Ojas Chaphekar

Lukas Licon

P6 Writeup

Submit a pdf containing the following:

**The final classification accuracy on the test set with transfer and without.**

Model: "sequential"

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Layer (type) Output Shape Param #

=================================================================

sequential (Sequential) (None, 64) 106632

dense\_layer (Dense) (None, 64) 4160

output\_layer (Dense) (None, 3) 195

=================================================================

Total params: 110987 (433.54 KB)

Trainable params: 4355 (17.01 KB)

Non-trainable params: 106632 (416.53 KB)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\* Evaluating transfered\_model

30/30 [==============================] - 1s 42ms/step - loss: 0.7943 - accuracy: 0.7658

\* Confusion Matrix for transfered\_model

30/30 [==============================] - 1s 42ms/step

[[1335 305 134]

[ 209 909 115]

[ 60 76 695]]

\* Training random\_model for 10 epochs

Model: "sequential\_1"

=================================================================

Total params: 106827 (417.29 KB)

Trainable params: 6411 (25.04 KB)

Non-trainable params: 100416 (392.25 KB)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Evaluating random\_model

30/30 [==============================] - 1s 45ms/step - loss: 1.1284 - accuracy: 0.3913

\* Confusion Matrix for random\_model

30/30 [==============================] - 1s 44ms/step

[[ 472 1098 204]

[ 307 810 116]

[ 190 421 220]]

**An image of each facial recognition class**







**Transfer vs. Random Model**



Basic\_Model.py Graph:

Model: "sequential"

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Layer (type) Output Shape Param #

=================================================================

rescaling (Rescaling) (None, 150, 150, 3) 0

conv2d (Conv2D) (None, 148, 148, 2) 56

max\_pooling2d (MaxPooling2 (None, 74, 74, 2) 0

D)

flatten (Flatten) (None, 10952) 0

dense (Dense) (None, 4) 43812

dense\_1 (Dense) (None, 3) 15

=================================================================

Total params: 43883 (171.42 KB)

Trainable params: 43883 (171.42 KB)

Non-trainable params: 0 (0.00 Byte)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Epoch 1/100

WARNING:tensorflow:From D:\CMPM146\myenv\lib\site-packages\keras\src\utils\tf\_utils.py:492: The name tf.ragged.RaggedTensorValue is deprecated. Please use tf.compat.v1.ragged.RaggedTensorValue instead.

WARNING:tensorflow:From D:\CMPM146\myenv\lib\site-packages\keras\src\engine\base\_layer\_utils.py:384: The name tf.executing\_eagerly\_outside\_functions is deprecated. Please use tf.compat.v1.executing\_eagerly\_outside\_functions instead.

32/32 [==============================] - 2s 44ms/step - loss: 1.0757 - accuracy: 0.4020 - val\_loss: 1.0534 - val\_accuracy: 0.4030

Epoch 2/100

32/32 [==============================] - 1s 39ms/step - loss: 1.0293 - accuracy: 0.4740 - val\_loss: 1.0218 - val\_accuracy: 0.5170

Epoch 3/100

32/32 [==============================] - 1s 39ms/step - loss: 0.9877 - accuracy: 0.5328 - val\_loss: 0.9886 - val\_accuracy: 0.5400

Epoch 4/100

32/32 [==============================] - 1s 40ms/step - loss: 0.9461 - accuracy: 0.5695 - val\_loss: 0.9911 - val\_accuracy: 0.4630

Epoch 5/100

32/32 [==============================] - 1s 39ms/step - loss: 0.9210 - accuracy: 0.5867 - val\_loss: 0.9355 - val\_accuracy: 0.5800

Epoch 6/100

32/32 [==============================] - 1s 38ms/step - loss: 0.8890 - accuracy: 0.6135 - val\_loss: 0.9416 - val\_accuracy: 0.5530

Epoch 7/100

32/32 [==============================] - 1s 39ms/step - loss: 0.8753 - accuracy: 0.6195 - val\_loss: 0.9350 - val\_accuracy: 0.5420

