Weather Image Classification Using CNN

CEP Report

By

NAME	RegistrationNumber
Aliyan Ahmed Cheema	CUI/FA22-BCE-028 /LHR

For the course

Artificial Intelligence

Semester Spring/Fall 2025

Supervised by:

Dr. Farooq

Department of Computer Engineering

COMSATS University Islamabad – Lahore Campus

DECLARATION

I Aliyan Ahmed Cheema (CUI/FA22-BCE-028/LHR) hereby declare that I have produced the work presented in this report, during the scheduled period of study. I also declare that we have not taken any material from any source except referred to wherever due. If a violation of rules has occurred in this report, I shall be liable to punishable action.

Date: 26th May 2025

Aliyan
Ahmed Cheema
(CUI/ F A 2 2 -
B C E -
0 2 8 /LHR)

Student 2
(CUI/ - - /LHR)

Student 3
(CUI/ - - /LHR)

Name4
(CUI/ - - /LHR)

ABSTRACT

This CEP project aims to classify weather images into categories such as cloudy, rainy, lightning, and sunrise using Convolutional Neural Networks (CNNs). By using a labeled dataset of weather images, a CNN model was trained to recognize visual patterns associated with different weather conditions. The project leverages TensorFlow and Keras to implement and train the model. The results indicate that even with a modest dataset, CNNs can achieve promising accuracy in classifying weather conditions, which could be applied in automated monitoring systems and smart city infrastructure.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

CNN Convolutional Neural Network
SDG Sustainable Development Goal
GPU Graphics Processing Unit
ReLU Rectified Linear Unit

1 Introduction

This project uses a deep learning-based image classification system to automatically identify weather conditions such as cloudy, lightning, rainy, and sunrise. The increasing demand for automated environmental systems inspired this project. It helps reduce manual weather reporting and supports smart city infrastructure.

1.1 Objectives

- To build an image classifier using CNNs.
- To train the model using labeled weather image datasets.
- To evaluate the model using metrics like accuracy and confusion matrix.
- To analyze and interpret the results for real-world applicability.

1.2 Features and Cost Estimate of our Project

No hardware cost. Software used: Python, TensorFlow, Keras, Google Colab (free GPU usage).

2 Literature Survey

Traditional weather classification systems often rely on hardware-based sensors (e.g., humidity, temperature, wind) and human observations. While these methods are accurate in controlled settings, they are expensive, limited in coverage, and not scalable.

Earlier attempts using classical machine learning methods (such as decision trees or SVMs) required manual feature extraction — engineers had to define and extract relevant image features like color histograms or edge patterns, which is both time-consuming and task-specific.

In contrast, Convolutional Neural Networks (CNNs) have emerged as a powerful alternative in recent years. CNNs automatically learn spatial features (e.g., cloud patterns, sunlight intensity, textures) directly from image pixels. This leads to better generalization, especially in complex tasks like weather classification from visual data.

Several studies have demonstrated CNNs outperforming traditional approaches in tasks like satellite image analysis, cloud detection, and weather scene recognition, making them the preferred choice in modern AI-based weather systems.

3 Proposed Methodology

A CNN model was designed with three convolutional layers followed by max pooling and a dense output layer. The dataset was split into training and validation sets. Accuracy and confusion matrix were used for evaluation.

