```
PS C:\Users\gudur\Downloads> python facial_expression_notebook_final.py
2025-10-04 14:54:56.150391: I tensorflow/core/util/port.cc:153] oneDNN custom operations are on. You may see slightly different numerical results due to floating-point round-off errors from different computation orders. To turn them off, set the environment variable `TF_ENABLE_ONEDNN_OPTS=0`.
2025-10-04 14:55:05.516160: I tensorflow/core/util/port.cc:153] oneDNN custom operations are on. You may see slightly different numerical results due to floating-point round-off errors from different computation orders. To turn them off, set the environment variable `TF_ENABLE_ONEDNN_OPTS=0`.
TensorFlow Version: 2.20.0
GPU Available: []
Libraries imported successfully!
Configuration set successfully!
Dataset path: C:\Users\gudur\Downloads\archive
Classes: ['angry', 'disgust', 'fear', 'happy', 'neutral', 'sad', 'surprise']
============================================================
LOADING DATASET
============================================================

Loading training data from C:\Users\gudur\Downloads\archive\train
Loading angry images from C:\Users\gudur\Downloads\archive\train\angry
  → Loaded 7990 angry images
Loading disgust images from C:\Users\gudur\Downloads\archive\train\disgust
  → Loaded 872 disgust images
Loading fear images from C:\Users\gudur\Downloads\archive\train\fear
  → Loaded 8194 fear images
Loading happy images from C:\Users\gudur\Downloads\archive\train\happy
  → Loaded 14430 happy images
Loading neutral images from C:\Users\gudur\Downloads\archive\train\neutral
  → Loaded 9930 neutral images
Loading sad images from C:\Users\gudur\Downloads\archive\train\sad
  → Loaded 9660 sad images
Loading surprise images from C:\Users\gudur\Downloads\archive\train\surprise
  → Loaded 6342 surprise images

Loading testing data from C:\Users\gudur\Downloads\archive\test
Loading angry images from C:\Users\gudur\Downloads\archive\test\angry
  → Loaded 1916 angry images
Loading disgust images from C:\Users\gudur\Downloads\archive\test\disgust
  → Loaded 222 disgust images
Loading fear images from C:\Users\gudur\Downloads\archive\test\fear
  → Loaded 2048 fear images
Loading happy images from C:\Users\gudur\Downloads\archive\test\happy
  → Loaded 3548 happy images
Loading neutral images from C:\Users\gudur\Downloads\archive\test\neutral
  → Loaded 2466 neutral images
Loading sad images from C:\Users\gudur\Downloads\archive\test\sad
  → Loaded 2494 sad images
Loading surprise images from C:\Users\gudur\Downloads\archive\test\surprise
  → Loaded 1662 surprise images

Dataset loaded successfully!
Training images: (57418, 48, 48, 1)
Training labels: (57418,)
Test images: (14356, 48, 48, 1)
Test labels: (14356,)

============================================================
DATA PREPROCESSING
```

==========
Data shape: (57418, 48, 48, 1)
Labels shape: (57418, 7)
Data shape: (14356, 48, 48, 1)
Labels shape: (14356, 7)

Training Set Class Distribution:
angry: 7990 images
disgust: 872 images
fear: 8194 images
happy: 14430 images
neutral: 9930 images
sad: 9660 images
surprise: 6342 images

Test Set Class Distribution:
angry: 1916 images
disgust: 222 images
fear: 2048 images
happy: 3548 images
neutral: 2466 images
sad: 2494 images
surprise: 1662 images

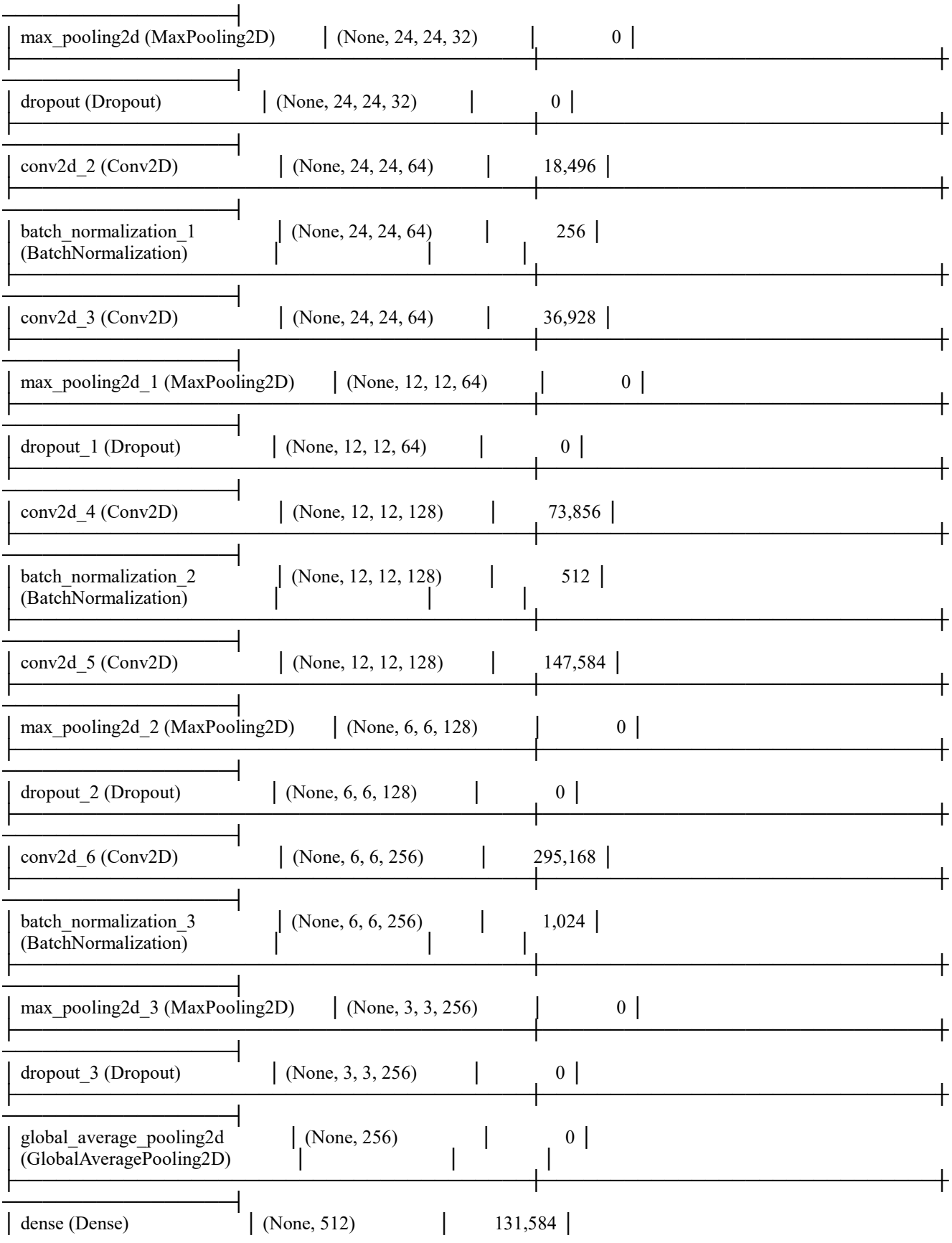
Final Dataset Split:
Training set: 45934 images
Validation set: 11484 images
Test set: 14356 images

Sample images from training set:

==========
DATA AUGMENTATION SETUP
==========
Data augmentation setup complete!