Epoch 8/100

32/32 [==============================] - 1s 39ms/step - loss: 0.8492 - accuracy: 0.6363 - val\_loss: 0.9080 - val\_accuracy: 0.5790

Epoch 9/100

32/32 [==============================] - 1s 40ms/step - loss: 0.8292 - accuracy: 0.6550 - val\_loss: 0.8820 - val\_accuracy: 0.6340

Epoch 10/100

32/32 [==============================] - 1s 39ms/step - loss: 0.8150 - accuracy: 0.6585 - val\_loss: 0.8865 - val\_accuracy: 0.5840

Epoch 11/100

32/32 [==============================] - 1s 40ms/step - loss: 0.7939 - accuracy: 0.6722 - val\_loss: 0.8699 - val\_accuracy: 0.6010

Epoch 12/100

32/32 [==============================] - 1s 41ms/step - loss: 0.7796 - accuracy: 0.6758 - val\_loss: 0.8882 - val\_accuracy: 0.5850

Epoch 13/100

32/32 [==============================] - 1s 40ms/step - loss: 0.7572 - accuracy: 0.6925 - val\_loss: 0.9489 - val\_accuracy: 0.5380

Epoch 14/100

32/32 [==============================] - 1s 41ms/step - loss: 0.7449 - accuracy: 0.6957 - val\_loss: 0.8453 - val\_accuracy: 0.6440

Epoch 15/100

32/32 [==============================] - 1s 41ms/step - loss: 0.7353 - accuracy: 0.7010 - val\_loss: 0.8395 - val\_accuracy: 0.6490

Epoch 16/100

32/32 [==============================] - 1s 40ms/step - loss: 0.7152 - accuracy: 0.7185 - val\_loss: 0.8633 - val\_accuracy: 0.6090

Epoch 17/100

32/32 [==============================] - 1s 40ms/step - loss: 0.6975 - accuracy: 0.7343 - val\_loss: 0.8809 - val\_accuracy: 0.6110

Epoch 18/100

32/32 [==============================] - 1s 40ms/step - loss: 0.6920 - accuracy: 0.7285 - val\_loss: 0.9504 - val\_accuracy: 0.5650

Epoch 19/100

32/32 [==============================] - 1s 40ms/step - loss: 0.6739 - accuracy: 0.7430 - val\_loss: 0.8352 - val\_accuracy: 0.6380

Epoch 20/100

32/32 [==============================] - 1s 39ms/step - loss: 0.6584 - accuracy: 0.7490 - val\_loss: 0.9108 - val\_accuracy: 0.6030

Epoch 21/100

32/32 [==============================] - 1s 41ms/step - loss: 0.6469 - accuracy: 0.7567 - val\_loss: 0.8298 - val\_accuracy: 0.6480

Epoch 22/100

32/32 [==============================] - 1s 40ms/step - loss: 0.6296 - accuracy: 0.7667 - val\_loss: 0.8428 - val\_accuracy: 0.6310

Epoch 23/100

32/32 [==============================] - 1s 41ms/step - loss: 0.6166 - accuracy: 0.7770 - val\_loss: 0.8564 - val\_accuracy: 0.6300

Epoch 24/100

32/32 [==============================] - 1s 39ms/step - loss: 0.6039 - accuracy: 0.7850 - val\_loss: 0.8226 - val\_accuracy: 0.6420

Epoch 25/100

32/32 [==============================] - 1s 40ms/step - loss: 0.5915 - accuracy: 0.7885 - val\_loss: 0.8597 - val\_accuracy: 0.6320

Epoch 26/100

32/32 [==============================] - 1s 41ms/step - loss: 0.5807 - accuracy: 0.7893 - val\_loss: 0.8620 - val\_accuracy: 0.6310

Epoch 27/100

32/32 [==============================] - 1s 39ms/step - loss: 0.5644 - accuracy: 0.7990 - val\_loss: 0.8425 - val\_accuracy: 0.6180

Epoch 28/100

32/32 [==============================] - 1s 41ms/step - loss: 0.5627 - accuracy: 0.8058 - val\_loss: 0.8486 - val\_accuracy: 0.6470

