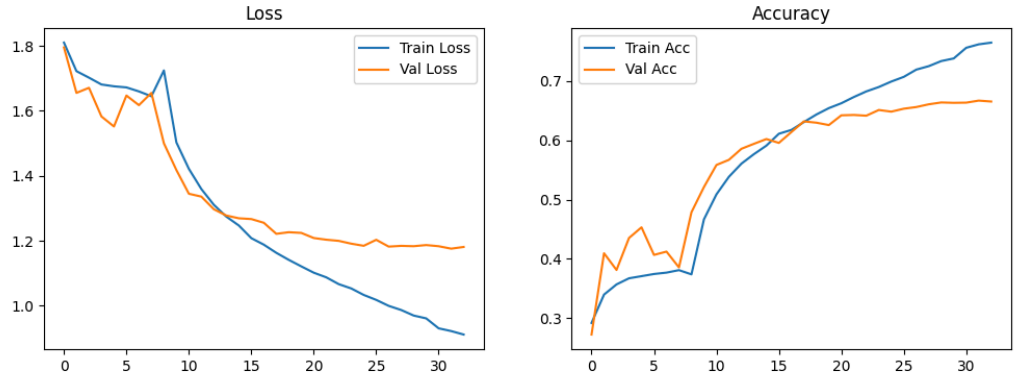
1st version

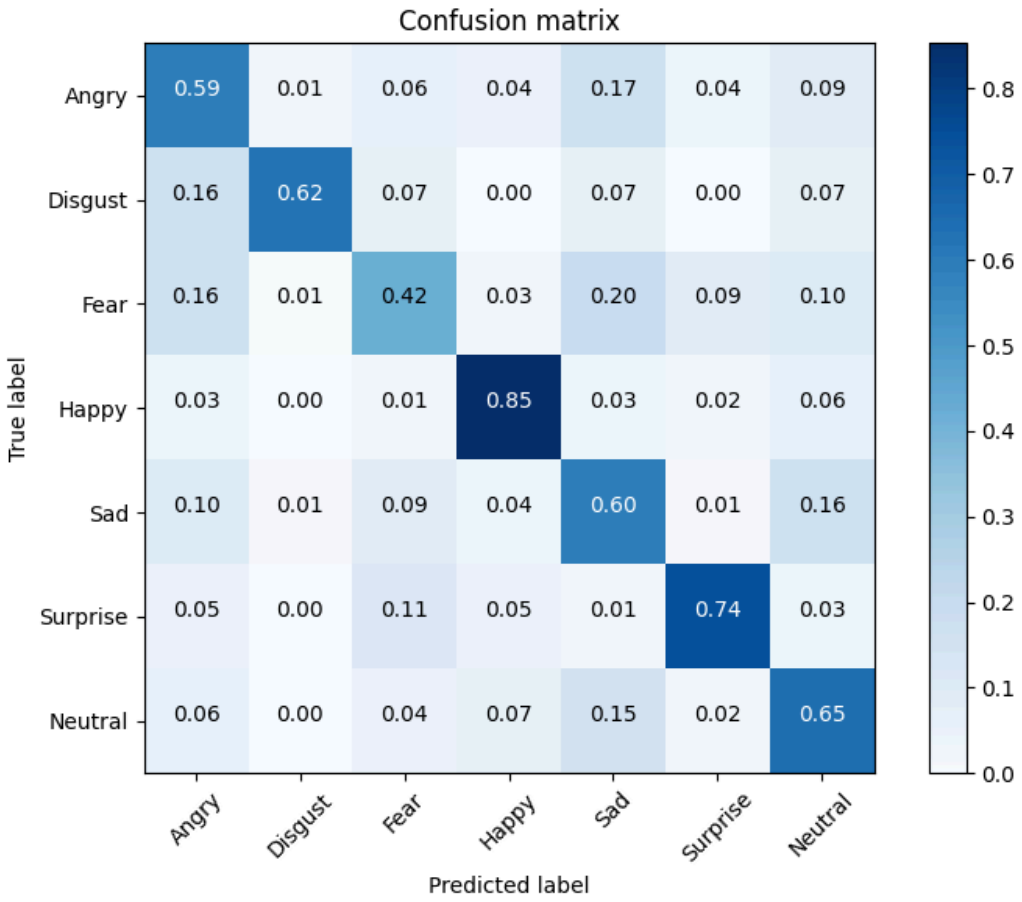
Training History:



Accuracy on test set:

```
(venv) PS C:\Users\Nedlanox\Desktop\UOW\AI_Modern\Project\transferLearning> python test_tl.py
2025-11-07 03:30:35.741888: I tensorflow/core/util/port.cc:153] oneDNN custom operations are on. You may see slightly different numerical re
sults due to floating-point round-off errors from different computation orders. To turn them off, set the environment variable `TF_ENABLE_ON
EDNN_OPTS=0`.
2025-11-07 03:30:38.996965: I tensorflow/core/util/port.cc:153] oneDNN custom operations are on. You may see slightly different numerical re
sults due to floating-point round-off errors from different computation orders. To turn them off, set the environment variable `TF_ENABLE_ON
EDNN_OPTS=0`.
2025-11-07 03:30:41.456432: I tensorflow/core/platform/cpu_feature_guard.cc:210] This TensorFlow binary is optimized to use available CPU in
structions in performance-critical operations.
