2.5: Visual Applications of Machine Learning January 6, 2025
Isaac Contreras

Real-World Applications of Machine Learning

Part 1 – Handwriting Recognition

After tuning the model, an 80% accuracy was achieved.

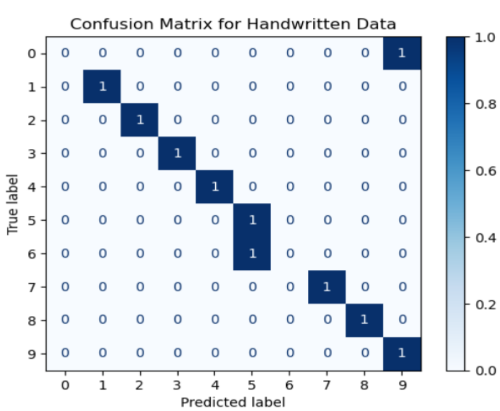
```
# Predict handwritten images
handwritten_predictions = model.predict(handwritten_images)

# Get predicted class for each image
predicted_digits = np.argmax(handwritten_predictions, axis=1)

# Print the results
print("Handwritten Image Predictions (After Retraining):")
for i, (true_label, predicted_label) in enumerate(zip(handwritten_labels, predicted_digits)):
    print(f"Image {i}: True Label = {true_label}, Predicted Label = {predicted_label}")

# Calculate accuracy on handwritten data
accuracy = np.mean(predicted_digits == handwritten_labels) * 100
print(f"Accuracy on handwritten data (After Retraining): {accuracy:.2f}%")
```

```
1/1 ━━━━━━━━━━━━━━━━━━━━ 1s 679ms/step
Handwritten Image Predictions (After Retraining):
Image 0: True Label = 0, Predicted Label = 9
Image 1: True Label = 1, Predicted Label = 1
Image 2: True Label = 2, Predicted Label = 2
Image 3: True Label = 3, Predicted Label = 3
Image 4: True Label = 4, Predicted Label = 4
Image 5: True Label = 5, Predicted Label = 5
Image 6: True Label = 6, Predicted Label = 5
Image 7: True Label = 7, Predicted Label = 7
Image 8: True Label = 8, Predicted Label = 8
Image 9: True Label = 9, Predicted Label = 9
Accuracy on handwritten data (After Retraining): 80.00%
```

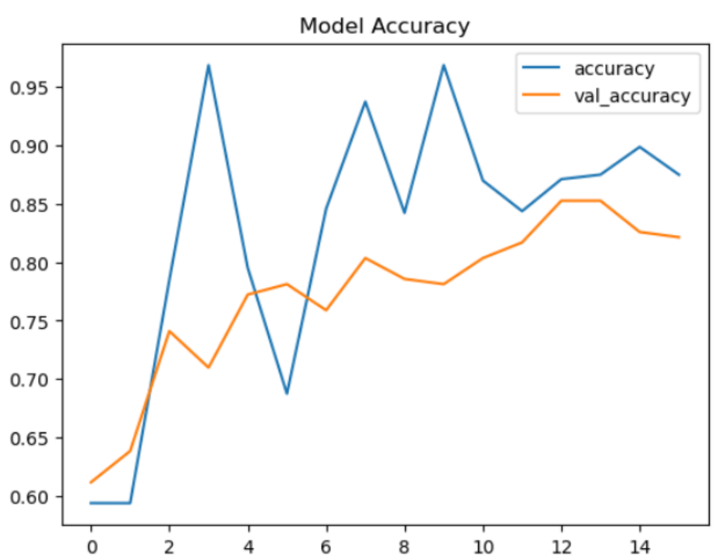


Part 2 – Radar Recognition

To classify weather images into categories the "Cloudy," "Rain," "Shine," and "Sunrise," I built and trained a Convolutional Neural Network (CNN) on a dataset of 1,125 images. The model used three convolutional layers with max-pooling and dropout for regularization, followed by dense layers to output predictions. The images were resized to 250x250 pixels, and the dataset was split into 80% training and 20% validation.

