East West University

LAB 07

CSE366(2) – Artificial Intelligence

[Fall 24]

Lab Report

Image Classification with Caltech-101 Dataset

Submitted by

Md. Sabbir Hossain

2020-2-60-107

Submitted to

Dr. Raihan Ul Islam

Associate Professor

Department of Computer Science and Engineering

East West University

Introduction:

This project pursues the goal of image classification using the Caltech-101 dataset where images belong to 101 categories. In this regard, the different featured deep learning approaches, including the traditional CNNs and YOLO (You Only Look Once) for the object detection tasks, are explored. The model uses three architectures from CNN: VGG19, ResNet50, and EfficientNetB0, besides YOLO, for evaluating and comparing their applicability to image classification.

The VGG19 model that is known widely for its simplicity, as well as for being deep, serves as the base benchmark. This model, consisting of nineteen layers, is one of the most used architectures in image classification tasks, and the major downside to it is that it is quite computationally expensive.

Therefore, we also employ ResNet50, which is said to be more advanced in that it has implemented residual connections. This allows deeper networks to be built without the major problem of vanishing gradients often found in normal CNNs. Thus, ResNet50 can be appropriate for more complex endeavors.

In addition to the above, we conduct experiments using EfficientNetB0 that incorporates a compound scaling method to optimize the performance of the model. By balancing the parameters depth, width, and resolution, EfficientNetB0 can attain high accuracy at fewer parameters when compared with traditional architectures of CNNs, rendering it both computationally and effective.

In addition to all those models, we factor in YOLO, which has been touted to be cutting-edge when it comes to detection of real-time objects. It provides a fast and efficient way to identify the objects in an image and finds a lot of applications in field operations that need both accuracy and speed.

Grad-CAM is one of the primary XAI strategies applied in this context to improve interpretability of the model. The way Grad-CAM visualization sheds light on the decision-making of these models is by showing the parts of the image that.

Objective:

The main aim of this project is to build, train, and evaluate several deep learning models using the Caltech-101 dataset for image classification, containing images from 101 different categories. In this regard, we consider three state-of-the-art CNNs: VGG19, ResNet50, and EfficientNetB0. Each model represents another approach to deep learning: VGG19 has a simple and effective architecture; ResNet50 uses residual connections to allow much deeper models; and EfficientNetB0 offers a better balance of model efficiency with accuracy.

Besides traditional CNN architectures, we also apply the YOLO model for real-time object detection. It shows that it is capable of fast and precise localization of objects in an image.

We further enhance the transparency and interpretability of these models by incorporating Explainable AI techniques, more specifically Grad-CAM. Grad-CAM helps to highlight regions of an image that are influencing model predictions, thus facilitating the interpretation and explainability of models' decision-making processes. This combination of deep learning models with XAI techniques enables both high performance in the results and further insight into the inner workings of models.

Training and validation accuracy/loss:

The model was trained for 10 epochs. During training, the training accuracy increased with a steady rise, showing that at each step, the model learned to classify images more and more accurately. Correspondingly, the training loss decreased over time, which means that the model fitted better to the training data while minimizing the error between its predictions and the true labels.

The validation accuracy followed the same trend as the training accuracy, with some plateauing after 10 epochs. This might be because the model has reached its capacity to generalize on the validation data and it showed signs of not improving further. The loss of validation also showed a similar trend. It decreased initially and, however, sometimes had small oscillations that could be due to the overfitting of the model or variances in the validation data. Such oscillations mean that while the model learned, its generalization to the validation set was not perfect after some number of epochs.

Insights:

It would appear that the model generalizes well, as can be seen by the increasing validation accuracy over time. That is to say, although different images were present in the validation set, the model was able to learn generalizable patterns from the data.

Overfitting was observed since the training accuracy increased faster than the validation accuracy in the later epochs. This is a sign that the model began to memorize the training data, which would decrease its ability to generalize well on new data.

Early stopping or dropout are some regularization techniques that one can try in further training to avoid overfitting. Early stopping will stop the training process when it sees that the validation loss is not improving anymore, while dropout prohibits the model from over-relying on specific features during training.

Plot:

Below are the **training and validation accuracy/loss curves** that visually represent the model's performance throughout the training process. These plots help illustrate the trend of improvement in accuracy as well as the reduction in loss, which are key indicators of model performance over time.