To enable the following instructions: SSE3 SSE4.1 SSE4.2 AVX AVX2 AVX_VNNI FMA, in other operations, rebuild TensorFlow with the appropriate
C:\Users\Nedlanox\Desktop\UOW\AI_Modern\Project\transferLearning\venv\lib\site-packages\keras\src\saving\saving_lib.py:797: UserWarning: Ski
pping variable loading for optimizer 'rmsprop', because it has 71 variables whereas the saved optimizer has 140 variables.
  saveable.load_own_variables(weights_store.get(inner_path))
Loaded transfer-learning model
Accuracy on TL test set: 66.06%
```

Confusion Matrix:



Terminal Output:

```
(venv) PS C:\Users\Nedlanox\Desktop\UOW\AI_Modern\Project\transferLearning>
(venv) PS C:\Users\Nedlanox\Desktop\UOW\AI_Modern\Project\transferLearning> python preprocessing_tl.py
Transfer-learning preprocessing done.
X: (35887, 224, 224, 3)  y: (35887, 7)
(venv) PS C:\Users\Nedlanox\Desktop\UOW\AI_Modern\Project\transferLearning> python train_tl.py
2025-11-06 21:27:40.390241: I tensorflow/core/util/port.cc:153] oneDNN custom operations are on. You may see slightly different numerical re
sults due to floating-point round-off errors from different computation orders. To turn them off, set the environment variable `TF_ENABLE_ON
EDNN_OPTS=0`.
2025-11-06 21:27:42.953169: I tensorflow/core/util/port.cc:153] oneDNN custom operations are on. You may see slightly different numerical re
sults due to floating-point round-off errors from different computation orders. To turn them off, set the environment variable `TF_ENABLE_ON
EDNN_OPTS=0`.
2025-11-06 21:28:24.983400: I tensorflow/core/platform/cpu_feature_guard.cc:210] This TensorFlow binary is optimized to use available CPU in
structions in performance-critical operations.
To enable the following instructions: SSE3 SSE4.1 SSE4.2 AVX AVX2 AVX_VNNI FMA, in other operations, rebuild TensorFlow with the appropriate
 compiler flags.
Downloading data from https://storage.googleapis.com/keras-applications/efficientnetb0_notop.h5
16705208/16705208 ━━━━━━━━━━━━━━━━━━━━ 2s 0us/step
Epoch 1/8
455/455 ━━━━━━━━━━━━━━━━━━━━ 0s 978ms/step - accuracy: 0.2624 - loss: 1.8651WARNING:absl:You are saving your model as an HDF5 file via `mode
l.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')`.
455/455 ━━━━━━━━━━━━━━━━━━━━ 506s 1s/step - accuracy: 0.2923 - loss: 1.8097 - val_accuracy: 0.2728 - val_loss: 1.7950 - learning_rate: 0.001
0
Epoch 2/8
455/455 ━━━━━━━━━━━━━━━━━━━━ 0s 940ms/step - accuracy: 0.3371 - loss: 1.7248WARNING:absl:You are saving your model as an HDF5 file via `mode
```

```
l.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')`.
455/455 ━━━━━━━━━━━━━━━━━━━━ 473s 1s/step - accuracy: 0.3400 - loss: 1.7219 - val_accuracy: 0.4096 - val_loss: 1.6551 - learning_rate: 0.001
0
Epoch 3/8
455/455 ━━━━━━━━━━━━━━━━━━━━ 488s 1s/step - accuracy: 0.3571 - loss: 1.7020 - val_accuracy: 0.3814 - val_loss: 1.6707 - learning_rate: 0.001
0
Epoch 4/8
455/455 ━━━━━━━━━━━━━━━━━━━━ 0s 936ms/step - accuracy: 0.3662 - loss: 1.6873WARNING:absl:You are saving your model as an HDF5 file via `mode
l.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')`.
455/455 ━━━━━━━━━━━━━━━━━━━━ 471s 1s/step - accuracy: 0.3674 - loss: 1.6810 - val_accuracy: 0.4353 - val_loss: 1.5825 - learning_rate: 0.001
0
Epoch 5/8
455/455 ━━━━━━━━━━━━━━━━━━━━ 0s 970ms/step - accuracy: 0.3729 - loss: 1.6793WARNING:absl:You are saving your model as an HDF5 file via `mode
l.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')`.
455/455 ━━━━━━━━━━━━━━━━━━━━ 485s 1s/step - accuracy: 0.3710 - loss: 1.6753 - val_accuracy: 0.4533 - val_loss: 1.5516 - learning_rate: 0.001
0
Epoch 6/8
455/455 ━━━━━━━━━━━━━━━━━━━━ 498s 1s/step - accuracy: 0.3747 - loss: 1.6719 - val_accuracy: 0.4068 - val_loss: 1.6470 - learning_rate: 0.001
0
Epoch 7/8
455/455 ━━━━━━━━━━━━━━━━━━━━ 474s 1s/step - accuracy: 0.3770 - loss: 1.6596 - val_accuracy: 0.4124 - val_loss: 1.6173 - learning_rate: 0.001
0
Epoch 8/8
```

```
455/455 ━━━━━━━━━━━━━━━━━━━━ 461s 1s/step - accuracy: 0.3811 - loss: 1.6447 - val_accuracy: 0.3858 - val_loss: 1.6551 - learning_rate: 0.001
0
Epoch 1/25
455/455 ━━━━━━━━━━━━━━━━━━━━ 0s 1s/step - accuracy: 0.3384 - loss: 1.8982WARNING:absl:You are saving your model as an HDF5 file via `model.s
ave()` 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')`.
455/455 ━━━━━━━━━━━━━━━━━━━━ 699s 2s/step - accuracy: 0.3741 - loss: 1.7241 - val_accuracy: 0.4786 - val_loss: 1.4998 - learning_rate: 1.000
0e-04
Epoch 2/25
455/455 ━━━━━━━━━━━━━━━━━━━━ 0s 1s/step - accuracy: 0.4536 - loss: 1.5258WARNING:absl:You are saving your model as an HDF5 file via `model.s
ave()` 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')`.
