# Automated Weather Classification Using Transfer Learning

1. **INTRODUCTION**

1.1 Project Overview

In this project, we are classifying various types of weather. These weathers are classified into 5 categories: Cloudy, Shine, Rain, Foggy& Sunrise. Transfer learning has become one of the most common techniques that have achieved better performance in many areas, especially in image analysis and classification. We used Transfer Learning techniques like Inception V3, VGG19, and Xception V3, which are more widely used as transfer learning methods in image analysis and are highly effective.

1.2 Purpose

The Automated Weather Classification using Transfer Learning project aims to develop a system that can automatically classify weather conditions based on input images using transfer learning techniques. The project aims to leverage the power of deep learning and pre-trained models to improve the accuracy and efficiency of weather classification. To develop a system that can automate and improve the accuracy of weather classification, providing valuable insights for various industries and domains.

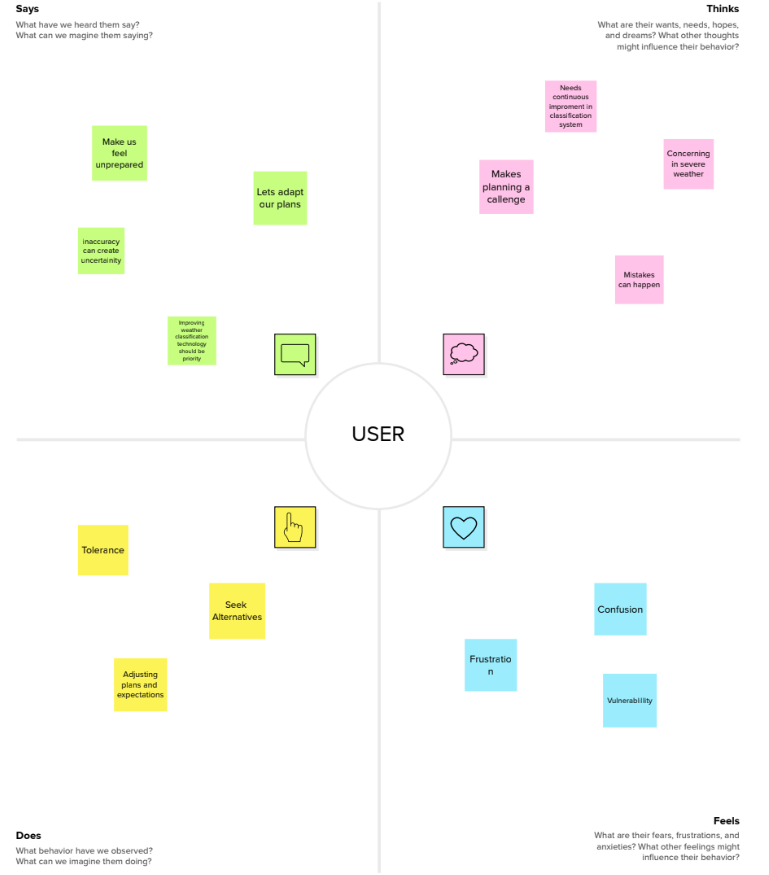
2. **IDEATION & PROPOSED SOLUTION**

2.1 Problem Statement Definition

The manual classification of weather conditions based on images is time-consuming, labor-intensive, and prone to human errors. Existing automated weather classification systems often lack accuracy and efficiency, hindering their practical applications. Therefore, there is a need for an automated weather classification system that can accurately and efficiently classify weather conditions based on input images using transfer learning techniques.