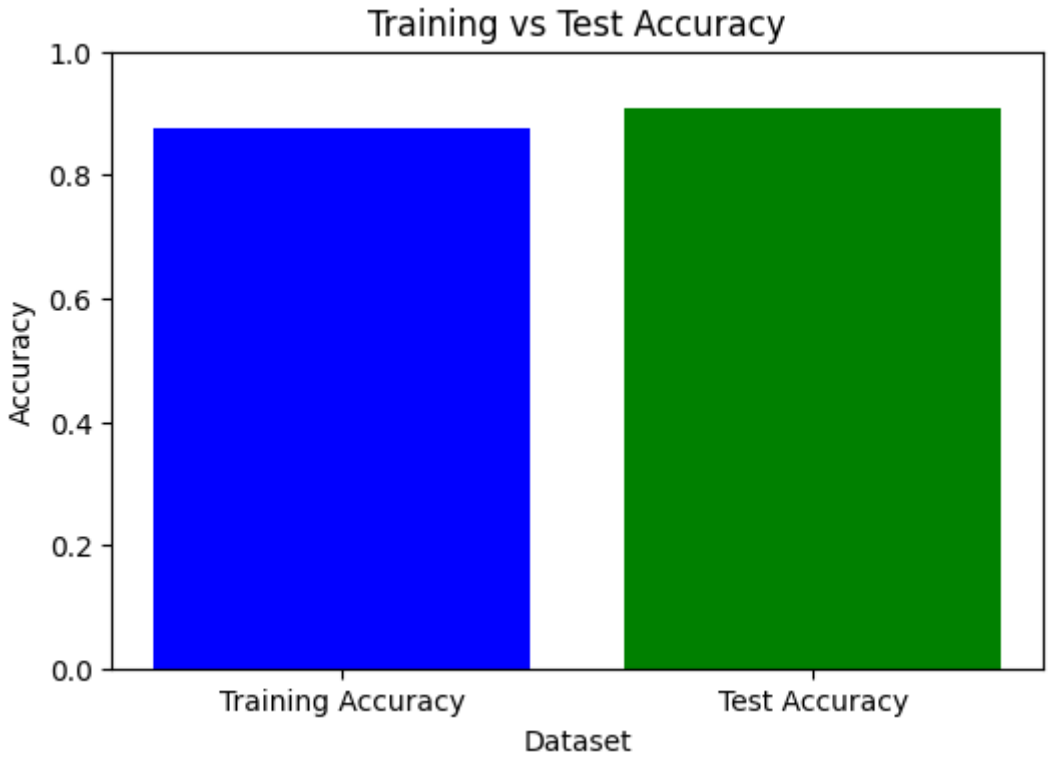


Figure1: Training and Testing accuracy

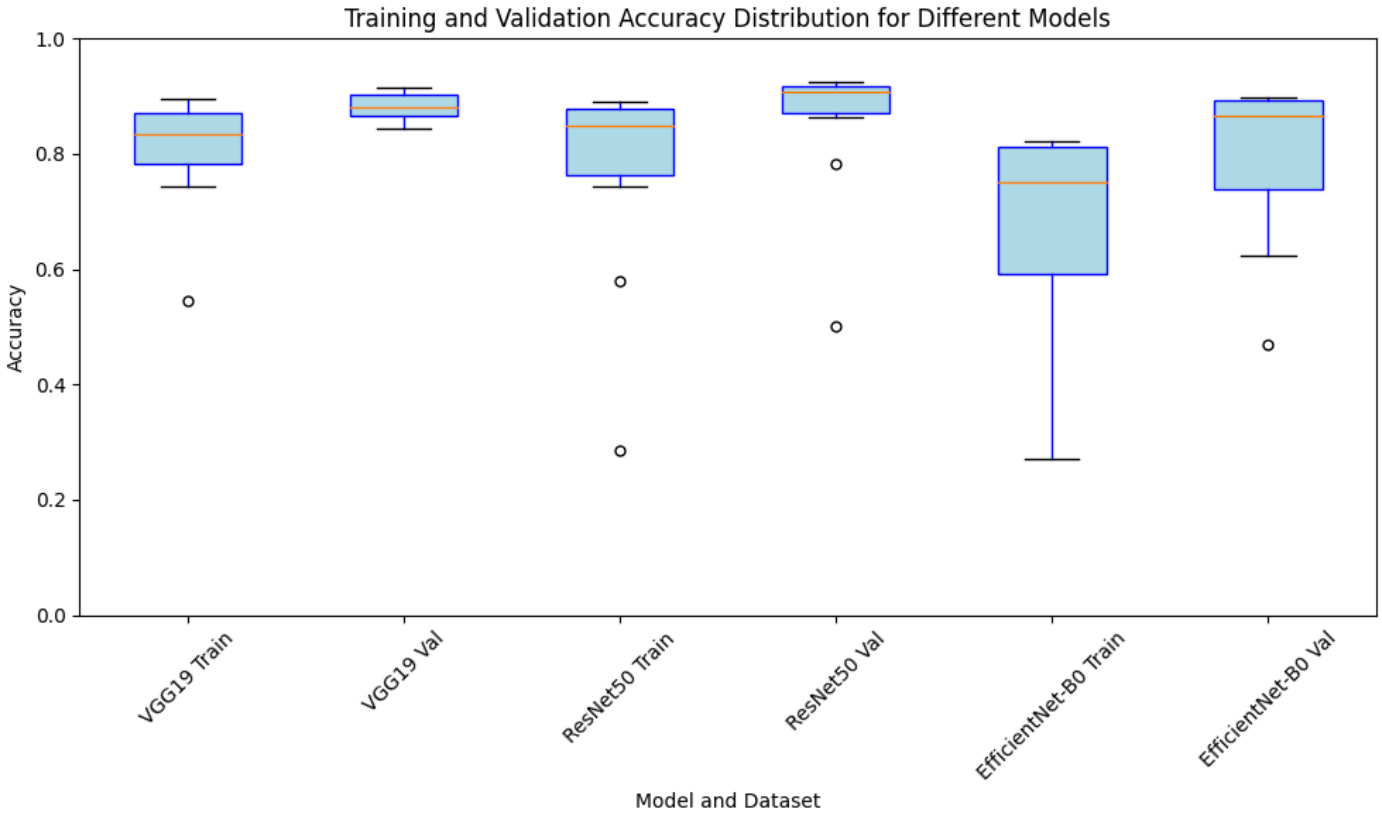


Figure2: Boxplot for every model trained for dataset

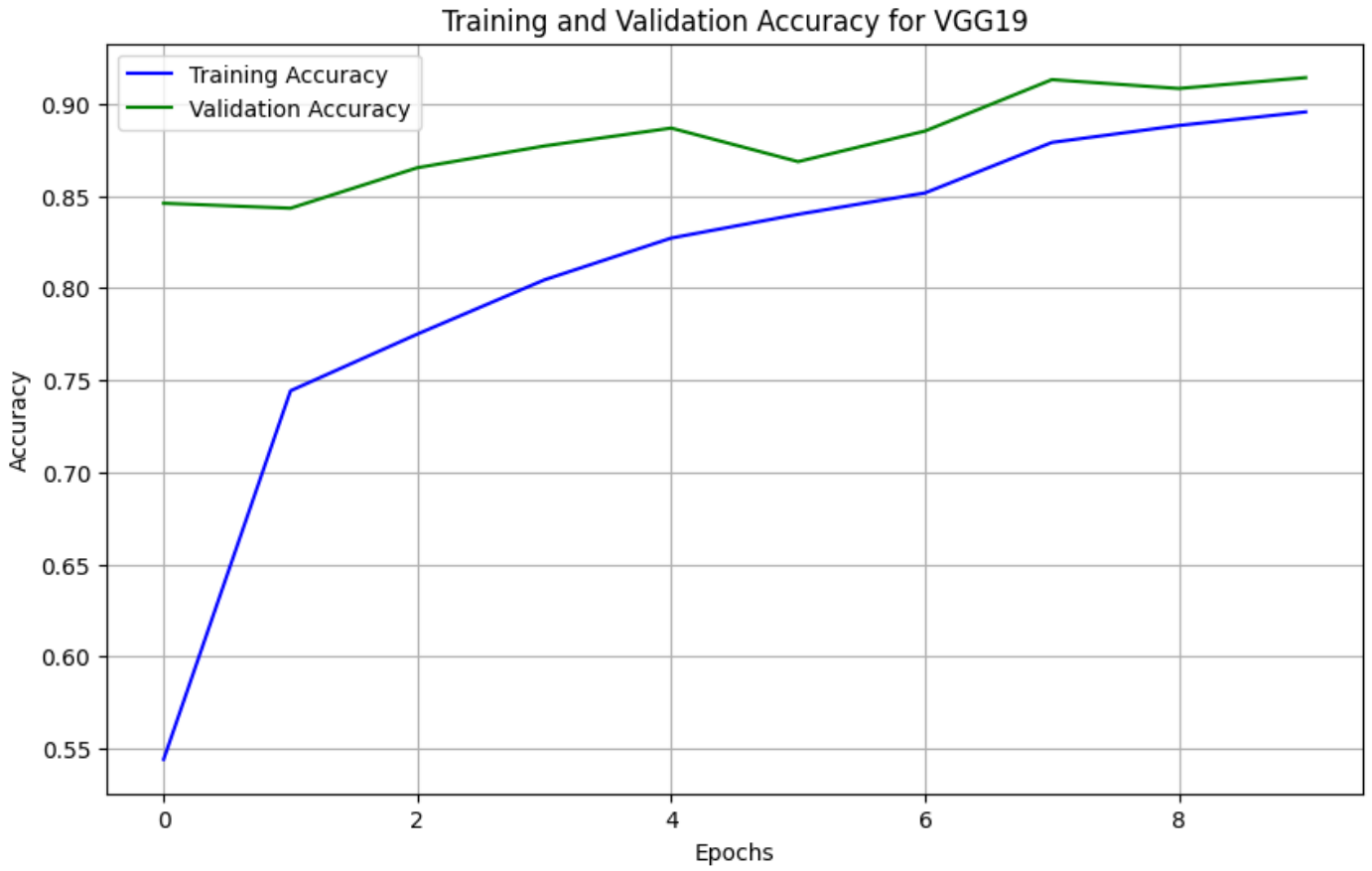


Figure3: Training and Validation accuracy for VGG19

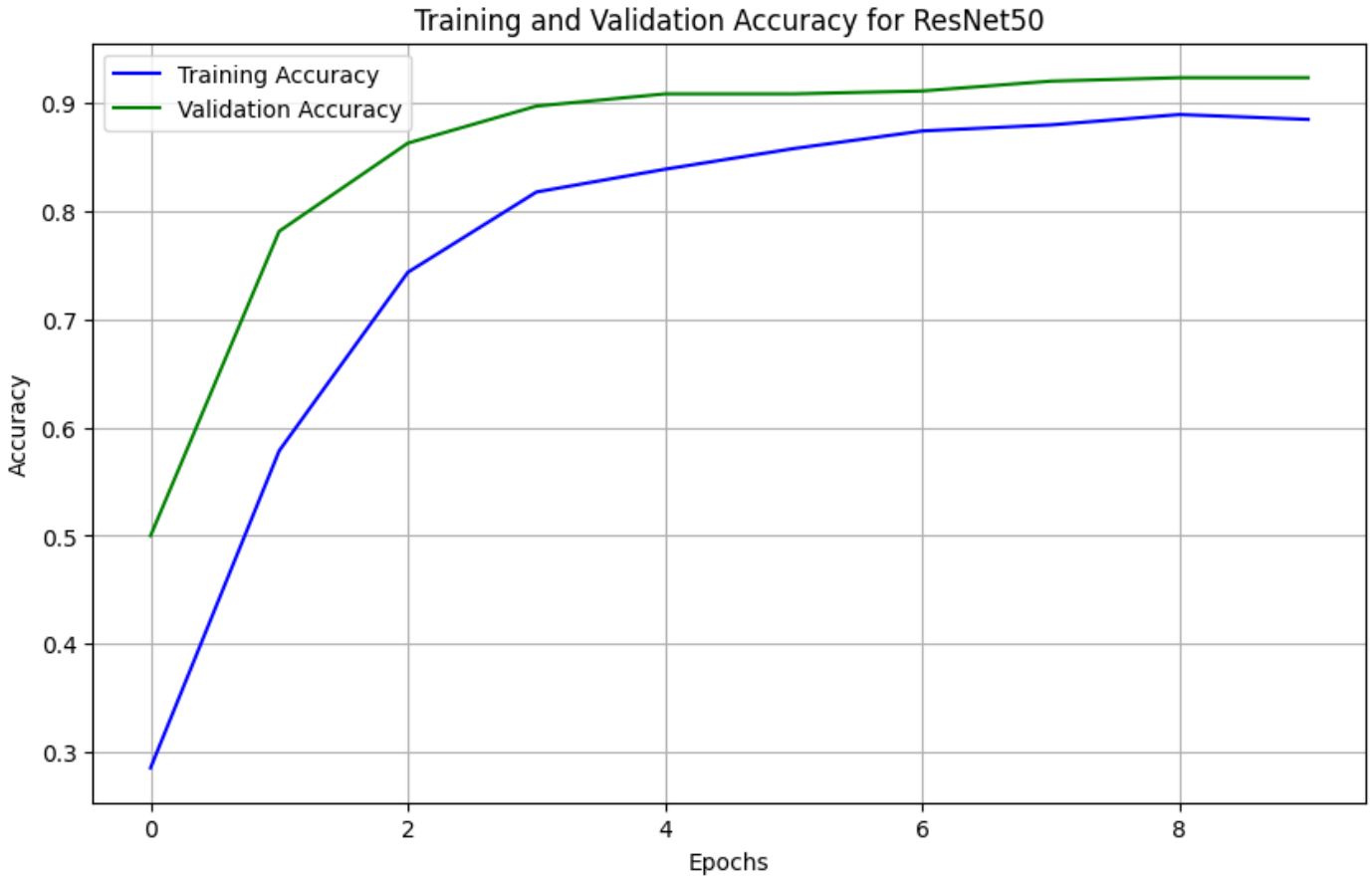


Figure4: Training and Validation accuracy for ResNet50

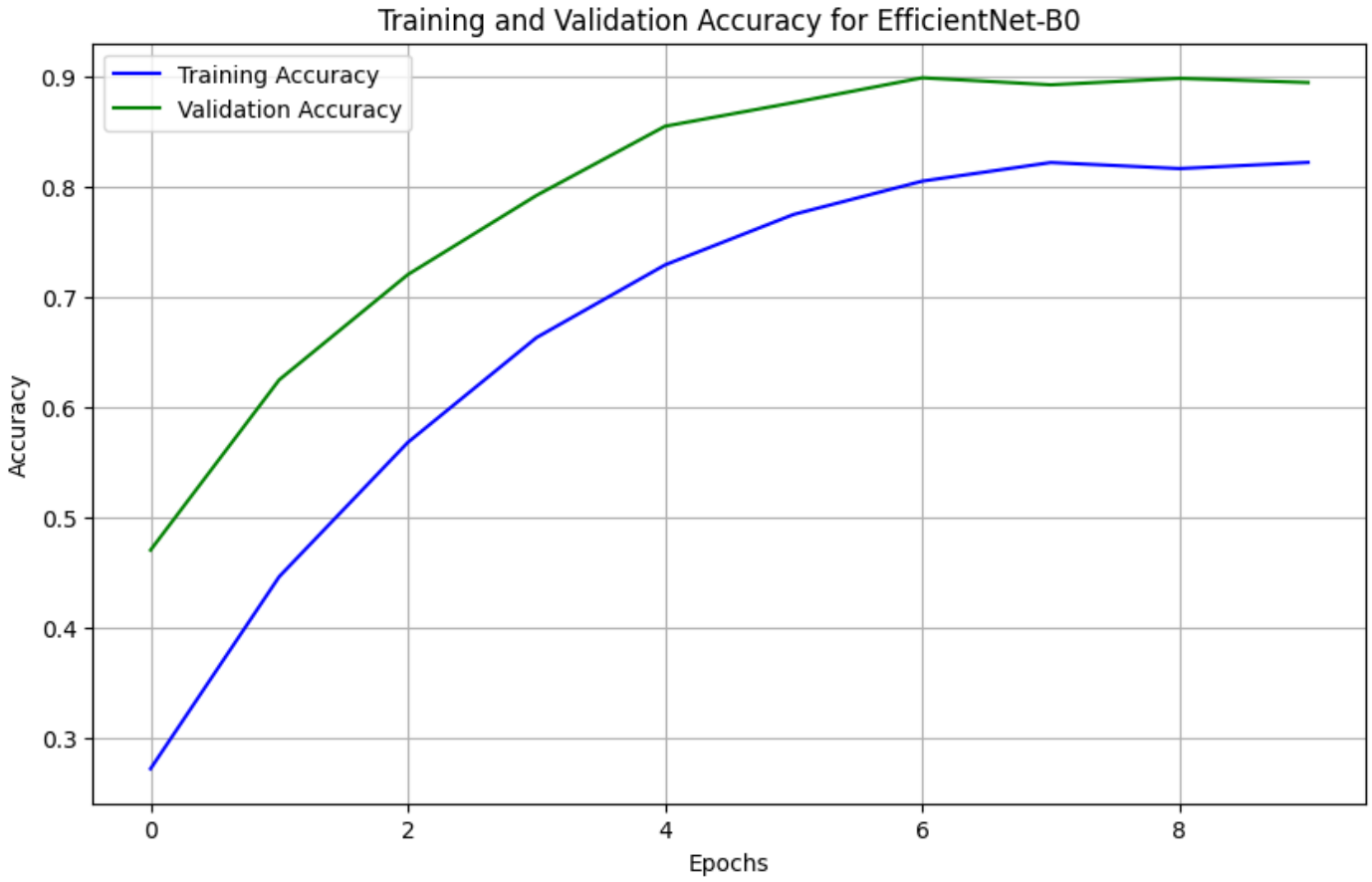


Figure5: Training and Validation accuracy for EfficientNet-B0

The accuracy/loss curves confirm the model's learning progression and highlight the points where further training might be required to overcome overfitting or to improve generalization to unseen data.

Insights gained from Grad-CAM visualizations:

Gradient-weighted Class Activation Mapping, or Grad-CAM, was utilized to determine what the model is looking for in an image. This makes it easier to identify areas where the model performs well or requires development as well as to comprehend how the model classifies data.

