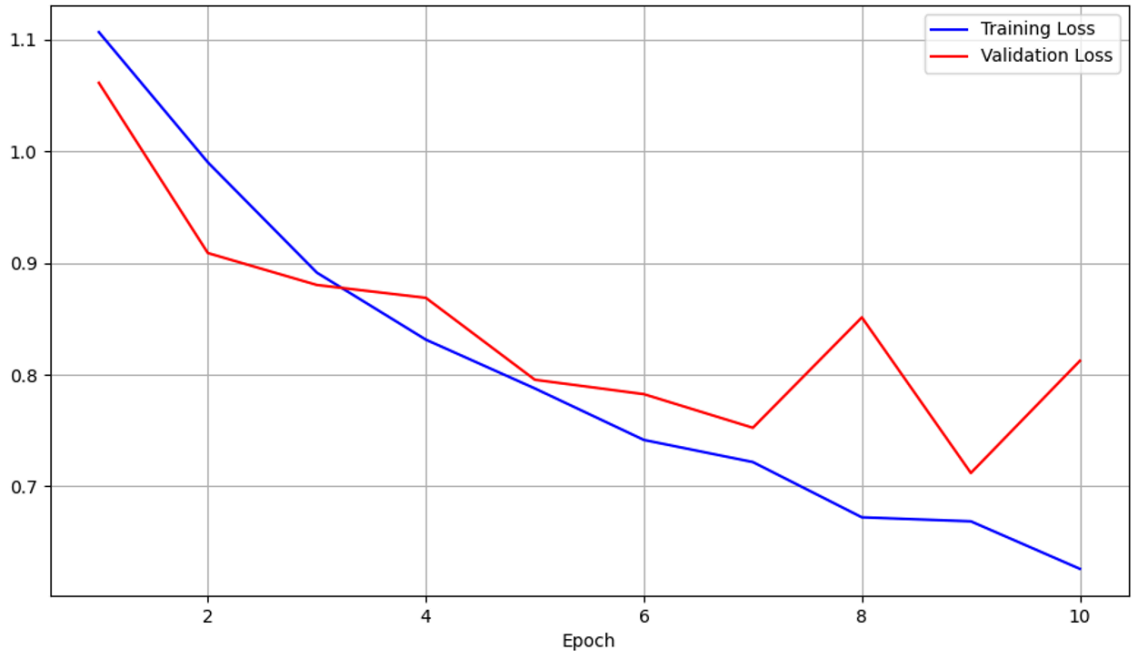
5. (Train your network)
Printout showing the shape of the network using model.print_summary():

Model: "sequential"

