Task 1:

Describe or draw the two CNN architectures you used for Task 1.

Model 1

Layer (type)	Output Shape	Param #
rescaling_2 (Rescaling)	(None, 100, 100, 3)	0
conv2d_4 (Conv2D)	(None, 98, 98, 32)	896
max_pooling2d_4 (MaxPooling2D)	(None, 49, 49, 32)	0
conv2d_5 (Conv2D)	(None, 47, 47, 64)	18496
max_pooling2d_5 (MaxPooling2D)	(None, 23, 23, 64)	0
flatten_2 (Flatten)	(None, 33856)	0
dense_4 (Dense)	(None, 128)	4333696
dropout_2 (Dropout)	(None, 128)	0
dense_5 (Dense)	(None, 10)	1290

Model 2

Layer (type)	Output Shape	Param #
rescaling_1 (Rescaling)	(None, 100, 100, 3)	0
conv2d_2 (Conv2D)	(None, 98, 98, 64)	1792
max_pooling2d_2 (MaxPooling2D)	(None, 49, 49, 64)	0
conv2d_3 (Conv2D)	(None, 47, 47, 128)	73856
max_pooling2d_3 (MaxPooling2D)	(None, 23, 23, 128)	0
flatten_1 (Flatten)	(None, 67712)	0
dense_2 (Dense)	(None, 256)	17334528
dropout_1 (Dropout)	(None, 256)	0
dense_3 (Dense)	(None, 10)	2570

Model 1 and 2 differ with the nature of the convolutional layers and the number of parameters utilized.

(ii) For each model, show in a table how training accuracy changed over epochs.

Training Accuracy	Model 1	Model 2
Epoch 1	0.4082757234573364	0.37150585651397705
Epoch 2	0.6193768382072449	0.5867147445678711
Epoch 3	0.7308886647224426	0.6977256536483765
Epoch 4	0.8166516423225403	0.7717663645744324
Epoch 5	0.8762649297714233	0.8253682255744934
Epoch 6	0.9162408709526062	0.8656447529792786
Epoch 7	0.9419897794723511	0.8921951651573181
Epoch 8	0.9544134140014648	0.9113315343856812
Epoch 9	0.9627291560173035	0.9223524928092957
Epoch 10	0.9679390788078308	0.9283639192581177

(iii) Show test accuracy for each model in a table.

Model	Test Accuracy
Model 1	0.6992126107215881
Model 2	0.7251968383789062

(iv) Show the confusion matrix of the more accurate model on the test set.

```
Confusion Matrix:
[[ 51   0   0   0   0   0   0   1   0   0]
 [  0 119   2   5   5   3   1   5   4   2]
 [  0   1  18   1   3   1   4   5   2   0]
 [  0   1   1  40   4   3   0   0  20   4]
 [  0   1   0   2  25   1   2   2   1   0]
 [  0   4   0   1   7  72   2   3   2   0]
 [  1   1   3   1   3   1  94   7   1   0]
 [  4   4   9   2   8   5   3 120   2   1]
 [  0   2   2  20  26   7   3   0 100   1]
 [  2  39   2  15  33  27   3  31  20 236]]
```

(v) Write some comments on the results comparing the two models, and some comments on the confusion matrix.

It is evident from both models' steady improvements in training accuracy across epochs that they are effectively learning from the training set. Model 2 shows a superior performance compared to Model 1, both in terms of training accuracy and test accuracy, indicating that it might be better able to recognize the underlying patterns and also generalizes better to unseen data.

The confusion matrix shows that, as indicated by the large counts along the diagonal, Model 1 performs rather well across several classes. However, as the off-diagonal elements show, there are still some misclassifications.

Task 2:

Mention which pre-trained model you used and what is its size, and what layer(s) you added on top of it.

I first used ResNet50V2 but it had a low accuracy (0.34) so I opted for EfficientNetV2S. It is 88MB with the minimum size as 128 pixels and the maximum size being 300 pixels. On top of the EfficientNetV2S base model, I have added the following custom layers for classification:

a. GlobalAveragePooling2D: This layer reduces the spatial dimensions of the feature maps from the previous layer by averaging.
b. Dense(256, activation='relu'): This fully connected layer consists of 256 units with ReLU activation, which introduces non-linearity into the network.
c. Dense(10, activation='softmax'): This is the output layer with 10 units, and utilizing softmax activation.