==========
MODEL CREATION
==========
2025-10-04 15:04:53.257953: I tensorflow/core/platform/cpu_feature_guard.cc:210] This TensorFlow binary is optimized to use available CPU instructions in performance-critical operations.
To enable the following instructions: SSE3 SSE4.1 SSE4.2 AVX AVX2 AVX512F AVX512_VNNI FMA, in other operations, rebuild TensorFlow with the appropriate compiler flags.
Model Architecture:
Model: "sequential"

| Layer (type) | Output Shape | Param # |
|---|---|---|
| conv2d (Conv2D) | (None, 48, 48, 32) | 320 |
| batch_normalization (BatchNormalization) | (None, 48, 48, 32) | 128 |
| conv2d_1 (Conv2D) | (None, 48, 48, 32) | 9,248 |


max_pooling2d (MaxPooling2D) | (None, 24, 24, 32) | 0
dropout (Dropout) | (None, 24, 24, 32) | 0
conv2d_2 (Conv2D) | (None, 24, 24, 64) | 18,496
batch_normalization_1 (BatchNormalization) | (None, 24, 24, 64) | 256
conv2d_3 (Conv2D) | (None, 24, 24, 64) | 36,928
max_pooling2d_1 (MaxPooling2D) | (None, 12, 12, 64) | 0
dropout_1 (Dropout) | (None, 12, 12, 64) | 0
conv2d_4 (Conv2D) | (None, 12, 12, 128) | 73,856
batch_normalization_2 (BatchNormalization) | (None, 12, 12, 128) | 512
conv2d_5 (Conv2D) | (None, 12, 12, 128) | 147,584
max_pooling2d_2 (MaxPooling2D) | (None, 6, 6, 128) | 0
dropout_2 (Dropout) | (None, 6, 6, 128) | 0
conv2d_6 (Conv2D) | (None, 6, 6, 256) | 295,168
batch_normalization_3 (BatchNormalization) | (None, 6, 6, 256) | 1,024
max_pooling2d_3 (MaxPooling2D) | (None, 3, 3, 256) | 0
dropout_3 (Dropout) | (None, 3, 3, 256) | 0
global_average_pooling2d (GlobalAveragePooling2D) | (None, 256) | 0
dense (Dense) | (None, 512) | 131,584

| | | |
|---|---|---|
| batch_normalization_4 (BatchNormalization) | (None, 512) | 2,048 |
| dropout_4 (Dropout) | (None, 512) | 0 |
| dense_1 (Dense) | (None, 256) | 131,328 |
| batch_normalization_5 (BatchNormalization) | (None, 256) | 1,024 |
| dropout_5 (Dropout) | (None, 256) | 0 |
| dense_2 (Dense) | (None, 7) | 1,799 |

Total params: 851,303 (3.25 MB)
Trainable params: 848,807 (3.24 MB)
Non-trainable params: 2,496 (9.75 KB)

==========
MODEL COMPILATION
==========
Model compiled with callbacks setup!

==========
MODEL TRAINING
==========
Starting model training...
Training for maximum 100 epochs
Training will stop early if validation accuracy doesn't improve
Steps per epoch: 1435
Validation steps: 358
Epoch 1/100
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 0s 233ms/step - accuracy: 0.2006 - loss: 2.2725
Epoch 1: val_accuracy improved from None to 0.28928, saving model to best_facial_expression_model.h5
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `keras.saving.save_model(model)`. This file format is considered legacy. We recommend using instead the native Keras format, e.g. `model.save('my_model.keras')` or `keras.saving.save_model(model, 'my_model.keras')`.
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 355s 243ms/step - accuracy: 0.2222 - loss: 2.0092 - val_accuracy: 0.2893 - val_loss: 1.7536 - learning_rate: 0.0010
Epoch 2/100
  1/1435 ━━━━━━━━━━━━━━━━━━━━ 4:07 173ms/step - accuracy: 0.3750 - loss: 1.8479
Epoch 2: val_accuracy did not improve from 0.28928
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 14s 9ms/step - accuracy: 0.3750 - loss: 1.8479 - val_accuracy: 0.2886 - val_loss: 1.7558 - learning_rate: 0.0010
Epoch 3/100
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 0s 157ms/step - accuracy: 0.2859 - loss: 1.7453
Epoch 3: val_accuracy improved from 0.28928 to 0.36749, saving model to best_facial_expression_model.h5
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `keras.saving.save_model(model)`. This file format is considered legacy. We recommend using instead the native Keras format, e.g. `model.save('my_model.keras')`

or `keras.saving.save_model(model, 'my_model.keras')`.
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 238s 166ms/step - accuracy: 0.3091 - loss: 1.7078 - val_accuracy: 0.3675 - val_loss: 1.6160 - learning_rate: 0.0010
Epoch 4/100
  1/1435 ━━━━━━━━━━━━━━━━━━━━ 3:58 167ms/step - accuracy: 0.4062 - loss: 1.5297
Epoch 4: val_accuracy improved from 0.36749 to 0.37893, saving model to best_facial_expression_model.h5
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `keras.saving.save_model(model)`. This file format is considered legacy. We recommend using instead the native Keras format, e.g. `model.save('my_model.keras')` or `keras.saving.save_model(model, 'my_model.keras')`.
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 13s 9ms/step - accuracy: 0.4062 - loss: 1.5297 - val_accuracy: 0.3789 - val_loss: 1.5954 - learning_rate: 0.0010
Epoch 5/100
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 0s 181ms/step - accuracy: 0.3934 - loss: 1.5629
Epoch 5: val_accuracy improved from 0.37893 to 0.44990, saving model to best_facial_expression_model.h5
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `keras.saving.save_model(model)`. This file format is considered legacy. We recommend using instead the native Keras format, e.g. `model.save('my_model.keras')` or `keras.saving.save_model(model, 'my_model.keras')`.
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 284s 198ms/step - accuracy: 0.4091 - loss: 1.5250 - val_accuracy: 0.4499 - val_loss: 1.4649 - learning_rate: 0.0010
Epoch 6/100
  1/1435 ━━━━━━━━━━━━━━━━━━━━ 4:15 178ms/step - accuracy: 0.4062 - loss: 1.6141
Epoch 6: val_accuracy did not improve from 0.44990
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 15s 10ms/step - accuracy: 0.4062 - loss: 1.6141 - val_accuracy: 0.4495 - val_loss: 1.4763 - learning_rate: 0.0010
Epoch 7/100
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 0s 275ms/step - accuracy: 0.4522 - loss: 1.4250
Epoch 7: val_accuracy improved from 0.44990 to 0.51178, saving model to best_facial_expression_model.h5
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `keras.saving.save_model(model)`. This file format is considered legacy. We recommend using instead the native Keras format, e.g. `model.save('my_model.keras')` or `keras.saving.save_model(model, 'my_model.keras')`.
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 421s 293ms/step - accuracy: 0.4654 - loss: 1.3982 - val_accuracy: 0.5118 - val_loss: 1.2562 - learning_rate: 0.0010
Epoch 8/100
  1/1435 ━━━━━━━━━━━━━━━━━━━━ 6:36 277ms/step - accuracy: 0.3438 - loss: 1.5607
Epoch 8: val_accuracy improved from 0.51178 to 0.51257, saving model to best_facial_expression_model.h5
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `keras.saving.save_model(model)`. This file format is considered legacy. We recommend using instead the native Keras format, e.g. `model.save('my_model.keras')` or `keras.saving.save_model(model, 'my_model.keras')`.
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 27s 19ms/step - accuracy: 0.3438 - loss: 1.5607 - val_accuracy: 0.5126 - val_loss: 1.2552 - learning_rate: 0.0010
Epoch 9/100
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 0s 282ms/step - accuracy: 0.4943 - loss: 1.3348
Epoch 9: val_accuracy improved from 0.51257 to 0.55936, saving model to best_facial_expression_model.h5
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `keras.saving.save_model(model)`. This file format is considered legacy. We recommend using instead the native Keras format, e.g. `model.save('my_model.keras')` or `keras.saving.save_model(model, 'my_model.keras')`.
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 485s 301ms/step - accuracy: 0.4976 - loss: 1.3196 - val_accuracy: 0.5594 - val_loss: 1.1547 - learning_rate: 0.0010
Epoch 10/100
  1/1435 ━━━━━━━━━━━━━━━━━━━━ 6:42 281ms/step - accuracy: 0.5000 - loss: 1.1560
Epoch 10: val_accuracy did not improve from 0.55936
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 40s 28ms/step - accuracy: 0.5000 - loss: 1.1560 - val_accuracy: 0.5577 - val_loss: 1.1549 - learning_rate: 0.0010
Epoch 11/100
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 0s 174ms/step - accuracy: 0.5131 - loss: 1.2869
Epoch 11: val_accuracy did not improve from 0.55936
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 265s 185ms/step - accuracy: 0.5185 - loss: 1.2690 - val_accuracy: 0.5259 - val_loss:

1.2383 - learning_rate: 0.0010
Epoch 12/100
  1/1435 ━━━━━━━━━━━━━━━━━━━━ 4:28 187ms/step - accuracy: 0.5312 - loss: 1.1804
Epoch 12: val_accuracy did not improve from 0.55936
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 14s 10ms/step - accuracy: 0.5312 - loss: 1.1804 - val_accuracy: 0.5273 - val_loss: 1.2360 - learning_rate: 0.0010
Epoch 13/100
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 0s 177ms/step - accuracy: 0.5271 - loss: 1.2484
Epoch 13: val_accuracy improved from 0.55936 to 0.57315, saving model to best_facial_expression_model.h5
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `keras.saving.save_model(model)`. This file format is considered legacy. We recommend using instead the native Keras format, e.g. `model.save('my_model.keras')` or `keras.saving.save_model(model, 'my_model.keras')`.
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 269s 188ms/step - accuracy: 0.5291 - loss: 1.2418 - val_accuracy: 0.5731 - val_loss: 1.0999 - learning_rate: 0.0010
Epoch 14/100
  1/1435 ━━━━━━━━━━━━━━━━━━━━ 4:26 186ms/step - accuracy: 0.5625 - loss: 0.9797
Epoch 14: val_accuracy improved from 0.57315 to 0.57332, saving model to best_facial_expression_model.h5
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `keras.saving.save_model(model)`. This file format is considered legacy. We recommend using instead the native Keras format, e.g. `model.save('my_model.keras')` or `keras.saving.save_model(model, 'my_model.keras')`.
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 15s 11ms/step - accuracy: 0.5625 - loss: 0.9797 - val_accuracy: 0.5733 - val_loss: 1.0998 - learning_rate: 0.0010
Epoch 15/100
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 0s 212ms/step - accuracy: 0.5441 - loss: 1.2069
Epoch 15: val_accuracy improved from 0.57332 to 0.58450, saving model to best_facial_expression_model.h5
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `keras.saving.save_model(model)`. This file format is considered legacy. We recommend using instead the native Keras format, e.g. `model.save('my_model.keras')` or `keras.saving.save_model(model, 'my_model.keras')`.
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 319s 223ms/step - accuracy: 0.5441 - loss: 1.2065 - val_accuracy: 0.5845 - val_loss: 1.0774 - learning_rate: 0.0010
Epoch 16/100
  1/1435 ━━━━━━━━━━━━━━━━━━━━ 4:35 192ms/step - accuracy: 0.5625 - loss: 1.1989
Epoch 16: val_accuracy did not improve from 0.58450
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 15s 10ms/step - accuracy: 0.5625 - loss: 1.1989 - val_accuracy: 0.5807 - val_loss: 1.0825 - learning_rate: 0.0010
Epoch 17/100
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 0s 204ms/step - accuracy: 0.5547 - loss: 1.1907
Epoch 17: val_accuracy did not improve from 0.58450
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 308s 215ms/step - accuracy: 0.5570 - loss: 1.1841 - val_accuracy: 0.5652 - val_loss: 1.1212 - learning_rate: 0.0010
Epoch 18/100
  1/1435 ━━━━━━━━━━━━━━━━━━━━ 4:41 196ms/step - accuracy: 0.5938 - loss: 0.9707
Epoch 18: val_accuracy did not improve from 0.58450
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 16s 11ms/step - accuracy: 0.5938 - loss: 0.9707 - val_accuracy: 0.5668 - val_loss: 1.1192 - learning_rate: 0.0010
Epoch 19/100
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 0s 187ms/step - accuracy: 0.5596 - loss: 1.1698
Epoch 19: val_accuracy improved from 0.58450 to 0.58886, saving model to best_facial_expression_model.h5
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `keras.saving.save_model(model)`. This file format is considered legacy. We recommend using instead the native Keras format, e.g. `model.save('my_model.keras')` or `keras.saving.save_model(model, 'my_model.keras')`.
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 285s 199ms/step - accuracy: 0.5613 - loss: 1.1698 - val_accuracy: 0.5889 - val_loss: 1.0835 - learning_rate: 0.0010
Epoch 20/100
  1/1435 ━━━━━━━━━━━━━━━━━━━━ 4:38 194ms/step - accuracy: 0.6562 - loss: 1.3172
Epoch 20: val_accuracy improved from 0.58886 to 0.59078, saving model to best_facial_expression_model.h5
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `keras.saving.save_model(model)`. This