Epoch 29/100

32/32 [==============================] - 1s 41ms/step - loss: 0.5419 - accuracy: 0.8085 - val\_loss: 0.8758 - val\_accuracy: 0.6200

Epoch 30/100

32/32 [==============================] - 1s 40ms/step - loss: 0.5303 - accuracy: 0.8217 - val\_loss: 0.8453 - val\_accuracy: 0.6540

Epoch 31/100

32/32 [==============================] - 1s 41ms/step - loss: 0.5163 - accuracy: 0.8242 - val\_loss: 0.8737 - val\_accuracy: 0.6370

Epoch 32/100

32/32 [==============================] - 1s 41ms/step - loss: 0.5031 - accuracy: 0.8328 - val\_loss: 0.8558 - val\_accuracy: 0.6240

Epoch 33/100

32/32 [==============================] - 1s 40ms/step - loss: 0.5016 - accuracy: 0.8298 - val\_loss: 0.8790 - val\_accuracy: 0.6300

Epoch 34/100

32/32 [==============================] - 1s 40ms/step - loss: 0.4805 - accuracy: 0.8435 - val\_loss: 0.9957 - val\_accuracy: 0.6090

Epoch 35/100

32/32 [==============================] - 1s 39ms/step - loss: 0.4782 - accuracy: 0.8407 - val\_loss: 0.9759 - val\_accuracy: 0.6090

Epoch 36/100

32/32 [==============================] - 1s 40ms/step - loss: 0.4653 - accuracy: 0.8457 - val\_loss: 0.8967 - val\_accuracy: 0.6450

Epoch 37/100

32/32 [==============================] - 2s 42ms/step - loss: 0.4625 - accuracy: 0.8468 - val\_loss: 0.8714 - val\_accuracy: 0.6230

Epoch 38/100

32/32 [==============================] - 1s 41ms/step - loss: 0.4451 - accuracy: 0.8593 - val\_loss: 0.9783 - val\_accuracy: 0.5910

Epoch 39/100

32/32 [==============================] - 1s 40ms/step - loss: 0.4375 - accuracy: 0.8620 - val\_loss: 0.8949 - val\_accuracy: 0.6500

Epoch 40/100

32/32 [==============================] - 2s 42ms/step - loss: 0.4225 - accuracy: 0.8683 - val\_loss: 0.9042 - val\_accuracy: 0.6470

Epoch 41/100

32/32 [==============================] - 2s 45ms/step - loss: 0.4149 - accuracy: 0.8725 - val\_loss: 0.9138 - val\_accuracy: 0.6480

Epoch 42/100

32/32 [==============================] - 2s 42ms/step - loss: 0.4119 - accuracy: 0.8665 - val\_loss: 0.9213 - val\_accuracy: 0.6290

Epoch 43/100

32/32 [==============================] - 2s 42ms/step - loss: 0.4065 - accuracy: 0.8705 - val\_loss: 0.9877 - val\_accuracy: 0.6390

Epoch 44/100

32/32 [==============================] - 2s 41ms/step - loss: 0.3946 - accuracy: 0.8765 - val\_loss: 0.9279 - val\_accuracy: 0.6350

Epoch 45/100

32/32 [==============================] - 1s 40ms/step - loss: 0.3780 - accuracy: 0.8860 - val\_loss: 0.9199 - val\_accuracy: 0.6420

Epoch 46/100

32/32 [==============================] - 1s 40ms/step - loss: 0.3751 - accuracy: 0.8842 - val\_loss: 1.0047 - val\_accuracy: 0.6280

Epoch 47/100

32/32 [==============================] - 2s 41ms/step - loss: 0.3632 - accuracy: 0.8880 - val\_loss: 1.0980 - val\_accuracy: 0.6000

Epoch 48/100

32/32 [==============================] - 2s 41ms/step - loss: 0.3598 - accuracy: 0.8928 - val\_loss: 0.9943 - val\_accuracy: 0.6020

