INTRODUCTION TO MACHINE LEARNING(COMPSCI 711) ASSIGNMENT 4 REPORT BY AMITH GEORGE DATED 05/06/2024

Task 1

Model 1

1. Conv2D Layer: The first layer is a 2D convolutional layer with 32 filters of size 3*3. It uses 'relu' (rectified linear unit) as the activation function. This layer is designed to perform feature extraction by sliding the filter across the input image.
2. MaxPooling2D Layer: This layer follows the convolutional layer and performs a max pooling operation with a pool size of 2*2. It reduces the spatial dimensions (height and width) of the input volume for the next convolutional layer.
3. Conv2D Layer: Another 2D convolutional layer with 64 filters of size 3*3, also using 'relu' as the activation function. This layer further abstracts the features extracted by the previous layers.
4. MaxPooling2D Layer: This max pooling layer also has a pool size of 2*2 , further reducing the dimensionality of the data.
5. Conv2D Layer: A third convolutional layer with 64 filters of size 3*3 and 'relu' activation.
6. Flatten Layer: This layer flattens the multi-dimensional input into a single dimension, preparing it for the fully connected layer.
7. Dense Layer: A fully connected layer that has 64 units and uses 'relu' activation function.
8. Dense Layer: The final layer is a fully connected layer with 10 units (corresponding to the number of classes) and uses 'softmax' activation for multi-class classification.

Model 2

1. Conv2D Layer: Similar to Model 1, it starts with a convolutional layer with 32 filters of 3*3 using 'relu' activation.
2. MaxPooling2D Layer: This layer performs max pooling with a pool size of 2*2.
3. Conv2D Layer: A convolutional layer with 64 filters of size 3*3 and 'relu' activation.
4. MaxPooling2D Layer: Follows with another 2*2 max pooling layer.
5. Conv2D Layer: Another convolutional layer but with 128 filters of 3*3 and 'relu' activation. It further increases the depth of the network to capture more complex features.
6. MaxPooling2D Layer: This max pooling layer also has a pool size of 2*2.
7. Flatten Layer: Flattens the output from the previous layers into a single-dimensional vector.
8. Dense Layer: A dense layer with 128 units and 'relu' activation.

9. Dropout Layer: A dropout layer is introduced with a dropout rate of 0.5 to prevent overfitting by randomly setting a fraction of input units to 0 at each update during training.
10. Dense Layer: The final dense layer has 10 units and uses 'softmax' activation function.

