Classification of Weather Images

Harish Neelam

harineelam10@gmail.com

CMSE Department

Michigan State University.

Table of Contents

1. Introduction

1.1. Importance

The value of data has increased significantly throughout the years. It can be in the form of Text, Numbers, Images, Audio, or Videos. Images certainly make up a large amount of the available data. This application can be used for different purposes. This application, together with decision support systems for traffic communication, afforestation, and weather prediction, will be helpful in situations where weather forecasts are inaccurate, such as when self-driving cars must operate safely in dangerous weather.

There are some previous works on this subject, but with fewer classes. This project focuses on eleven classes of weather images and to come up with the best model to predict the image as accurately as possible after training with many images. Since this application has importance in real-life situations, accuracy plays a vital role in avoiding harmful incidents. To accomplish the business goal, the model should be able to predict the rain even if it cannot determine the rain's speed from the image. This project can serve as the foundation for further weather prediction applications when a computer is required but human expertise can't be used and in situations where human expertise may be biased. In this project, it is assumed that weather may be classified in eleven different ways, from safe to dangerous, and it makes sense to divide it into these two categories, since the weather can be dangerous sometimes.

1.2. Objectives

This project aims to build a machine learning model to accurately classify the images into 11 classes namely dew, frost, glaze, rime, snow, hail, rain, lightning, rainbow, and sandstorm. In this project, the model will be trained on thousands of weather images to predict the weather condition in the Image. Finally, the objective is to develop a web Application which predicts the weather condition in the image uploaded by the user.

1.3. Deep Learning

The main concerns in the field of computer vision include object recognition, segmentation, and classification of images. An ML model must be capable of classifying the image among the other types before it can predict the image. Advanced techniques like machine learning and deep learning should be implemented to efficiently analyze the images that are obtained by sensors and cameras. Machine learning techniques like deep learning can be used to find features in imagery. It makes use of a neural network, a multi-layered computer architecture created to resemble the functioning of the human brain. Each layer can extract one or more unique characteristics from an image. This project involves more than 6000 images with 11 different types, it is important to capture every event in the image to accurately classify them. Hence, Deep learning techniques will be very useful in this case to extract the features of an image. Figure 1.3a describes the deep learning architecture with Input, some hidden layers and output of 11 classes.

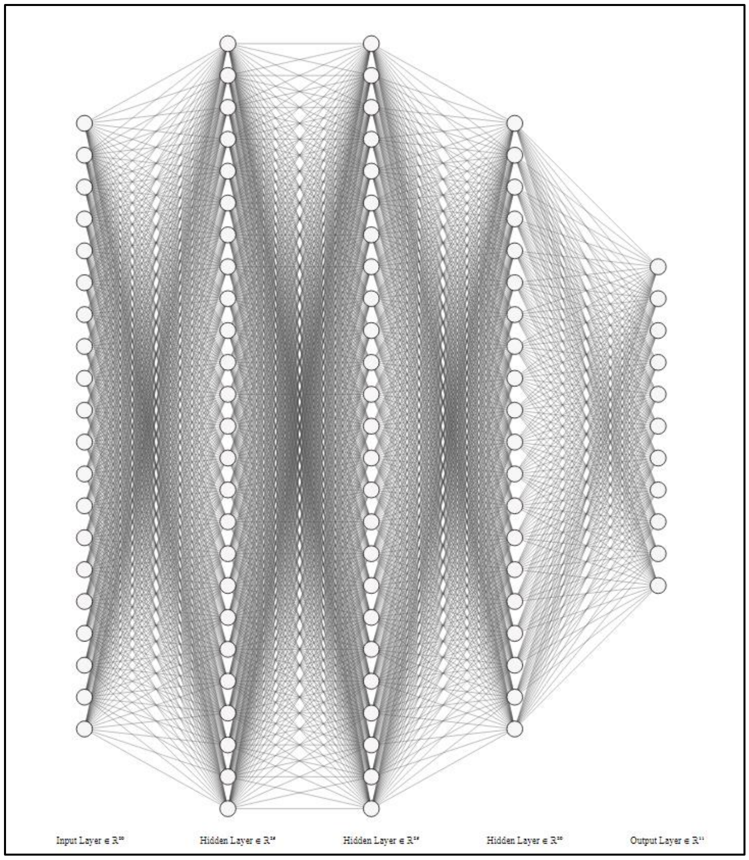


Fig. 1.3a. Deep Learning Architecture.

1.4. Workflow

In order to train a model, thousands of images with different weather conditions are required. Obtaining the dataset, analyzing it and preprocessing it would be the first step in the project. Building machine learning models, training them with the data and observing their performance would be the next step. Finally, the best model is selected and is used to build the web application as shown in Figure 1.4a.

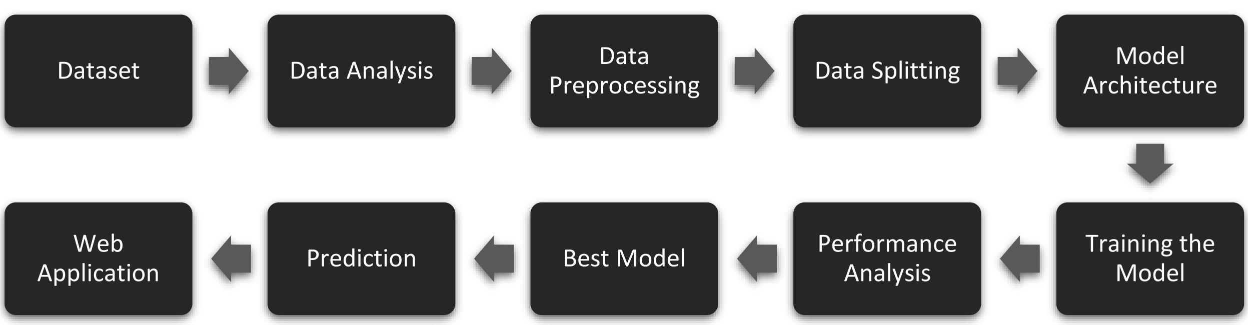


Fig. 1.4a. Workflow of the Project.

2. Data Analysis

2.1. Dataset

The required data is obtained from Kaggle[1] which has 6862 images of 11 classes(dew, fog, frost, glaze, hail, lighting, rain, rainbow, rime, sandstorm, snow) published by Harvard Dataverse in 2021. The images of these classes are shown in Figure 2.1a.