455/455 ━━━━━━━━━━━━━━━━━━━━ 727s 1s/step - accuracy: 0.4667 - loss: 1.5028 - val_accuracy: 0.5211 - val_loss: 1.4177 - learning_rate: 1.000
0e-04
Epoch 3/25
455/455 ━━━━━━━━━━━━━━━━━━━━ 0s 1s/step - accuracy: 0.5082 - loss: 1.4283WARNING:absl:You are saving your model as an HDF5 file via `model.s
ave()` 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')`.
455/455 ━━━━━━━━━━━━━━━━━━━━ 670s 1s/step - accuracy: 0.5087 - loss: 1.4213 - val_accuracy: 0.5582 - val_loss: 1.3447 - learning_rate: 1.000
0e-04
Epoch 4/25
455/455 ━━━━━━━━━━━━━━━━━━━━ 0s 1s/step - accuracy: 0.5343 - loss: 1.3692WARNING:absl:You are saving your model as an HDF5 file via `model.s
ave()` 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')`.
455/455 ━━━━━━━━━━━━━━━━━━━━ 671s 1s/step - accuracy: 0.5383 - loss: 1.3591 - val_accuracy: 0.5669 - val_loss: 1.3358 - learning_rate: 1.000
0e-04
```

```
0e-04
Epoch 10/25
455/455 ━━━━━━━━━━━━━━━━━━━━ 0s 1s/step - accuracy: 0.6247 - loss: 1.1733WARNING:absl:You are saving your model as an HDF5 file via `model.s
ave()` 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')`.
455/455 ━━━━━━━━━━━━━━━━━━━━ 690s 2s/step - accuracy: 0.6304 - loss: 1.1629 - val_accuracy: 0.6316 - val_loss: 1.2213 - learning_rate: 1.000
0e-04
Epoch 11/25
455/455 ━━━━━━━━━━━━━━━━━━━━ 679s 1s/step - accuracy: 0.6431 - loss: 1.1412 - val_accuracy: 0.6294 - val_loss: 1.2263 - learning_rate: 1.000
0e-04
Epoch 12/25
455/455 ━━━━━━━━━━━━━━━━━━━━ 679s 1s/step - accuracy: 0.6539 - loss: 1.1212 - val_accuracy: 0.6254 - val_loss: 1.2242 - learning_rate: 1.000
0e-04
Epoch 13/25
455/455 ━━━━━━━━━━━━━━━━━━━━ 0s 1s/step - accuracy: 0.6616 - loss: 1.1083WARNING:absl:You are saving your model as an HDF5 file via `model.s
ave()` 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')`.
455/455 ━━━━━━━━━━━━━━━━━━━━ 680s 1s/step - accuracy: 0.6621 - loss: 1.1017 - val_accuracy: 0.6418 - val_loss: 1.2084 - learning_rate: 1.000
0e-04
Epoch 14/25
455/455 ━━━━━━━━━━━━━━━━━━━━ 0s 1s/step - accuracy: 0.6727 - loss: 1.0831WARNING:absl:You are saving your model as an HDF5 file via `model.s
ave()` 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')`.
455/455 ━━━━━━━━━━━━━━━━━━━━ 678s 1s/step - accuracy: 0.6726 - loss: 1.0874 - val_accuracy: 0.6424 - val_loss: 1.2033 - learning_rate: 1.000
0e-04
Epoch 15/25
```

```
Epoch 15/25
455/455 ━━━━━━━━━━━━━━━━━━━━ 676s 1s/step - accuracy: 0.6821 - loss: 1.0665 - val_accuracy: 0.6412 - val_loss: 1.1997 - learning_rate: 1.000
0e-04
Epoch 16/25
455/455 ━━━━━━━━━━━━━━━━━━━━ 0s 1s/step - accuracy: 0.6910 - loss: 1.0493WARNING:absl:You are saving your model as an HDF5 file via `model.s
ave()` 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')`.
455/455 ━━━━━━━━━━━━━━━━━━━━ 680s 1s/step - accuracy: 0.6895 - loss: 1.0532 - val_accuracy: 0.6508 - val_loss: 1.1910 - learning_rate: 1.000
0e-04
Epoch 17/25
455/455 ━━━━━━━━━━━━━━━━━━━━ 678s 1s/step - accuracy: 0.6987 - loss: 1.0333 - val_accuracy: 0.6480 - val_loss: 1.1845 - learning_rate: 1.000
0e-04
Epoch 18/25
455/455 ━━━━━━━━━━━━━━━━━━━━ 0s 1s/step - accuracy: 0.7048 - loss: 1.0180WARNING:absl:You are saving your model as an HDF5 file via `model.s
ave()` 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')`.
455/455 ━━━━━━━━━━━━━━━━━━━━ 677s 1s/step - accuracy: 0.7066 - loss: 1.0180 - val_accuracy: 0.6529 - val_loss: 1.2029 - learning_rate: 1.000
0e-04
Epoch 19/25
455/455 ━━━━━━━━━━━━━━━━━━━━ 0s 1s/step - accuracy: 0.7181 - loss: 1.0008WARNING:absl:You are saving your model as an HDF5 file via `model.s
ave()` 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')`.
455/455 ━━━━━━━━━━━━━━━━━━━━ 680s 1s/step - accuracy: 0.7187 - loss: 0.9996 - val_accuracy: 0.6557 - val_loss: 1.1820 - learning_rate: 1.000
0e-04
Epoch 20/25
455/455 ━━━━━━━━━━━━━━━━━━━━ 0s 1s/step - accuracy: 0.7274 - loss: 0.9867WARNING:absl:You are saving your model as an HDF5 file via `model.s
ave()` 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')`.