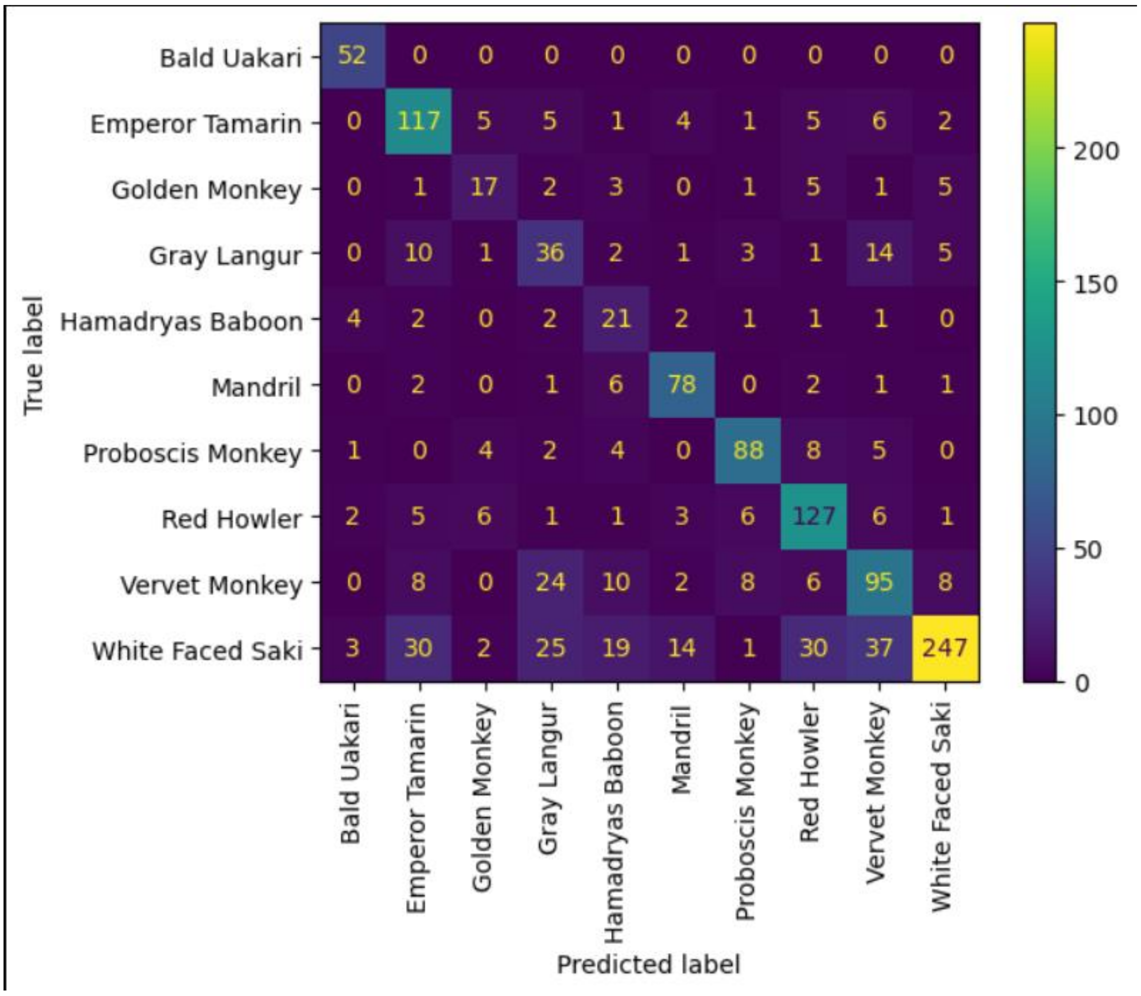
Training Accuracy Table Over Epochs

Epoch	Model 1 Accuracy	Model 2 Accuracy
1	0.148282	0.255886
2	0.234946	0.372608
3	0.357579	0.456968
4	0.503356	0.527903
5	0.626390	0.586715
6	0.731490	0.635107
7	0.783489	0.670674
8	0.806833	0.698828
9	0.856026	0.726480
10	0.893898	0.738002
11	0.907624	0.763551
12	0.902916	0.790302
13	0.907124	0.796513
14	0.916942	0.790903
15	0.933273	0.818956
16	0.927963	0.826871
17	0.941489	0.829977
18	0.946599	0.841799
19	0.932973	0.834686
20	0.940487	0.851217

Testing Accuracy Table

Model	Testing Accuracy
1	0.531496
2	0.691339

Confusion Matrix of Best Model (Model 2)



True label \ Predicted label	Bald Uakari	Emperor Tamarin	Golden Monkey	Gray Langur	Hamadryas Baboon	Mandril	Proboscis Monkey	Red Howler	Vervet Monkey	White Faced Saki
Bald Uakari	52	0	0	0	0	0	0	0	0	0
Emperor Tamarin	0	117	5	5	1	4	1	5	6	2
Golden Monkey	0	1	17	2	3	0	1	5	1	5
Gray Langur	0	10	1	36	2	1	3	1	14	5
Hamadryas Baboon	4	2	0	2	21	2	1	1	1	0
Mandril	0	2	0	1	6	78	0	2	1	1
Proboscis Monkey	1	0	4	2	4	0	88	8	5	0
Red Howler	2	5	6	1	1	3	6	127	6	1
Vervet Monkey	0	8	0	24	10	2	8	6	95	8
White Faced Saki	3	30	2	25	19	14	1	30	37	247

Training Accuracy :

Model 1's improvement over epochs is significant, starting from 14.8% and going up to about 94.0%. This shows a steady and robust learning process, although it starts at a lower baseline compared to Model 2.

Model 2 starts at a higher accuracy of 25.6% and reaches up to 85.1% by the 20th epoch. Not only does it start off higher, but it also shows a more rapid initial improvement, which indicates that its architecture might be capturing the features more effectively or it may have a better configuration for this specific task.

We can infer at the 20th epoch when comparing the training accuracy of both models, Model 1 has greater training accuracy compared to Model 2.