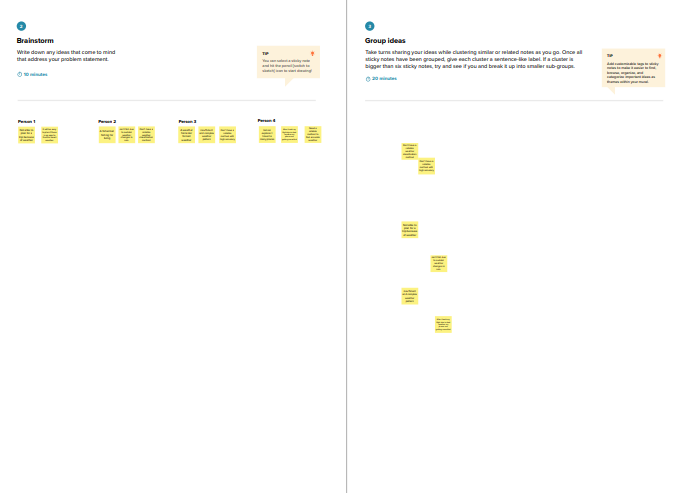
2.2 Empathy Map Canvas

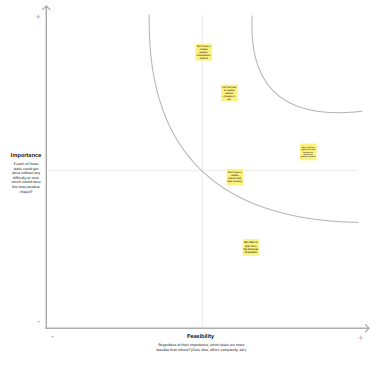
The Empathy map canvas, Automated Weather Classification using Transfer Learning project can be designed and developed to address their needs, alleviate pain points, and provide a solution that satisfies their goals and motivations.



2.3 Ideation & Brainstorming

These ideation and brainstorming points can serve as starting points to explore innovative ideas and directions for the Automated Weather Classification using Transfer Learning project. The aim is to develop a robust and effective system that addresses the challenges of accurate and automated weather classification.





2.4 Proposed Solution

The proposed solution for the Automated Weather Classification using Transfer Learning project is a comprehensive system that leverages transfer learning techniques and deep learning models to automate and improve the accuracy of weather classification. The system consists of the following key components:

* **Data Collection and Preprocessing** - Collect a diverse and representative dataset of weather-related images, including various weather conditions, geographic locations, and seasons.
* **Transfer Learning Model Selection and Fine-tuning** - Select a pre-trained deep learning model, such as VGG, ResNet, or Inception, that has been trained on a large-scale dataset, including general images or related tasks.
* **Training and Validation** - Split the dataset into training and validation sets to train and evaluate the model's performance.
* **Model Evaluation and Iterative Improvement** - Evaluate the trained model's performance using various metrics, such as accuracy, precision, recall, and F1 score, on the validation set.
* **User Interface and Integration** - Develop a user-friendly interface that allows users to upload weather images and obtain classification results.
* **Scalability and Deployment** - Ensure the solution is scalable to handle large-scale datasets, increasing volumes of weather images, and diverse weather conditions.
* **Continuous Learning and Updates** - Implement mechanisms for continuous learning, allowing the system to adapt to new weather patterns, environmental changes, and evolving user requirements.

3. **REQUIREMENT ANALYSIS**

3.1 Functional requirement

Following are the functional requirements of the proposed solution.

|  |  |  |
| --- | --- | --- |
| **FR No.** | **Functional Requirement (Epic)** | **Sub Requirement (Story / Sub-Task)** |
| FR-1 | User Image query | Upload the image from the device |