455/455 ━━━━━━━━━━━━━━━━━━━━ 677s 1s/step - accuracy: 0.7246 - loss: 0.9869 - val_accuracy: 0.6604 - val_loss: 1.1843 - learning_rate: 1.000
0e-04
Epoch 21/25
455/455 ━━━━━━━━━━━━━━━━━━━━ 0s 1s/step - accuracy: 0.7322 - loss: 0.9681WARNING:absl:You are saving your model as an HDF5 file via `model.s
ave()` 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')`.
455/455 ━━━━━━━━━━━━━━━━━━━━ 681s 1s/step - accuracy: 0.7331 - loss: 0.9698 - val_accuracy: 0.6635 - val_loss: 1.1832 - learning_rate: 1.000
0e-04
Epoch 22/25
455/455 ━━━━━━━━━━━━━━━━━━━━ 679s 1s/step - accuracy: 0.7376 - loss: 0.9611 - val_accuracy: 0.6628 - val_loss: 1.1868 - learning_rate: 1.000
0e-04
Epoch 23/25
455/455 ━━━━━━━━━━━━━━━━━━━━ 677s 1s/step - accuracy: 0.7555 - loss: 0.9305 - val_accuracy: 0.6632 - val_loss: 1.1832 - learning_rate: 5.000
0e-05
Epoch 24/25
455/455 ━━━━━━━━━━━━━━━━━━━━ 0s 1s/step - accuracy: 0.7621 - loss: 0.9227WARNING:absl:You are saving your model as an HDF5 file via `model.s
ave()` 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')`.
455/455 ━━━━━━━━━━━━━━━━━━━━ 676s 1s/step - accuracy: 0.7613 - loss: 0.9221 - val_accuracy: 0.6666 - val_loss: 1.1757 - learning_rate: 5.000
0e-05
Epoch 25/25
455/455 ━━━━━━━━━━━━━━━━━━━━ 677s 1s/step - accuracy: 0.7642 - loss: 0.9114 - val_accuracy: 0.6650 - val_loss: 1.1809 - learning_rate: 5.000
0e-05
Saved transfer-learning model to disk
```

```
0e-05
Saved transfer-learning model to disk
Saved training_history_tl.png
(venv) PS C:\Users\Nedlanox\Desktop\UOW\AI_Modern\Project\transferLearning> python test_tl.py
2025-11-07 03:30:35.741888: I tensorflow/core/util/port.cc:153] oneDNN custom operations are on. You may see slightly different numerical re
sults due to floating-point round-off errors from different computation orders. To turn them off, set the environment variable `TF_ENABLE_ON
EDNN_OPTS=0`.
2025-11-07 03:30:38.996965: I tensorflow/core/util/port.cc:153] oneDNN custom operations are on. You may see slightly different numerical re
sults due to floating-point round-off errors from different computation orders. To turn them off, set the environment variable `TF_ENABLE_ON
EDNN_OPTS=0`.
2025-11-07 03:30:41.456432: I tensorflow/core/platform/cpu_feature_guard.cc:210] This TensorFlow binary is optimized to use available CPU in
structions in performance-critical operations.
To enable the following instructions: SSE3 SSE4.1 SSE4.2 AVX AVX2 AVX_VNNI FMA, in other operations, rebuild TensorFlow with the appropriate
C:\Users\Nedlanox\Desktop\UOW\AI_Modern\Project\transferLearning\venv\lib\site-packages\keras\src\saving\saving_lib.py:797: UserWarning: Ski
pping variable loading for optimizer 'rmsprop', because it has 71 variables whereas the saved optimizer has 140 variables.
  saveable.load_own_variables(weights_store.get(inner_path))
Loaded transfer-learning model
Accuracy on TL test set: 66.06%
Saved predy.npy and truey.npy for confmatrix.py
(venv) PS C:\Users\Nedlanox\Desktop\UOW\AI_Modern\Project\transferLearning> python confmatrix.py
Confusion matrix plot saved as confusion_matrix.png
(venv) PS C:\Users\Nedlanox\Desktop\UOW\AI_Modern\Project\transferLearning>
```

PS C:\Users\Nedlanox\Desktop\UOW\AI_Modern\Project\transferLearning> python -m venv venv

```
PS C:\Users\Nedlanox\Desktop\UOW\AI_Modern\Project\transferLearning> Set-ExecutionPolicy
-Scope CurrentUser -ExecutionPolicy RemoteSigned
PS C:\Users\Nedlanox\Desktop\UOW\AI_Modern\Project\transferLearning>
.\venv\Scripts\activate
>>
(venv) PS C:\Users\Nedlanox\Desktop\UOW\AI_Modern\Project\transferLearning> pip install
tensorflow keras numpy scikit-learn pandas opencv-python matplotlib
Collecting tensorflow
  Downloading tensorflow-2.20.0-cp310-cp310-win_amd64.whl (331.7 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━
331.7/331.7 MB 6.6 MB/s eta 0:00:00
Collecting keras
  Downloading keras-3.12.0-py3-none-any.whl (1.5 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━
1.5/1.5 MB 15.6 MB/s eta 0:00:00
Collecting numpy
  Downloading numpy-2.2.6-cp310-cp310-win_amd64.whl (12.9 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━
12.9/12.9 MB 13.6 MB/s eta 0:00:00
Collecting scikit-learn
  Downloading scikit_learn-1.7.2-cp310-cp310-win_amd64.whl (8.9 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━
8.9/8.9 MB 13.8 MB/s eta 0:00:00
Collecting pandas
  Downloading pandas-2.3.3-cp310-cp310-win_amd64.whl (11.3 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━