Testing Accuracy :

When it comes to the testing accuracy, Model 2 outperforms Model 1 as Model 2 has a higher test accuracy of 69% and Model 1 has only 53% test accuracy.

This suggests that Model 2 is not only learning faster but also generalizing better to new, unseen data from the test set.

Confusion Matrix Analysis

Strong Performance on Some Classes:

- Bald Uakari: This class shows excellent accuracy with all predictions correctly classified (52/52).
- White Faced Saki: This class also performs very well with most predictions correctly classified (247 out of several predictions).

Problems with Misclassification:

- Emperor Tamarin: This class has significant misclassifications, with only 117 correctly classified out of many predictions. Many of these are misclassified as Golden Monkey, Gray Langur, and White Faced Saki.
- White Faced Saki often misclassifies other classes as itself, indicating potential bias or overfitting to this class due to unique or dominating features.

Moderate to Low Accuracy for Certain Classes:

- Proboscis Monkey: This class is often confused with other classes like the Golden Monkey, Gray Langur, and White Faced Saki.
- Red Howler and Vervet Monkey: There is considerable confusion between these two classes, possibly due to similar visual features that the model fails to distinguish properly.

Task 2

Fine-tuned model : VGG 16

This model constructs a classification system utilizing the VGG16 pre-trained model as its foundation. To adapt the pre-trained model for a new task, the last layer, which is responsible for categorizing images into 1,000 classes, is eliminated. Instead, additional layers are incorporated to create a new classifier capable of categorizing images into 5 distinct classes. The model's size is 528 MB.

Initially, a global average pooling layer is appended, computing the average of each feature map within the output of VGG16's final convolutional layer. Subsequently, this pooling layer's output is fed into a dense layer comprising 1024 neurons, employing the ReLU activation function. Finally, a dense layer consisting of 5 neurons and employing the softmax activation function is appended as the output layer.

The subsequent step involves setting all layers within the pre-trained VGG16 model as non-trainable, preventing their modification during the training phase. Subsequently, the model undergoes compilation utilizing categorical cross-entropy loss and accuracy metrics.

For training, the model undergoes 14 epochs on a training dataset with a batch size of 32. During each epoch, the training accuracy is recorded. Additionally, the model's performance is evaluated on a distinct test set, with the resulting testing accuracy also printed.

