Image Based-Emotion Facial Recognition

Presented by
A. B. S. Hasan - u3237372 & Somya Shukla– u3238876

Outline

1. Introduction & Project Overview
2. Data Description & Analysis Techniques
3. Results & Insights
4. Conclusion

Project Overview

Problem Statement

1) Develop a model to recognizing seven different emotions from image

Motivation

1) Enhancing human-computer interactions, mental health support, and user experiences.

2) Improving communication, empathy, and overall well-being. Contributing to artificial intelligence development and addressing real-world needs.

Relation to Previous Work

- ***“A ResNet deep learning based facial recognition design for future multimedia applications” by Durga & Rajesh***
  - Their research emphasizes how important sophisticated neural networks are to accurately identify emotions.

- ***“Facial expression recognition from image based on hybrid features understanding” by Wang, F., Lv, J., Ying, G., Chen, S., & Zhang***
  - Talked about a method focused on hybrid feature understanding for facial expression recognition. This approach combines traditional image-based features with modern techniques, indicating the importance of feature engineering

- ***“Deep learning-based facial emotion recognition for human–computer interaction applications” by T. Chowdary, Nguyen, and Hemanth***
  - Investigated the use of deep learning for face emotion recognition, demonstrating how it might improve computer-human interaction

Project Approach

Data preprocessing

Training Strategy

Model Architecture

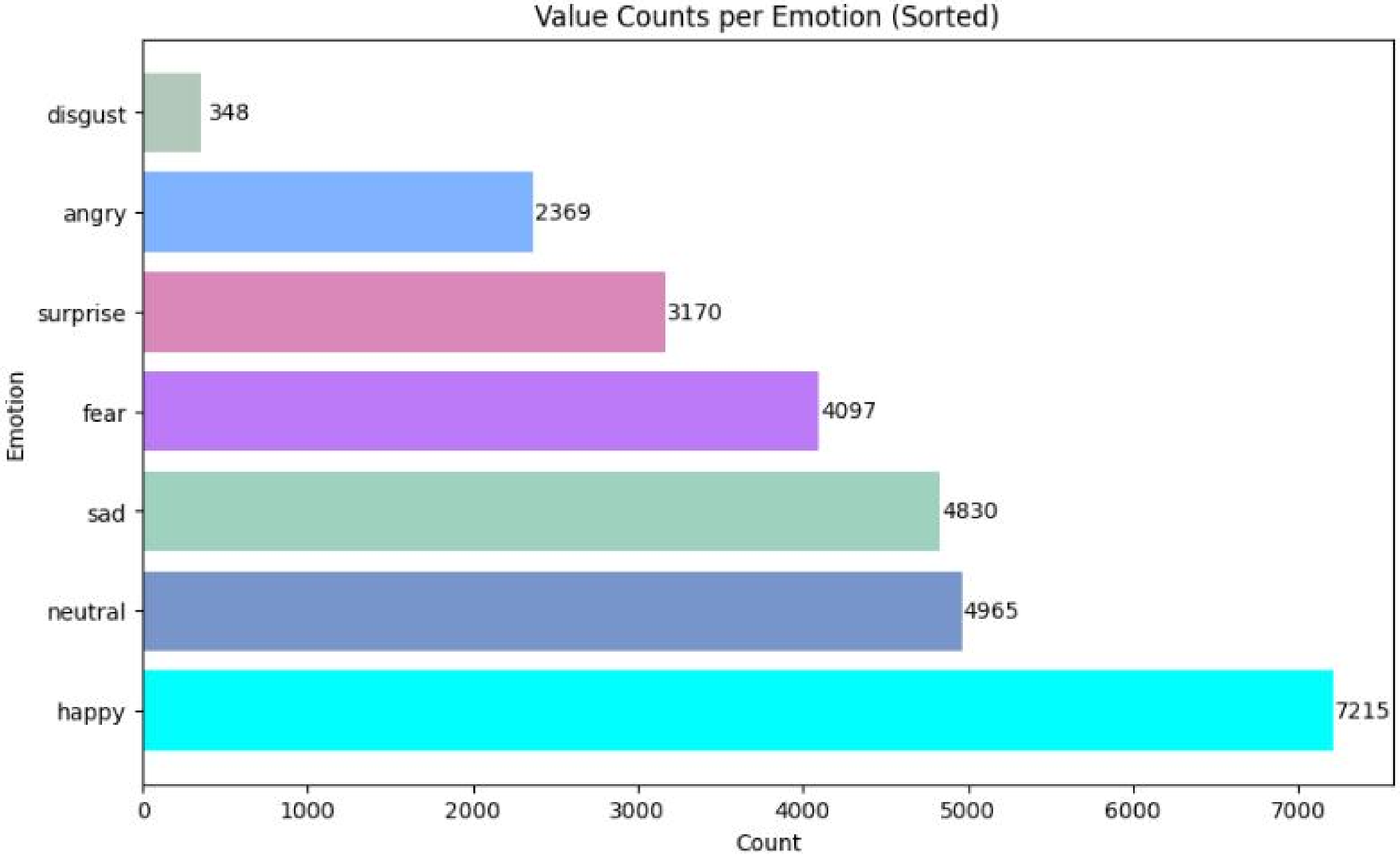
Evaluation metrics

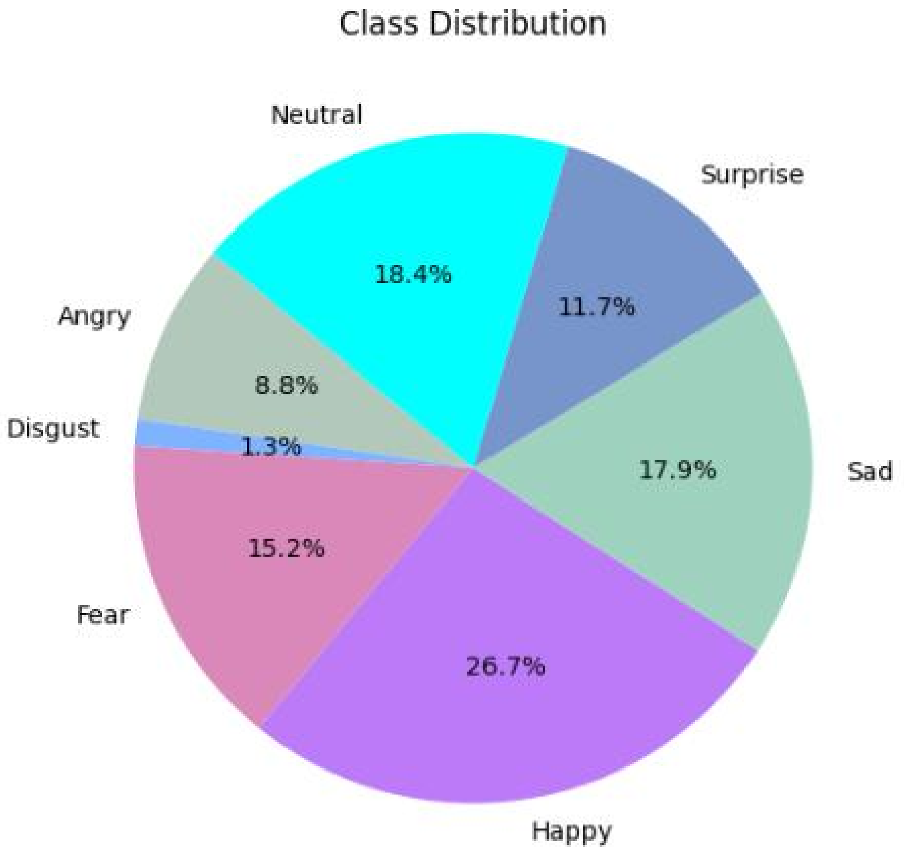
Progress and achievements

Data Source

- Kaggle Dataset by MANAS SAMBARE
- Original Data Set : 7 Emotions, 34156 Images
- Emotion List: Angry, Disgust, Fear, Happy, Sad, Surprise, Neutral
- This dataset is consist of 48x48 pixel grayscale images of faces

Training Class Distribution





Performance Comparison

Model Name	Test Accuracy	F1	Precision	Recall
Custom CNN Model	29.17%	10.88%	7.42%	22.51%
ResNet50	24.71%	9.80%	6.00%	24.71%
MobileNet	**42.98%**	**18.23%**	**18.23%**	**20.51%**
VGG16	38.62%	17.99%	18.10%	20.03%
Inception3	34.01%	17.24%	17.63%	20.35%