Epoch 49/100

32/32 [==============================] - 1s 41ms/step - loss: 0.3464 - accuracy: 0.8970 - val\_loss: 0.9497 - val\_accuracy: 0.6410

Epoch 50/100

32/32 [==============================] - 1s 40ms/step - loss: 0.3521 - accuracy: 0.8965 - val\_loss: 1.0885 - val\_accuracy: 0.6170

Epoch 51/100

32/32 [==============================] - 1s 41ms/step - loss: 0.3329 - accuracy: 0.9028 - val\_loss: 1.3059 - val\_accuracy: 0.5720

Epoch 52/100

32/32 [==============================] - 2s 42ms/step - loss: 0.3292 - accuracy: 0.9062 - val\_loss: 0.9960 - val\_accuracy: 0.6130

Epoch 53/100

32/32 [==============================] - 2s 42ms/step - loss: 0.3222 - accuracy: 0.9068 - val\_loss: 1.1041 - val\_accuracy: 0.6300

Epoch 54/100

32/32 [==============================] - 1s 40ms/step - loss: 0.3130 - accuracy: 0.9103 - val\_loss: 1.0479 - val\_accuracy: 0.6270

Epoch 55/100

32/32 [==============================] - 1s 40ms/step - loss: 0.3026 - accuracy: 0.9150 - val\_loss: 1.1613 - val\_accuracy: 0.6240

Epoch 56/100

32/32 [==============================] - 1s 41ms/step - loss: 0.3079 - accuracy: 0.9153 - val\_loss: 0.9896 - val\_accuracy: 0.6230

Epoch 57/100

32/32 [==============================] - 1s 40ms/step - loss: 0.2976 - accuracy: 0.9160 - val\_loss: 1.1040 - val\_accuracy: 0.6060

Epoch 58/100

32/32 [==============================] - 1s 39ms/step - loss: 0.2925 - accuracy: 0.9150 - val\_loss: 1.1446 - val\_accuracy: 0.6130

Epoch 59/100

32/32 [==============================] - 1s 40ms/step - loss: 0.2832 - accuracy: 0.9225 - val\_loss: 1.1804 - val\_accuracy: 0.6090

Epoch 60/100

32/32 [==============================] - 1s 40ms/step - loss: 0.2835 - accuracy: 0.9193 - val\_loss: 1.1035 - val\_accuracy: 0.6180

Epoch 61/100

32/32 [==============================] - 1s 41ms/step - loss: 0.2654 - accuracy: 0.9305 - val\_loss: 1.2134 - val\_accuracy: 0.6180

Epoch 62/100

32/32 [==============================] - 1s 39ms/step - loss: 0.2678 - accuracy: 0.9260 - val\_loss: 1.1509 - val\_accuracy: 0.6340

Epoch 63/100

32/32 [==============================] - 1s 39ms/step - loss: 0.2653 - accuracy: 0.9285 - val\_loss: 1.1408 - val\_accuracy: 0.6310

Epoch 64/100

32/32 [==============================] - 2s 42ms/step - loss: 0.2508 - accuracy: 0.9333 - val\_loss: 1.2398 - val\_accuracy: 0.6310

Epoch 65/100

32/32 [==============================] - 2s 42ms/step - loss: 0.2521 - accuracy: 0.9330 - val\_loss: 1.3167 - val\_accuracy: 0.6280

Epoch 66/100

32/32 [==============================] - 1s 40ms/step - loss: 0.2501 - accuracy: 0.9323 - val\_loss: 1.3311 - val\_accuracy: 0.5820

Epoch 67/100

32/32 [==============================] - 1s 40ms/step - loss: 0.2401 - accuracy: 0.9360 - val\_loss: 1.1806 - val\_accuracy: 0.6280

Epoch 68/100

32/32 [==============================] - 1s 40ms/step - loss: 0.2346 - accuracy: 0.9388 - val\_loss: 1.2652 - val\_accuracy: 0.6270

Epoch 69/100

32/32 [==============================] - 1s 39ms/step - loss: 0.2291 - accuracy: 0.9405 - val\_loss: 1.2324 - val\_accuracy: 0.6300