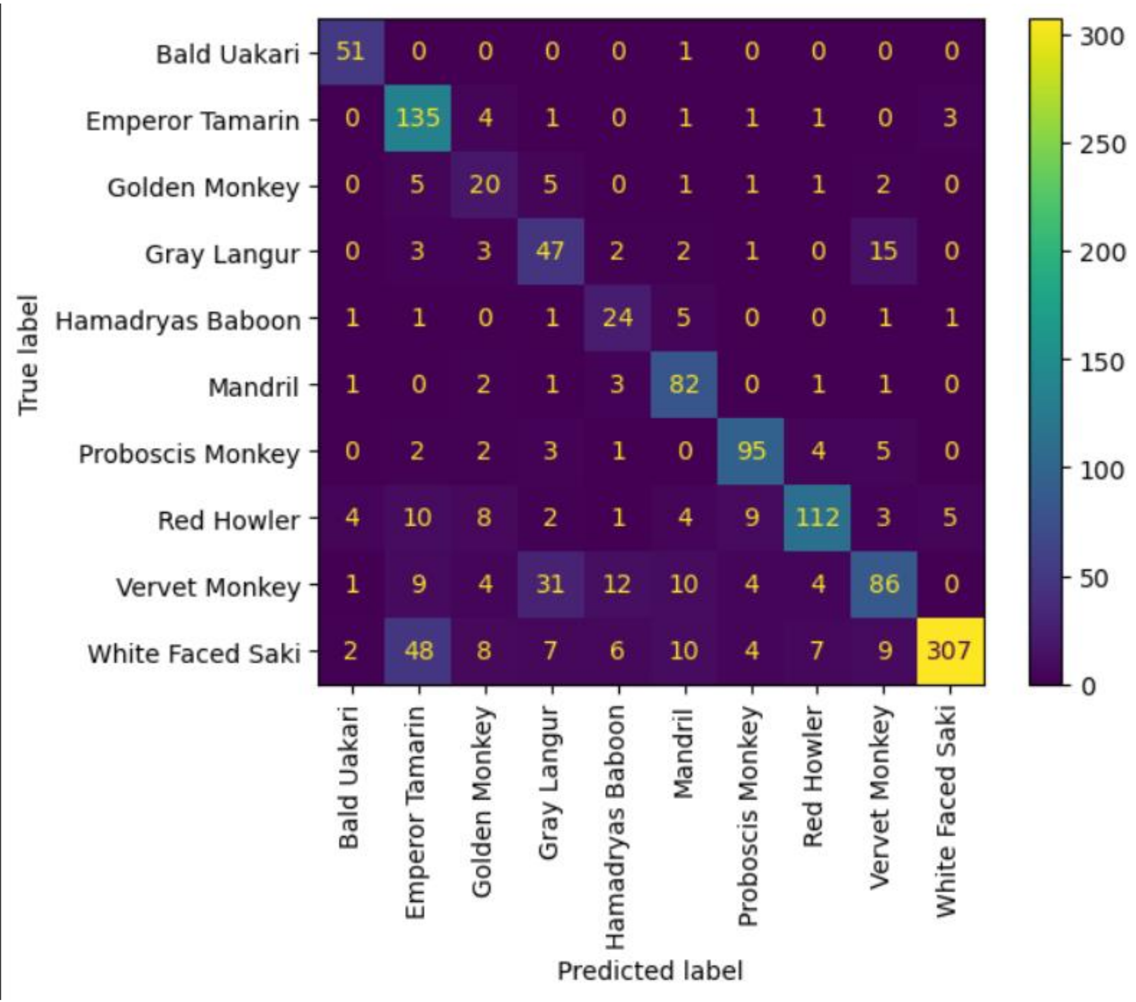
Training Accuracy Table Over Epochs

Epoch	Fine-tuned Model Accuracy
1	0.597135
2	0.788498
3	0.858732
4	0.899609
5	0.926260
6	0.936680
7	0.944495
8	0.954514
9	0.958120
10	0.961126
11	0.961327
12	0.965134
13	0.970344
14	0.968841

<u>Testing Accuracy Table</u>

Model	Testing Accuracy
2 (Best Model of Task 1)	0.691339
VGG 16 (Fine-tuned Model)	0.755118

<u>Confusion Matrix of Fine-tuned Model(VGG 16)</u>



True label \ Predicted label	Bald Uakari	Emperor Tamarin	Golden Monkey	Gray Langur	Hamadryas Baboon	Mandril	Proboscis Monkey	Red Howler	Vervet Monkey	White Faced Saki
Bald Uakari	51	0	0	0	0	1	0	0	0	0
Emperor Tamarin	0	135	4	1	0	1	1	1	0	3
Golden Monkey	0	5	20	5	0	1	1	1	2	0
Gray Langur	0	3	3	47	2	2	1	0	15	0
Hamadryas Baboon	1	1	0	1	24	5	0	0	1	1
Mandril	1	0	2	1	3	82	0	1	1	0
Proboscis Monkey	0	2	2	3	1	0	95	4	5	0
Red Howler	4	10	8	2	1	4	9	112	3	5
Vervet Monkey	1	9	4	31	12	10	4	4	86	0
White Faced Saki	2	48	8	7	6	10	4	7	9	307

Training Accuracy:

The fine-tuned model shows robust improvement over epochs in training accuracy. It starts at around 59.7% and progresses impressively to about 96.8% by the 14th epoch. This demonstrates the model's effective learning and its ability to adapt well over time with fine-tuning, capturing the features necessary to distinguish between different monkey species more accurately.

Testing Accuracy:

The fine-tuned model achieves a testing accuracy of 75.5%, which is a significant improvement over the best model from Task 1, which had a testing accuracy of 69.1%.

Comparison with Task 1's Best Model:

The fine-tuned model not only improves in testing accuracy but also shows more stability and consistency in its predictions across different classes, evidenced by the denser diagonal in the confusion matrix.

The better generalization to unseen data suggests that fine-tuning has effectively leveraged the deeper, more complex features that are pivotal in differentiating between similar classes.

Confusion Matrix Analysis

Strong Performers:

- Classes like Bald Uakari, White Faced Saki, and Proboscis Monkey show very high accuracy with nearly all their instances correctly classified.

