CNN Assignment Report

1 - Executive Summary

The aim of this assignment is to build a NN model to classify landscape images into different categories based on the weather on the days when those images were captured. To achieve the aim, CNN model is used on the provided training data of 505 images in 3 categories of the weather, namely, sunny, rainy, and cloudy. By categorizing, augmenting, & resizing the training data, a successful accuracy of 73.3% is achieved on the testing data of total 45 images in the three categories.

2 - Model Building

2.1 - Data Description

Training data contains total of 505 images in the three categories, sunny, cloudy, and rainy. The distribution of the number of images in each category can be seen in the image (a).

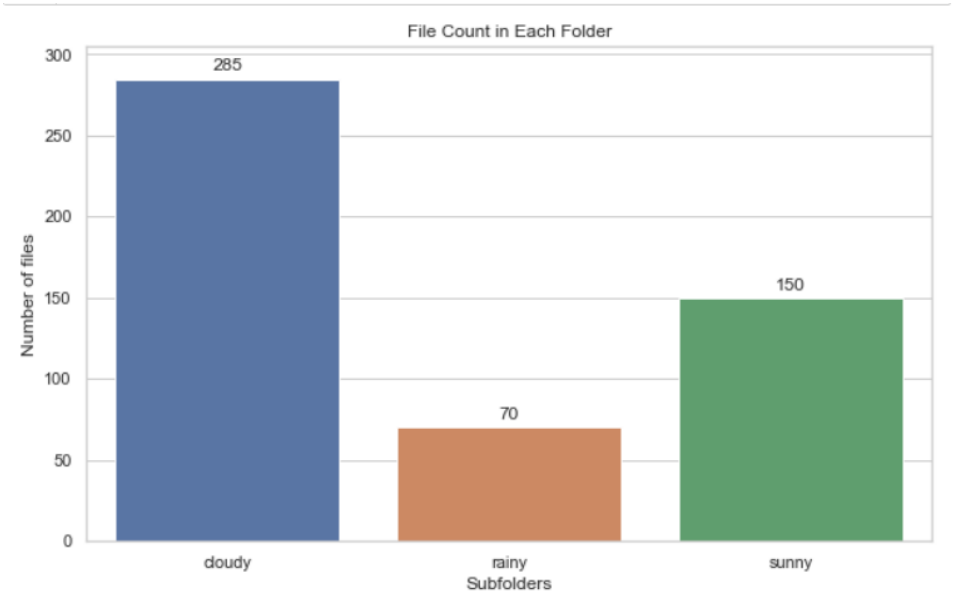

Image (a)

The testing data has 45 images, 15 images in each category.

The size of the images in training data set is varying. The smallest pixel height and width are 158 & 111.

The following is an example of images contained in the dataset chosen randomly (image (b)):

Image (b)

The following is an example of images in training dataset with their corresponding labels (image (c)):

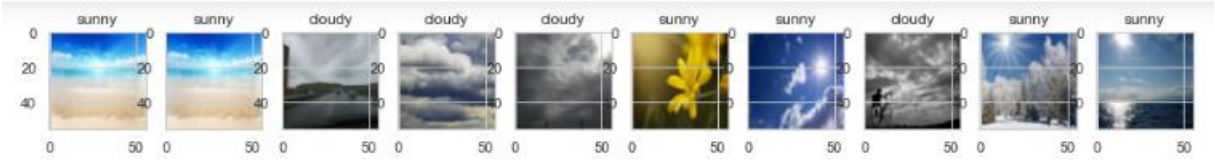

Image (c)

The following is an example of images in testing dataset with their corresponding labels (image (d)):

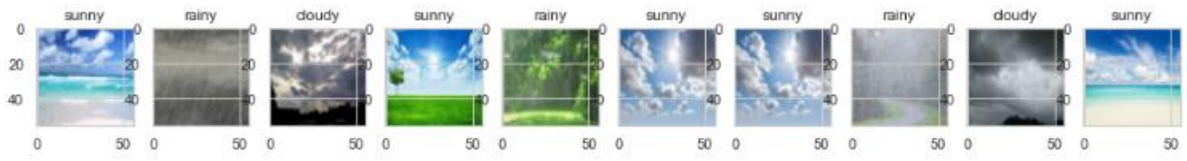

Image (d)

2.2 - Data Preparation

Following are the steps executed to prepare the data:

1. Resized all images from varying dimensions to (56, 56)
2. Converted all the images to array to be fed into CNN model
3. Rescaled the training and testing data from the range [0, 255] to [0, 1]
4. One-hot encoded the labels

2.3 - Basic Model

The simple CNN is configured with 2 n filters, (3, 3) filter size, (2, 2) pool filter size, 10 epochs, and 0.2 validation split. And it is tried 10 times.