file format is considered legacy. We recommend using instead the native Keras format, e.g. `model.save('my_model.keras')` or `keras.saving.save_model(model, 'my_model.keras')`.
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 17s 12ms/step - accuracy: 0.6562 - loss: 1.3172 - val_accuracy: 0.5908 - val_loss: 1.0745 - learning_rate: 0.0010
Epoch 21/100
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 0s 263ms/step - accuracy: 0.5671 - loss: 1.1540
Epoch 21: val_accuracy improved from 0.59078 to 0.60038, saving model to best_facial_expression_model.h5
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `keras.saving.save_model(model)`. This file format is considered legacy. We recommend using instead the native Keras format, e.g. `model.save('my_model.keras')` or `keras.saving.save_model(model, 'my_model.keras')`.
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 399s 278ms/step - accuracy: 0.5663 - loss: 1.1536 - val_accuracy: 0.6004 - val_loss: 1.0467 - learning_rate: 0.0010
Epoch 22/100
1/1435 ━━━━━━━━━━━━━━━━━━━━ 4:51 203ms/step - accuracy: 0.4688 - loss: 1.2214
Epoch 22: val_accuracy improved from 0.60038 to 0.60065, saving model to best_facial_expression_model.h5
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `keras.saving.save_model(model)`. This file format is considered legacy. We recommend using instead the native Keras format, e.g. `model.save('my_model.keras')` or `keras.saving.save_model(model, 'my_model.keras')`.
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 17s 12ms/step - accuracy: 0.4688 - loss: 1.2214 - val_accuracy: 0.6006 - val_loss: 1.0456 - learning_rate: 0.0010
Epoch 23/100
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 0s 197ms/step - accuracy: 0.5757 - loss: 1.1371
Epoch 23: val_accuracy did not improve from 0.60065
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 301s 209ms/step - accuracy: 0.5736 - loss: 1.1390 - val_accuracy: 0.5905 - val_loss: 1.0568 - learning_rate: 0.0010
Epoch 24/100
1/1435 ━━━━━━━━━━━━━━━━━━━━ 4:49 202ms/step - accuracy: 0.5625 - loss: 1.2601
Epoch 24: val_accuracy did not improve from 0.60065
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 18s 12ms/step - accuracy: 0.5625 - loss: 1.2601 - val_accuracy: 0.5928 - val_loss: 1.0530 - learning_rate: 0.0010
Epoch 25/100
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 0s 164ms/step - accuracy: 0.5729 - loss: 1.1282
Epoch 25: val_accuracy improved from 0.60065 to 0.61444, saving model to best_facial_expression_model.h5
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `keras.saving.save_model(model)`. This file format is considered legacy. We recommend using instead the native Keras format, e.g. `model.save('my_model.keras')` or `keras.saving.save_model(model, 'my_model.keras')`.
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 249s 174ms/step - accuracy: 0.5744 - loss: 1.1302 - val_accuracy: 0.6144 - val_loss: 1.0229 - learning_rate: 0.0010
Epoch 26/100
1/1435 ━━━━━━━━━━━━━━━━━━━━ 4:13 177ms/step - accuracy: 0.6562 - loss: 0.9929
Epoch 26: val_accuracy did not improve from 0.61444
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 13s 9ms/step - accuracy: 0.6562 - loss: 0.9929 - val_accuracy: 0.6127 - val_loss: 1.0281 - learning_rate: 0.0010
Epoch 27/100
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 0s 160ms/step - accuracy: 0.5855 - loss: 1.1141
Epoch 27: val_accuracy improved from 0.61444 to 0.62238, saving model to best_facial_expression_model.h5
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `keras.saving.save_model(model)`. This file format is considered legacy. We recommend using instead the native Keras format, e.g. `model.save('my_model.keras')` or `keras.saving.save_model(model, 'my_model.keras')`.
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 243s 169ms/step - accuracy: 0.5826 - loss: 1.1191 - val_accuracy: 0.6224 - val_loss: 0.9905 - learning_rate: 0.0010
Epoch 28/100
1/1435 ━━━━━━━━━━━━━━━━━━━━ 4:00 168ms/step - accuracy: 0.5000 - loss: 1.1325
Epoch 28: val_accuracy did not improve from 0.62238
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 14s 9ms/step - accuracy: 0.5000 - loss: 1.1325 - val_accuracy: 0.6220 - val_loss: 0.9928 - learning_rate: 0.0010
Epoch 29/100

1435/1435 ━━━━━━━━━━━━━━━━━━━━ 0s 161ms/step - accuracy: 0.5864 - loss: 1.1068
Epoch 29: val_accuracy did not improve from 0.62238
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 244s 170ms/step - accuracy: 0.5840 - loss: 1.1077 - val_accuracy: 0.6197 - val_loss: 1.0285 - learning_rate: 0.0010
Epoch 30/100
  1/1435 ━━━━━━━━━━━━━━━━━━━━ 4:00 168ms/step - accuracy: 0.6875 - loss: 0.8251
Epoch 30: val_accuracy did not improve from 0.62238
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 13s 9ms/step - accuracy: 0.6875 - loss: 0.8251 - val_accuracy: 0.6159 - val_loss: 1.0333 - learning_rate: 0.0010
Epoch 31/100
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 0s 164ms/step - accuracy: 0.5918 - loss: 1.0955
Epoch 31: val_accuracy improved from 0.62238 to 0.63478, saving model to best_facial_expression_model.h5
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `keras.saving.save_model(model)`. This file format is considered legacy. We recommend using instead the native Keras format, e.g. `model.save('my_model.keras')` or `keras.saving.save_model(model, 'my_model.keras')`.
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 249s 174ms/step - accuracy: 0.5922 - loss: 1.0948 - val_accuracy: 0.6348 - val_loss: 0.9753 - learning_rate: 0.0010
Epoch 32/100
  1/1435 ━━━━━━━━━━━━━━━━━━━━ 4:22 183ms/step - accuracy: 0.5625 - loss: 0.8554
Epoch 32: val_accuracy did not improve from 0.63478
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 14s 10ms/step - accuracy: 0.5625 - loss: 0.8554 - val_accuracy: 0.6344 - val_loss: 0.9754 - learning_rate: 0.0010
aving.save_model(model)`. This file format is considered legacy. We recommend using instead the native Keras format, e.g. `model.save('my_model.keras')` or `keras.saving.save_model(model, 'my_model.keras')`.
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 312s 217ms/step - accuracy: 0.5951 - loss: 1.0807 - val_accuracy: 0.6397 - val_loss: 0.9536 - learning_rate: 0.0010
Epoch 36/100
  1/1435 ━━━━━━━━━━━━━━━━━━━━ 4:32 190ms/step - accuracy: 0.5938 - loss: 1.0404
Epoch 36: val_accuracy improved from 0.63966 to 0.64010, saving model to best_facial_expression_model.h5
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `keras.saving.save_model(model)`. This file format is considered legacy. We recommend using instead the native Keras format, e.g. `model.save('my_model.keras')` or `keras.saving.save_model(model, 'my_model.keras')`.
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 15s 10ms/step - accuracy: 0.5938 - loss: 1.0404 - val_accuracy: 0.6401 - val_loss: 0.9530 - learning_rate: 0.0010
Epoch 37/100
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 0s 265ms/step - accuracy: 0.5951 - loss: 1.0791
Epoch 37: val_accuracy improved from 0.64010 to 0.64473, saving model to best_facial_expression_model.h5
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `keras.saving.save_model(model)`. This file format is considered legacy. We recommend using instead the native Keras format, e.g. `model.save('my_model.keras')` or `keras.saving.save_model(model, 'my_model.keras')`.
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 408s 284ms/step - accuracy: 0.5956 - loss: 1.0778 - val_accuracy: 0.6447 - val_loss: 0.9471 - learning_rate: 0.0010
Epoch 38/100
  1/1435 ━━━━━━━━━━━━━━━━━━━━ 6:30 273ms/step - accuracy: 0.6562 - loss: 1.3078
Epoch 38: val_accuracy did not improve from 0.64473
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 40s 28ms/step - accuracy: 0.6562 - loss: 1.3078 - val_accuracy: 0.6441 - val_loss: 0.9453 - learning_rate: 0.0010
Epoch 39/100
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 0s 305ms/step - accuracy: 0.6068 - loss: 1.0607
Epoch 39: val_accuracy did not improve from 0.64473
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 467s 326ms/step - accuracy: 0.6039 - loss: 1.0674 - val_accuracy: 0.6385 - val_loss: 0.9467 - learning_rate: 0.0010
Epoch 40/100
  1/1435 ━━━━━━━━━━━━━━━━━━━━ 7:09 300ms/step - accuracy: 0.5312 - loss: 1.3955
Epoch 40: val_accuracy did not improve from 0.64473
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 30s 21ms/step - accuracy: 0.5312 - loss: 1.3955 - val_accuracy: 0.6387 - val_loss: 0.9463 - learning_rate: 0.0010