11.3/11.3 MB 13.9 MB/s eta 0:00:00
Collecting opencv-python
  Using cached opencv_python-4.12.0.88-cp37-abi3-win_amd64.whl (39.0 MB)
Collecting matplotlib
  Downloading matplotlib-3.10.7-cp310-cp310-win_amd64.whl (8.1 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━
8.1/8.1 MB 14.8 MB/s eta 0:00:00
Collecting absl-py>=1.0.0
  Using cached absl_py-2.3.1-py3-none-any.whl (135 kB)
Collecting packaging
  Downloading packaging-25.0-py3-none-any.whl (66 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━
66.5/66.5 kB ? eta 0:00:00
Collecting requests<3,>=2.21.0
  Downloading requests-2.32.5-py3-none-any.whl (64 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━
64.7/64.7 kB ? eta 0:00:00
Collecting ml_dtypes<1.0.0,>=0.5.1
```

```
  Using cached ml_dtypes-0.5.3-cp310-cp310-win_amd64.whl (206 kB)
Collecting h5py>=3.11.0
  Downloading h5py-3.15.1-cp310-cp310-win_amd64.whl (2.9 MB)
     ---------------------------------------- 2.9/2.9 MB 10.8 MB/s eta 0:00:00
Collecting google_pasta>=0.1.1
  Using cached google_pasta-0.2.0-py3-none-any.whl (57 kB)
Collecting libclang>=13.0.0
  Using cached libclang-18.1.1-py2.py3-none-win_amd64.whl (26.4 MB)
Collecting grpcio<2.0,>=1.24.3
  Downloading grpcio-1.76.0-cp310-cp310-win_amd64.whl (4.7 MB)
     ---------------------------------------- 4.7/4.7 MB 7.2 MB/s eta 0:00:00
Collecting gast!=0.5.0,!=0.5.1,!=0.5.2,>=0.2.1
  Using cached gast-0.6.0-py3-none-any.whl (21 kB)
Collecting astunparse>=1.6.0
  Using cached astunparse-1.6.3-py2.py3-none-any.whl (12 kB)
Collecting six>=1.12.0
  Using cached six-1.17.0-py2.py3-none-any.whl (11 kB)
Collecting protobuf>=5.28.0
  Downloading protobuf-6.33.0-cp310-abi3-win_amd64.whl (436 kB)
     ---------------------------------------- 436.9/436.9 kB 26.7 MB/s eta 0:00:00
Collecting wrapt>=1.11.0
  Downloading wrapt-2.0.0-cp310-cp310-win_amd64.whl (60 kB)
     ---------------------------------------- 60.3/60.3 kB ? eta 0:00:00
Requirement already satisfied: setuptools in
c:\users\nedlanox\desktop\uow\ai_modern\project\transferlearning\venv\lib\site-packages (from
tensorflow) (65.5.0)
Collecting opt_einsum>=2.3.2
  Using cached opt_einsum-3.4.0-py3-none-any.whl (71 kB)
Collecting tensorboard~=2.20.0
  Downloading tensorboard-2.20.0-py3-none-any.whl (5.5 MB)
     ---------------------------------------- 5.5/5.5 MB 14.7 MB/s eta 0:00:00
Collecting flatbuffers>=24.3.25
  Using cached flatbuffers-25.9.23-py2.py3-none-any.whl (30 kB)
Collecting termcolor>=1.1.0
  Downloading termcolor-3.2.0-py3-none-any.whl (7.7 kB)
Collecting typing_extensions>=3.6.6
  Downloading typing_extensions-4.15.0-py3-none-any.whl (44 kB)
     ---------------------------------------- 44.6/44.6 kB ? eta 0:00:00
```

```
Collecting namex
  Downloading namex-0.1.0-py3-none-any.whl (5.9 kB)
Collecting rich
  Downloading rich-14.2.0-py3-none-any.whl (243 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━
243.4/243.4 kB 14.6 MB/s eta 0:00:00
Collecting optree
  Downloading optree-0.17.0-cp310-cp310-win_amd64.whl (304 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━
304.4/304.4 kB 18.4 MB/s eta 0:00:00
Collecting joblib>=1.2.0
  Downloading joblib-1.5.2-py3-none-any.whl (308 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━
308.4/308.4 kB 18.6 MB/s eta 0:00:00
Collecting threadpoolctl>=3.1.0
  Downloading threadpoolctl-3.6.0-py3-none-any.whl (18 kB)
Collecting scipy>=1.8.0
  Downloading scipy-1.15.3-cp310-cp310-win_amd64.whl (41.3 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━
41.3/41.3 MB 13.1 MB/s eta 0:00:00
Collecting pytz>=2020.1
  Downloading pytz-2025.2-py2.py3-none-any.whl (509 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━
509.2/509.2 kB 31.2 MB/s eta 0:00:00
Collecting python-dateutil>=2.8.2
  Using cached python_dateutil-2.9.0.post0-py2.py3-none-any.whl (229 kB)
Collecting tzdata>=2022.7
  Downloading tzdata-2025.2-py2.py3-none-any.whl (347 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━
347.8/347.8 kB ? eta 0:00:00
Collecting pyparsing>=3
  Downloading pyparsing-3.2.5-py3-none-any.whl (113 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━
113.9/113.9 kB ? eta 0:00:00
Collecting cycler>=0.10
  Using cached cycler-0.12.1-py3-none-any.whl (8.3 kB)
Collecting contourpy>=1.0.1
  Downloading contourpy-1.3.2-cp310-cp310-win_amd64.whl (221 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━