The accuracy can be seen in the following image (e):

	Trial Number	Training Accuracy	Validation Accuracy
0	1.0	0.908416	0.247525
1	2.0	0.628713	0.306931
2	3.0	0.893564	0.178218
3	4.0	0.861386	0.277228
4	5.0	0.628713	0.306931
5	6.0	0.868812	0.257426
6	7.0	0.933168	0.287129
7	8.0	0.60396	0.306931
8	9.0	0.955446	0.287129
9	10.0	0.846535	0.277228
10	Mean	0.812871	0.273267
11	Median	0.861386	0.277228
12	Max	0.955446	0.306931

Image (e)

Although the simple model has performed decently on the training data, but it achieves poor accuracy in validation data.

2.4 - Challenges

Following are the list of challenges due to which the model performed poorly on validation data:

1. There was a class imbalance among the categories (image (f)). The model has to be trained equally for all categories to avoid overfitting.

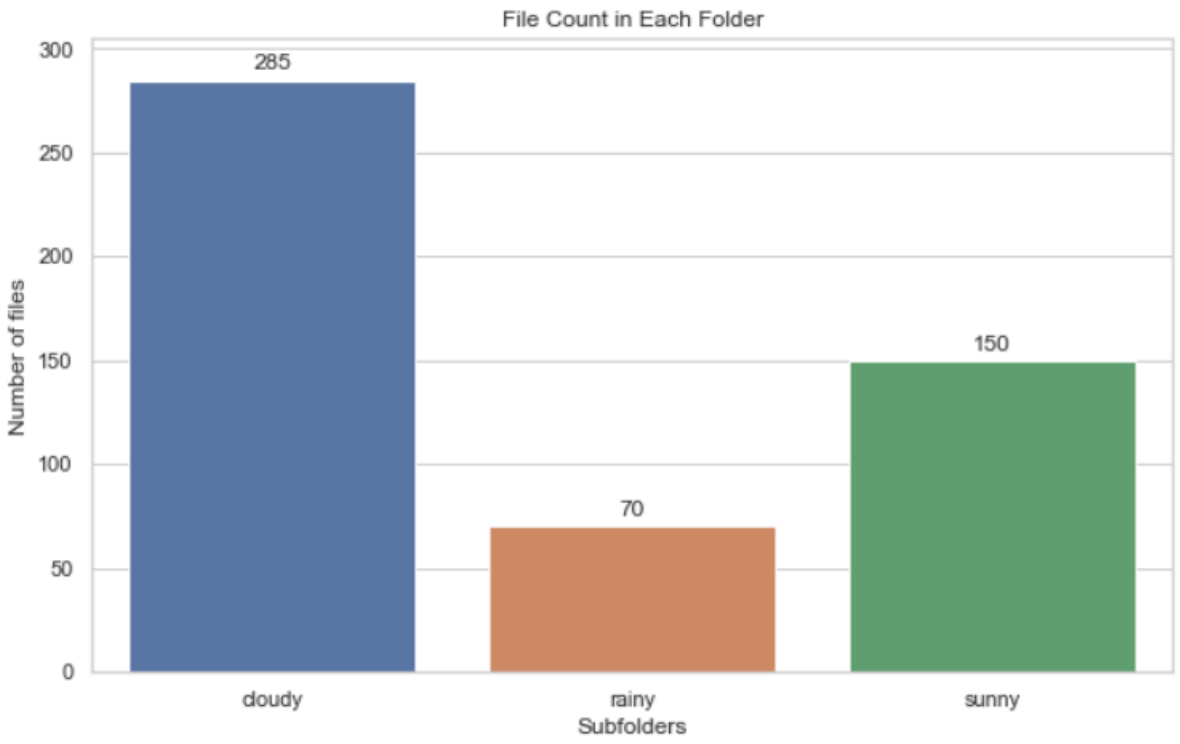

Image (f)

2. Even though the number images of cloudy weather are highest among the three categories, it is still not sufficient for training the model adequately.

3 - Data Augmentation and Model Optimization

3.1 - Data Augmentation

To increase the images artificially, data augmentation is used. The three methods incorporated for augmenting images are rotating 90 degrees, flipping up, and flipping down, chosen randomly to random images.

The training set is augmented by the factor of 2, i.e., the number of images is increased by 2 times the number of images in the category with highest number of images.