Significant Misclassifications:

- Emperor Tamarin shows considerable confusion, with 135 correct classifications but notable misclassifications with Golden Monkey and White Faced Saki.
- White Faced Saki, while predominantly classified correctly, has its instances frequently confused with Emperor Tamarin.

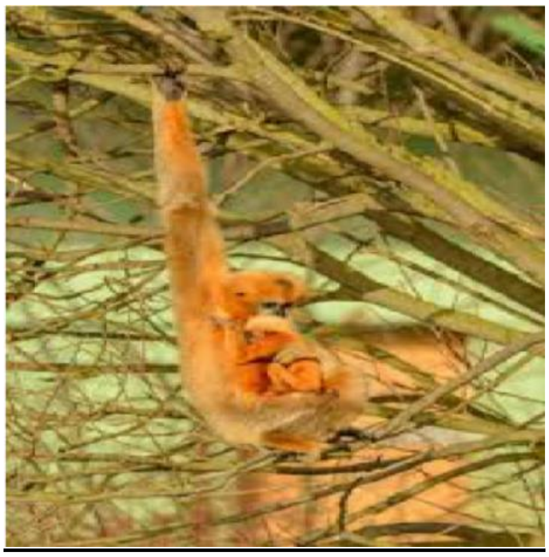
Moderate Confusions:

- Classes like Red Howler and Vervet Monkey show moderate confusion with other classes, though to a lesser extent compared to the non-fine-tuned models.

General Improvement: Compared to previous models, this fine-tuned version generally shows a reduced number of misclassifications across all classes, indicating better generalization and feature learning.

Task 3

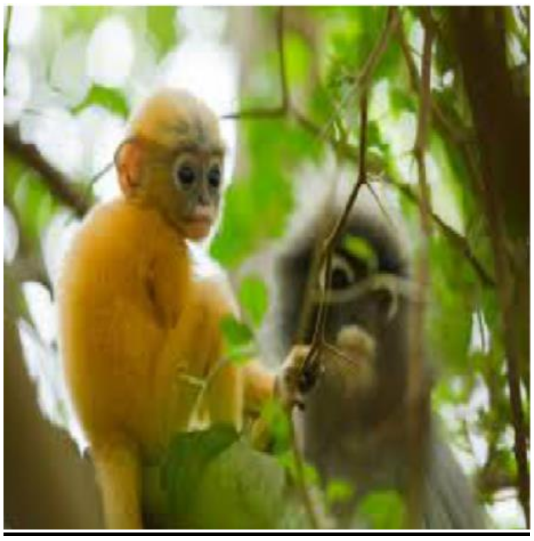
Error Analysis



Correct Class : Golden Monkey

Predicted class by the better model of Task 1 : Red Howler

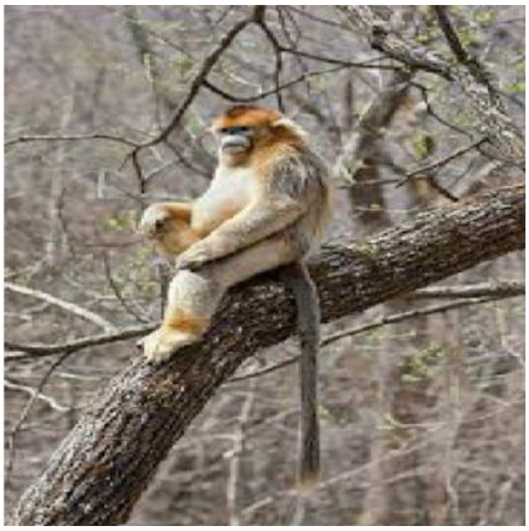
Predicted class by the fine-tuned model : Proboscis Monkey