Epoch 41/100
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 0s 310ms/step - accuracy: 0.6075 - loss: 1.0591
Epoch 41: val_accuracy did not improve from 0.64473
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 462s 322ms/step - accuracy: 0.6064 - loss: 1.0602 - val_accuracy: 0.6421 - val_loss: 0.9579 - learning_rate: 0.0010
Epoch 42/100
  1/1435 ━━━━━━━━━━━━━━━━━━━━ 5:04 212ms/step - accuracy: 0.4688 - loss: 1.1764
Epoch 42: val_accuracy did not improve from 0.64473
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 19s 13ms/step - accuracy: 0.4688 - loss: 1.1764 - val_accuracy: 0.6417 - val_loss: 0.9572 - learning_rate: 0.0010
Epoch 43/100
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 0s 171ms/step - accuracy: 0.6047 - loss: 1.0523
Epoch 43: val_accuracy did not improve from 0.64473
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 258s 180ms/step - accuracy: 0.6084 - loss: 1.0506 - val_accuracy: 0.6361 - val_loss: 0.9677 - learning_rate: 0.0010
Epoch 44/100
  1/1435 ━━━━━━━━━━━━━━━━━━━━ 4:21 182ms/step - accuracy: 0.6875 - loss: 0.9756
Epoch 44: val_accuracy did not improve from 0.64473
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 13s 9ms/step - accuracy: 0.6875 - loss: 0.9756 - val_accuracy: 0.6360 - val_loss: 0.9666 - learning_rate: 0.0010
Epoch 45/100
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 0s 192ms/step - accuracy: 0.6107 - loss: 1.0466
Epoch 45: val_accuracy improved from 0.64473 to 0.64481, saving model to best_facial_expression_model.h5
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `keras.saving.save_model(model)`. This file format is considered legacy. We recommend using instead the native Keras format, e.g. `model.save('my_model.keras')` or `keras.saving.save_model(model, 'my_model.keras')`.
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 293s 205ms/step - accuracy: 0.6111 - loss: 1.0473 - val_accuracy: 0.6448 - val_loss: 0.9558 - learning_rate: 0.0010
Epoch 46/100
  1/1435 ━━━━━━━━━━━━━━━━━━━━ 4:29 188ms/step - accuracy: 0.7500 - loss: 0.9144
Epoch 46: ReduceLROnPlateau reducing learning rate to 0.00020000000949949026.

Epoch 46: val_accuracy did not improve from 0.64481
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 15s 11ms/step - accuracy: 0.7500 - loss: 0.9144 - val_accuracy: 0.6439 - val_loss: 0.9557 - learning_rate: 0.0010
Epoch 47/100
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 0s 168ms/step - accuracy: 0.6203 - loss: 1.0240
Epoch 47: val_accuracy improved from 0.64481 to 0.66288, saving model to best_facial_expression_model.h5
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `keras.saving.save_model(model)`. This file format is considered legacy. We recommend using instead the native Keras format, e.g. `model.save('my_model.keras')` or `keras.saving.save_model(model, 'my_model.keras')`.
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 256s 178ms/step - accuracy: 0.6243 - loss: 1.0110 - val_accuracy: 0.6629 - val_loss: 0.8918 - learning_rate: 2.0000e-04
Epoch 48/100
  1/1435 ━━━━━━━━━━━━━━━━━━━━ 4:33 191ms/step - accuracy: 0.6562 - loss: 1.1027
Epoch 48: val_accuracy did not improve from 0.66288
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 15s 10ms/step - accuracy: 0.6562 - loss: 1.1027 - val_accuracy: 0.6625 - val_loss: 0.8915 - learning_rate: 2.0000e-04
Epoch 49/100
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 0s 204ms/step - accuracy: 0.6333 - loss: 0.9980
Epoch 49: val_accuracy improved from 0.66288 to 0.67214, saving model to best_facial_expression_model.h5
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `keras.saving.save_model(model)`. This file format is considered legacy. We recommend using instead the native Keras format, e.g. `model.save('my_model.keras')` or `keras.saving.save_model(model, 'my_model.keras')`.
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 311s 216ms/step - accuracy: 0.6335 - loss: 0.9944 - val_accuracy: 0.6721 - val_loss: 0.8780 - learning_rate: 2.0000e-04
Epoch 50/100

```
  1/1435 ━━━━━━━━━━━━━━━━━━━━ 4:36 193ms/step - accuracy: 0.5625 - loss: 1.1045
Epoch 50: val_accuracy improved from 0.67214 to 0.67266, saving model to best_facial_expression_model.h5
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `keras.saving.save_model(model)`. This file format is considered legacy. We recommend using instead the native Keras format, e.g. `model.save('my_model.keras')` or `keras.saving.save_model(model, 'my_model.keras')`.
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 16s 11ms/step - accuracy: 0.5625 - loss: 1.1045 - val_accuracy: 0.6727 - val_loss: 0.8775 - learning_rate: 2.0000e-04
Epoch 51/100
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 0s 186ms/step - accuracy: 0.6334 - loss: 0.9894
Epoch 51: val_accuracy did not improve from 0.67266
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 283s 197ms/step - accuracy: 0.6337 - loss: 0.9906 - val_accuracy: 0.6720 - val_loss: 0.8733 - learning_rate: 2.0000e-04
Epoch 52/100
  1/1435 ━━━━━━━━━━━━━━━━━━━━ 4:29 188ms/step - accuracy: 0.7500 - loss: 0.8619
Epoch 52: val_accuracy improved from 0.67266 to 0.67292, saving model to best_facial_expression_model.h5
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `keras.saving.save_model(model)`. This file format is considered legacy. We recommend using instead the native Keras format, e.g. `model.save('my_model.keras')` or `keras.saving.save_model(model, 'my_model.keras')`.
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 16s 11ms/step - accuracy: 0.7500 - loss: 0.8619 - val_accuracy: 0.6729 - val_loss: 0.8721 - learning_rate: 2.0000e-04
Epoch 53/100
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 0s 159ms/step - accuracy: 0.6344 - loss: 0.9874
Epoch 53: val_accuracy did not improve from 0.67292
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 241s 168ms/step - accuracy: 0.6339 - loss: 0.9860 - val_accuracy: 0.6721 - val_loss: 0.8779 - learning_rate: 2.0000e-04
Epoch 54/100
  1/1435 ━━━━━━━━━━━━━━━━━━━━ 3:49 160ms/step - accuracy: 0.6875 - loss: 0.7854
Epoch 54: val_accuracy did not improve from 0.67292
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 14s 10ms/step - accuracy: 0.6875 - loss: 0.7854 - val_accuracy: 0.6720 - val_loss: 0.8785 - learning_rate: 2.0000e-04
Epoch 55/100
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 0s 245ms/step - accuracy: 0.6389 - loss: 0.9804
Epoch 55: val_accuracy improved from 0.67292 to 0.67764, saving model to best_facial_expression_model.h5
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `keras.saving.save_model(model)`. This file format is considered legacy. We recommend using instead the native Keras format, e.g. `model.save('my_model.keras')` or `keras.saving.save_model(model, 'my_model.keras')`.
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 364s 254ms/step - accuracy: 0.6358 - loss: 0.9827 - val_accuracy: 0.6776 - val_loss: 0.8670 - learning_rate: 2.0000e-04
Epoch 56/100
  1/1435 ━━━━━━━━━━━━━━━━━━━━ 4:47 200ms/step - accuracy: 0.6250 - loss: 0.9322
Epoch 56: val_accuracy improved from 0.67764 to 0.67807, saving model to best_facial_expression_model.h5
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `keras.saving.save_model(model)`. This file format is considered legacy. We recommend using instead the native Keras format, e.g. `model.save('my_model.keras')` or `keras.saving.save_model(model, 'my_model.keras')`.
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 13s 9ms/step - accuracy: 0.6250 - loss: 0.9322 - val_accuracy: 0.6781 - val_loss: 0.8666 - learning_rate: 2.0000e-04
Epoch 57/100
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 0s 186ms/step - accuracy: 0.6324 - loss: 0.9809
Epoch 57: val_accuracy did not improve from 0.67807
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 281s 195ms/step - accuracy: 0.6343 - loss: 0.9807 - val_accuracy: 0.6713 - val_loss: 0.8759 - learning_rate: 2.0000e-04
Epoch 58/100
  1/1435 ━━━━━━━━━━━━━━━━━━━━ 4:22 183ms/step - accuracy: 0.6875 - loss: 0.8839
Epoch 58: val_accuracy did not improve from 0.67807
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 13s 9ms/step - accuracy: 0.6875 - loss: 0.8839 - val_accuracy: 0.6711 - val_loss: 0.8764 - learning_rate: 2.0000e-04
Epoch 59/100
```