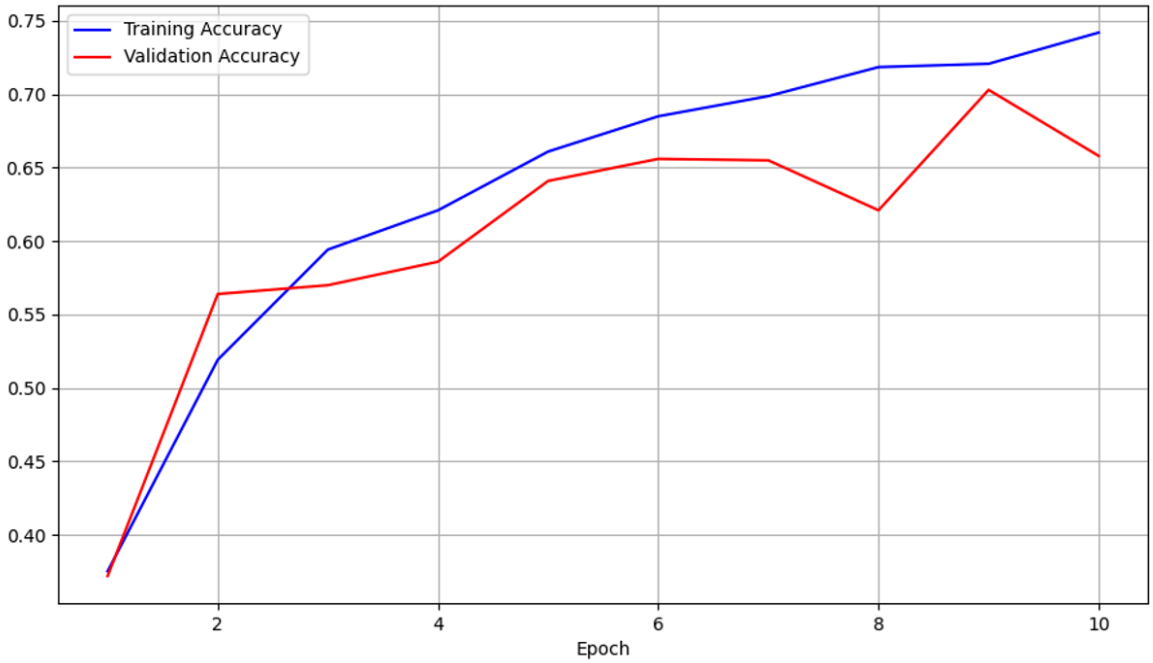
Layer (type)	Output Shape	Param #
rescaling (Rescaling)	(None, 150, 150, 3)	0
conv2d (Conv2D)	(None, 148, 148, 8)	224
max_pooling2d (MaxPooling2D)	(None, 74, 74, 8)	0
conv2d_1 (Conv2D)	(None, 72, 72, 16)	1168
max_pooling2d_1 (MaxPooling2D)	(None, 36, 36, 16)	0
conv2d_2 (Conv2D)	(None, 34, 34, 32)	4640
max_pooling2d_2 (MaxPooling2D)	(None, 17, 17, 32)	0
max_pooling2d_3 (MaxPooling2D)	(None, 8, 8, 32)	0
flatten (Flatten)	(None, 2048)	0
dense (Dense)	(None, 32)	65568
dense_1 (Dense)	(None, 3)	99

Total params: 71,699
Trainable params: 71,699
Non-trainable params: 0

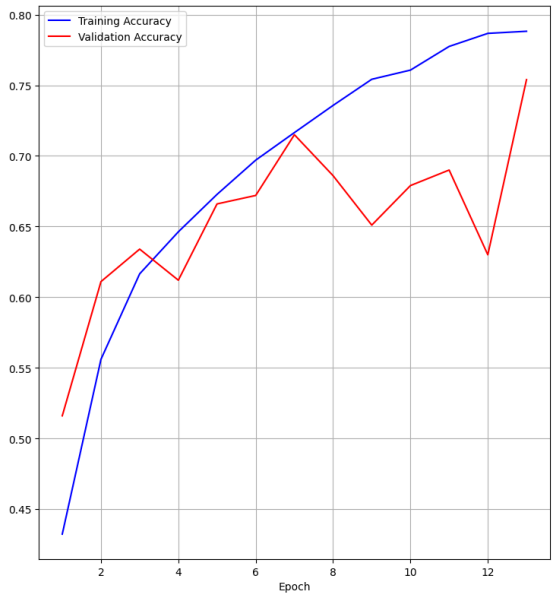
Plot showing training and validation loss as function of epoch:

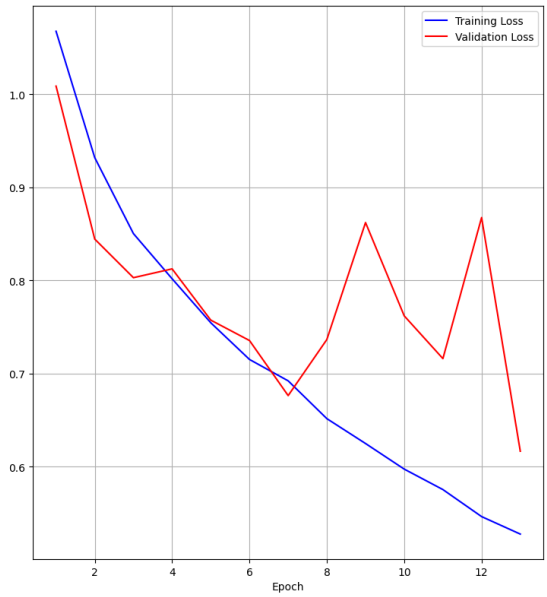


Plot showing accuracy against the training and validation:



Accuracy and loss of best learned model (13 epochs):





Best .keras filed saved and will submit separately

6. Employ Hyper-parameter optimization

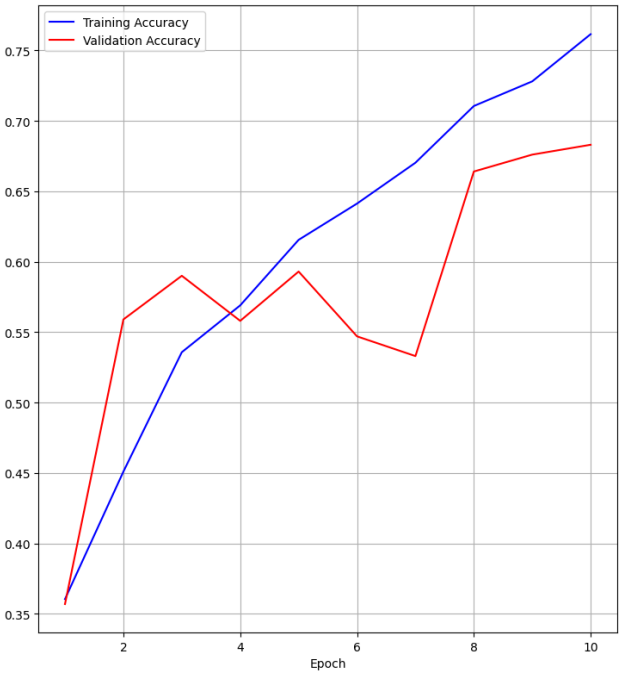
In the hyperparameter optimization process, we experimented with increasing the number of filters in the convolutional layers from 8, 16, and 32 to 32, 64, and 128. Additionally, a dropout layer with a rate of 0.5 was added to reduce overfitting and increased the size of the dense layer from 32 to 128 neurons. The adjustments led to improved model accuracy (up to 72%) and reduced overfitting, as observed through more stable validation accuracy during training.

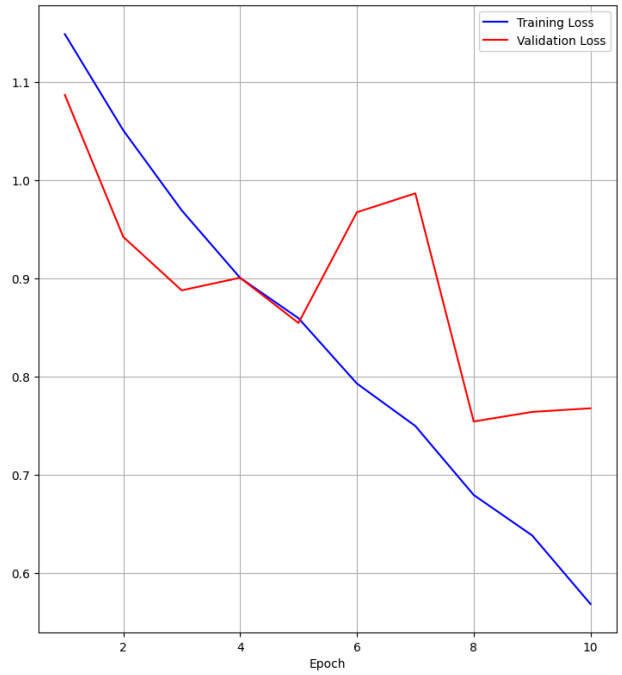
Model: "sequential"