Following is the representation of class balance, before (image (g)) and after (image (h)) the data augmentation:

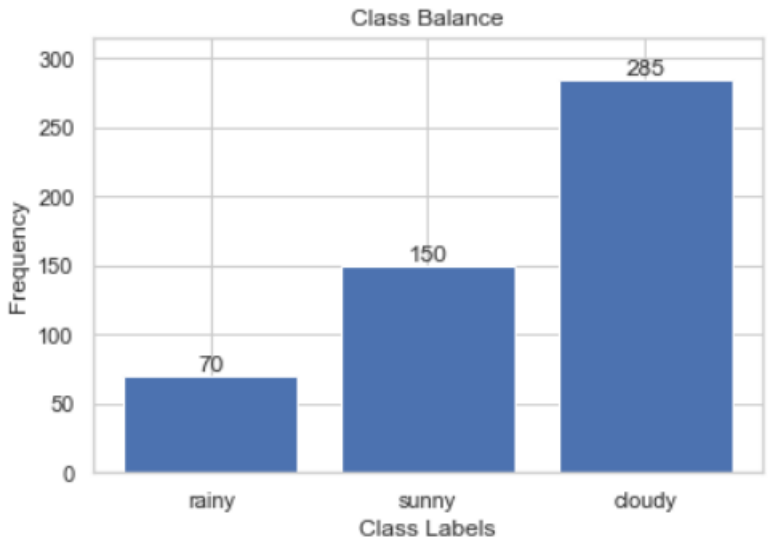

Image (g)

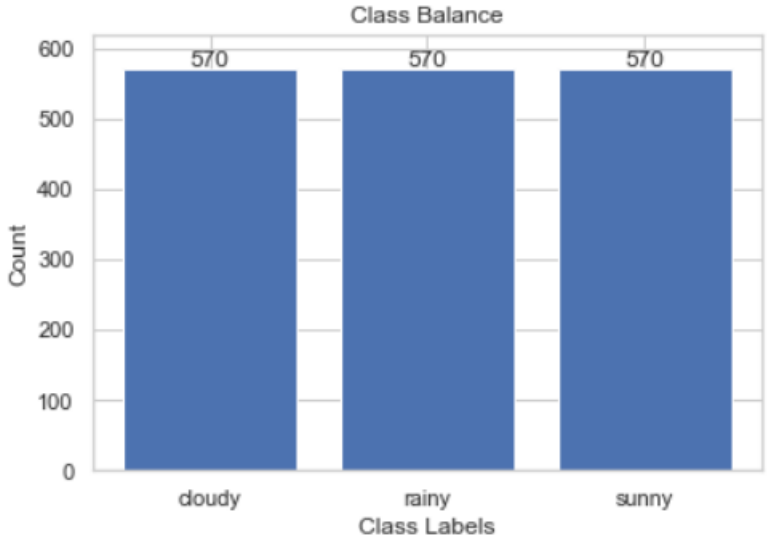

Image (h)

3.2 - Basic Model

A simple CNN model is tried again on the augmented data with the same configuration as before.

The following is the resultant accuracy (image (i)):

	Trial Number	Training Accuracy	Validation Accuracy
0	1.0	0.746345	0.540936
1	2.0	0.898392	0.473684
2	3.0	0.935673	0.704678
3	4.0	0.42617	0.0
4	5.0	0.609649	0.497076
5	6.0	0.753655	0.625731
6	7.0	0.928363	0.74269
7	8.0	0.41155	0.0
8	9.0	0.914474	0.763158
9	10.0	0.964181	0.663743
10	Mean	0.758845	0.50117
11	Median	0.758845	0.540936
12	Max	0.964181	0.763158

Image (i)

The accuracy on the validation is better this time. But is it still not sufficient.

3.3 - Hyperparameter Tuning

For hyperparameter tuning the model was tried 5 times on 5 epochs with the following configurations with all possible combination:

Filter size list = (3, 3) & (5, 5)

Pool size list = (2, 2), (4, 4) & (7, 7)

N filters list = 1, 4 & 16

Learning rate list = 0.01 & 0.001

The total possible combination from the above configurations is 36.

4 - Final Model

The optimal values achieved for the optimal model are as follows:

Optimal Filter Size: (3, 3)

Optimal Pool Size: (2, 2)

Optimal Number of Filters: 4

Optimal Learning Rate: 0.001

The following is the performance of the optimal model on training data (image (j)):

	Trial Number	Training Accuracy	Validation Accuracy
0	1.0	0.817251	0.540936
1	2.0	0.873538	0.54386
2	3.0	0.804094	0.684211
3	4.0	0.877924	0.792398
4	5.0	0.83114	0.625731
5	6.0	0.815058	0.719298
6	7.0	0.88962	0.736842
7	8.0	0.812135	0.839181
8	9.0	0.796053	0.614035
9	10.0	0.756579	0.549708
10	Mean	0.827339	0.66462
11	Median	0.817251	0.66462
12	Max	0.88962	0.839181

Image (j)

The testing accuracy achieved from this model is 73.3%

5 - Way Forward

Due to the memory constraint on the personal PC and time constraint to run each model, the model could not be trained on pixels more than (58, 58). The bigger size of pixel might result in better testing accuracy.

Also, hyperparameter tuning had to be done with lesser epochs and each model had to be tried lesser number of times, and the variety in the parameter values is also restricted due to the same constraints.

By overcoming these constraints, the training accuracy can be increased by a greater extent.