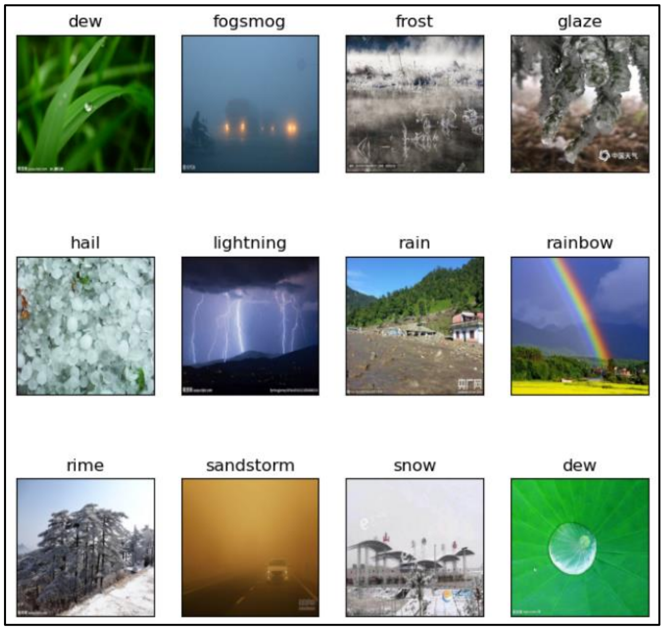
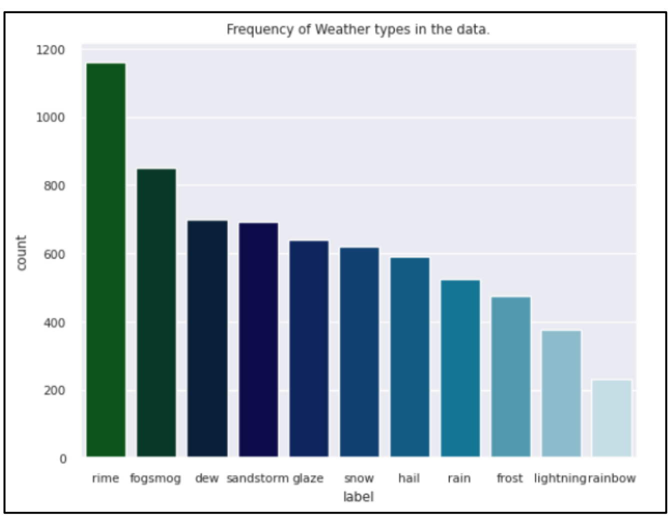


Fig. 2.1a. Sample Images in the Dataset.

2.2. Exploratory Data Analysis

The Dataset was explored using Matplotlib and Seaborn to identify any patterns or trends in the data. Figures 2.2a (Bar Chart) And 2.2b (Pie Chart) illustrates the distribution of target variables in the data. The distribution of new target variables is shown in Figure 2.2c.



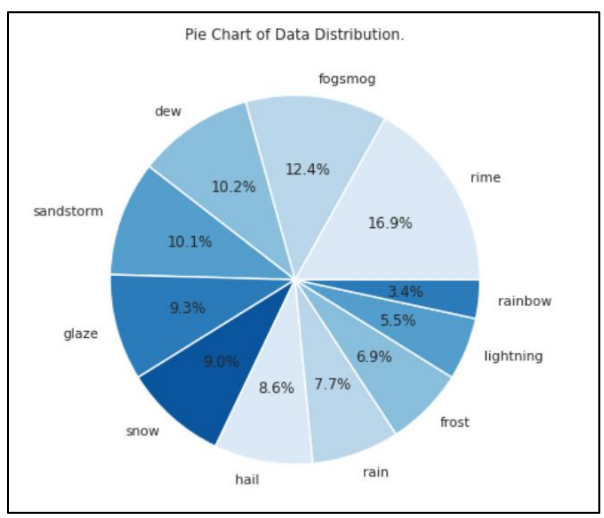


Fig. 2.2a. Bar Chart of Class Distribution.

Fig. 2.2b. Pie Chart of Class Distribution.

2.3. Data Splitting

The Data was then split into Training and Testing sets(70% Training data and 30% Testing data). The Training data was further split into training and validation data with a ratio of 80:20. The validation set is used to check the performance of the model while training with the training data to improve its performance. The following figures (2.3a, 2.3b) show the distribution of data in training and testing data respectively to see the class distribution.

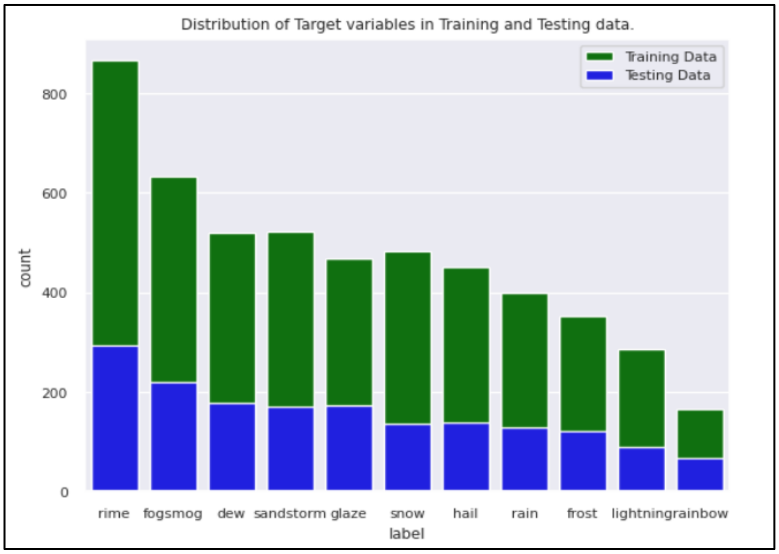


Fig. 2.3a. Bar Graph of Data Distribution in Training and Testing Data.

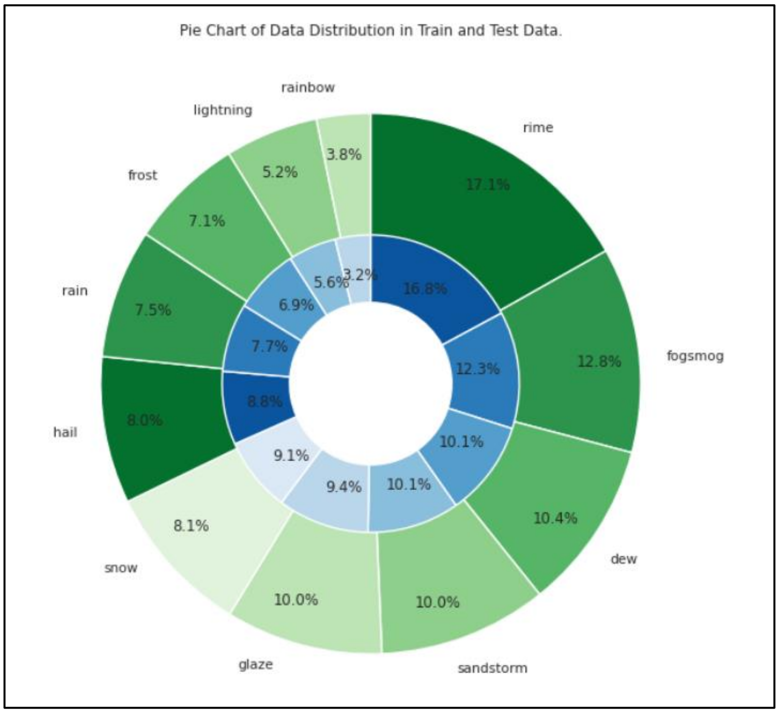


Fig. 2.3b. Nested Pie Chart of Data Distribution in Training and Testing Data.

3. Data Preprocessing

3.1. Image Preprocessing

Resizing: Resizing refers to the process of changing the size of an image. It is crucial to choose an appropriate interpolation method while resizing an image to preserve its quality as much as possible. In this case, the images were resized to 100x100x3.

Interpolation: In Image processing, interpolation is used to create new pixels when resizing an image to a different size. Lanczos interpolation is used to resize the images to 100x100x3, because it is considered a good method for image resizing as it helps to preserve image quality and sharpness. Unlike other interpolation methods, Lanczos interpolation uses a more complex formula to estimate the values of new pixels based on the surrounding pixels.

