accuracy of training data is 96% and the accuracy of testing data is also 96%.
- The loss on during training start from 0.19 and ends on 0.07.

2.2.4. Problem:

Our model is correctly classifying the healthy peeper leave images on unseen data but unable to predict bacterial leaves which means that our model is biased. This problem is raised due to the more occurrences of the healthy peeper images about 25% more than the other class which causes the over fitting and giving the accuracy of 100% on seen data whereas less than 50% accuracy on unseen data

Solution to the Problem:

To remove the biasness we have to modify the data to make the both classes equal.

2.2.5. Finding of AlexNet Classifier trained on Balanced Dataset:

This time we have modified the data and distributed it equally among both of the classes. Now we have 997 healthy and 997 bacterial images.

We distributed the data as following:

1. Total images: 1994
2. Training data: 1600
3. Testing data: 244
4. Validation data: 150

When we train our model on this data set by setting the epochs to 6. We get the following results:

- We are having the accuracy on training up to 0.9685 and loss is 0.0788.
- We are having accuracy on testing up to 0.96875 and loss is 0.06290.
- Now we are able to make accurate predictions on unseen data taken from the internet with different sizes and different backgrounds. The results were much better than the previous model but still it does not giving the satisfying results as it is not much able to accurately classify images correctly. We Gave 6 images to it for prediction it classified 3 images correctly and 3 were predicted wrong.

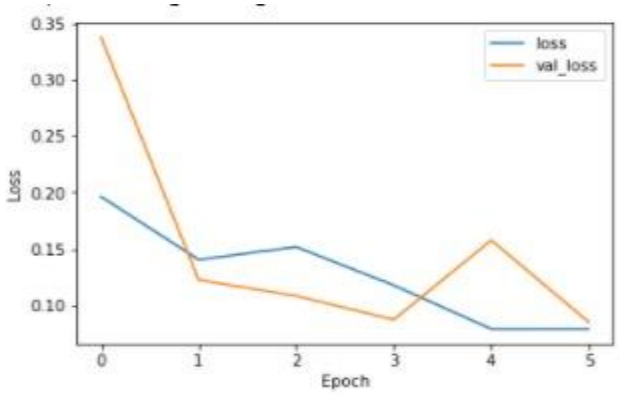

Figure 9: Loss During Training Data

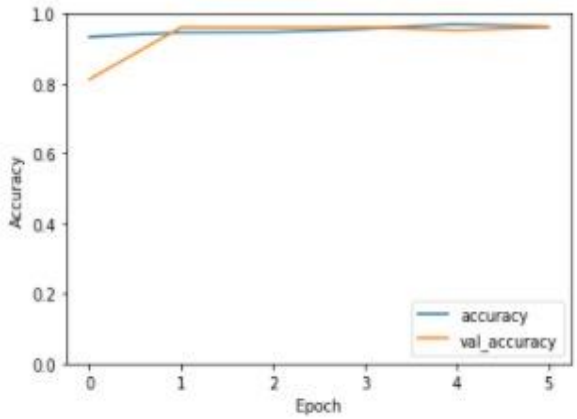

Figure 10: Accuracy During Training Data

3. Conclusion:

We noticed that our dataset is not much extensive (complex and huge data), when we have used our own classifier which is much simpler consist of 6 layers. It gives us more accurate results, 99% accuracy on training data and 99% on testing data from the Kaggle. It also give 60% accuracy on unseen data from the internet.

On the other hand AlexNet architecture is not giving good results as it was designed for large dataset (15 million images and 22000 categories). In our point of view for small dataset as we are having in this problem simple CNN of 4-5 layers can efficiently classify the classes. But for complex problems where the dataset is large like CIFAR-10 problem AlexNet is useful.

One point we also not that when data is increased alexnet works more efficiently gradually. Another one is that for small dataset few number of epochs are sufficient for classification.

4. Glossary

- **Precision**

  Precision is the ability of a classifier not to label an instance positive that is actually negative**.**

  Precision = True Positive/(True Positive + False Positive)

- **Recall**

  Recall is the ability of a classifier to find all positive instances.

  Recall = True Positive/(True Positive +False Negative)

- **F1-score**

  The F1 score is a weighted harmonic mean of precision and recall such that the best score is 1.0 and the worst is 0.0.

  F1 Score = 2*(Recall * Precision) / (Recall + Precision)

  (Kohli)

Bibliography

Kohli, S. (n.d.). *Understanding a Classification Report For Your Machine Learning Model.* Retrieved from Medium: https://medium.com/@kohlishivam5522/understanding-a-classification-report-for-your-machine-learning-model-88815e2ce397