Correct Class : Golden Monkey

Predicted class by the better model of Task 1 : Red Howler

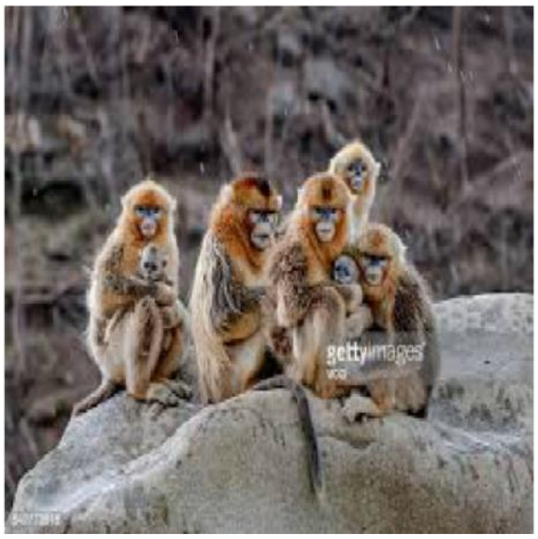
Predicted class by the fine-tuned model : Emperor Tamarin



Correct Class : Golden Monkey

Predicted class by the better model of Task 1 : Vervet Monkey

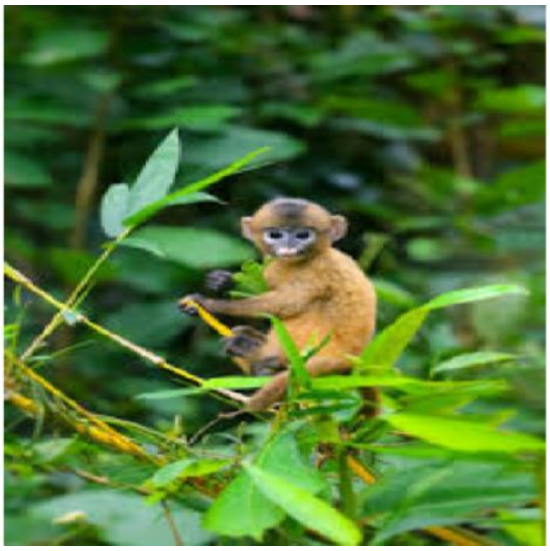
Predicted class by the fine-tuned model : Golden Monkey



Correct Class : Golden Monkey

Predicted class by the better model of Task 1 : Golden Monkey

Predicted class by the fine-tuned model : Golden Monkey



Correct Class : Golden Monkey

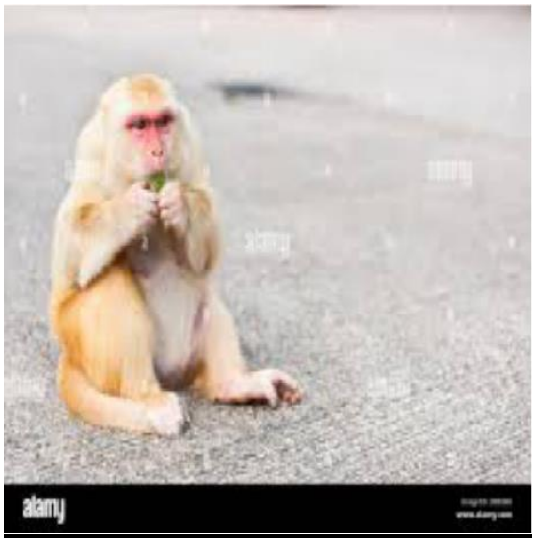
Predicted class by the better model of Task 1 : Gray Langur

Predicted class by the fine-tuned model : Golden Monkey

Correct Class : Golden Monkey

Predicted class by the better model of Task 1 : Golden Monkey

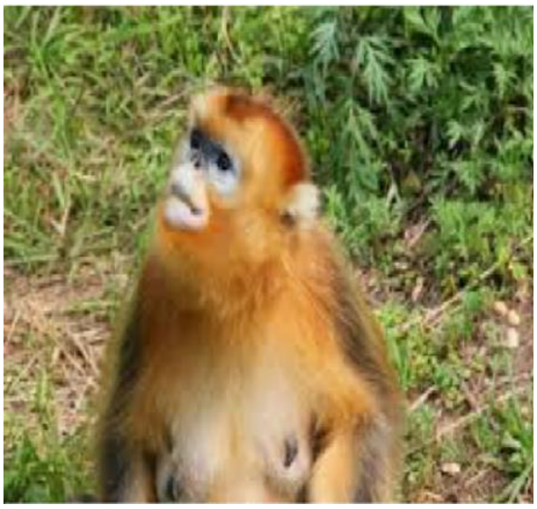
Predicted class by the fine-tuned model : Golden Monkey