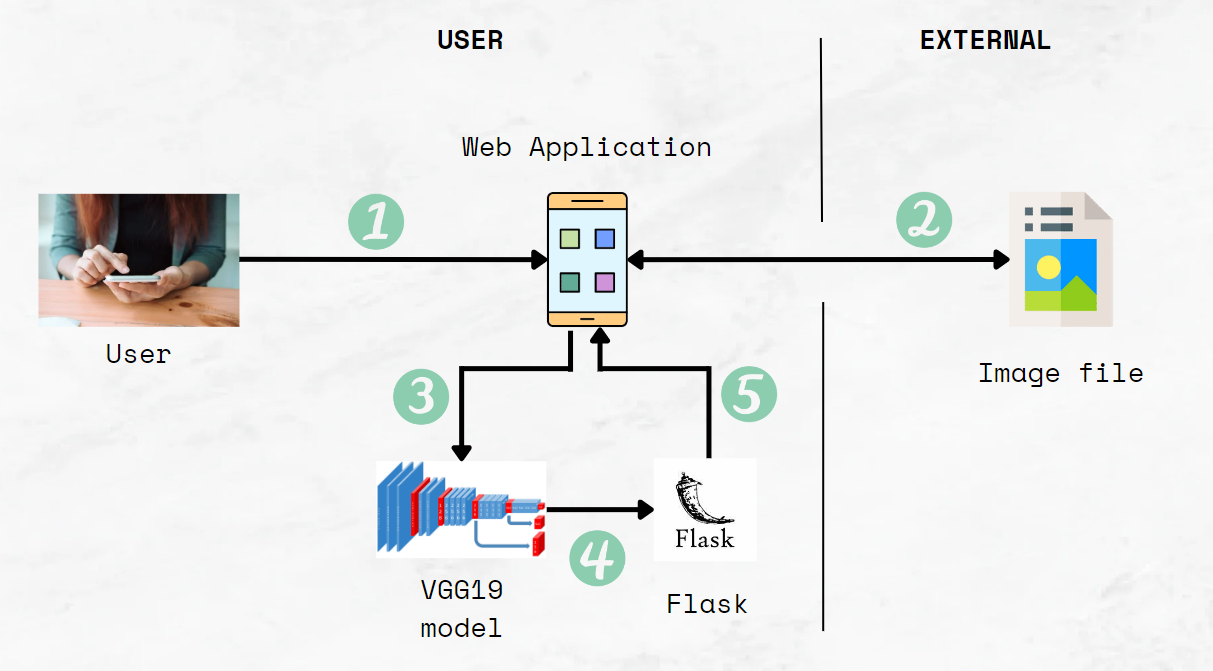
3.2 Non-Functional requirements

Following are the non-functional requirements of the proposed solution.

|  |  |  |
| --- | --- | --- |
| **FR No.** | **Non-Functional Requirement** | **Description** |
| NFR-1 | **Usability** | Design an intuitive and visually appealing user interface that is easy to navigate and understand. Use clear and concise labels, icons, and visual cues to guide users through the weather classification process. Consider incorporating user-centered design principles to ensure the interface meets the needs and expectations of the target users. |
| NFR-2 | **Security** | Implement secure storage practices for user data, including weather images and personal information. Store data in encrypted databases or file systems with restricted access. When transmitting data over networks, use secure protocols (e.g., HTTPS) to prevent unauthorized interception or tampering. |
| NFR-3 | **Reliability** | Ensure the availability of high-quality and reliable training data for the transfer learning model. Thoroughly clean and preprocess the data to remove noise, inconsistencies, or biases that could impact the accuracy of the weather classification. Regularly update and maintain the training dataset to reflect current weather patterns. |
| NFR-4 | **Performance** | Optimize the transfer learning model architecture to balance accuracy and computational efficiency. Consider lightweight model architectures that can provide accurate weather classification while minimizing computational resources and inference time. Techniques like model pruning, quantization, or knowledge distillation can be applied to reduce model size and improve performance. |
| NFR-5 | **Availability** | Implement redundancy and fault tolerance measures to minimize the impact of hardware failures, network outages, or software errors. This can include deploying the system across multiple servers or cloud instances with load balancing and failover mechanisms. Use distributed architectures or replication techniques to ensure data availability and consistency. |
| NFR-6 | **Scalability** | Design the system to leverage distributed computing techniques, such as parallel processing or distributed data storage. Break down computationally intensive tasks and distribute them across multiple nodes or instances to improve performance and accommodate higher workloads. |

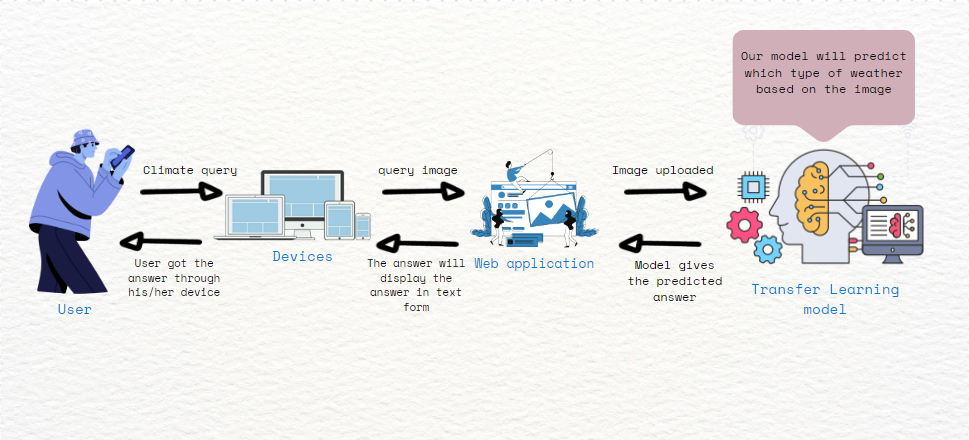
**4. PROJECT DESIGN**

4.1 Data Flow Diagrams

The described data flow represents the overall process of automated weather classification using transfer learning, encompassing data collection, preprocessing, model training, inference, and result presentation. It enables the system to learn from data and make accurate weather predictions based on input images. ****

1. User configures credentials for the Automated Weather Classification service and starts the web application .
2. User selects the image file to process and load.
3. The VGG19 extracts features from the image and predicts the answer.
4. The predicted answer will pass to the flask code.
5. The flask code will pass the predicted answer to the web application.