Color Space Conversion: Different color spaces represent colors in different ways, and some color spaces are better suited for specific applications than others. For example, the RGB color space is commonly used for displaying images on screens. There is evidence that RGB color spaced images train well, so the images were converted to the RGB color space. Figures 3.1a and 3.2b show the images before and after Image Preprocessing.

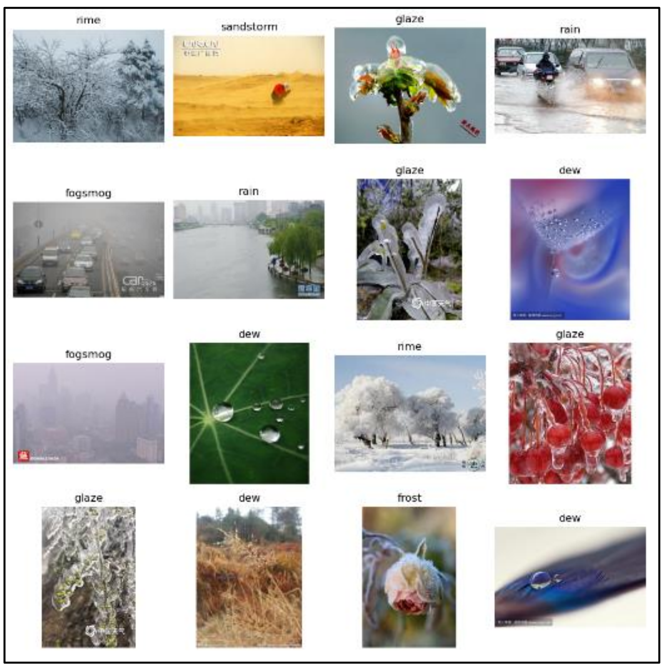


Fig. 3.1.a Images before Preprocessing.

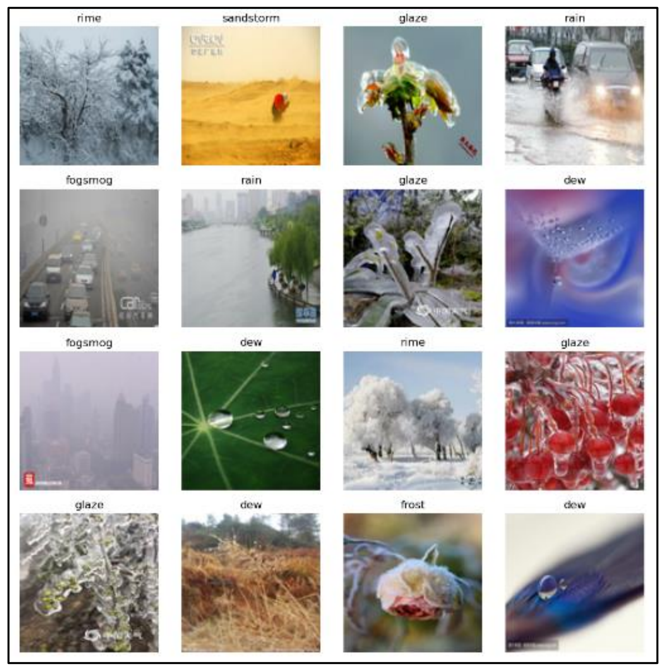


Fig. 3.1b Images after Preprocessing.

3.2. Normalization

Each image is normalized before training the model. Normalization can help to improve the performance and accuracy of the machine learning model by ensuring that the input data is on a consistent scale and distribution. It also helps to prevent the influence of any outliers or extreme values in the data. The images were converted to lists, and the mean and standard deviation of the training set were calculated as shown in the Equation 3.2a. Each image array was then subtracted by the mean and divided by the standard deviation. These mean and standard deviation values were stored to be used during prediction.

$$X = \frac{X - X\_train\_mean}{X\_train\_std} \quad \text{--- Eq. 3.2a}$$

3.3. One-Hot Encoding

Label encoding is a technique used to convert categorical data into numerical values. In machine learning, algorithms typically require numeric input data, so label encoding is often used to preprocess the data before it can be used to train a model. Label encoding assigns a unique integer value to each category in a categorical variable. It is used to encode the Target variable.

One-hot encoding is a technique used to represent categorical variables as binary vectors, where each category is represented as a vector of all zeros except for one at the index corresponding to the category as shown in the following Equation 3.3a.

$$Class\ 0 = [1,0,0,0,0,0,0,0,0,0,0]$$

$$Class\ 3 = [0,0,0,1,0,0,0,0,0,0,0] \quad \text{--- Eq. 3.3a}$$

4. Scikit-Learn Estimators

The following are some of the Scikit-Learn estimators used for this data as baseline models. The initial data obtained has images with different sizes and labels. The images were resized to 32x32x3 and flattened to 3,072 pixels. The data now has 3,072-pixel features and label column as target variable. This data is used to perform the following estimators.

4.1. Random Forest

A set of decision trees that grow in a selection of randomly chosen data subspaces are called random forests and are a method for creating classification ensembles. According to experimental findings, random forest classifiers are capable of accurately classifying data in high-dimensional domains with a wide range of classes. Recently, image classification and bioinformatics have shown increasing interest in random forests. The popularity and accuracy of this estimator made it my first option for this project to classify images. Unfortunately, this model did not perform well on this data. This was able to provide accuracy measure of 33.14% on average when ran multiple times. This may be because there were many features(3,072pixels) and 11 classes. The accuracy was 50.60% when the target variable is grouped to 3 classes.

4.2. Support Vector Machine Classifier

Support vector machines (SVMs) are a group of supervised learning techniques for classifying data, performing regression analysis, and identifying outliers. In high dimensional spaces, it performs very well. This was able to offer an accuracy of just 38.12% and took a huge amount of time to fit the model. Perhaps this is because there are many features and less samples.

5. Deep Learning Models

The Scikit-Learn estimators couldn’t produce good results, resulting in poor performance. Deep learning models were proven to be the best in such cases where Scikit-Learn estimators didn't perform well. There are many pre-trained models and convolutional neural networks with different numbers of hidden layers that can be used here. Some of them are as follows:

5.1. Basic CNN

A Basic CNN model is designed, and this model expects input images with a larger size of (100, 100, 3). The model architecture consists of multiple layers of convolutional, batch normalization, average pooling, and dropout layers, which are all used to extract important features from the input images and reduce the chances of overfitting. The final layer is a dense layer with 11 output nodes, which corresponds to the number of classes that the model can predict. The activation function used for all the convolutional and dense layers, except the last one, is ReLU. The last dense layer uses a SoftMax activation function, which outputs a probability distribution over the classes. The below figures(5.1a, 5.2b and 5.2c) show the results obtained by using basic CNN model.

```
CNN Training: accuracy = 0.788536
CNN Validation: accuracy = 0.730419
CNN Test: accuracy = 0.709913
```

Fig. 5.1a. Accuracies of CNN Model.