1435/1435 ━━━━━━━━━━━━━━━━━━━━ 0s 161ms/step - accuracy: 0.6433 - loss: 0.9691
Epoch 59: val_accuracy did not improve from 0.67807
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 245s 170ms/step - accuracy: 0.6397 - loss: 0.9734 - val_accuracy: 0.6769 - val_loss: 0.8673 - learning_rate: 2.0000e-04
Epoch 60/100
  1/1435 ━━━━━━━━━━━━━━━━━━━━ 3:57 166ms/step - accuracy: 0.7812 - loss: 0.7944
Epoch 60: val_accuracy did not improve from 0.67807
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 13s 9ms/step - accuracy: 0.7812 - loss: 0.7944 - val_accuracy: 0.6767 - val_loss: 0.8667 - learning_rate: 2.0000e-04
Epoch 61/100
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 0s 162ms/step - accuracy: 0.6376 - loss: 0.9708
Epoch 61: val_accuracy did not improve from 0.67807
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 248s 173ms/step - accuracy: 0.6376 - loss: 0.9711 - val_accuracy: 0.6747 - val_loss: 0.8712 - learning_rate: 2.0000e-04
Epoch 62/100
  1/1435 ━━━━━━━━━━━━━━━━━━━━ 4:21 182ms/step - accuracy: 0.6250 - loss: 1.1969
Epoch 62: val_accuracy did not improve from 0.67807
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 15s 10ms/step - accuracy: 0.6250 - loss: 1.1969 - val_accuracy: 0.6747 - val_loss: 0.8727 - learning_rate: 2.0000e-04
Epoch 63/100
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 0s 193ms/step - accuracy: 0.6385 - loss: 0.9700
Epoch 63: val_accuracy did not improve from 0.67807
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 293s 204ms/step - accuracy: 0.6430 - loss: 0.9678 - val_accuracy: 0.6758 - val_loss: 0.8687 - learning_rate: 2.0000e-04
Epoch 64/100
  1/1435 ━━━━━━━━━━━━━━━━━━━━ 4:26 186ms/step - accuracy: 0.7188 - loss: 0.9292
Epoch 64: ReduceLROnPlateau reducing learning rate to 4.0000001899898055e-05.

Epoch 64: val_accuracy did not improve from 0.67807
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 16s 11ms/step - accuracy: 0.7188 - loss: 0.9292 - val_accuracy: 0.6755 - val_loss: 0.8695 - learning_rate: 2.0000e-04
Epoch 65/100
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 0s 179ms/step - accuracy: 0.6440 - loss: 0.9570
Epoch 65: val_accuracy did not improve from 0.67807
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 271s 188ms/step - accuracy: 0.6423 - loss: 0.9592 - val_accuracy: 0.6772 - val_loss: 0.8600 - learning_rate: 4.0000e-05
Epoch 66/100
  1/1435 ━━━━━━━━━━━━━━━━━━━━ 4:17 180ms/step - accuracy: 0.5938 - loss: 1.0127
Epoch 66: val_accuracy did not improve from 0.67807
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 13s 9ms/step - accuracy: 0.5938 - loss: 1.0127 - val_accuracy: 0.6774 - val_loss: 0.8600 - learning_rate: 4.0000e-05
Epoch 67/100
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 0s 175ms/step - accuracy: 0.6466 - loss: 0.9537
Epoch 67: val_accuracy improved from 0.67807 to 0.67860, saving model to best_facial_expression_model.h5
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `keras.saving.save_model(model)`. This file format is considered legacy. We recommend using instead the native KeraEpoch 63: val_accuracy did not improve from 0.67807
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 293s 204ms/step - accuracy: 0.6430 - loss: 0.9678 - val_accuracy: 0.6758 - val_loss: 0.8687 - learning_rate: 2.0000e-04
Epoch 64/100
  1/1435 ━━━━━━━━━━━━━━━━━━━━ 4:26 186ms/step - accuracy: 0.7188 - loss: 0.9292
Epoch 64: ReduceLROnPlateau reducing learning rate to 4.0000001899898055e-05.

Epoch 64: val_accuracy did not improve from 0.67807
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 16s 11ms/step - accuracy: 0.7188 - loss: 0.9292 - val_accuracy: 0.6755 - val_loss: 0.8695 - learning_rate: 2.0000e-04
Epoch 65/100