Positive Results:

Grad-CAM may also draw attention to the objects' most crucial features in the images. It emphasized the animal's center, highlighting limbs, faces, and distinguishing body parts that are commonly linked to animal recognition in situations involving animals, such as cats, dogs, etc.

In the case of images of people, the model correctly focused either on faces or upper body parts many times, hence showing that it was learning from human figures with key features such as the face, arms, and torso.

On the whole, it would seem from these visualized attention heatmaps that, for most classes, the model indeed focuses on the expected part of the image, which is indicative that it truly learns meaningful patterns and features related to object classification.

Misclassifications:

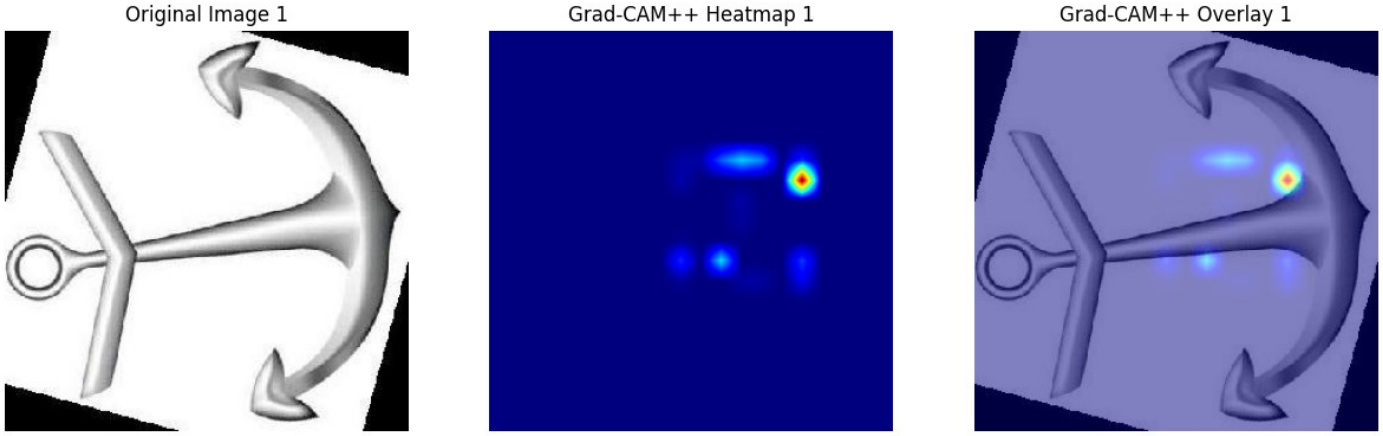