Layer (type)	Output Shape	Param #
rescaling (Rescaling)	(None, 150, 150, 3)	0
conv2d (Conv2D)	(None, 148, 148, 32)	896
max_pooling2d (MaxPooling2D)	(None, 74, 74, 32)	0
conv2d_1 (Conv2D)	(None, 72, 72, 64)	18496
max_pooling2d_1 (MaxPooling2D)	(None, 36, 36, 64)	0

conv2d_2 (Conv2D)	(None, 34, 34, 128)	73856
max_pooling2d_2 (MaxPooling 2D)	(None, 17, 17, 128)	0
dropout (Dropout)	(None, 17, 17, 128)	0
flatten (Flatten)	(None, 36992)	0
dense (Dense)	(None, 128)	4735104
dense_1 (Dense)	(None, 3)	387

Plots Showing Training and Validation Loss as Function of Epoch, and Accuracy Against the Training and Validation of Best Learned Model:





7. Tic-tac-toe Moves:

Tic-Tac-Toe move trace:
| | | |

| | | |
| | | |
Turn: X
Player X took position (1, 0).
X		
Turn: O
reference:
row 0 is neutral.
row 1 is happy.
row 2 is surprise.
1/1 [==============================] - 0s 115ms/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.
1/1 [==============================] - 0s 128ms/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	O	
Turn: X
Player X took position (0, 1).
	X	
X	O	
Turn: O
reference:
row 0 is neutral.
row 1 is happy.
row 2 is surprise.

1/1 [==============================] - 0s 67ms/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.
1/1 [==============================] - 0s 66ms/step
Emotion detected as happy (col 1). Enter 'text' to use text input instead (0, 1 or 2). Otherwise, press Enter to continue.

Position (1, 1) is already taken.
	X	
X	O	
Turn: O
reference:
row 0 is neutral.
row 1 is happy.
row 2 is surprise.

1/1 [==============================] - 0s 159ms/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.

1/1 [==============================] - 0s 112ms/step
Emotion detected as happy (col 1). Enter 'text' to use text input instead (0, 1 or 2). Otherwise, press Enter to continue.

Position (1, 1) is already taken.
	X	
X	O	
Turn: O
reference:
row 0 is neutral.
row 1 is happy.
row 2 is surprise.
1/1 [==============================] - 0s 65ms/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.
1/1 [==============================] - 0s 135ms/step
Emotion detected as happy (col 1). Enter 'text' to use text input instead (0, 1 or 2). Otherwise, press Enter to continue.

Position (1, 1) is already taken.
	X	
X	O	
Turn: O
reference:
row 0 is neutral.
row 1 is happy.
row 2 is surprise.
1/1 [==============================] - 0s 70ms/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.
1/1 [==============================] - 0s 97ms/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 (1, 2).
	X	
X	O	O
Turn: X
Player X took position (2, 2).
	X	
X	O	O
		X
Turn: O
reference:
row 0 is neutral.
row 1 is happy.
row 2 is surprise.
1/1 [==============================] - 0s 73ms/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.
1/1 [==============================] - 0s 76ms/step
Emotion detected as happy (col 1). Enter 'text' to use text input instead (0, 1 or 2). Otherwise, press Enter to continue.
Position (1, 1) is already taken.
	X	
X	O	O
		X
Turn: O
reference:
row 0 is neutral.
row 1 is happy.
row 2 is surprise.
1/1 [==============================] - 0s 155ms/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.
1/1 [==============================] - 0s 98ms/step
Emotion detected as happy (col 1). Enter 'text' to use text input instead (0, 1 or 2). Otherwise, press Enter to continue.

Position (1, 1) is already taken.
	X	
X	O	O
		X
Turn: O
reference:
row 0 is neutral.
row 1 is happy.
row 2 is surprise.
1/1 [==============================] - 0s 138ms/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.
1/1 [==============================] - 0s 133ms/step
Emotion detected as surprise (col 2). Enter 'text' to use text input instead (0, 1 or 2). Otherwise, press Enter to continue.