1435/1435 ━━━━━━━━━━━━━━━━━━━━ 0s 179ms/step - accuracy: 0.6440 - loss: 0.9570
Epoch 65: val_accuracy did not improve from 0.67807
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 271s 188ms/step - accuracy: 0.6423 - loss: 0.9592 - val_accuracy: 0.6772 - val_loss: 0.8600 - learning_rate: 4.0000e-05
Epoch 66/100
  1/1435 ━━━━━━━━━━━━━━━━━━━━ 4:17 180ms/step - accuracy: 0.5938 - loss: 1.0127
Epoch 66: val_accuracy did not improve from 0.67807
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 13s 9ms/step - accuracy: 0.5938 - loss: 1.0127 - val_accuracy: 0.6774 - val_loss: 0.8600 - learning_rate: 4.0000e-05
Epoch 67/100
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 0s 175ms/step - accuracy: 0.6466 - loss: 0.9537
Epoch 67: val_accuracy improved from 0.67807 to 0.67860, saving model to best_facial_expression_model.h5
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `keras.saving.save_model(model)`. This file format is considered legacy. We recommend using instead the native KeraEpoch 65/100
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 0s 179ms/step - accuracy: 0.6440 - loss: 0.9570
Epoch 65: val_accuracy did not improve from 0.67807
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 271s 188ms/step - accuracy: 0.6423 - loss: 0.9592 - val_accuracy: 0.6772 - val_loss: 0.8600 - learning_rate: 4.0000e-05
Epoch 66/100
  1/1435 ━━━━━━━━━━━━━━━━━━━━ 4:17 180ms/step - accuracy: 0.5938 - loss: 1.0127
Epoch 66: val_accuracy did not improve from 0.67807
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 13s 9ms/step - accuracy: 0.5938 - loss: 1.0127 - val_accuracy: 0.6774 - val_loss: 0.8600 - learning_rate: 4.0000e-05
Epoch 67/100
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 0s 175ms/step - accuracy: 0.6466 - loss: 0.9537
Epoch 67: val_accuracy improved from 0.67807 to 0.67860, saving model to best_facial_expression_model.h5
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `keras.saving.save_model(model)`. This file format is considered legacy. We recommend using instead the native Kera1435/1435 ━━━━━━━━━━━━━━━━━━━━ 0s 175ms/step - accuracy: 0.6466 - loss: 0.9537
Epoch 67: val_accuracy improved from 0.67807 to 0.67860, saving model to best_facial_expression_model.h5
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `keras.saving.save_model(model)`. This file format is considered legacy. We recommend using instead the native Keras format, e.g. `model.save('my_model.keras')` or `keras.saving.save_model(model, 'my_model.keras')`.
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 267s 186ms/step - accuracy: 0.6448 - loss: 0.9578 - val_accuracy: 0.6786 - val_loss: 0.8588 - learning_rate: 4.0000e-05
Epoch 68/100
  1/1435 ━━━━━━━━━━━━━━━━━━━━ 4:45 199ms/step - accuracy: 0.8125 - loss: 0.6131
Epoch 68: val_accuracy did not improve from 0.67860
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 16s 11ms/step - accuracy: 0.8125 - loss: 0.6131 - val_accuracy: 0.6782 - val_loss: 0.8595 - learning_rate: 4.0000e-05
Epoch 69/100
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 0s 186ms/step - accuracy: 0.6451 - loss: 0.9539
Epoch 69: val_accuracy did not improve from 0.67860
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 281s 195ms/step - accuracy: 0.6465 - loss: 0.9559 - val_accuracy: 0.6783 - val_loss: 0.8571 - learning_rate: 4.0000e-05
Epoch 70/100
  1/1435 ━━━━━━━━━━━━━━━━━━━━ 4:49 202ms/step - accuracy: 0.7188 - loss: 0.8072
Epoch 70: val_accuracy did not improve from 0.67860
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 13s 9ms/step - accuracy: 0.7188 - loss: 0.8072 - val_accuracy: 0.6783 - val_loss: 0.8569 - learning_rate: 4.0000e-05
Epoch 71/100
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 0s 159ms/step - accuracy: 0.6504 - loss: 0.9493
Epoch 71: val_accuracy did not improve from 0.67860
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 241s 168ms/step - accuracy: 0.6459 - loss: 0.9560 - val_accuracy: 0.6778 - val_loss: 0.8612 - learning_rate: 4.0000e-05
Epoch 72/100
  1/1435 ━━━━━━━━━━━━━━━━━━━━ 4:10 175ms/step - accuracy: 0.7188 - loss: 0.7783

Epoch 72: val_accuracy did not improve from 0.67860
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 13s 9ms/step - accuracy: 0.7188 - loss: 0.7783 - val_accuracy: 0.6775 - val_loss: 0.8614 - learning_rate: 4.0000e-05
Epoch 73/100
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 0s 160ms/step - accuracy: 0.6403 - loss: 0.9618
Epoch 73: val_accuracy did not improve from 0.67860
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 242s 169ms/step - accuracy: 0.6440 - loss: 0.9560 - val_accuracy: 0.6781 - val_loss: 0.8544 - learning_rate: 4.0000e-05
Epoch 74/100
  1/1435 ━━━━━━━━━━━━━━━━━━━━ 3:58 166ms/step - accuracy: 0.8438 - loss: 0.7393
Epoch 74: val_accuracy did not improve from 0.67860
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 14s 9ms/step - accuracy: 0.8438 - loss: 0.7393 - val_accuracy: 0.6782 - val_loss: 0.8545 - learning_rate: 4.0000e-05
Epoch 75/100
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 0s 160ms/step - accuracy: 0.6427 - loss: 0.9611
Epoch 75: val_accuracy improved from 0.67860 to 0.67964, saving model to best_facial_expression_model.h5
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `keras.saving.save_model(model)`. This file format is considered legacy. We recommend using instead the native Keras format, e.g. `model.save('my_model.keras')` or `keras.saving.save_model(model, 'my_model.keras')`.
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 243s 169ms/step - accuracy: 0.6437 - loss: 0.9581 - val_accuracy: 0.6796 - val_loss: 0.8542 - learning_rate: 4.0000e-05
Epoch 76/100
  1/1435 ━━━━━━━━━━━━━━━━━━━━ 4:43 198ms/step - accuracy: 0.5938 - loss: 0.9740
Epoch 76: val_accuracy improved from 0.67964 to 0.67973, saving model to best_facial_expression_model.h5
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `keras.saving.save_model(model)`. This file format is considered legacy. We recommend using instead the native Keras format, e.g. `model.save('my_model.keras')` or `keras.saving.save_model(model, 'my_model.keras')`.
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 14s 9ms/step - accuracy: 0.5938 - loss: 0.9740 - val_accuracy: 0.6797 - val_loss: 0.8550 - learning_rate: 4.0000e-05
Epoch 77/100
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 0s 166ms/step - accuracy: 0.6474 - loss: 0.9594
Epoch 77: val_accuracy did not improve from 0.67973
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 253s 176ms/step - accuracy: 0.6482 - loss: 0.9534 - val_accuracy: 0.6789 - val_loss: 0.8557 - learning_rate: 4.0000e-05
Epoch 78/100
  1/1435 ━━━━━━━━━━━━━━━━━━━━ 4:36 193ms/step - accuracy: 0.6250 - loss: 0.8782
Epoch 78: val_accuracy did not improve from 0.67973
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 15s 10ms/step - accuracy: 0.6250 - loss: 0.8782 - val_accuracy: 0.6793 - val_loss: 0.8545 - learning_rate: 4.0000e-05
Epoch 79/100
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 0s 185ms/step - accuracy: 0.6446 - loss: 0.9476
Epoch 79: val_accuracy improved from 0.67973 to 0.67999, saving model to best_facial_expression_model.h5
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `keras.saving.save_model(model)`. This file format is considered legacy. We recommend using instead the native Keras format, e.g. `model.save('my_model.keras')` or `keras.saving.save_model(model, 'my_model.keras')`.
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 281s 196ms/step - accuracy: 0.6463 - loss: 0.9535 - val_accuracy: 0.6800 - val_loss: 0.8525 - learning_rate: 4.0000e-05
Epoch 80/100
  1/1435 ━━━━━━━━━━━━━━━━━━━━ 4:48 201ms/step - accuracy: 0.6250 - loss: 0.9497
Epoch 80: val_accuracy improved from 0.67999 to 0.68043, saving model to best_facial_expression_model.h5
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `keras.saving.save_model(model)`. This file format is considered legacy. We recommend using instead the native Keras format, e.g. `model.save('my_model.keras')` or `keras.saving.save_model(model, 'my_model.keras')`.
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 16s 11ms/step - accuracy: 0.6250 - loss: 0.9497 - val_accuracy: 0.6804 - val_loss: 0.8517 - learning_rate: 4.0000e-05
Epoch 81/100
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 0s 181ms/step - accuracy: 0.6497 - loss: 0.9454