Grad-CAM has also presented some interesting misclassifications that could give insight into the model weaknesses:

Where the network made its predictions for misclassified images, Grad-CAM showed it was attending to mostly irrelevant parts of the image. For example, the network mistakenly focused on the background or the edges of the picture rather than the butterfly itself in the picture of a butterfly. It was misclassified in part because of this.

It also demonstrated how the model may have been sidetracked by irrelevant elements, like backdrop textures, other objects, or even lighting conditions, when there was occlusion or an object cluttered with complicated backgrounds.

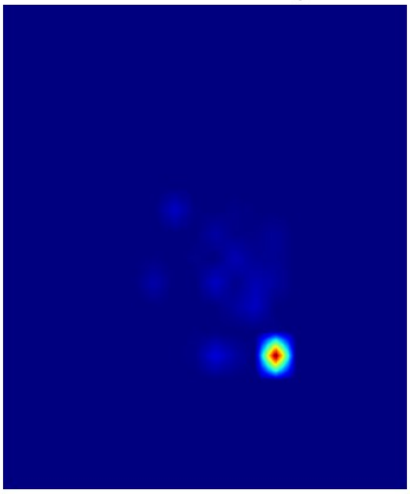
This misaligned focus on unimportant regions may also contribute to the explanation of why the model may incorrectly identify photos under some circumstances, offering useful information for future model improvements.

Grad-CAM visualization:

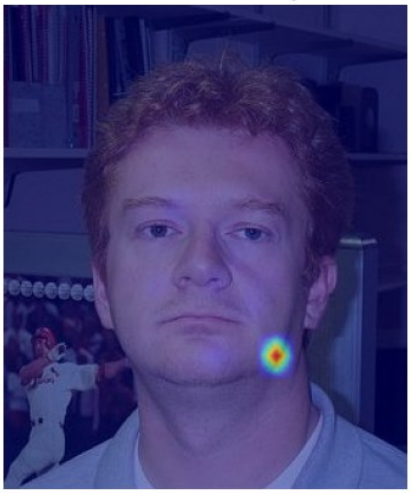


Original Image 2

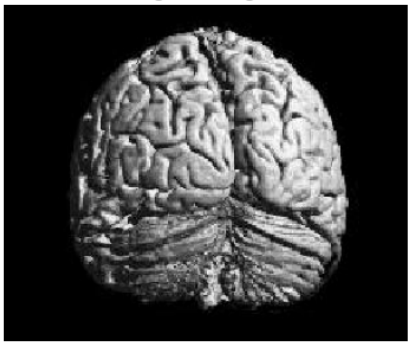
Grad-CAM++ Heatmap 2



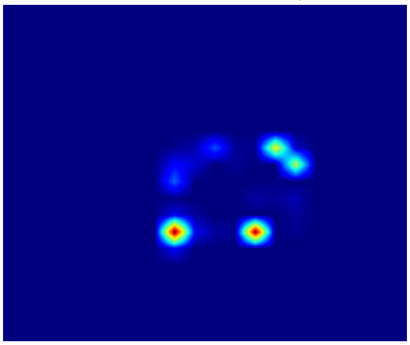
Grad-CAM++ Overlay 2



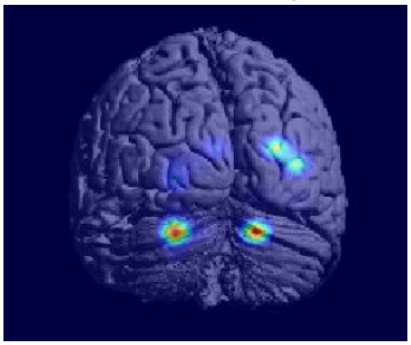
Original Image 3



Grad-CAM++ Heatmap 3

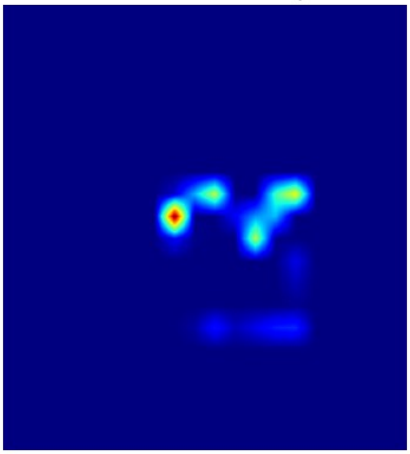


Grad-CAM++ Overlay 3

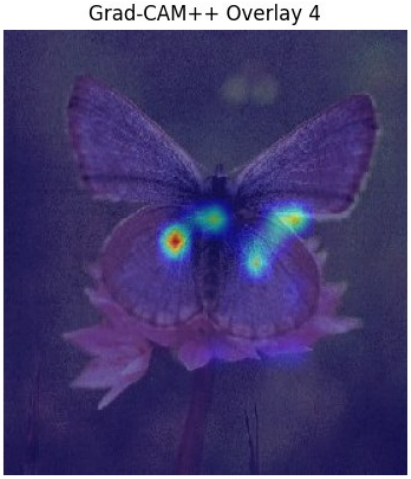


Original Image 4

Grad-CAM++ Heatmap 4