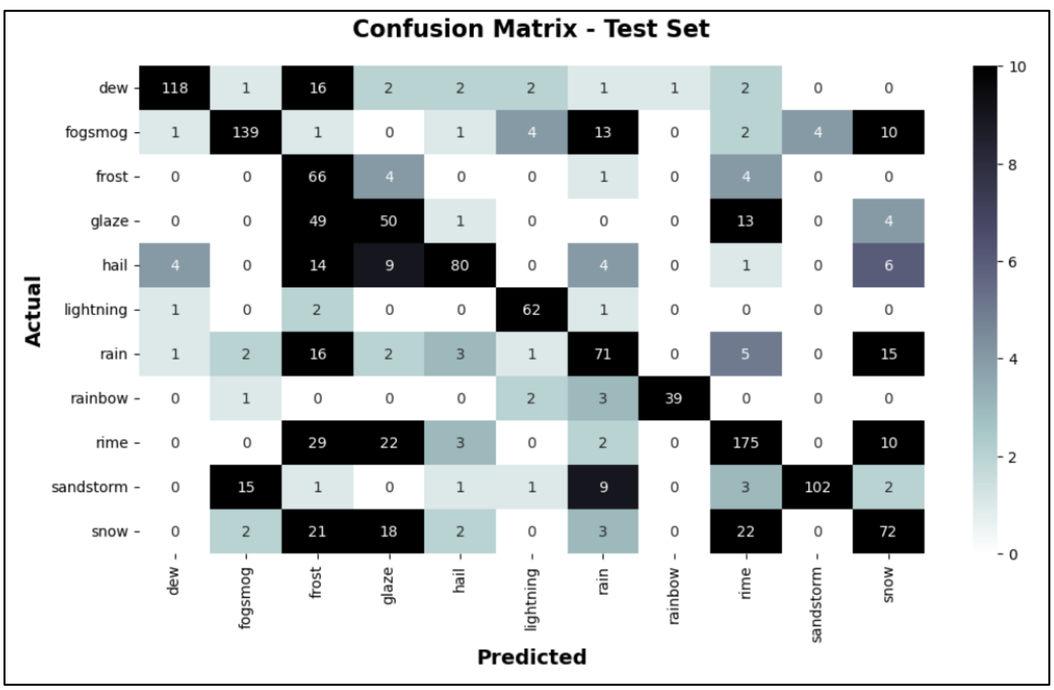


Fig. 5.1b. Confusion Matrix of CNN Model.

	precision	recall	f1-score	support
0	0.94	0.81	0.87	145
1	0.87	0.79	0.83	175
2	0.31	0.88	0.46	75
3	0.47	0.43	0.45	117
4	0.86	0.68	0.76	118
5	0.86	0.94	0.90	66
6	0.66	0.61	0.63	116
7	0.97	0.87	0.92	45
8	0.77	0.73	0.75	241
9	0.96	0.76	0.85	134
10	0.61	0.51	0.56	140
accuracy			0.71	1372
macro avg	0.75	0.73	0.72	1372
weighted avg	0.76	0.71	0.72	1372

Fig. 5.1c. Classification Report of CNN Model.

This model gave an accuracy of **70.99%,** which is not bad but can be improved. The model was adjusted by increasing and decreasing the number of hidden layers, changing the number of filters, and adjusting the input shape. The best results were achieved with 26 hidden layers and an input shape of 100x100x3.

5.2. ResNet50

The ResNet50 model is loaded with pre-trained weights on the ImageNet dataset and its last layer, which performs classification into the 1,000 ImageNet classes, is removed. The remaining convolutional layers in the ResNet50 model are set as non-trainable, so that only the dense layers added later will be trained. Then, a Flatten layer is added to the output of the ResNet50 model to convert the 2D output tensor from the convolutional layers into a 1D vector. Two Dense layers are added on top of the flattened output. Finally, a model object is created, which takes the input of the ResNet50 model and outputs the probability distribution over the 11 classes generated by the two dense layers. This was able to produce an accuracy of **55.95%,** which is less than Basic CNN model as shown in the following figures 5.2a and 5.2b.

```
Training: accuracy = 0.582249
Validation: accuracy = 0.532778
Test: accuracy = 0.559495
```

Fig. 5.2a. Accuracies of ResNet50 Model.

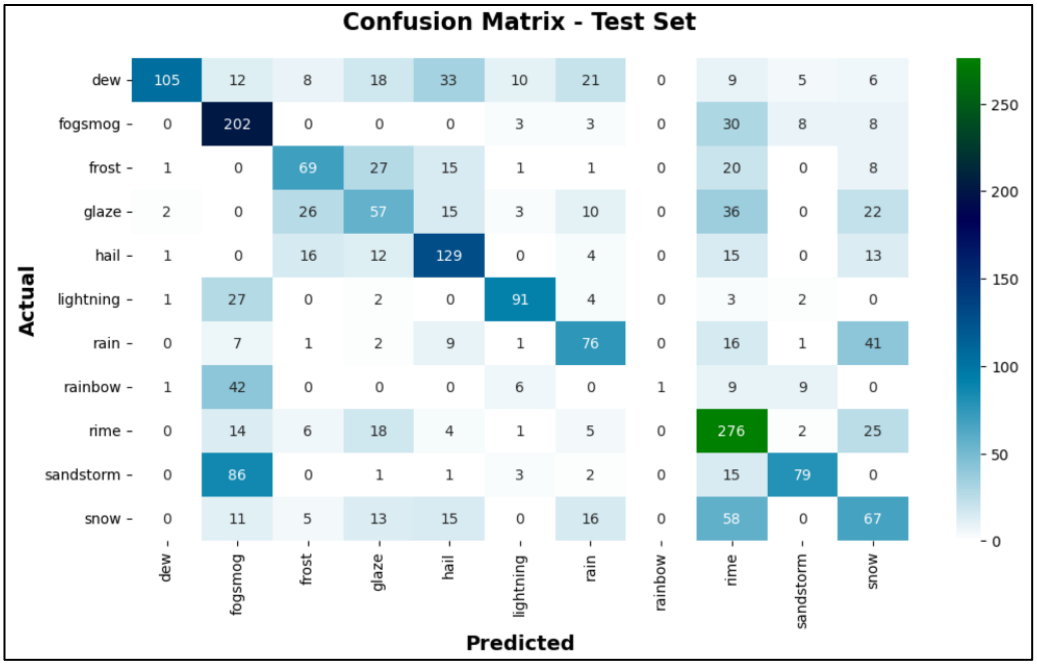


Fig. 5.2b. Confusion Matrix of ResNet50 Model.

ResNet is a more complex architecture compared to basic CNN models, and it has been shown to perform well on many computer vision tasks. However, ResNet may perform worse than a basic CNN model in some cases due to overfitting. ResNet has many parameters, which can lead to overfitting if the model is not properly regularized. In contrast, basic CNN models have fewer parameters and may be less prone to overfitting.

5.3. InceptionV3

InceptionV3 is a deep learning model for image classification that was developed by Google. InceptionV3 has been shown to achieve state-of-the-art performance on large-scale image classification datasets such as ImageNet. InceptionV3 also has a relatively small number of parameters compared to some other deep learning models, which makes it computationally efficient and faster to train. This makes it a good choice for applications where speed and efficiency are important.

The InceptionV3 model was also used to classify weather images with 11 classes. However, it was only able to classify the images with **62.02%** accuracy as shown in the following figures(5.3a and 5.3b), which could be due to some of the potential drawbacks of the model. One of these drawbacks is that it can be difficult to train on smaller datasets, as the model may overfit the training data. Additionally, the InceptionV3 architecture can be complex and difficult to interpret, which may make it harder to optimize and fine-tune for specific applications.

```
Training: accuracy = 0.663977
Validation: accuracy = 0.603538
Test: accuracy = 0.620204
```

Fig. 5.3a. Accuracies of InceptionV3 Model.