221.2/221.2 kB 14.1 MB/s eta 0:00:00
Collecting fonttools>=4.22.0
  Downloading fonttools-4.60.1-cp310-cp310-win_amd64.whl (2.3 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━
2.3/2.3 MB 13.2 MB/s eta 0:00:00
```

```
Collecting kiwisolver>=1.3.1
  Downloading kiwisolver-1.4.9-cp310-cp310-win_amd64.whl (73 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━
73.7/73.7 kB ? eta 0:00:00
Collecting pillow>=8
  Downloading pillow-12.0.0-cp310-cp310-win_amd64.whl (7.0 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━
7.0/7.0 MB 9.1 MB/s eta 0:00:00
Collecting wheel<1.0,>=0.23.0
  Using cached wheel-0.45.1-py3-none-any.whl (72 kB)
Collecting urllib3<3,>=1.21.1
  Downloading urllib3-2.5.0-py3-none-any.whl (129 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━
129.8/129.8 kB 8.0 MB/s eta 0:00:00
Collecting idna<4,>=2.5
  Downloading idna-3.11-py3-none-any.whl (71 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━
71.0/71.0 kB ? eta 0:00:00
Collecting charset_normalizer<4,>=2
  Downloading charset_normalizer-3.4.4-cp310-cp310-win_amd64.whl (107 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━
107.2/107.2 kB ? eta 0:00:00
Collecting certifi>=2017.4.17
  Downloading certifi-2025.10.5-py3-none-any.whl (163 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━
163.3/163.3 kB 9.6 MB/s eta 0:00:00
Collecting werkzeug>=1.0.1
  Using cached werkzeug-3.1.3-py3-none-any.whl (224 kB)
Collecting markdown>=2.6.8
  Downloading markdown-3.10-py3-none-any.whl (107 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━
107.7/107.7 kB ? eta 0:00:00
Collecting tensorboard-data-server<0.8.0,>=0.7.0
  Using cached tensorboard_data_server-0.7.2-py3-none-any.whl (2.4 kB)
Collecting pygments<3.0.0,>=2.13.0
  Downloading pygments-2.19.2-py3-none-any.whl (1.2 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━
1.2/1.2 MB 15.6 MB/s eta 0:00:00
Collecting markdown-it-py>=2.2.0
  Downloading markdown_it_py-4.0.0-py3-none-any.whl (87 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━
87.3/87.3 kB ? eta 0:00:00
Collecting mdurl~=0.1
  Using cached mdurl-0.1.2-py3-none-any.whl (10.0 kB)
```

```
Collecting MarkupSafe>=2.1.1
  Downloading markupsafe-3.0.3-cp310-cp310-win_amd64.whl (15 kB)
Installing collected packages: pytz, namex, libclang, flatbuffers, wrapt, wheel, urllib3, tzdata,
typing_extensions, threadpoolctl, termcolor, tensorboard-data-server, six, pyparsing, pygments,
protobuf, pillow, packaging, opt_einsum, numpy, mdurl, MarkupSafe, markdown, kiwisolver,
joblib, idna, gast, fonttools, cycler, charset_normalizer, certifi, absl-py, werkzeug, scipy,
requests, python-dateutil, optree, opencv-python, ml_dtypes, markdown-it-py, h5py, grpcio,
google_pasta, contourpy, astunparse, tensorboard, scikit-learn, rich, pandas, matplotlib, keras,
tensorflow
Successfully installed MarkupSafe-3.0.3 absl-py-2.3.1 astunparse-1.6.3 certifi-2025.10.5
charset_normalizer-3.4.4 contourpy-1.3.2 cycler-0.12.1 flatbuffers-25.9.23 fonttools-4.60.1
gast-0.6.0 google_pasta-0.2.0 grpcio-1.76.0 h5py-3.15.1 idna-3.11 joblib-1.5.2 keras-3.12.0
kiwisolver-1.4.9 libclang-18.1.1 markdown-3.10 markdown-it-py-4.0.0 matplotlib-3.10.7
mdurl-0.1.2 ml_dtypes-0.5.3 namex-0.1.0 numpy-2.2.6 opencv-python-4.12.0.88
opt_einsum-3.4.0 optree-0.17.0 packaging-25.0 pandas-2.3.3 pillow-12.0.0 protobuf-6.33.0
pygments-2.19.2 pyparsing-3.2.5 python-dateutil-2.9.0.post0 pytz-2025.2 requests-2.32.5
rich-14.2.0 scikit-learn-1.7.2 scipy-1.15.3 six-1.17.0 tensorboard-2.20.0
tensorboard-data-server-0.7.2 tensorflow-2.20.0 termcolor-3.2.0 threadpoolctl-3.6.0
typing_extensions-4.15.0 tzdata-2025.2 urllib3-2.5.0 werkzeug-3.1.3 wheel-0.45.1 wrapt-2.0.0

[notice] A new release of pip is available: 23.0.1 -> 25.3
[notice] To update, run: python.exe -m pip install --upgrade pip
(venv) PS C:\Users\Nedlanox\Desktop\UOW\AI_Modern\Project\transferLearning> python -c
"from tensorflow.keras.models import model_from_json; from
tensorflow.keras.applications.efficientnet import preprocess_input; print('OK')"
>>
2025-11-06 21:22:03.818838: 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`.
2025-11-06 21:23:44.754056: 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`.
OK
(venv) PS C:\Users\Nedlanox\Desktop\UOW\AI_Modern\Project\transferLearning>
(venv) PS C:\Users\Nedlanox\Desktop\UOW\AI_Modern\Project\transferLearning> python
preprocessing_tl.py
Transfer-learning preprocessing done.