Epoch 70/100

32/32 [==============================] - 1s 40ms/step - loss: 0.2288 - accuracy: 0.9408 - val\_loss: 1.2963 - val\_accuracy: 0.6280

Epoch 71/100

32/32 [==============================] - 1s 40ms/step - loss: 0.2118 - accuracy: 0.9477 - val\_loss: 1.2925 - val\_accuracy: 0.6200

Epoch 72/100

32/32 [==============================] - 1s 40ms/step - loss: 0.2209 - accuracy: 0.9427 - val\_loss: 1.2337 - val\_accuracy: 0.6300

Epoch 73/100

32/32 [==============================] - 1s 40ms/step - loss: 0.2066 - accuracy: 0.9485 - val\_loss: 1.2018 - val\_accuracy: 0.6340

Epoch 74/100

32/32 [==============================] - 1s 39ms/step - loss: 0.2079 - accuracy: 0.9480 - val\_loss: 1.3936 - val\_accuracy: 0.6170

Epoch 75/100

32/32 [==============================] - 1s 41ms/step - loss: 0.2027 - accuracy: 0.9498 - val\_loss: 1.1901 - val\_accuracy: 0.6260

Epoch 76/100

32/32 [==============================] - 1s 40ms/step - loss: 0.2142 - accuracy: 0.9398 - val\_loss: 1.3755 - val\_accuracy: 0.6290

Epoch 77/100

32/32 [==============================] - 1s 40ms/step - loss: 0.1896 - accuracy: 0.9538 - val\_loss: 1.3522 - val\_accuracy: 0.6300

Epoch 78/100

32/32 [==============================] - 1s 40ms/step - loss: 0.1915 - accuracy: 0.9535 - val\_loss: 1.4155 - val\_accuracy: 0.6230

Epoch 79/100

32/32 [==============================] - 1s 39ms/step - loss: 0.1907 - accuracy: 0.9548 - val\_loss: 1.3594 - val\_accuracy: 0.6340

Epoch 80/100

32/32 [==============================] - 1s 40ms/step - loss: 0.1879 - accuracy: 0.9538 - val\_loss: 1.4286 - val\_accuracy: 0.6290

Epoch 81/100

32/32 [==============================] - 1s 40ms/step - loss: 0.1798 - accuracy: 0.9585 - val\_loss: 1.3106 - val\_accuracy: 0.6290

Epoch 82/100

32/32 [==============================] - 1s 40ms/step - loss: 0.1825 - accuracy: 0.9535 - val\_loss: 1.4026 - val\_accuracy: 0.6330

Epoch 83/100

32/32 [==============================] - 1s 40ms/step - loss: 0.1781 - accuracy: 0.9542 - val\_loss: 1.5461 - val\_accuracy: 0.6300

Epoch 84/100

32/32 [==============================] - 1s 39ms/step - loss: 0.1738 - accuracy: 0.9580 - val\_loss: 1.4240 - val\_accuracy: 0.6310

Epoch 85/100

32/32 [==============================] - 1s 40ms/step - loss: 0.1690 - accuracy: 0.9590 - val\_loss: 1.3293 - val\_accuracy: 0.6320

Epoch 86/100

32/32 [==============================] - 1s 40ms/step - loss: 0.1622 - accuracy: 0.9588 - val\_loss: 1.3746 - val\_accuracy: 0.6180

Epoch 87/100

32/32 [==============================] - 1s 39ms/step - loss: 0.1654 - accuracy: 0.9610 - val\_loss: 1.3906 - val\_accuracy: 0.6300

Epoch 88/100

32/32 [==============================] - 1s 40ms/step - loss: 0.1548 - accuracy: 0.9622 - val\_loss: 1.5284 - val\_accuracy: 0.6290

Epoch 89/100

32/32 [==============================] - 1s 41ms/step - loss: 0.1577 - accuracy: 0.9603 - val\_loss: 1.5614 - val\_accuracy: 0.6170

Epoch 90/100

32/32 [==============================] - 1s 39ms/step - loss: 0.1572 - accuracy: 0.9610 - val\_loss: 1.4447 - val\_accuracy: 0.6340