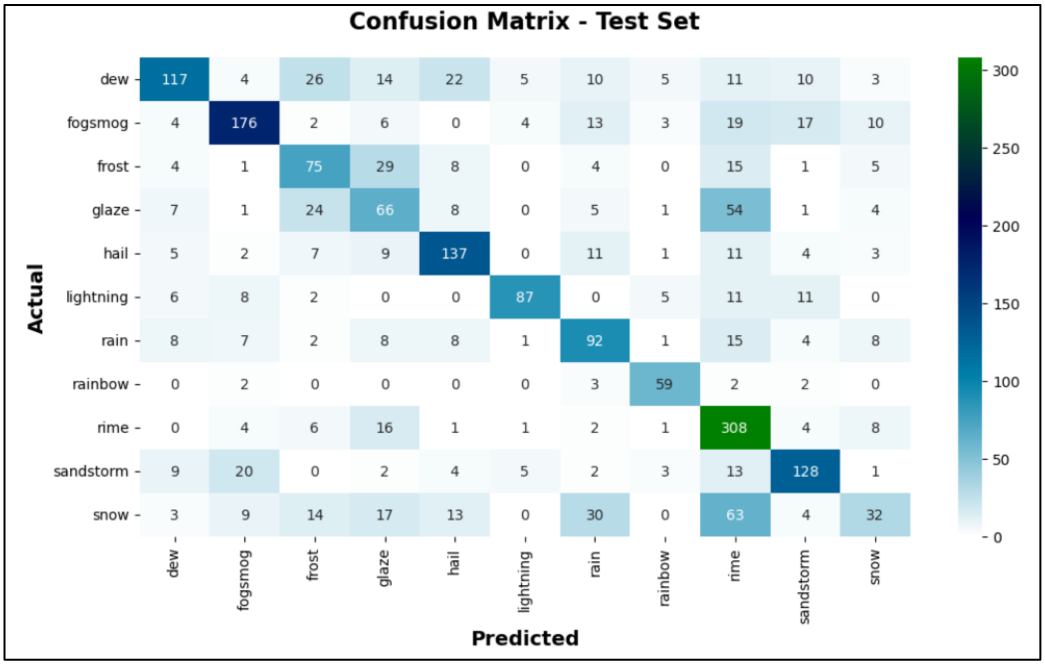


Fig. 5.3b. Confusion Matrix of InceptionV3 Model.

5.4. VGG16

The Visual Geometry Group (VGG) at the University of Oxford created the deep learning model VGG16 for classifying images. The accuracy of VGG16 in picture classification tasks is one of its key advantages[4]. 13 convolutional layers and 3 fully linked layers make up the VGG16 architecture. To extract features from the input image, the convolutional layers utilize tiny 3x3 filters with a stride of 1 and a padding of 1. The fully connected layers output a probability distribution over the classes using a SoftMax activation function. Additionally, the design of VGG16 is quite uniform and straightforward to understand, making it simple to customize for particular applications. Due to its simplicity, it is also less likely to overfit smaller datasets. Since VGG16 has already been trained on large image classification datasets, it can be fine-tuned for specific applications using smaller datasets. This pre-training can enhance the accuracy and generalizability of the model.

5.4.1. Target Variable with 11 classes

This model performed very well on this data, achieving an accuracy of **80.38%,** results can be seen in Figures 5.4.1a, 5.4.1b and 5.4.1c. This is notably higher than the performance of other models. Therefore, it is a good choice to proceed with this model for predicting the weather in the images, with some modifications.

```
Training: accuracy = 0.865435
Validation: accuracy = 0.780437
Test: accuracy = 0.803788
```

Fig. 5.4.1a. Accuracies of VGG16 Model with 11 classes.

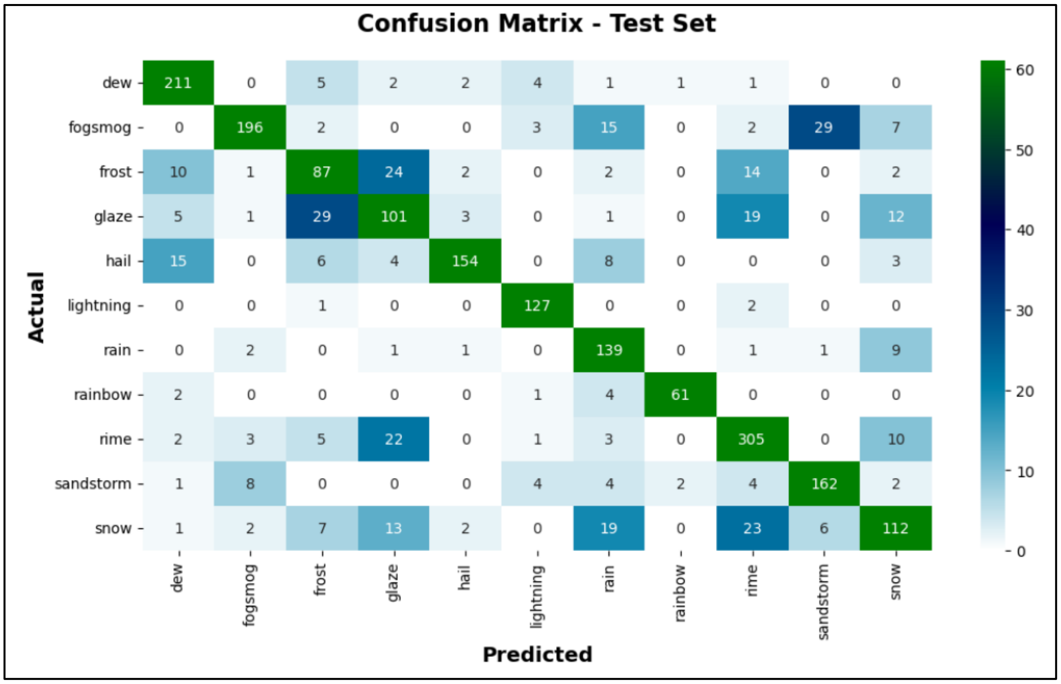


Fig. 5.4.1b. Confusion Matrix of VGG16 model with 11 classes.

	precision	recall	f1-score	support
0	0.85	0.93	0.89	227
1	0.92	0.77	0.84	254
2	0.61	0.61	0.61	142
3	0.60	0.59	0.60	171
4	0.94	0.81	0.87	190
5	0.91	0.98	0.94	130
6	0.71	0.90	0.79	154
7	0.95	0.90	0.92	68
8	0.82	0.87	0.84	351
9	0.82	0.87	0.84	187
10	0.71	0.61	0.65	185
accuracy			0.80	2059
macro avg	0.80	0.80	0.80	2059
weighted avg	0.81	0.80	0.80	2059

Fig. 5.4.1c. Classification Report of VGG16 Model with 11 classes.

5.4.2. Target Variable with 8 classes and 7 classes

Based on the above classification report, we can conclude that the classes **Frost, Glaze, Snow,** and **Rime** are confusing from one another. Therefore, these four classes were grouped together to train the model more efficiently with the other classes. However, creating an alternate model for each of these four classes could be a useful strategy. The VGG16 model was used to classify these 8 weather classes, as it had the highest accuracy among other models. It was able to predict the 8 classes with **85.13%** accuracy.