Correct Class : Golden Monkey

Predicted class by the better model of Task 1 : Hamadryas Baboon

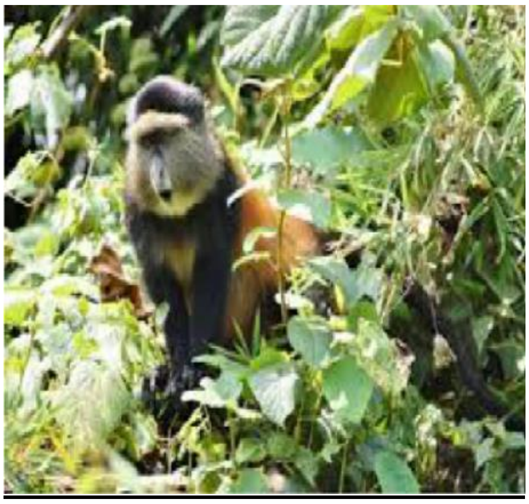
Predicted class by the fine-tuned model : Golden Monkey



Correct Class : Golden Monkey

Predicted class by the better model of Task 1 : Golden Monkey

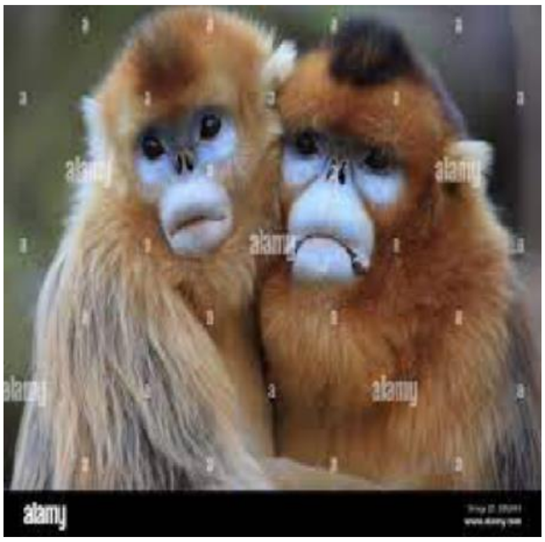
Predicted class by the fine-tuned model : Golden Monkey



Correct Class : Golden Monkey

Predicted class by the better model of Task 1 : White Faced Saki

Predicted class by the fine-tuned model : Emperor Tamarin



Correct Class : Golden Monkey

Predicted class by the better model of Task 1 : Golden Monkey

Predicted class by the fine-tuned model : Golden Monkey

Given the performance of the best model from Task 1 and the fine-tuned model on a set of images for the Golden Monkey class, where the former correctly identified 4 out of 10 and the latter correctly identified 7 out of 10, we can explore several qualitative factors that may be influencing their predictions.

Reasons for Mistakes by the Best Model of Task 1

Generalization Issues:

- Feature Sensitivity: The model might be too sensitive to specific features that aren't necessarily discriminative of the Golden Monkey class. For example, background elements or lighting conditions in the training images might be coincidentally similar and not correctly learned as irrelevant.
- Overfitting: If the model has overfitted to the training data, it may struggle to generalize to new examples that differ slightly from those it was trained on, leading to incorrect predictions.

Data Imbalance or Quality:

- If the training data did not include a diverse enough set of examples of the Golden Monkey, the model might not learn to recognize the class under different poses, lighting, or backgrounds. Insufficient examples or low-quality images can hinder the model's ability to learn robust features.

Reasons for Improved Predictions by the Fine-Tuned Model

Enhanced Feature Learning:

- Deep Learning Capabilities: Fine-tuning, especially on models like VGG-16, allows the network to adjust its deep, pre-trained convolutional features to better suit the specific dataset. This can lead to more accurate identification of the unique characteristics of the Golden Monkey.
- Refined Feature Representation: Fine-tuning typically updates the weights in the higher layers of the network, which are responsible for capturing the more specific features of the dataset's classes, potentially leading to better performance on challenging examples.

Better Generalization:

- Regularization and Dropout: If applied during fine-tuning, techniques like dropout can help prevent overfitting by making the network less sensitive to the noise in the training data, thereby improving its ability to generalize to new data.
- Augmented Training Data: Fine-tuning often involves using augmented data, which can help the model learn to ignore irrelevant variances like background and focus more on the critical features of the Golden Monkey.