4.2 Solution & Technical Architecture

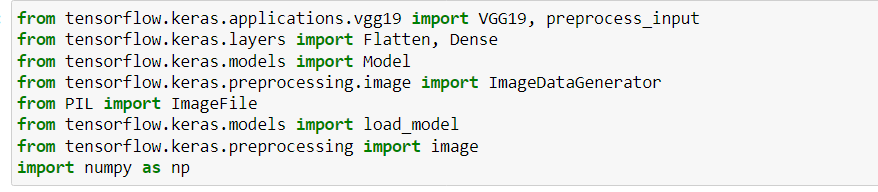


4.3 User Stories

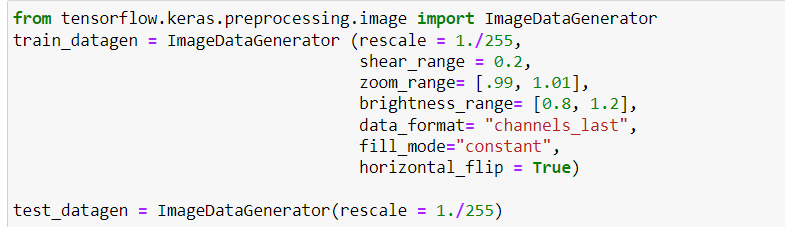
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **User Type** | **Functional Requirement**  **(Epic)** | **User Story Number** | **UserStory/Task** | **Acceptance criteria** | **Priority** | **Team member** |
| Customer(Mobileuser) | Dashboard | USN-1 | As a user, In the web application can see few details about the weather classification | I can see examples of weather images | High | Shailesh |
| Customer (Web user) | Predict image | USN-2 | As a user, I can upload images and get what type of weather it is | Upload the image from the device | High | Rithiesh |
| Administrator | Manage the product | USN-3 | As a developer, I want to build a user-friendly web application that allows users to upload images and receive feedback on how to sort it properly. | We can detect weather classification using an image. | High | Kailash |
| Weather forecast | Predict image | USN-4 | As a weather forecaster, I am able to predict the weather with high accuracy | Improve the performance of the model | High | Barath |
| Farmer | Predict image | USN-5 | I can make informed decisions about crop management, irrigation, and pest control based on predicted weather conditions. | I select the high performance model | High | Aravind |

**5. CODING & SOLUTIONING (Explain the features added in the project along with code)**

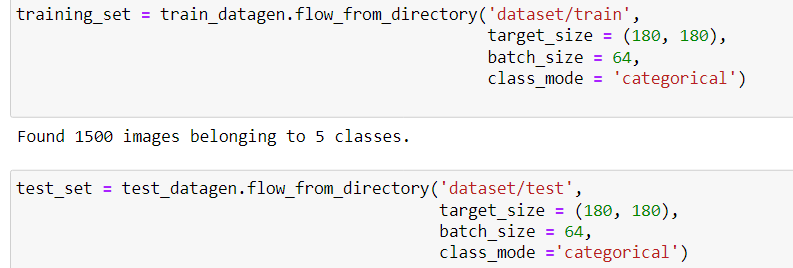
Import the necessary libraries



An instance of the ImageDataGenerator class can be constructed for train and test.

****

Apply ImageDataGenerator functionality to the Train set and Test set by using the following code. For Training set using flow\_from\_directory function. This function will return batches of images from the subdirectories Arguments

****

Set the image size

****

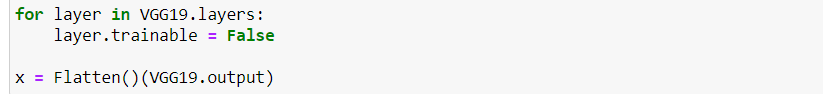
For one of the models, we will use it as a simple feature extractor by freezing all the five convolution blocks to make sure their weights don’t get updated after each epoch as we train our own model.

Here, we have considered images of dimensions (180,180,3).