Epoch 91/100

32/32 [==============================] - 1s 40ms/step - loss: 0.1476 - accuracy: 0.9663 - val\_loss: 1.6311 - val\_accuracy: 0.6020

Epoch 92/100

32/32 [==============================] - 2s 42ms/step - loss: 0.1541 - accuracy: 0.9603 - val\_loss: 1.5313 - val\_accuracy: 0.6250

Epoch 93/100

32/32 [==============================] - 1s 40ms/step - loss: 0.1456 - accuracy: 0.9640 - val\_loss: 1.7667 - val\_accuracy: 0.5980

Epoch 94/100

32/32 [==============================] - 2s 42ms/step - loss: 0.1560 - accuracy: 0.9580 - val\_loss: 1.7840 - val\_accuracy: 0.6190

Epoch 95/100

32/32 [==============================] - 1s 40ms/step - loss: 0.1461 - accuracy: 0.9657 - val\_loss: 1.5384 - val\_accuracy: 0.5510

Epoch 96/100

32/32 [==============================] - 2s 42ms/step - loss: 0.1587 - accuracy: 0.9557 - val\_loss: 1.6107 - val\_accuracy: 0.6360

Epoch 97/100

32/32 [==============================] - 2s 43ms/step - loss: 0.1451 - accuracy: 0.9617 - val\_loss: 1.6699 - val\_accuracy: 0.6310

Epoch 98/100

32/32 [==============================] - 2s 43ms/step - loss: 0.1374 - accuracy: 0.9680 - val\_loss: 1.6547 - val\_accuracy: 0.5780

Epoch 99/100

32/32 [==============================] - 2s 42ms/step - loss: 0.1396 - accuracy: 0.9645 - val\_loss: 1.9563 - val\_accuracy: 0.5970

Epoch 100/100

32/32 [==============================] - 1s 41ms/step - loss: 0.1284 - accuracy: 0.9697 - val\_loss: 1.6809 - val\_accuracy: 0.6220

\* Evaluating basic\_model

30/30 [==============================] - 1s 33ms/step - loss: 1.6679 - accuracy: 0.6248

\* Confusion Matrix for basic\_model

30/30 [==============================] - 1s 34ms/step

[[1152 365 257]

[ 331 659 243]

[ 138 106 587]]

\* Model saved as results/basic\_model\_100\_epochs\_timestamp\_1708575451.keras

Hyperoptimized Graph:

Model: "sequential"

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Layer (type) Output Shape Param #

=================================================================

rescaling (Rescaling) (None, 150, 150, 3) 0

conv2d (Conv2D) (None, 148, 148, 4) 112

max\_pooling2d (MaxPooling2 (None, 74, 74, 4) 0

D)

conv2d\_1 (Conv2D) (None, 72, 72, 8) 296

max\_pooling2d\_1 (MaxPoolin (None, 36, 36, 8) 0

g2D)

conv2d\_2 (Conv2D) (None, 34, 34, 16) 1168

max\_pooling2d\_2 (MaxPoolin (None, 17, 17, 16) 0

g2D)

dropout (Dropout) (None, 17, 17, 16) 0

conv2d\_3 (Conv2D) (None, 15, 15, 32) 4640

max\_pooling2d\_3 (MaxPoolin (None, 7, 7, 32) 0

g2D)

dropout\_1 (Dropout) (None, 7, 7, 32) 0

flatten (Flatten) (None, 1568) 0

dense (Dense) (None, 64) 100416

dense\_1 (Dense) (None, 3) 195

=================================================================

Total params: 106827 (417.29 KB)

Trainable params: 106827 (417.29 KB)

Non-trainable params: 0 (0.00 Byte)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Epoch 1/10

WARNING:tensorflow:From D:\CMPM146\myenv\lib\sitepackages\

keras\src\utils\tf\_utils.py:492: The name

tf.ragged.RaggedTensorValue is deprecated. Please use

tf.compat.v1.ragged.RaggedTensorValue instead.