Since all these weather types fall under the same category, it made sense to group them together into one class called **Cold**. This included the previously grouped Frost, Glaze, Snow, and Rime classes, as well as the **Dew** class. With these 7 classes as the target variable, the model achieved an accuracy of 87.36%.

5.4.3. Target Variable with 3 classes(Multi-level Hierarchical Classification Model):

A new model was built to predict the 5 weather classes that were grouped together. Using the same VGG16 model, the accuracy was **79.85%.** However, since all these images are closely related and have a white background, more data may be needed to accurately predict these classes.

The images were divided into three groups based on the weather: Rainy, Dusty, and Cold. The images labeled as rain, hail, lightning, and rainbow were mapped to Rainy weather, while the images labeled as sandstorm and fog/smog were mapped to Dusty weather. All the other images were mapped to Cold weather. A model was built to first predict the main class of weather: **Rainy, Cold,** or **Dusty**. Based on this result, sub-models were trained using respective images to predict the type of weather and whether or not it is safe. This approach is called **Multi-level Hierarchical Classification Model**. The first level of the model would classify the images into high-level groups, and the second level would classify sub-groups within each group. The training process for the models is depicted in Figure 5.4.3a.

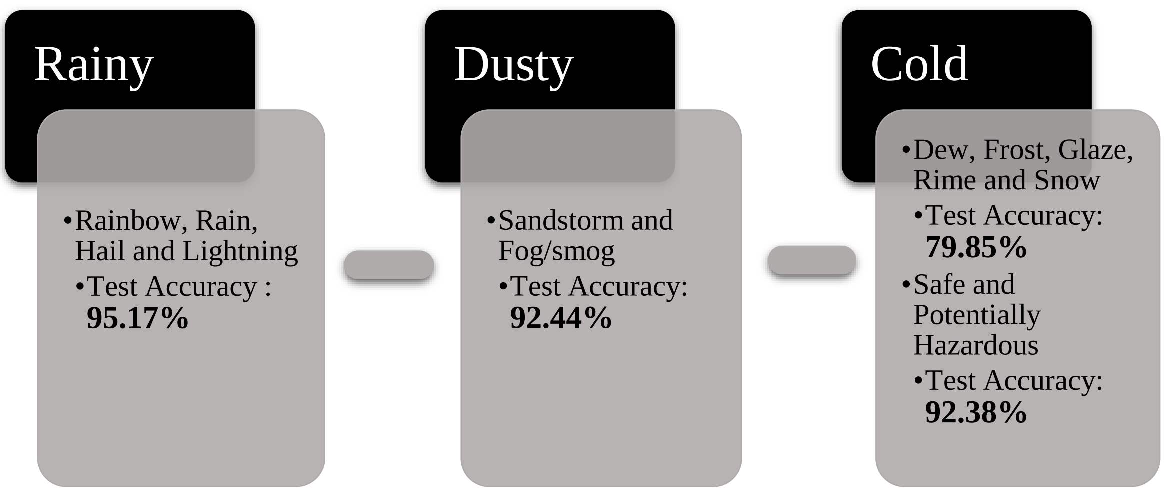


Fig. 5.4.3a. Training process of VGG16 Model with 3 classes.

The Accuracy of the model which predicts Rainy, Dusty or Cold Weathers is **93.29%** as shown in the Figure 5.4.3b. The Confusion Matrix is shown in Figure 5.4.3c.

```
Training: accuracy = 0.982041
Validation: accuracy = 0.939646
Test: accuracy = 0.932977
```

Fig. 5.4.3b. Accuracies of VGG16 Model with 3 classes.

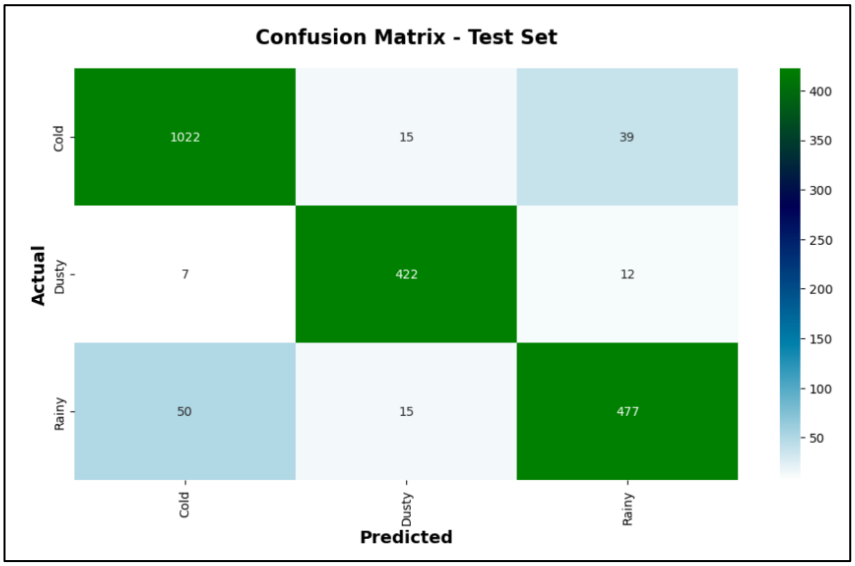


Fig. 5.4.3c. Confusion Matrix of VGG16 Model with 3 classes.

6. Results and Observations

It has been observed that when the model is trained with three classes, the accuracy is high. The final model consists of a main model that classifies three labels (Rainy, Dusty, and Cold), and sub-models that classify Rainy Weather into four classes (Rain, Rainbow, Lightning, and Hail), Dusty Weather into two classes (Sandstorm and Fog/Smog), and Cold Weather in two ways. The first model classifies weather into five classes (Dew, Snow, Rime, Glaze, and Frost), while the other model classifies it into Safe or Potentially Hazardous weather. Therefore, the weather in the image is classified into eleven classes and also into three classes (Safe, Potentially Hazardous, and Dangerous). The results of all the models are as shown in Table 6a.

The models that classify Rainy and Dusty weather produced accuracies of **95.17%** and **92.44%,** respectively. The model that classifies Cold weather into 5 classes gave an accuracy of about **79.85%.** The classification report and confusion matrix are shown in figures 6a and 6b, respectively.

Table 6a. Results of Different Classification models with 11 classes.

Classification Model	Accuracy
Random Forest	33.14%
Support Vector Classifier	38.12%
Basic CNN	70.99%
ResNet50	55.95%
InceptionV3	62.02%
VGG16	80.38%

	precision	recall	f1-score	support
0	0.94	0.88	0.91	207
1	0.67	0.79	0.72	140
2	0.71	0.70	0.70	171
3	0.84	0.88	0.86	360
4	0.85	0.73	0.79	200
accuracy			0.81	1078
macro avg	0.80	0.80	0.80	1078
weighted avg	0.82	0.81	0.81	1078

Fig. 6a. Classification Report of Model with 5 classes of Cold Weather.