Position (1, 2) is already taken.
	X	
X	O	O
		X
Turn: O
reference:
row 0 is neutral.
row 1 is happy.
row 2 is surprise.
1/1 [==============================] - 0s 216ms/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 122ms/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
	O	X
Turn: X
Player X took position (2, 0).
	X	
X	O	O
X	O	X
Turn: O
reference:
row 0 is neutral.
row 1 is happy.
row 2 is surprise.
1/1 [==============================] - 0s 131ms/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.
1/1 [==============================] - 0s 130ms/step
Emotion detected as happy (col 1). Enter 'text' to use text input instead (0, 1 or 2). Otherwise, press Enter to continue.

Position (1, 1) is already taken.
	X	
X	O	O
X	O	X
Turn: O
reference:
row 0 is neutral.
row 1 is happy.
row 2 is surprise.
1/1 [==============================] - 0s 66ms/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.
1/1 [==============================] - 0s 136ms/step
Emotion detected as happy (col 1). Enter 'text' to use text input instead (0, 1 or 2). Otherwise, press Enter to continue.

Position (1, 1) is already taken.
	X	
X	O	O
X	O	X
Turn: O
reference:
row 0 is neutral.
row 1 is happy.
row 2 is surprise.
1/1 [==============================] - 0s 122ms/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 153ms/step
Emotion detected as neutral (col 0). Enter 'text' to use text input instead (0, 1 or 2). Otherwise, press Enter to continue.

Position (2, 0) is already taken.
	X	
X	O	O
X	O	X
Turn: O
reference:
row 0 is neutral.
row 1 is happy.
row 2 is surprise.
1/1 [==============================] - 0s 86ms/step
Emotion detected as neutral (row 0). 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 62ms/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 (0, 0).
O	X	
X	O	O
X	O	X
Turn: X
Player X took position (0, 2).
O	X	X
X	O	O
X	O	X
The game has ended in draw!

Game Model Questions:

1. **How well did the interface work? (and why it didn't for Mylo specifically)**
   For me personally (Mylo) the interface worked perfectly fine for happy and surprised, but

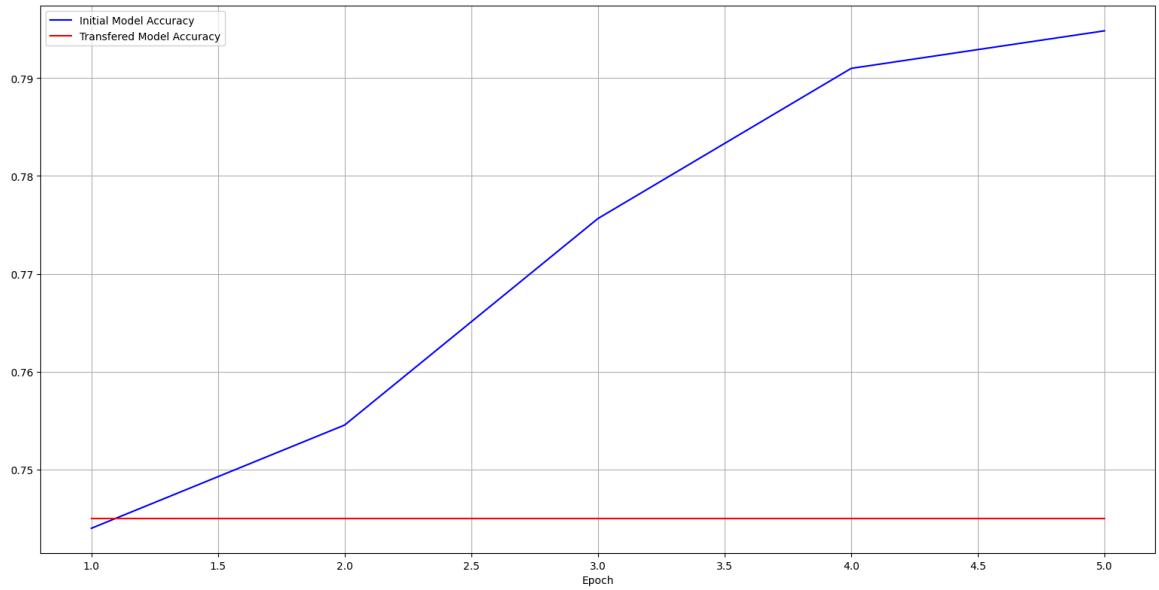
it never recognized my face as a neutral face. If I tried to make a neutral face it 100% got recognized as happy. However when I had my parents try it it got their neutral face every time they did one (as can be seen in the game moves by many happy faces in a row followed finally by a few neutral faces in a row.). My project partner and family and I theorize that it is because my nose is very crooked from breaking it many times, and that this could make the model not recognize my face as neutral due to this irregularity.