MobileNet Model Structure

```python
# Define the number of classes for your classification task
num_classes = 7

# Load the pre-trained MobileNet model with weights pre-trained on ImageNet
base_model = MobileNet(weights='imagenet', include_top=False, input_shape=(image_size, image_size, 3

# Freeze the layers of the pre-trained model
for layer in base_model.layers:
    layer.trainable = False

# Create a custom top model with additional dense layers and dropout
x = base_model.output
x = GlobalAveragePooling2D()(x)
x = Dense(1024, activation='relu')(x)
x = Dropout(0.5)(x)
x = Dense(512, activation='relu')(x)
x = Dropout(0.5)(x)
predictions = Dense(num_classes, activation='softmax')(x)

# Combine the base model and the custom top model
model = Model(inputs=base_model.input, outputs=predictions)

# Define early stopping callback
early_stopping = EarlyStopping(monitor='val_loss', patience=5, restore_best_weights=True)

# Compile the model
model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])

# Print a summary of the model architecture
model.summary()
```

1) **Data Pre-Processing**

Random horizontal and vertical shifts (20%)

Random rotation (5 degrees)

Random shear (20%), Horizontal flip & Vertical flip

2) **Model Architecture**

- MobileNet (pre-trained on ImageNet)
- Global Average Pooling
- Dense (1024 units) + Dropout 50%
- Dense (512 units) + Dropout 50%
- Dense (output layer with num_classes units and softmax activation)

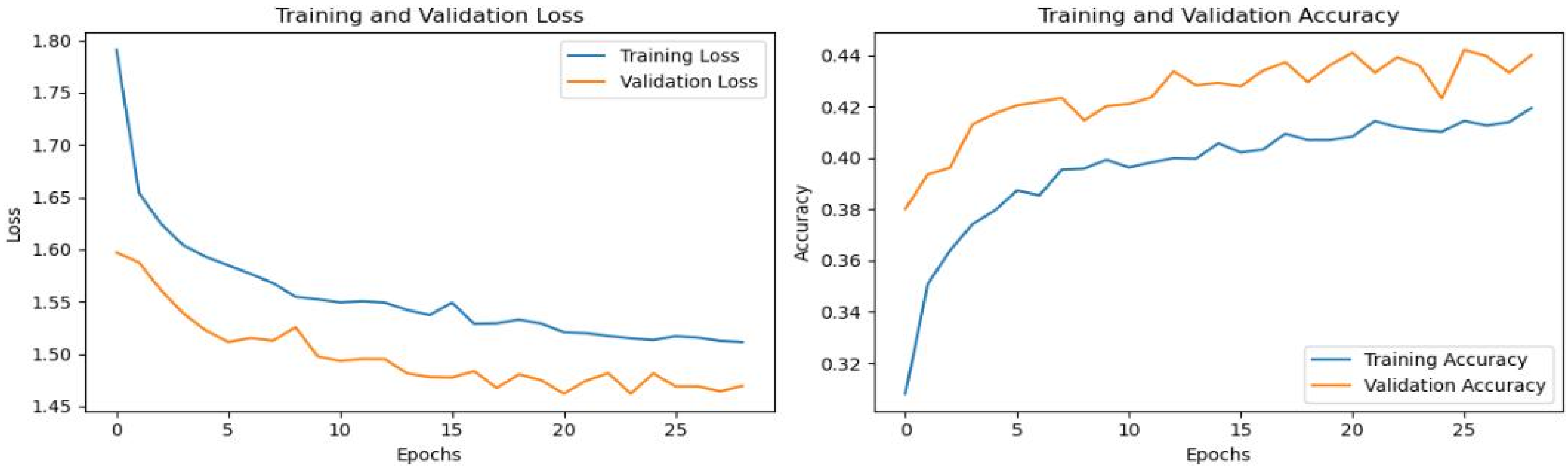
3) **Training**

- You used an early stopping callback to prevent overfitting, which monitors the validation loss.
- Compiled the model using the Adam optimizer and categorical cross-entropy loss.
- Trained the model for 50 epochs.

4) **Evaluation**

- Evaluated the model on the testing dataset and printed the test accuracy.

MobileNet : Training vs Validation



- Used early stopping to prevent overfitting. Both training and testing loss decreased over time.

MobileNet Model Report

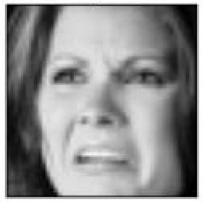
- The model failed to correctly classify the 'Disgust' class.
- This class is particularly challenging to classify, even for human observers.
- Most of the instances were misclassified as 'Happy'.