Epoch 81: val_accuracy did not improve from 0.68043
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 272s 189ms/step - accuracy: 0.6471 - loss: 0.9498 - val_accuracy: 0.6794 - val_loss: 0.8556 - learning_rate: 4.0000e-05
Epoch 82/100
  1/1435 ━━━━━━━━━━━━━━━━━━━━ 3:51 161ms/step - accuracy: 0.6562 - loss: 0.9911
Epoch 82: val_accuracy did not improve from 0.68043
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 13s 9ms/step - accuracy: 0.6562 - loss: 0.9911 - val_accuracy: 0.6790 - val_loss: 0.8552 - learning_rate: 4.0000e-05
Epoch 83/100
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 0s 163ms/step - accuracy: 0.6439 - loss: 0.9523
Epoch 83: val_accuracy did not improve from 0.68043
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 247s 172ms/step - accuracy: 0.6449 - loss: 0.9532 - val_accuracy: 0.6775 - val_loss: 0.8525 - learning_rate: 4.0000e-05
Epoch 84/100
  1/1435 ━━━━━━━━━━━━━━━━━━━━ 3:55 164ms/step - accuracy: 0.6875 - loss: 0.9204
Epoch 84: val_accuracy did not improve from 0.68043
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 13s 9ms/step - accuracy: 0.6875 - loss: 0.9204 - val_accuracy: 0.6775 - val_loss: 0.8526 - learning_rate: 4.0000e-05
Epoch 85/100
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 0s 176ms/step - accuracy: 0.6464 - loss: 0.9504
Epoch 85: val_accuracy did not improve from 0.68043
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 268s 187ms/step - accuracy: 0.6449 - loss: 0.9527 - val_accuracy: 0.6785 - val_loss: 0.8517 - learning_rate: 4.0000e-05
Epoch 86/100
  1/1435 ━━━━━━━━━━━━━━━━━━━━ 4:16 179ms/step - accuracy: 0.7500 - loss: 0.6886
Epoch 86: val_accuracy did not improve from 0.68043
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 16s 11ms/step - accuracy: 0.7500 - loss: 0.6886 - val_accuracy: 0.6782 - val_loss: 0.8523 - learning_rate: 4.0000e-05
Epoch 87/100
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 0s 209ms/step - accuracy: 0.6443 - loss: 0.9521
Epoch 87: val_accuracy did not improve from 0.68043
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 317s 221ms/step - accuracy: 0.6444 - loss: 0.9509 - val_accuracy: 0.6800 - val_loss: 0.8510 - learning_rate: 4.0000e-05
Epoch 88/100
  1/1435 ━━━━━━━━━━━━━━━━━━━━ 4:49 202ms/step - accuracy: 0.6250 - loss: 1.0136
Epoch 88: val_accuracy did not improve from 0.68043
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 17s 12ms/step - accuracy: 0.6250 - loss: 1.0136 - val_accuracy: 0.6803 - val_loss: 0.8514 - learning_rate: 4.0000e-05
Epoch 89/100
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 0s 161ms/step - accuracy: 0.6456 - loss: 0.9523
Epoch 89: val_accuracy did not improve from 0.68043
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 244s 170ms/step - accuracy: 0.6463 - loss: 0.9509 - val_accuracy: 0.6789 - val_loss: 0.8538 - learning_rate: 4.0000e-05
Epoch 90/100
  1/1435 ━━━━━━━━━━━━━━━━━━━━ 4:11 175ms/step - accuracy: 0.5625 - loss: 1.0445
Epoch 90: val_accuracy did not improve from 0.68043
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 13s 9ms/step - accuracy: 0.5625 - loss: 1.0445 - val_accuracy: 0.6790 - val_loss: 0.8530 - learning_rate: 4.0000e-05
Epoch 91/100
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 0s 161ms/step - accuracy: 0.6513 - loss: 0.9461
Epoch 91: val_accuracy did not improve from 0.68043
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 244s 170ms/step - accuracy: 0.6492 - loss: 0.9486 - val_accuracy: 0.6790 - val_loss: 0.8547 - learning_rate: 4.0000e-05
Epoch 92/100
  1/1435 ━━━━━━━━━━━━━━━━━━━━ 4:19 181ms/step - accuracy: 0.7188 - loss: 0.8634
Epoch 92: val_accuracy did not improve from 0.68043
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 13s 9ms/step - accuracy: 0.7188 - loss: 0.8634 - val_accuracy: 0.6787 - val_loss: 0.8547 -

learning_rate: 4.0000e-05
Epoch 93/100
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 0s 161ms/step - accuracy: 0.6403 - loss: 0.9600
Epoch 93: val_accuracy improved from 0.68043 to 0.68104, saving model to best_facial_expression_model.h5
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `keras.saving.save_model(model)`. This file format is considered legacy. We recommend using instead the native Keras format, e.g. `model.save('my_model.keras')` or `keras.saving.save_model(model, 'my_model.keras')`.
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 245s 171ms/step - accuracy: 0.6475 - loss: 0.9470 - val_accuracy: 0.6810 - val_loss: 0.8516 - learning_rate: 4.0000e-05
Epoch 94/100
  1/1435 ━━━━━━━━━━━━━━━━━━━━ 4:28 187ms/step - accuracy: 0.5312 - loss: 1.3525
Epoch 94: val_accuracy improved from 0.68104 to 0.68139, saving model to best_facial_expression_model.h5
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `keras.saving.save_model(model)`. This file format is considered legacy. We recommend using instead the native Keras format, e.g. `model.save('my_model.keras')` or `keras.saving.save_model(model, 'my_model.keras')`.
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 14s 9ms/step - accuracy: 0.5312 - loss: 1.3525 - val_accuracy: 0.6814 - val_loss: 0.8515 - learning_rate: 4.0000e-05
Epoch 95/100
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 0s 162ms/step - accuracy: 0.6477 - loss: 0.9504
Epoch 95: ReduceLROnPlateau reducing learning rate to 8.000000525498762e-06.

Epoch 95: val_accuracy did not improve from 0.68139
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 246s 171ms/step - accuracy: 0.6478 - loss: 0.9512 - val_accuracy: 0.6798 - val_loss: 0.8528 - learning_rate: 4.0000e-05
Epoch 96/100
  1/1435 ━━━━━━━━━━━━━━━━━━━━ 3:55 164ms/step - accuracy: 0.5938 - loss: 0.9733
Epoch 96: val_accuracy did not improve from 0.68139
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 13s 9ms/step - accuracy: 0.5938 - loss: 0.9733 - val_accuracy: 0.6799 - val_loss: 0.8526 - learning_rate: 8.0000e-06
Epoch 97/100
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 0s 163ms/step - accuracy: 0.6534 - loss: 0.9371
Epoch 97: val_accuracy did not improve from 0.68139
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 248s 172ms/step - accuracy: 0.6502 - loss: 0.9449 - val_accuracy: 0.6788 - val_loss: 0.8530 - learning_rate: 8.0000e-06
Epoch 98/100
  1/1435 ━━━━━━━━━━━━━━━━━━━━ 4:07 173ms/step - accuracy: 0.7188 - loss: 0.8702
Epoch 98: val_accuracy did not improve from 0.68139
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 13s 9ms/step - accuracy: 0.7188 - loss: 0.8702 - val_accuracy: 0.6789 - val_loss: 0.8534 - learning_rate: 8.0000e-06
Epoch 99/100
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 0s 161ms/step - accuracy: 0.6493 - loss: 0.9446
Epoch 99: val_accuracy did not improve from 0.68139
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 245s 171ms/step - accuracy: 0.6500 - loss: 0.9466 - val_accuracy: 0.6791 - val_loss: 0.8516 - learning_rate: 8.0000e-06
Epoch 100/100
  1/1435 ━━━━━━━━━━━━━━━━━━━━ 3:52 162ms/step - accuracy: 0.6250 - loss: 0.9970
Epoch 100: val_accuracy did not improve from 0.68139
1435/1435 ━━━━━━━━━━━━━━━━━━━━ 13s 9ms/step - accuracy: 0.6250 - loss: 0.9970 - val_accuracy: 0.6792 - val_loss: 0.8511 - learning_rate: 8.0000e-06
Restoring model weights from the end of the best epoch: 94.

Training completed!