WARNING:tensorflow:From D:\CMPM146\myenv\lib\sitepackages\

keras\src\engine\base\_layer\_utils.py:384: The name

tf.executing\_eagerly\_outside\_functions is deprecated. Please use

tf.compat.v1.executing\_eagerly\_outside\_functions instead.

32/32 [==============================] - 3s 73ms/step - loss: 1.0854 -

accuracy: 0.3875 - val\_loss: 1.0539 - val\_accuracy: 0.4470

Epoch 2/10

32/32 [==============================] - 2s 70ms/step - loss: 0.9845 -

accuracy: 0.5180 - val\_loss: 0.8942 - val\_accuracy: 0.6160

Epoch 3/10

32/32 [==============================] - 2s 69ms/step - loss: 0.9161 -

accuracy: 0.5822 - val\_loss: 0.8591 - val\_accuracy: 0.6470

Epoch 4/10

32/32 [==============================] - 2s 69ms/step - loss: 0.8627 -

accuracy: 0.6170 - val\_loss: 0.8584 - val\_accuracy: 0.6070

Epoch 5/10

32/32 [==============================] - 2s 70ms/step - loss: 0.8266 -

accuracy: 0.6438 - val\_loss: 0.8034 - val\_accuracy: 0.6500

Epoch 6/10

32/32 [==============================] - 2s 69ms/step - loss: 0.7941 -

accuracy: 0.6597 - val\_loss: 0.8144 - val\_accuracy: 0.6200

Epoch 7/10

32/32 [==============================] - 2s 69ms/step - loss: 0.7827 -

accuracy: 0.6615 - val\_loss: 0.7374 - val\_accuracy: 0.6870

Epoch 8/10

32/32 [==============================] - 2s 70ms/step - loss: 0.7388 -

accuracy: 0.6823 - val\_loss: 0.7248 - val\_accuracy: 0.7020

Epoch 9/10

32/32 [==============================] - 2s 69ms/step - loss: 0.7263 -

accuracy: 0.6950 - val\_loss: 0.6959 - val\_accuracy: 0.7160

Epoch 10/10

32/32 [==============================] - 2s 69ms/step - loss: 0.7039 -

accuracy: 0.7032 - val\_loss: 0.7318 - val\_accuracy: 0.6790

\* Evaluating basic\_model

30/30 [==============================] - 1s 40ms/step - loss: 0.6633 -

accuracy: 0.7220

\* Confusion Matrix for basic\_model

30/30 [==============================] - 1s 40ms/step

[[1531 160 83]

[ 456 724 53]

[ 183 132 516]]



PS C:\Users\ojas\Desktop\CMPM 146\P6\ python run.py

| | | |

| | | |

| | | |

Turn: X

Player X took position (1, 2).

| | | |

| | |X|

| | | |

Turn: O

reference:

row 0 is neutral.

row 1 is happy.

row 2 is surprise.

1/1 [==============================] - 0s 178ms/step

Emotion detected as surprise (row 2). Enter 'text' to use text input instead (0, 1 or 2). Otherwise,

press Enter to continue.

reference:

col 0 is neutral.

col 1 is happy.

col 2 is surprise.

1/1 [==============================] - 0s 106ms/step

Emotion detected as surprise (col 2). Enter 'text' to use text input instead (0, 1 or 2). Otherwise,

press Enter to continue.

Player O took position (2, 2).

| | | |

| | |X|

| | |O|

Turn: X

Player X took position (0, 2).

| | |X|

| | |X|

| | |O|

Turn: O

reference:

row 0 is neutral.

row 1 is happy.

row 2 is surprise.

1/1 [==============================] - 0s 112ms/step

Emotion detected as surprise (row 2). Enter 'text' to use text input instead (0, 1 or 2). Otherwise,

press Enter to continue.

reference:

col 0 is neutral.

col 1 is happy.

col 2 is surprise.

1/1 [==============================] - 0s 106ms/step

Emotion detected as happy (col 1). Enter 'text' to use text input instead (0, 1 or 2). Otherwise,

press Enter to continue.