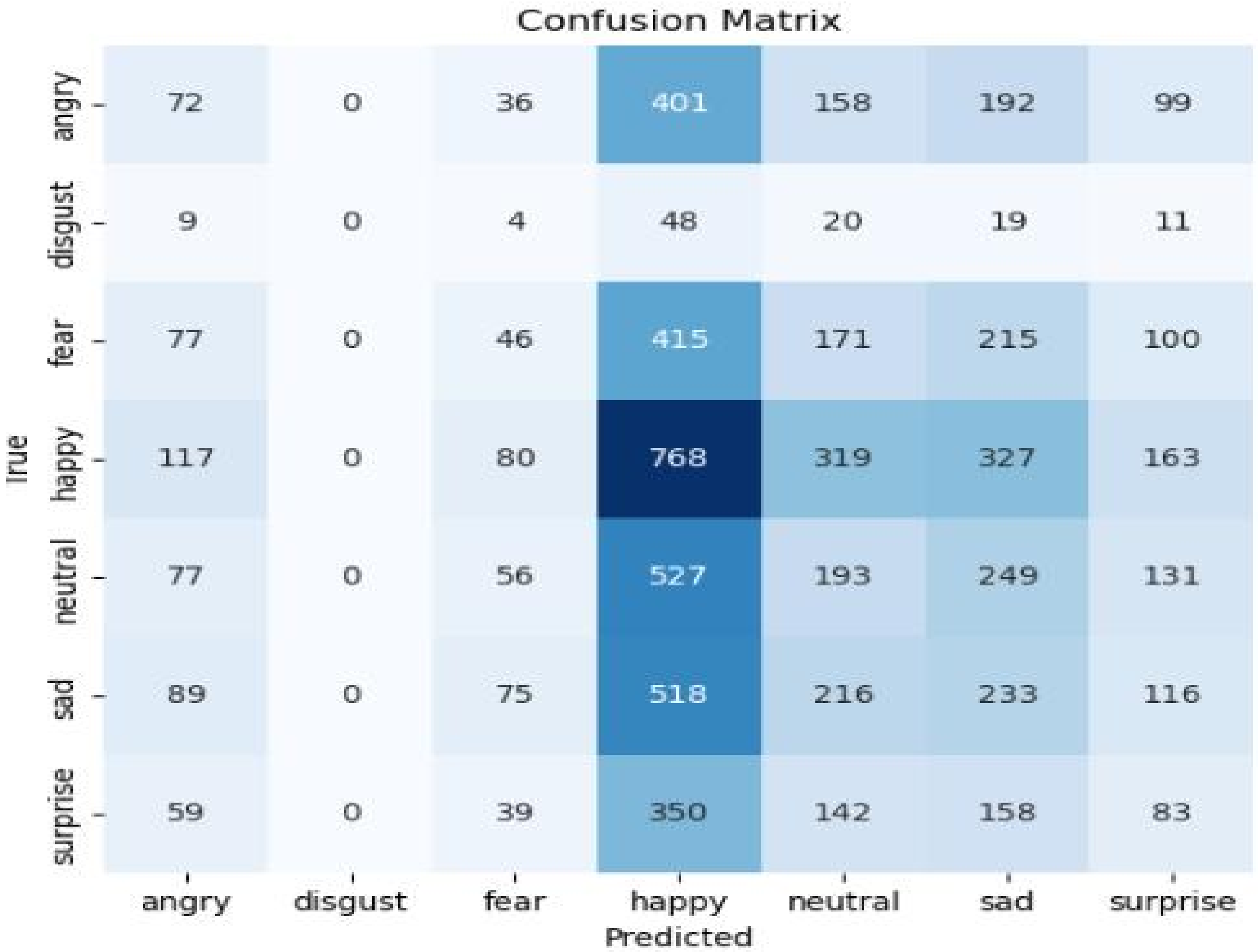
	precision	recall	f1-score	support
angry	0.14	0.07	0.10	958
disgust	1.00	0.00	0.00	111
fear	0.12	0.04	0.06	1024
happy	0.25	0.43	0.32	1774
neutral	0.18	0.18	0.18	1233
sad	0.20	0.22	0.21	1247
surprise	0.12	0.10	0.11	831
accuracy			0.20	7178
macro avg	0.29	0.15	0.14	7178
weighted avg	0.19	0.20	0.18	7178