X: (35887, 224, 224, 3)  y: (35887, 7)
(venv) PS C:\Users\Nedlanox\Desktop\UOW\AI_Modern\Project\transferLearning> python
train_tl.py
2025-11-06 21:27:40.390241: 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`.
2025-11-06 21:27:42.953169: 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`.
2025-11-06 21:28:24.983400: 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 AVX_VNNI FMA, in other operations, rebuild TensorFlow with the appropriate compiler flags.
Downloading data from https://storage.googleapis.com/keras-applications/efficientnetb0_notop.h5
16705208/16705208 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 2s 0us/step
Epoch 1/8
455/455 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 0s 978ms/step - accuracy: 0.2624 - loss: 1.8651WARNING: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')`.
455/455 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 506s 1s/step - accuracy: 0.2923 - loss: 1.8097 - val_accuracy: 0.2728 - val_loss: 1.7950 - learning_rate: 0.0010
Epoch 2/8
455/455 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 0s 940ms/step - accuracy: 0.3371 - loss: 1.7248WARNING: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')`.
455/455 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 473s 1s/step - accuracy: 0.3400 - loss: 1.7219 - val_accuracy: 0.4096 - val_loss: 1.6551 - learning_rate: 0.0010
Epoch 3/8
455/455 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 488s 1s/step - accuracy: 0.3571 - loss: 1.7020 - val_accuracy: 0.3814 - val_loss: 1.6707 - learning_rate: 0.0010
Epoch 4/8
455/455 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 0s 936ms/step - accuracy: 0.3662 - loss: 1.6873WARNING: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')`.
455/455 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 471s 1s/step - accuracy: 0.3674 - loss: 1.6810 - val_accuracy: 0.4353 - val_loss: 1.5825 - learning_rate: 0.0010
Epoch 5/8
455/455 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 0s 970ms/step - accuracy: 0.3729 - loss: 1.6793WARNING: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')`.
455/455 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 485s 1s/step - accuracy: 0.3710 - loss: 1.6753 - val_accuracy: 0.4533 - val_loss: 1.5516 - learning_rate: 0.0010
Epoch 6/8
455/455 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 498s 1s/step - accuracy: 0.3747 - loss: 1.6719 - val_accuracy: 0.4068 - val_loss: 1.6470 - learning_rate: 0.0010
Epoch 7/8
455/455 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 474s 1s/step - accuracy: 0.3770 - loss: 1.6596 - val_accuracy: 0.4124 - val_loss: 1.6173 - learning_rate: 0.0010
Epoch 8/8
455/455 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 461s 1s/step - accuracy: 0.3811 - loss: 1.6447 - val_accuracy: 0.3858 - val_loss: 1.6551 - learning_rate: 0.0010
Epoch 1/25
455/455 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 0s 1s/step - accuracy: 0.3384 - loss: 1.8982WARNING: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')`.
455/455 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 699s 2s/step - accuracy: 0.3741 - loss: 1.7241 - val_accuracy: 0.4786 - val_loss: 1.4998 - learning_rate: 1.0000e-04
Epoch 2/25
455/455 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 0s 1s/step - accuracy: 0.4536 - loss: 1.5258WARNING: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')`.
455/455 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 727s 1s/step - accuracy: 0.4667 - loss: 1.5028 - val_accuracy: 0.5211 - val_loss: 1.4177 - learning_rate: 1.0000e-04
Epoch 3/25
455/455 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 0s 1s/step - accuracy: 0.5082 - loss: 1.4283WARNING: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')`.
455/455 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 670s 1s/step - accuracy: 0.5087 - loss: 1.4213 - val_accuracy: 0.5582 - val_loss: 1.3447 - learning_rate: 1.0000e-04
Epoch 4/25
455/455 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 0s 1s/step - accuracy: 0.5343 - loss: 1.3692WARNING: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')`.

455/455 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 671s 1s/step - accuracy: 0.5383 - loss: 1.3591 - val_accuracy: 0.5669 - val_loss: 1.3358 - learning_rate: 1.0000e-04
Epoch 5/25
455/455 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 0s 1s/step - accuracy: 0.5534 - loss: 1.3222WARNING: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')`.
455/455 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 674s 1s/step - accuracy: 0.5604 - loss: 1.3106 - val_accuracy: 0.5854 - val_loss: 1.2970 - learning_rate: 1.0000e-04
Epoch 6/25
455/455 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 0s 1s/step - accuracy: 0.5754 - loss: 1.2787WARNING: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')`.
455/455 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 671s 1s/step - accuracy: 0.5764 - loss: 1.2742 - val_accuracy: 0.5938 - val_loss: 1.2776 - learning_rate: 1.0000e-04
Epoch 7/25
455/455 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 0s 1s/step - accuracy: 0.5901 - loss: 1.2533WARNING: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')`.
455/455 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 677s 1s/step - accuracy: 0.5908 - loss: 1.2470 - val_accuracy: 0.6019 - val_loss: 1.2692 - learning_rate: 1.0000e-04
Epoch 8/25
455/455 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 672s 1s/step - accuracy: 0.6108 - loss: 1.2080 - val_accuracy: 0.5954 - val_loss: 1.2668 - learning_rate: 1.0000e-04
Epoch 9/25
455/455 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 0s 1s/step - accuracy: 0.6101 - loss: 1.1931WARNING: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')`.
455/455 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 679s 1s/step - accuracy: 0.6172 - loss: 1.1880 - val_accuracy: 0.6136 - val_loss: 1.2554 - learning_rate: 1.0000e-04
Epoch 10/25
455/455 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 0s 1s/step - accuracy: 0.6247 - loss: 1.1733WARNING: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')`.