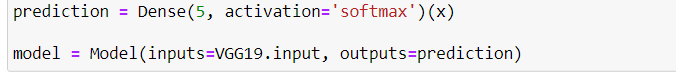
Also, we have assigned include\_top = False because we are using the convolution layer for features extraction and wants to train fully connected layer for our images classification(since it is not the part of Imagenet dataset)

Flatten layer flattens the input. Does not affect the batch size.

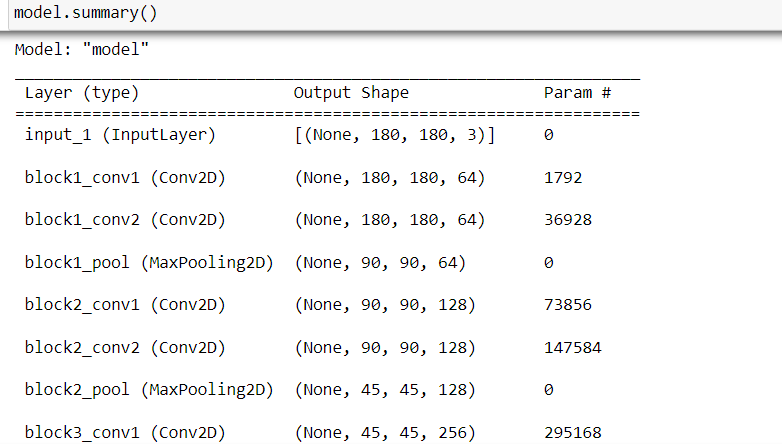
****

****

A dense layer is a deeply connected neural network layer. It is the most common and frequently used layer. Let us create a model object named model with inputs as VGG19.input and output as dense layer.

****

summary to get the full information about the model and its layers.

****

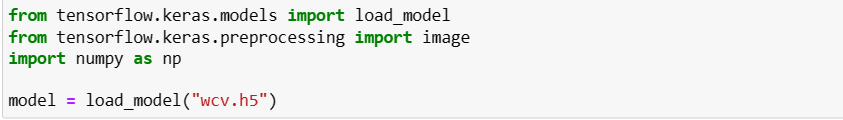
Train our model with our image dataset. The model is trained for 20 epochs and after every epoch, the current model state is saved if the model has the least loss encountered till that time. We can see that the training loss decreases in almost every epoch till 10 epochs and probably there is further scope to improve the model. it\_generator functions used to train a deep learning neural network

****

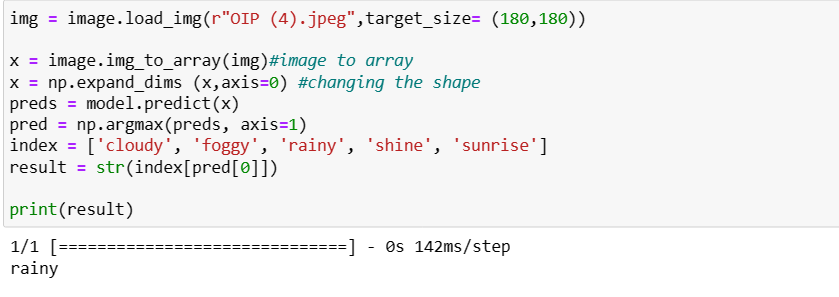
The model is saved with .h5 extension as follows

****

Evaluation is a process during the development of the model to check whether the model is the best fit for the given problem and corresponding data. Load the saved model using load\_model