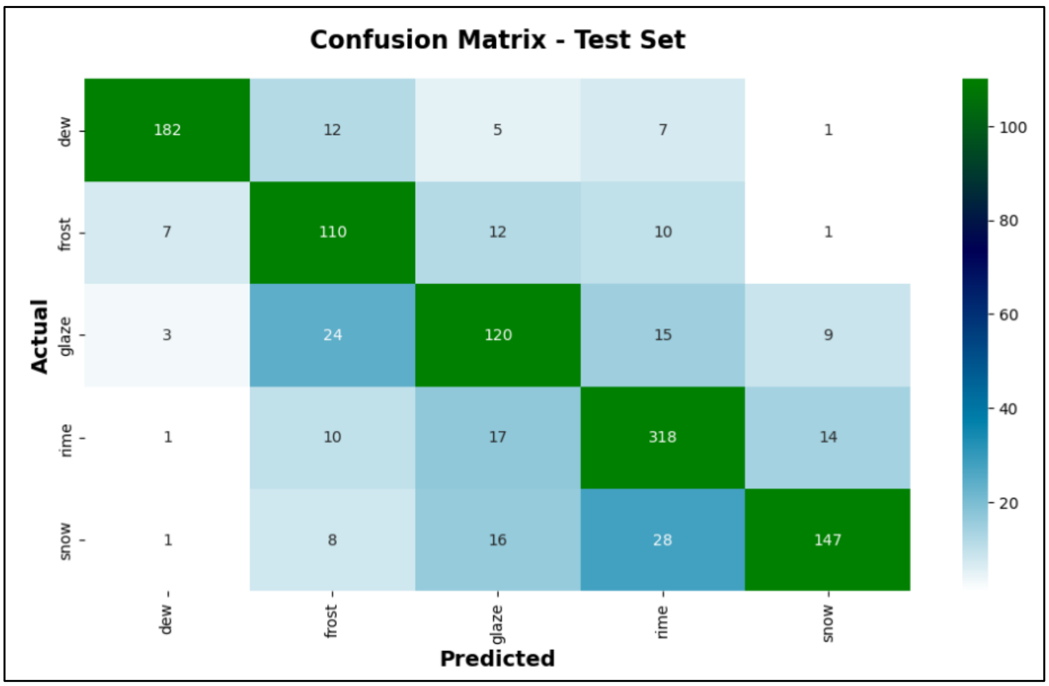


Fig. 6b. Confusion Matrix of Model with 5 classes of Cold Weather.

From the results, it can be seen that glaze is often confused with frost, and snow is often confused with rime. Frost and glaze have very poor precision, possibly because the images of these two weathers resemble each other.

7. Prediction

The Best model, VGG16, was chosen to perform the prediction on the image. The flow of the prediction is shown in Figure 7a.

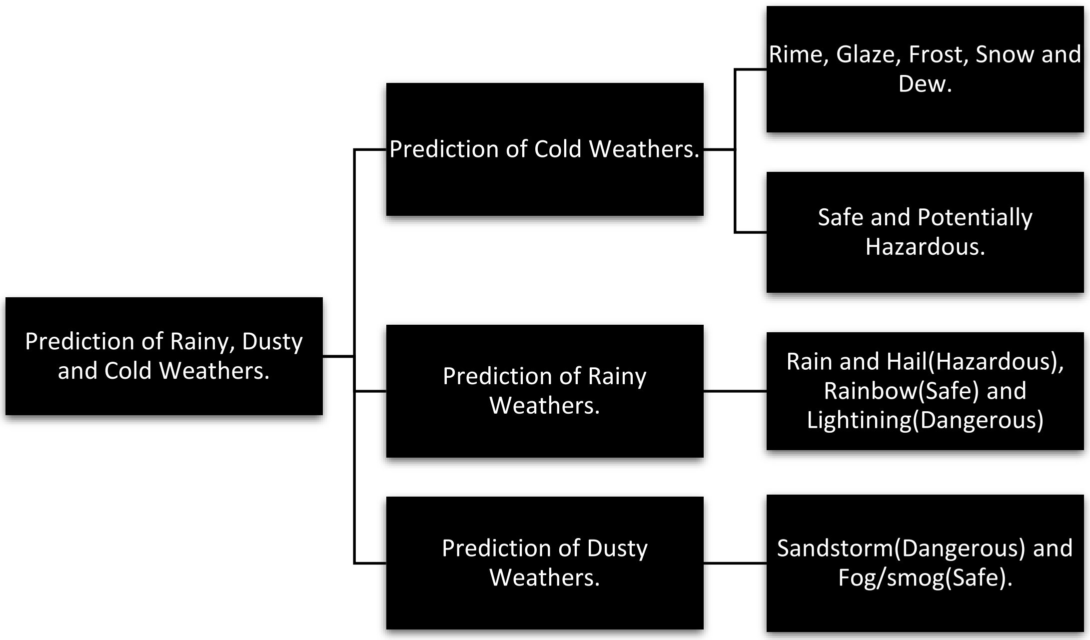


Fig. 7a. Flow of Prediction.

Finally, the model will be able to predict the 11 classes of weather and whether they are Safe, Potentially Hazardous, or Dangerous. The Trained models are stored and loaded when predicting images in the web application.

8. Conclusion

This study aims to build a machine learning model that accurately classifies weather images into 11 classes and predicts whether they are safe, potentially hazardous, or dangerous. The model is trained on thousands of images and is intended to be used for weather forecasting and hazard prediction in situations where human expertise cannot be used. The importance of this application lies in its potential to avoid harmful incidents by accurately predicting weather conditions. The best model, **VGG16**, achieved high accuracies for rainy and dusty weather, but classification of cold weather had lower accuracy due to similarities between frost and glaze. The trained models are stored and loaded in the web application for image prediction. Overall, this approach shows promise for weather forecasting and hazard prediction in practical applications.

9. References

[1]Xiao, Haixia (Harvard Dataverse). (2021). Weather phenomenon database (WEAPD), Version 1. Retrieved 20 February 2023 from www.kaggle.com/datasets/jehanbhathena/weather-dataset.