Player O took position (2, 1).

| | |X|

| | |X|

| |O|O|

Turn: X

Player X took position (0, 1).

| |X|X|

| | |X|

| |O|O|

Turn: O

reference:

row 0 is neutral.

row 1 is happy.

row 2 is surprise.

WARNING:tensorflow:5 out of the last 5 calls to <function

Model.make\_predict\_function.<locals>.predict\_function at 0x00000221184D2710> triggered

tf.function retracing. Tracing is expensive and the excessive number of tracings could be due to

(1) creating @tf.function repeatedly in a loop, (2) passing tensors with different shapes, (3)

passing Python objects instead of tensors. For (1), please define your @tf.function outside of the

loop. For (2), @tf.function has reduce\_retracing=True option that can avoid unnecessary

retracing. For (3), please refer to

<https://www.tensorflow.org/guide/function#controlling_retracing> and

<https://www.tensorflow.org/api_docs/python/tf/function> for more details.

1/1 [==============================] - 0s 101ms/step

Emotion detected as happy (row 1). Enter 'text' to use text input instead (0, 1 or 2). Otherwise,

press Enter to continue.

reference:

col 0 is neutral.

col 1 is happy.

col 2 is surprise.

WARNING:tensorflow:6 out of the last 6 calls to <function

Model.make\_predict\_function.<locals>.predict\_function at 0x00000221184D3490> triggered

tf.function retracing. Tracing is expensive and the excessive number of tracings could be due to

(1) creating @tf.function repeatedly in a loop, (2) passing tensors with different shapes, (3)

passing Python objects instead of tensors. For (1), please define your @tf.function outside of the

loop. For (2), @tf.function has reduce\_retracing=True option that can avoid unnecessary

retracing. For (3), please refer to

<https://www.tensorflow.org/guide/function#controlling_retracing> and

<https://www.tensorflow.org/api_docs/python/tf/function> for more details.

1/1 [==============================] - 0s 113ms/step

Emotion detected as happy (col 1). Enter 'text' to use text input instead (0, 1 or 2). Otherwise,

press Enter to continue.

Player O took position (1, 1).

| |X|X|

| |O|X|

| |O|O|

Turn: X

Player X took position (1, 0).

| |X|X|

|X|O|X|

| |O|O|

Turn: O

reference:

row 0 is neutral.

row 1 is happy.

row 2 is surprise.

1/1 [==============================] - 0s 108ms/step

Emotion detected as surprise (row 2). Enter 'text' to use text input instead (0, 1 or 2). Otherwise,

press Enter to continue.

reference:

col 0 is neutral.

col 1 is happy.

col 2 is surprise.

1/1 [==============================] - 0s 111ms/step

Emotion detected as neutral (col 0). Enter 'text' to use text input instead (0, 1 or 2). Otherwise,

press Enter to continue.

Player O took position (2, 0).

| |X|X|

|X|O|X|

|O|O|O|

Player O has won!

The interface actually worked alright. For the most part, it recognized the facial expressions pretty well. It didn’t recognize facial expressions with the same accuracy. A possibly reason would be because overfitting is occurring.

def \_get\_emotion(self, img) -> int:

# Your code goes here

#

# img an np array of size NxN (square), each pixel is a value between 0 to 255

# you have to resize this to image\_size before sending to your model

# to show the image here, you can use:

# import matplotlib.pyplot as plt

# plt.imshow(img, cmap='gray', vmin=0, vmax=255)

# plt.show()

#

# You have to use your saved model, use resized img as input, and get one classification value out of it

# The classification value should be 0, 1, or 2 for neutral, happy or surprise respectively

my\_model = models.load\_model('results/basic\_model\_20\_epochs\_timestamp\_1708644495.keras')

resized\_img = np.resize(img, (image\_size[0], image\_size[1], 3)).reshape(150, 150, 3)

resized\_img = np.expand\_dims(resized\_img, axis=0)

result = my\_model.predict(resized\_img)

# return an integer (0, 1 or 2), otherwise the code will throw an error

print(result)

return int(np.argmax(result[0]))

A graph with red and blue lines

Description automatically generated