455/455 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 690s 2s/step - accuracy: 0.6304 - loss: 1.1629 - val_accuracy: 0.6316 - val_loss: 1.2213 - learning_rate: 1.0000e-04
Epoch 11/25
455/455 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 679s 1s/step - accuracy: 0.6431 - loss: 1.1412 - val_accuracy: 0.6294 - val_loss: 1.2263 - learning_rate: 1.0000e-04
Epoch 12/25
455/455 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 679s 1s/step - accuracy: 0.6539 - loss: 1.1212 - val_accuracy: 0.6254 - val_loss: 1.2242 - learning_rate: 1.0000e-04
Epoch 13/25
455/455 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 0s 1s/step - accuracy: 0.6616 - loss: 1.1083WARNING: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')`.
455/455 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 680s 1s/step - accuracy: 0.6621 - loss: 1.1017 - val_accuracy: 0.6418 - val_loss: 1.2084 - learning_rate: 1.0000e-04
Epoch 14/25
455/455 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 0s 1s/step - accuracy: 0.6727 - loss: 1.0831WARNING: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')`.
455/455 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 678s 1s/step - accuracy: 0.6726 - loss: 1.0874 - val_accuracy: 0.6424 - val_loss: 1.2033 - learning_rate: 1.0000e-04
Epoch 15/25
455/455 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 676s 1s/step - accuracy: 0.6821 - loss: 1.0665 - val_accuracy: 0.6412 - val_loss: 1.1997 - learning_rate: 1.0000e-04
Epoch 16/25
455/455 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 0s 1s/step - accuracy: 0.6910 - loss: 1.0493WARNING: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')`.
455/455 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 680s 1s/step - accuracy: 0.6895 - loss: 1.0532 - val_accuracy: 0.6508 - val_loss: 1.1910 - learning_rate: 1.0000e-04
Epoch 17/25
455/455 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 678s 1s/step - accuracy: 0.6987 - loss: 1.0333 - val_accuracy: 0.6480 - val_loss: 1.1845 - learning_rate: 1.0000e-04
Epoch 18/25
455/455 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 0s 1s/step - accuracy: 0.7048 - loss: 1.0180WARNING: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')`.

455/455 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 677s 1s/step - accuracy: 0.7066 - loss: 1.0180 - val_accuracy: 0.6529 - val_loss: 1.2029 - learning_rate: 1.0000e-04
Epoch 19/25
455/455 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 0s 1s/step - accuracy: 0.7181 - loss: 1.0008WARNING: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')`.
455/455 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 680s 1s/step - accuracy: 0.7187 - loss: 0.9996 - val_accuracy: 0.6557 - val_loss: 1.1820 - learning_rate: 1.0000e-04
Epoch 20/25
455/455 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 0s 1s/step - accuracy: 0.7274 - loss: 0.9867WARNING: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')`.
455/455 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 677s 1s/step - accuracy: 0.7246 - loss: 0.9869 - val_accuracy: 0.6604 - val_loss: 1.1843 - learning_rate: 1.0000e-04
Epoch 21/25
455/455 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 0s 1s/step - accuracy: 0.7322 - loss: 0.9681WARNING: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')`.
455/455 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 681s 1s/step - accuracy: 0.7331 - loss: 0.9698 - val_accuracy: 0.6635 - val_loss: 1.1832 - learning_rate: 1.0000e-04
Epoch 22/25
455/455 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 679s 1s/step - accuracy: 0.7376 - loss: 0.9611 - val_accuracy: 0.6628 - val_loss: 1.1868 - learning_rate: 1.0000e-04
Epoch 23/25
455/455 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 677s 1s/step - accuracy: 0.7555 - loss: 0.9305 - val_accuracy: 0.6632 - val_loss: 1.1832 - learning_rate: 5.0000e-05
Epoch 24/25
455/455 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 0s 1s/step - accuracy: 0.7621 - loss: 0.9227WARNING: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')`.
455/455 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 676s 1s/step - accuracy: 0.7613 - loss: 0.9221 - val_accuracy: 0.6666 - val_loss: 1.1757 - learning_rate: 5.0000e-05
Epoch 25/25
455/455 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 677s 1s/step - accuracy: 0.7642 - loss: 0.9114 - val_accuracy: 0.6650 - val_loss: 1.1809 - learning_rate: 5.0000e-05
Saved transfer-learning model to disk

```
Saved training_history_tl.png
(venv) PS C:\Users\Nedlanox\Desktop\UOW\AI_Modern\Project\transferLearning> python
test_tl.py
2025-11-07 03:30:35.741888: 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`.
2025-11-07 03:30:38.996965: 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`.
2025-11-07 03:30:41.456432: 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 AVX_VNNI FMA, in other
operations, rebuild TensorFlow with the
appropriateC:\Users\Nedlanox\Desktop\UOW\AI_Modern\Project\transferLearning\venv\lib\site-
packages\keras\src\saving\saving_lib.py:797: UserWarning: Skipping variable loading for
optimizer 'rmsprop', because it has 71 variables whereas the saved optimizer has 140 variables.
  saveable.load_own_variables(weights_store.get(inner_path))
Loaded transfer-learning model
Accuracy on TL test set: 66.06%
Saved predy.npy and truey.npy for confmatrix.py
(venv) PS C:\Users\Nedlanox\Desktop\UOW\AI_Modern\Project\transferLearning> python
confmatrix.py
Confusion matrix plot saved as confusion_matrix.png
(venv) PS C:\Users\Nedlanox\Desktop\UOW\AI_Modern\Project\transferLearning>
```