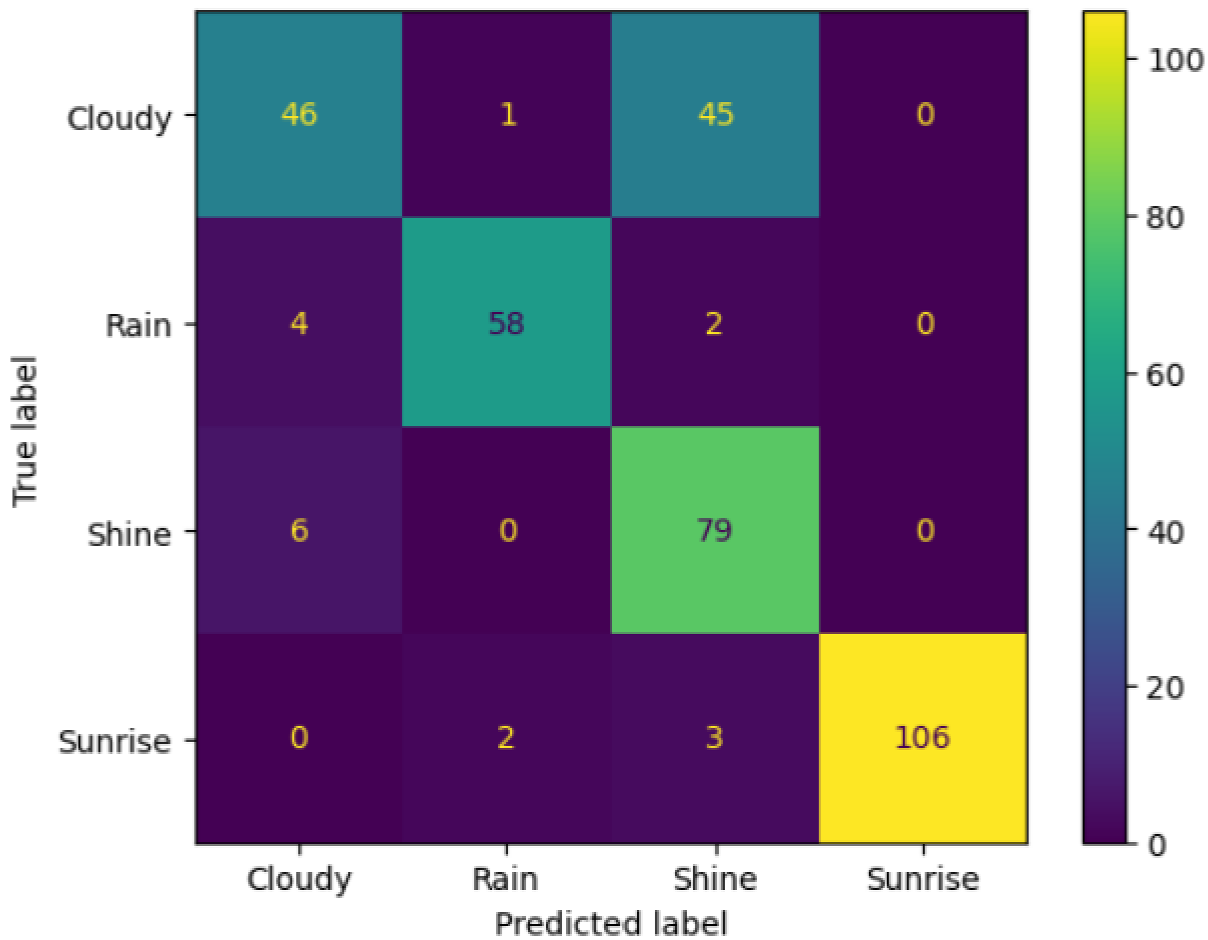
The model was trained for 16 epochs, with results showing:

- Training Accuracy: 87.5%
- Validation Accuracy: 82.1%
- Training Loss: 0.30
- Validation Loss: 0.42



The performance indicates that the model learned the patterns in the training data well, though the slight gap between training and validation metrics suggests some overfitting.

A confusion matrix was generated to better understand the model's strengths and weaknesses. The model performed well in classifying "Sunrise" and "Shine" images, but struggled to distinguish between "Cloudy" and "Shine" or "Cloudy" and "Rain." This highlights areas for improvement, such as refining the dataset or further tuning hyperparameters.



The plots of training and validation accuracy and loss over epochs showed steady improvements in accuracy but some fluctuation in validation loss, reflecting the challenges of generalizing to unseen data.

Proposal for Using GANs in Weather Prediction:

Generative Adversarial Networks (GANs) could greatly enhance weather prediction and analysis. For example:

1. GANs can generate synthetic satellite images to monitor storms and precipitation patterns, which would be useful for real-time weather tracking.
2. They could simulate future climate scenarios based on historical data, helping predict the frequency and severity of extreme weather events.
3. GANs could automate the segmentation and labeling of radar and satellite imagery to identify meteorological phenomena like hurricanes or storm systems.

These applications have been explored in recent research, such as:

- GANs for Climate Modeling
- AI for Climate Change

This project demonstrates the potential of CNNs in weather classification and points to the vast opportunities GANs offer for advancing weather prediction and analysis.