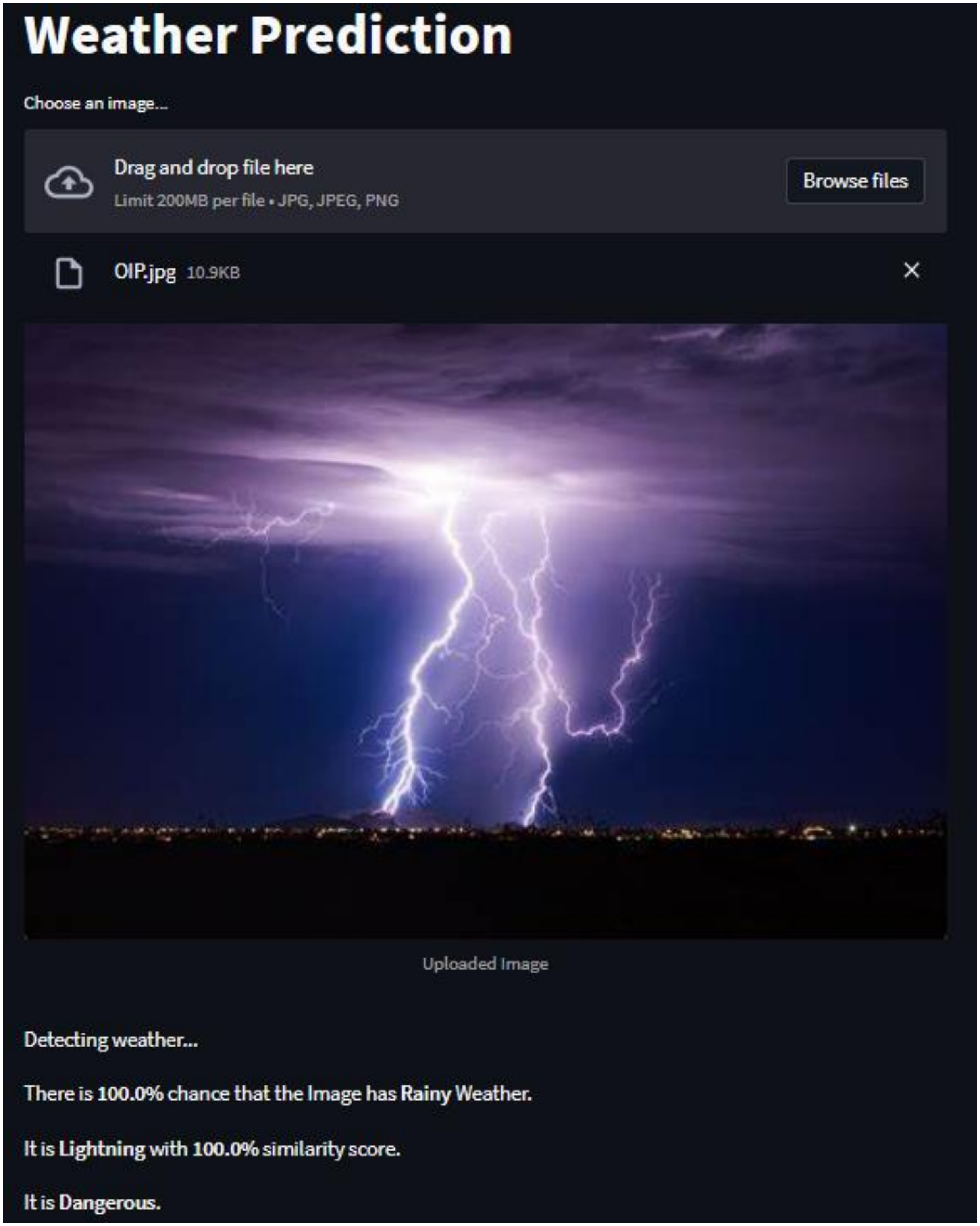
[2]K. Padmavathi, H. Neelam, M. P. K. Reddy, P. Yadlapalli, K. S. Veerella and K. Pampari, "Melanoma Detection Using Deep Learning," 2022 International Conference on Computer Communication and Informatics (ICCCI), Coimbatore, India, 2022, pp. 1-4, doi: 10.1109/ICCCI54379.2022.9741010.

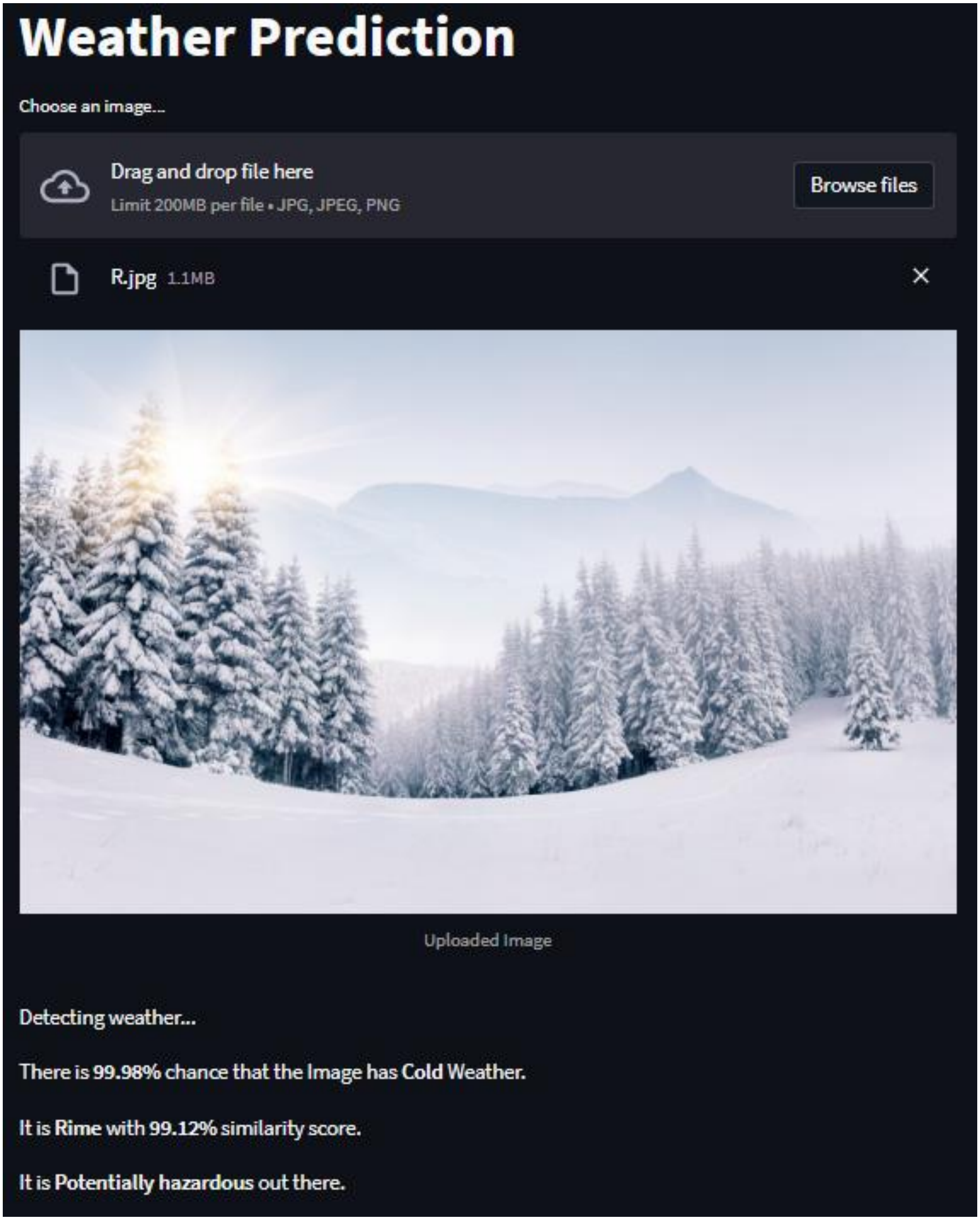
[3]Hassan, Aya Gamal Nasr et al. "Image classification based deep learning: A Review." Aswan University Journal of Sciences and Technology (2022): n. pag.

[4]J. Tao, Y. Gu, J. Sun, Y. Bie and H. Wang, "Research on vgg16 convolutional neural network feature classification algorithm based on Transfer Learning," 2021 2nd China International SAR Symposium (CISS), Shanghai, China, 2021, pp. 1-3, doi: 10.23919/CISS51089.2021.9652277.

10. Appendix

- GitHub Repository. **Harishneelam/Weather-Image-Classification (github.com).**
- The following are the images from the web-application predicting weather from the images.


Weather Prediction
Choose an image...
Drag and drop file here
Limit 200MB per file • JPG, JPEG, PNG
Browse files
OIP.jpg 10.9KB
Uploaded Image
Detecting weather...
There is 100.0% chance that the Image has Rainy Weather.
It is Lightning with 100.0% similarity score.
It is Dangerous.


Weather Prediction
Choose an image...
Drag and drop file here
Limit 200MB per file • JPG, JPEG, PNG
Browse files
R.jpg 1.1MB
Uploaded Image
Detecting weather...
There is 99.98% chance that the Image has Cold Weather.
It is Rime with 99.12% similarity score.
It is Potentially hazardous out there.