disgust

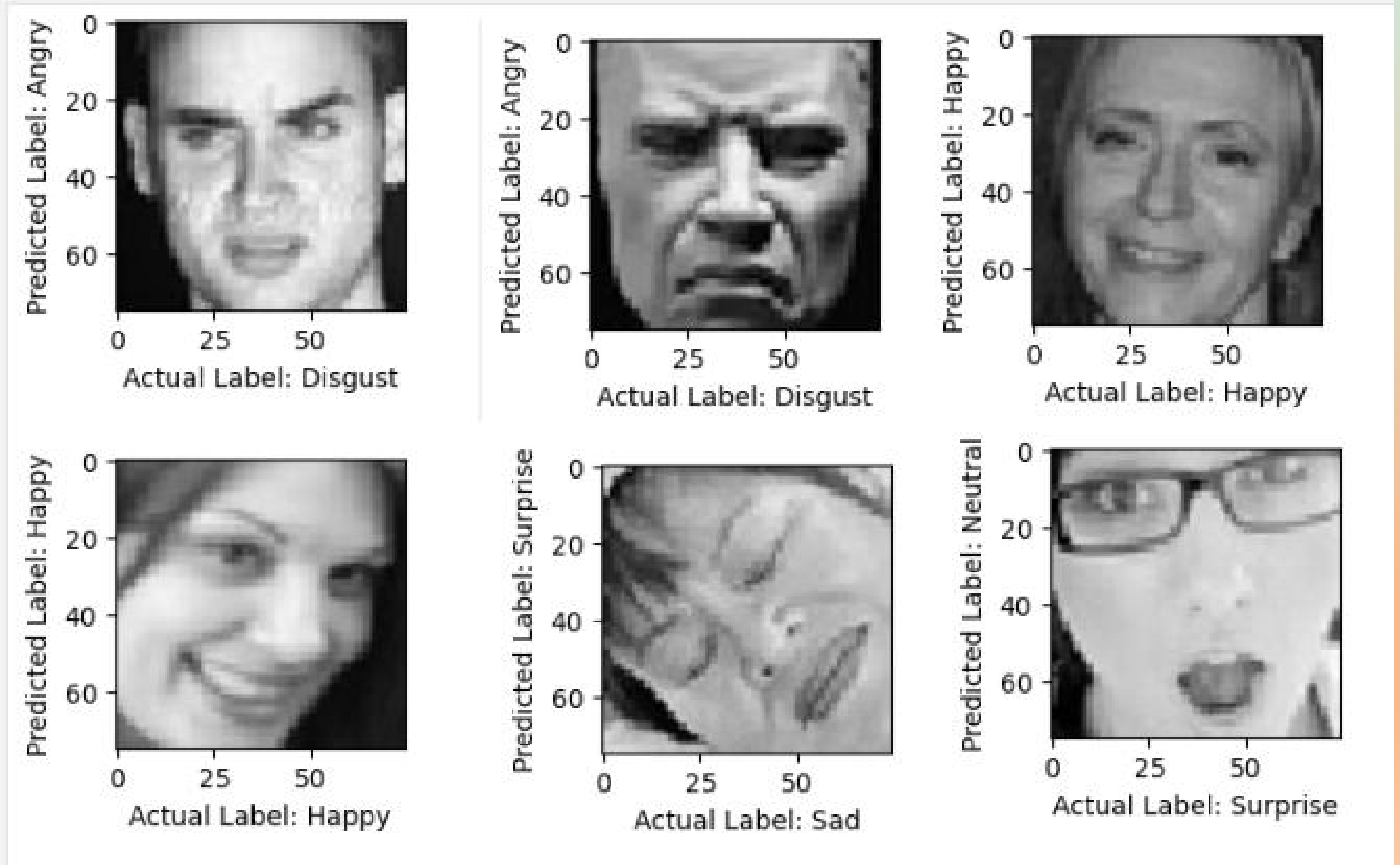

disgust


disgust


disgust



MobileNet Model's Performance



Challenges and Future Work

Challenges

- Long Training Time
- Computational Resources
- Model Selection
- Data Imbalance

Future Work

- Try Advanced CNN Architectures like EfficientNet, Vision Transformers (ViT), or DenseNet.
- Hybrid Models: Explore the potential of combining CNNs with other model types for improved performance.
- Hyperparameter Tuning: Fine-tune hyperparameters to optimize model performance.
- Data Augmentation and Dataset Size Expansion: Apply data augmentation techniques to increase dataset size and diversity.
- Further Evaluation of CNN Architecture Reconfigurations: Continuously assess and adjust the CNN architecture for enhanced results.

Q&A