Show in a table how training accuracy changed over epochs.

Epochs	Training accuracy
1	0.8261697292327881
2	0.8843803405761719
3	0.9102294445037842
4	0.9240556955337524
5	0.9400861859321594
6	0.9467989206314087
7	0.9552149176597595
8	0.9609257578849792
9	0.9616270661354065
10	0.9672377705574036

Show the test accuracy of the fine-tuned model and the better model of Task 1 in a table.

Model	Test Accuracy
better model	0.7251968383789062
fine-tuned model	0.8992125988006592

Show the confusion matrix of the fine-tuned model.

```
Confusion Matrix:
[[ 52   0   0   0   0   0   0   0   0   0]
 [  0 135   3   3   0   1   0   2   0   2]
 [  1   0  28   1   1   0   1   2   0   1]
 [  0   1   2  59   0   1   0   0   7   3]
 [  0   0   0   0  31   3   0   0   0   0]
 [  0   0   0   0   0  91   0   0   0   0]
 [  1   0   0   0   1   0 108   2   0   0]
 [  4   1   1   3   1   4   0 143   1   0]
 [  1   3   4  17   6   5   2   2 121   0]
 [  1   6   3   7   2   6   1   4   4 374]]
```

(v) Write some comments on the results comparing the fine-tuned model and the better model of Task 1, and some comments on the confusion matrix of the fine-tuned model.

The fine-tuned model achieves a significantly higher test accuracy (89.9%) compared to the better model from Task 1 (72.5%). The increase in test accuracy shows that, the fine-tuned model has either learnt more discriminative features or has more effectively adapted to the target task.

Compared to the better model, the fine-tuned model shows high counts along the diagonal, suggesting successful classification for the majority of classes. Also, there are few misclassifications recorded.

Task 3:

Include the 10 images in the report along with their correct classes, predicted classes by the better model of Task 1, and predicted classes by the fine-tuned model.

Predicted classes by the better model of Task 1: 7

Predicted classes by the fine-tuned model: 0

Correct class: 0

Predicted classes by the better model of Task 1: 4

Predicted classes by the fine-tuned model: 1

Correct class: 1

Predicted classes by the better model of Task 1: 3

Predicted classes by the fine-tuned model: 3

Correct class: 1

Predicted classes by the better model of Task 1: 3

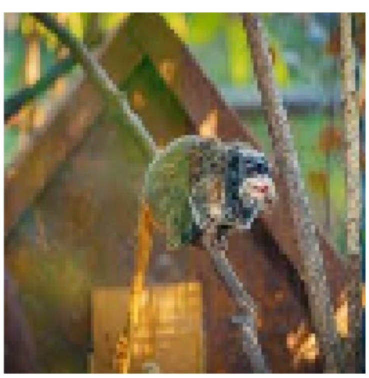
Predicted classes by the fine-tuned model: 1

Correct class: 1

Predicted classes by the better model of Task 1: 8

Predicted classes by the fine-tuned model: 1

Correct class: 1



Predicted classes by the better model of Task 1: 8

Predicted classes by the fine-tuned model: 9

Correct class: 1

Predicted classes by the better model of Task 1: 6

Predicted classes by the fine-tuned model: 7

Correct class: 1

Predicted classes by the better model of Task 1: 8

Predicted classes by the fine-tuned model: 1

Correct class: 1

Predicted classes by the better model of Task 1: 7

Predicted classes by the fine-tuned model: 1

Correct class: 1

Predicted classes by the better model of Task 1: 3

Predicted classes by the fine-tuned model: 1

Correct class: 1

Give possible qualitative reasons why the better model of Task 1 may be making those mistakes, and some qualitative reasons why the fine-tuned model may or may not be improving.

Qualitative reasons for mistakes by the better model

1. Image complexity – The better model may struggle with complex images which are either too noisy for it so it is not able to capture the features correctly.
2. Lack of robustness – The better model could misclassify because it is less robust to variations in background, lighting, or object orientation.
3. Training data – It's likely that the better model did not receive enough varied training examples to make a good generalization to all possible test image variations.

Qualitative reasons why the fine-tuned model may or may not be improving

1. Training data – the fine-tuned model is exposed to a large variety of image variations while training and that is why it is better at capturing more intricate patterns in the data leading to better performance
2. Model complexity – since the fine-tuned model is more complex and has additional layers, it performs better. It is able to capture more features that was not possible for the better model.