****

Taking an image as input and checking the results

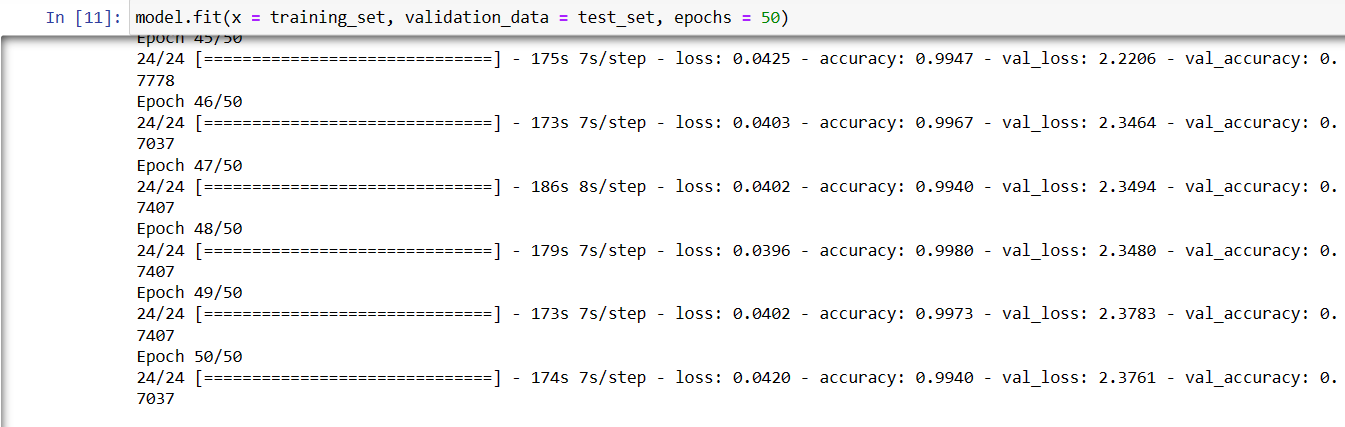
****

So, our model has predicted the class of input correctly as rainy.

**6. RESULTS**

6.1 Performance Metrics

We got an accuracy of 99.4% in the Training set and 74% in Test



**7. ADVANTAGES & DISADVANTAGES**

Advantages of Automated Weather Classification using Transfer Learning:

* Improved Accuracy: Transfer learning allows leveraging pre-trained deep learning models that have been trained on large-scale datasets, enabling better generalization and accuracy in weather classification.
* Reduced Training Time: By utilizing pre-trained models as a starting point, transfer learning significantly reduces the time and computational resources required for training weather classification models from scratch.
* Handling Limited Data: Transfer learning helps address the challenge of limited labeled weather data by transferring knowledge from related tasks or datasets, enhancing the model's performance even with a smaller dataset.
* Scalability: The use of transfer learning enables the system to handle a wide range of weather conditions, including variations in lighting, seasons, and geographical locations, making it scalable for diverse weather classification tasks.
* Interpretability: Transfer learning models can provide insights into the classification process by visualizing learned features, contributing to the interpretability of weather classification results.

Disadvantages of Automated Weather Classification using Transfer Learning:

* Dependency on Pre-trained Models: The performance of the weather classification system heavily relies on the quality and suitability of the pre-trained models used for transfer learning. Inadequate or mismatched pre-trained models may lead to suboptimal results.
* Overfitting Risk: Fine-tuning a pre-trained model with a small or biased weather dataset may increase the risk of overfitting, where the model becomes excessively specialized to the training data and fails to generalize well to new or unseen weather images.
* Limited Adaptability: Pre-trained models might have learned features that are not directly applicable to weather classification. Fine-tuning and transfer learning methods may not always capture the specific nuances and complexities of weather patterns, limiting adaptability in some cases.
* Bias Transfer: If the pre-trained model has inherent biases from the original training dataset, those biases can be transferred to the weather classification model, potentially leading to biased results.
* Limited Transparency: Deep learning models used in transfer learning can be considered as "black boxes," making it challenging to understand and explain the decision-making process behind weather classification predictions.

**8. CONCLUSION**

In conclusion, the Automated Weather Classification using Transfer Learning project presents a promising solution for automating weather classification tasks. By leveraging transfer learning techniques and pre-trained deep learning models, the project aims to enhance weather classification's accuracy, scalability, and efficiency. Throughout the project, various components were developed and integrated, including data collection and preprocessing, model selection and fine-tuning, training, and validation, and the implementation of a user-friendly interface. The system allows users to upload weather images and obtain accurate weather classification results.

**9. FUTURE SCOPE**

The Automated Weather Classification using Transfer Learning project has a promising future scope for further improvements and advancements. Some potential future directions and areas of focus include: Integration of Additional Data Sources, Ensembling and Model Fusion, Advanced Transfer Learning Techniques, Handling Multi-label Classification, Interpretability and Explainability, Integration with IoT and Sensor Networks, and Mobile and Edge Deployment can continue to evolve and contribute to advancements in weather classification technology, providing more accurate and efficient tools for weather forecasting, climate monitoring, and related applications.

**10. APPENDIX**

GitHub

[IBM--12018-1682491633/ at main · naanmudhalvan-SI/IBM--12018-1682491633 · GitHub](https://github.com/naanmudhalvan-SI/IBM--12018-1682491633?search=1)

Project Video Demo Link

https://youtu.be/FUxDUgkxMLM