2. **Did it predict it as well in the model?**
   Excluding the problems Mylo had with having the neutral face recognized, it actually predicted it better than the model did.

Get Emotion Code:

```
        resized_img = cv2.resize(img, (150, 150))

        if len(resized_img.shape) == 2:
            resized_img = cv2.cvtColor(resized_img, cv2.COLOR_GRAY2RGB)

        model_path = "./results/basic_model_10_70_percent.keras"
        loaded_model = models.load_model(model_path)
        prediction = loaded_model.predict(np.expand_dims(resized_img,
axis=0))

        emotion_class = np.argmax(prediction)

        return int(emotion_class)
```

8. Transfer Learning:



*labels flipped

Epoch 1/5
33/33 [==============================] - 27s 659ms/step - loss: 0.6117 - accuracy: 0.7108 - val_loss: 0.5478 - val_accuracy: 0.7440
Epoch 2/5
33/33 [==============================] - 25s 635ms/step - loss: 0.5334 - accuracy: 0.7613 - val_loss: 0.5018 - val_accuracy: 0.7546
Epoch 3/5
33/33 [==============================] - 25s 634ms/step - loss: 0.4801 - accuracy: 0.7860 - val_loss: 0.4672 - val_accuracy: 0.7756
Epoch 4/5
33/33 [==============================] - 23s 589ms/step - loss: 0.4489 - accuracy: 0.7977 - val_loss: 0.4436 - val_accuracy: 0.7910
Epoch 5/5
33/33 [==============================] - 23s 607ms/step - loss: 0.4309 - accuracy: 0.8066 - val_loss: 0.4282 - val_accuracy: 0.7948

Random model:
Epoch 1/5
33/33 [==============================] - 25s 620ms/step - loss: 0.6073 - accuracy: 0.7239 - val_loss: 0.5846 - val_accuracy: 0.7450
Epoch 2/5
33/33 [==============================] - 23s 598ms/step - loss: 0.5822 - accuracy: 0.7424 - val_loss: 0.5767 - val_accuracy: 0.7450
Epoch 3/5

33/33 [==============================] - 25s 665ms/step - loss: 0.5762 - accuracy: 0.7424 - val_loss: 0.5697 - val_accuracy: 0.7450
Epoch 4/5
33/33 [==============================] - 31s 795ms/step - loss: 0.5696 - accuracy: 0.7424 - val_loss: 0.5631 - val_accuracy: 0.7450
Epoch 5/5
33/33 [==============================] - 29s 723ms/step - loss: 0.5660 - accuracy: 0.7424 - val_loss: 0.5572 - val_accuracy: 0.7450

* Evaluating random_model
5/5 [==============================] - 2s 366ms/step - loss: 0.6781 - accuracy: 0.6250

Summaries of both models:
The random model serves as a baseline for comparison against the transfer learning model. The random model is identical to the original model, but with randomized weights, allowing us to assess the impact of using pre-trained weights. On the other hand, the transfer model builds upon the original model by removing its final layer and adding a new fully connected layer tailored to the new classification task, which has two output categories instead of the original three. This approach leverages the pre-trained weights from the earlier layers, enabling the model to fine-tune its performance on the new task with less data and training time.

Facial Recognition Classes:

1. **Happy** 
2. **Surprised** 
3. **Neutral** 


Normal
Bacterial Pneumonia
Viral Pneumonia