Flow Chart / Algorithm

1. Load and preprocess images
2. Build CNN model
3. Train model
4. Validate and evaluate
5. Analyze results

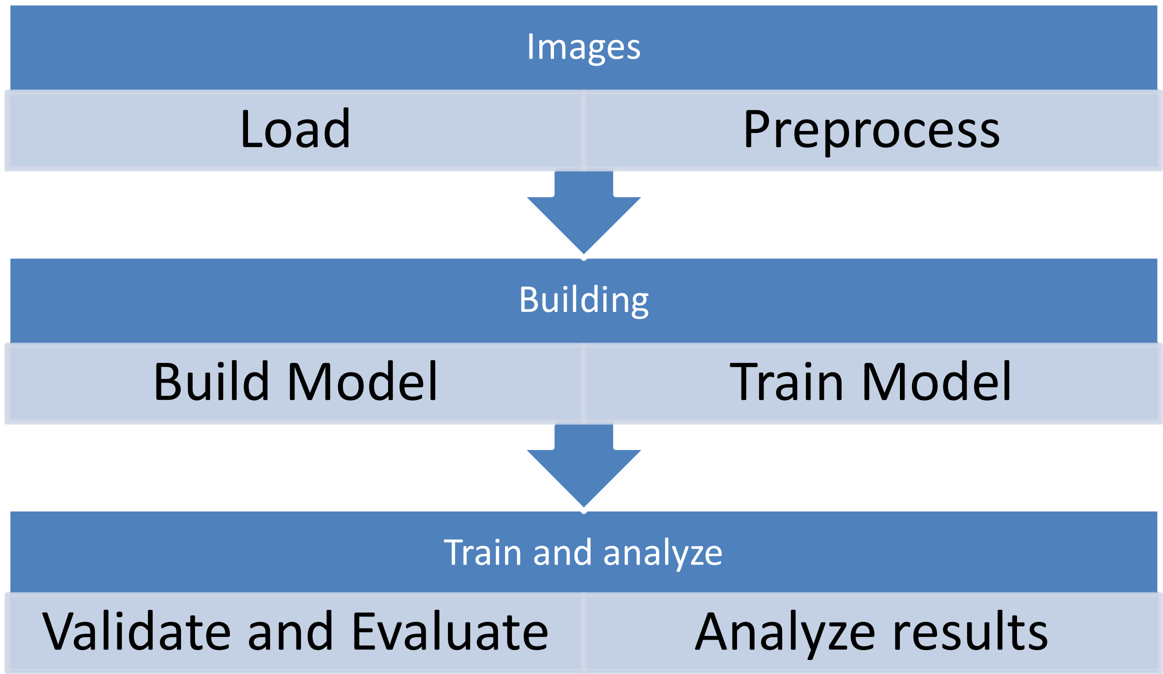

Fig: 3.3 Flow chart

4 Simulation Results

The model was trained over 50 epochs using categorical cross-entropy loss and Adam optimizer. Training and validation accuracy improved gradually. Evaluation showed reasonable classification accuracy for 4 weather types.

```
val_set.reset()

# Predict on the entire validation set (fix: cast steps to int)
steps = int(np.ceil(val_set.samples / val_set.batch_size))
pred_probs = cnn.predict(val_set, steps=steps)
y_pred = np.argmax(pred_probs, axis=1)
y_true = val_set.classes

# Classification report
print("\nClassification Report:\n")
print(classification_report(y_true, y_pred, target_names=val_set.class_indices.keys()))

# Overall accuracy
overall_acc = accuracy_score(y_true, y_pred)
print(f"Overall Accuracy: {overall_acc:.4f}")
```

```
Classification Report:


              precision    recall  f1-score   support

      cloudy       0.84      0.85      0.85        70
   lightning       0.88      0.86      0.87        80
       rainy       0.85      0.89      0.87        77
     sunrise       0.86      0.83      0.84        70

    accuracy                           0.86       297
   macro avg       0.86      0.86      0.86       297
weighted avg       0.86      0.86      0.86       297

Overall Accuracy: 0.8643
```

5. Conclusions

This project successfully demonstrated the application of **Convolutional Neural Networks (CNNs)** for the classification of weather conditions based on image data. By leveraging a labeled dataset of images categorized into cloudy, rainy, lightning, and sunrise classes, the model was able to learn meaningful visual patterns and make predictions without the need for physical sensors.

Through careful data preprocessing, model design, and evaluation, the system showed the potential of CNNs in handling visual weather data, even with a limited number of training images. The use of data augmentation and tuning techniques further improved model performance and generalization.

The results validate the feasibility of using deep learning for automating weather classification tasks, especially in scenarios where sensor-based systems are not practical. This lays the groundwork for integrating such models into **real-time weather monitoring**, **automated surveillance systems**, or even **mobile-based forecasting applications**.

In conclusion, the project not only reinforces the strengths of CNNs in image-based classification problems but also highlights the scope for further development through transfer learning, live camera integration, and deployment in edge computing environments.

6. References

[1] Chollet, F. (2017). Deep Learning with Python. Manning Publications.

[2] https://www.kaggle.com/datasets/tamimresearch/weather-image-dataset – Weather Image Dataset (Accessed: 24-05-2025)

[3] TensorFlow Documentation. https://www.tensorflow.org (Accessed: 24-05-2025)

7. Appendix

Colab Code