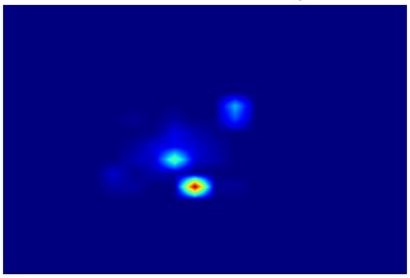


Grad-CAM++ Overlay 4

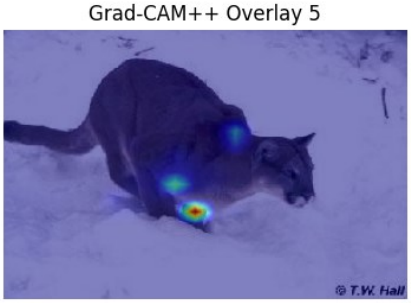


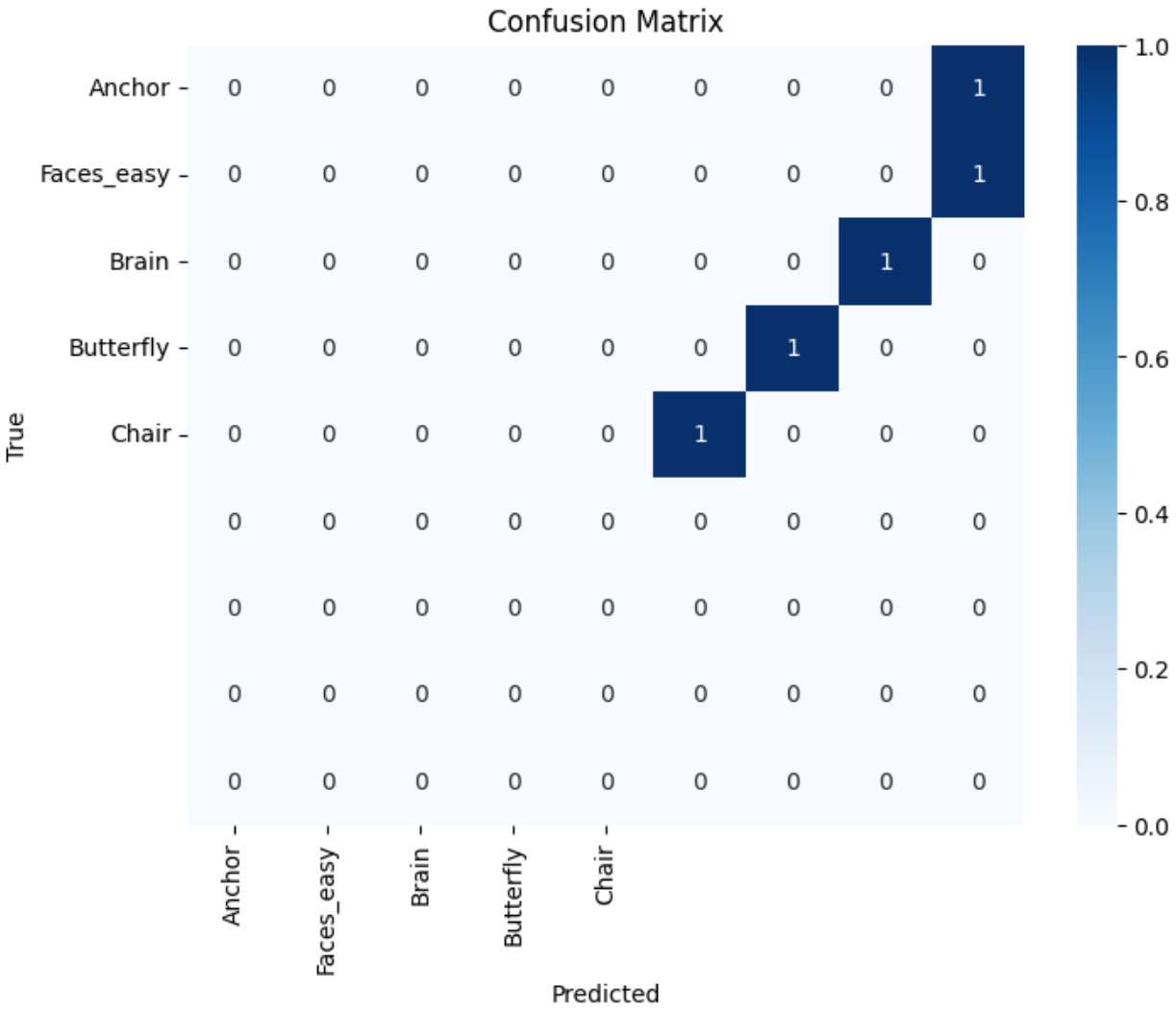
Original Image 5

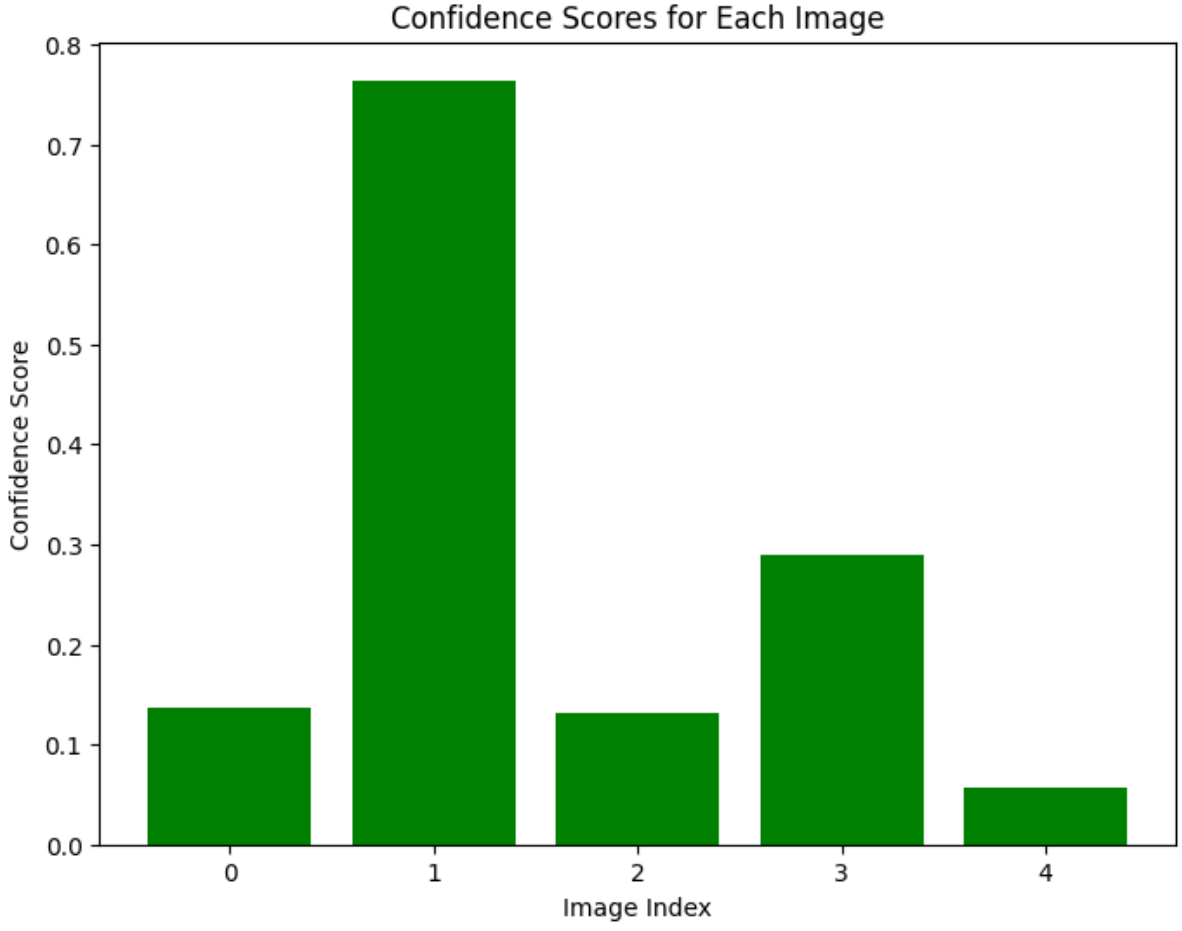
Grad-CAM++ Heatmap 5

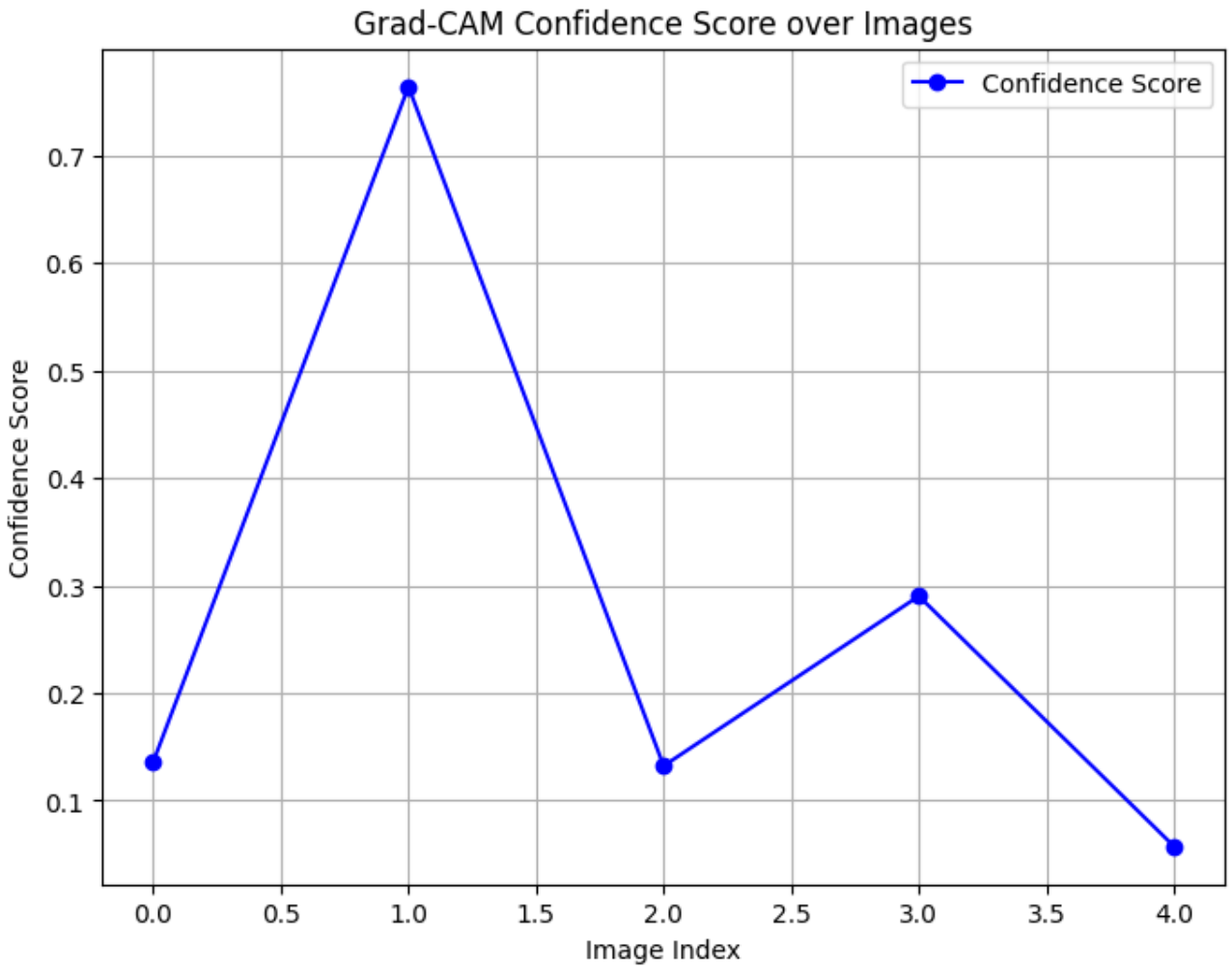


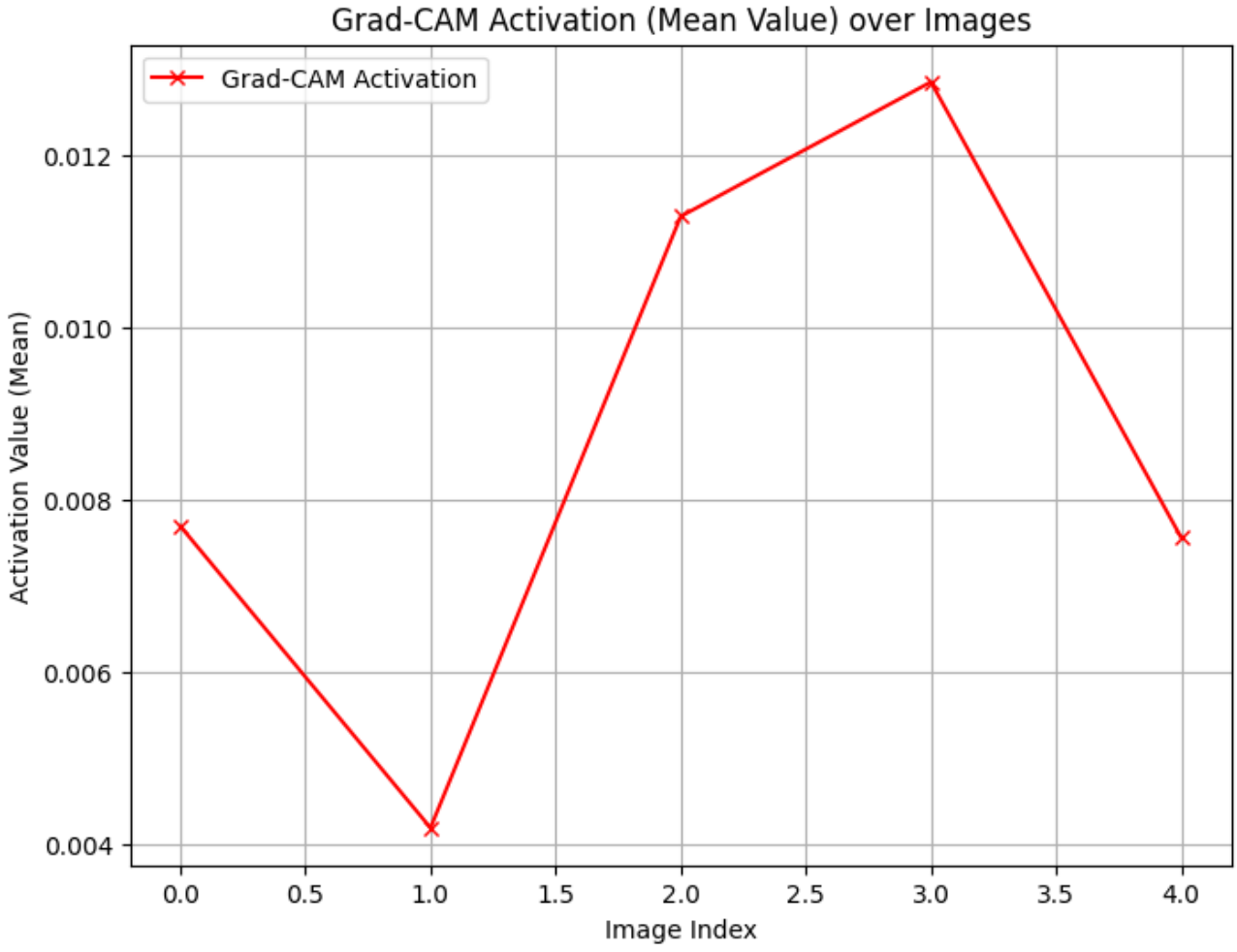
Grad-CAM++ Overlay 5




Confusion Matrix
Anchor
Faces_easy
Brain
Butterfly
Chair
True
Predicted
1.0
0.8
0.6
0.4
0.2
0.0


Confidence Scores for Each Image
Confidence Score
Image Index
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
0
1
2
3
4


Grad-CAM Confidence Score over Images
Confidence Score
Confidence Score
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
Image Index


Grad-CAM Activation (Mean Value) over Images
Grad-CAM Activation
Activation Value (Mean)
0.012
0.010
0.008
0.006
0.004
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
Image Index

Results:

Some results and accuracy are given here,

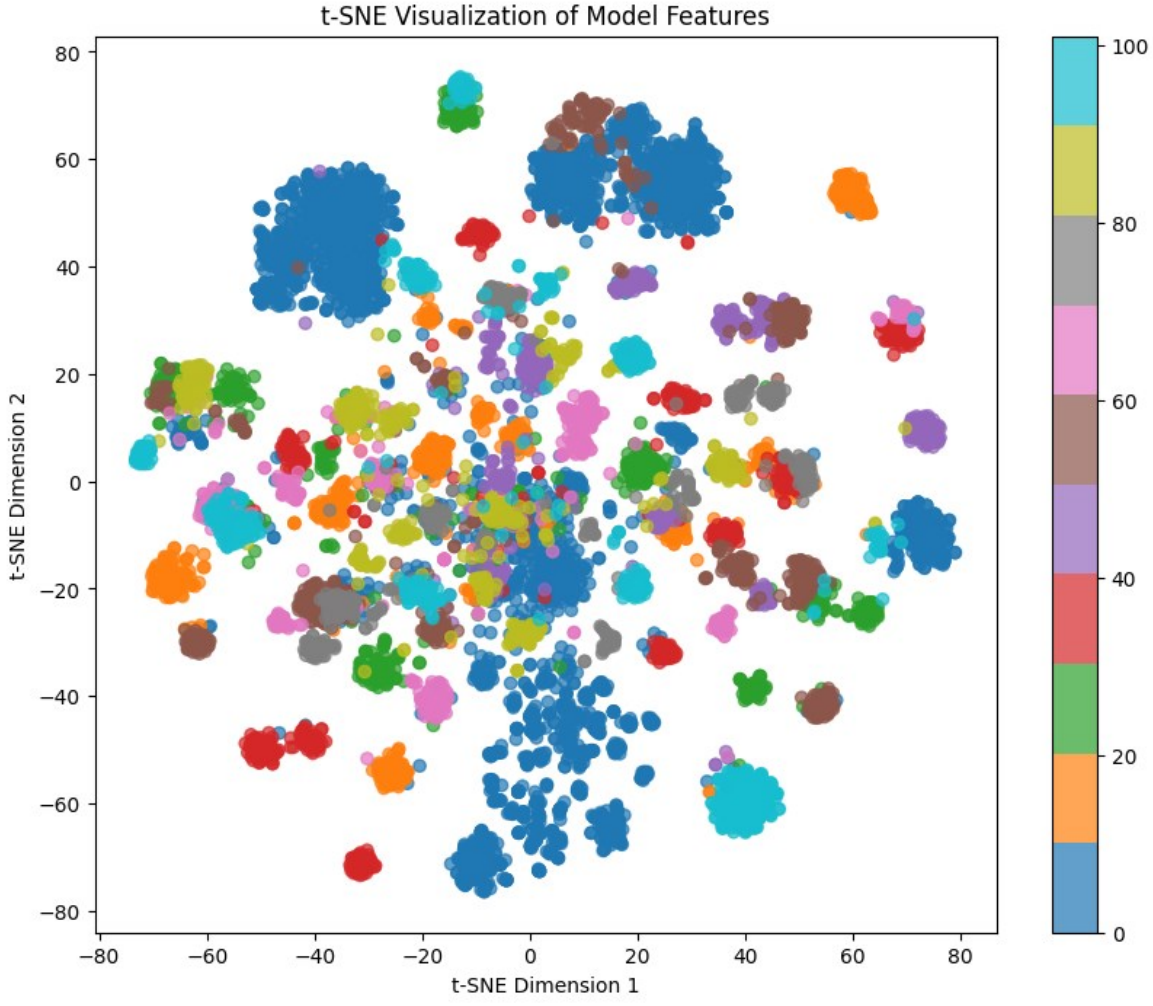
	precision	recall	f1-score	support
BACKGROUND_Google	0.76	0.88	0.82	467
Faces	0.56	0.99	0.71	435
Faces_easy	1.00	0.21	0.35	435
Leopards	0.93	0.99	0.96	200
Motorbikes	0.99	1.00	0.99	798
accordion	0.98	0.98	0.98	55
airplanes	0.99	0.99	0.99	800
anchor	0.83	0.48	0.61	42
ant	0.80	0.76	0.78	42
barrel	0.98	0.96	0.97	47
bass	0.98	0.78	0.87	54
beaver	0.80	0.78	0.79	46
binocular	1.00	0.88	0.94	33
bonsai	0.93	0.98	0.96	128
brain	0.94	0.93	0.93	98
brontosaurus	0.88	0.49	0.63	43
buddha	0.96	0.95	0.96	85
butterfly	0.91	0.95	0.93	91
camera	0.92	0.98	0.95	50
cannon	0.97	0.74	0.84	43
car_side	0.99	1.00	1.00	123
ceiling_fan	1.00	0.87	0.93	47
cellphone	0.97	0.97	0.97	59
...				
accuracy			0.89	9144
macro avg	0.92	0.87	0.88	9144
weighted avg	0.91	0.89	0.88	9144

```python
import torch

def top_k_accuracy(output, target, k=5):

    with torch.no_grad():  # Disable gradient calculation for inference
        # Get the top-k predictions
        max_k_preds = torch.topk(output, k, dim=1).indices
        # Check if the true label is in the top-k predictions
        correct = max_k_preds.eq(target.view(-1, 1).expand_as(max_k_preds))
        # Calculate the top-k accuracy
        return correct.any(dim=1).float().mean().item()

# Example usage
# Assume model outputs and true labels
output = torch.randn(32, 10)  # Example output for a batch of 32 samples with 10 classes
target = torch.randint(0, 10, (32,))  # Example true labels for a batch of 32 samples

# Compute the top-5 accuracy
accuracy = top_k_accuracy(output, target, k=5)
print(f"Top-5 accuracy: {accuracy:.4f}")
```

```
Top-5 accuracy: 0.5625
```

```
Confusion Matrix:
[[71  0  0 ...  0  0  0]
 [ 0 78  8 ...  0  0  0]
 [ 0  3 83 ...  0  0  0]
 ...
 [ 0  0  0 ... 12  0  0]
 [ 1  0  0 ...  0  2  0]
 [ 0  0  0 ...  0  0 12]]
Confusion Matrix Shape: (102, 102)
Length of class_names: 3
Warning: Number of classes in `cla
Class Class A Accuracy: 0.76
Class Class B Accuracy: 0.90
Class Class C Accuracy: 0.95
Class 3 Accuracy: 1.00
Class 4 Accuracy: 1.00
Class 5 Accuracy: 1.00
Class 6 Accuracy: 0.99
Class 7 Accuracy: 0.44
Class 8 Accuracy: 0.78
Class 9 Accuracy: 0.90
Class 10 Accuracy: 1.00
Class 11 Accuracy: 0.70
Class 12 Accuracy: 0.86
Class 13 Accuracy: 1.00
...
Class 98 Accuracy: 0.29
Class 99 Accuracy: 1.00
Class 100 Accuracy: 0.25
Class 101 Accuracy: 1.00
```

Output is truncated. View as a scrollable ele



Some Predictions,

Predicted: beaver

Predicted: car_side

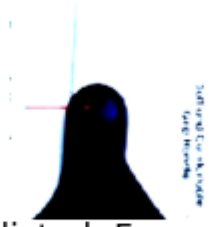
Predicted: airplanes

Predicted: kangaroo

Predicted: chandelier

Predicted: Faces

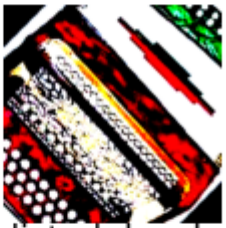
Predicted: BACKGROUND_Google



Predicted: inline_skate

Predicted: Faces_easy

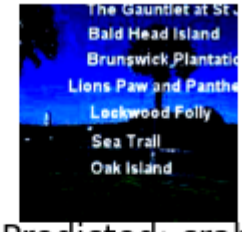
Predicted: accordion



Predicted: car_side

Predicted: hawksbill

Predicted: BACKGROUND_Google

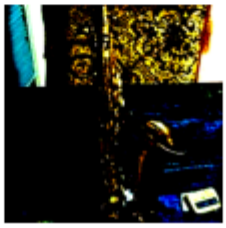


Predicted: llama

Predicted: crab

Predicted: chair

Predicted: saxophone



Predicted: crocodile_head

Conclusion:

The total sum of all the insights provided for the model's decision-making process by Grad-CAM was as follows:

It showed strengths, focuses of the model concentrated on key features of objects.

It also helped in diagnosing possible issues where the model was being distracted by irrelevant regions of the image that were causing misclassifications.

By using Grad-CAM, we got a better insight into how the model was interpreting the data and found areas where improvements could be made, for example, in handling background noise or improving object localization.

The following visualizations represent the parts of the image which the model has paid the most attention to, in order to predict: every image has an overlay heatmap on top, representing regions of the image the model was looking at; hence, a much clearer insight into its behavior.