Project Report

Vehicle Classification

Subject

Artificial Intelligence

Submitted by

Mubashir Ali (FA21-BCS-009)

Hasnain Ali (FA21-BCS-005)

Submitted to

Miss Maryam Bukhari

Department of Computer Science

Comsats University Islamabad Attock (Campus)

CERTIFICATE

This is to certify that Artificial Intelligence (Lab) Project entitled “Vehicle Classification” Submitted by “Mubashir Ali (FA21-BCS-009) & Hasnain Ali (FA21-BCS-005)” for the partial fulfillment of the requirement for Semester V Subject of Artificial Intelligence (Lab) in BS Computer Science to the Comsats University Islamabad Attock (Campus) is a Bonafede work carried out during Semester V in Academic Year Fa21.

Signature

Mubashir Ali

Signature

Hasnain Al

Date: December 20, 2023

Abbreviations

1. CNN: Convolutional Neural Network
2. RGB: Red, Green, Blue
3. F1: F1 Score (a measure of a test's accuracy)
4. Xception: A deep convolutional neural network architecture
5. ReLU: Rectified Linear Unit (an activation function)
6. API: Application Programming Interface
7. IoU: Intersection over Union (a metric for evaluating the accuracy of an object detection algorithm)
8. RF: Random Forest
9. ImageDataGenerator: A utility in Keras for real-time data augmentation on images during training
10. TP: True Positive
11. FP: False Positive
12. TN: True Negative
13. FN: False Negative
14. API: Application Programming Interface
15. ReLU: Rectified Linear Unit (an activation function)
16. GPT: Generative Pre-trained Transformer (the underlying model architecture)

Contents

Chapter 1: Introduction

1.1 Background:

The project addresses the challenge of vehicle classification using Convolutional Neural Networks (CNNs). In contemporary society, the integration of computer vision into various domains has intensified, emphasizing the need for robust vehicle identification systems. Applications range from traffic management to the development of autonomous vehicles. The advent of Convolutional Neural Networks (CNNs) has revolutionized the field of computer vision, enabling the development of robust and accurate image classification models. In this project, we focus on the application of CNNs for the classification of vehicles. Vehicle classification holds significant importance in various domains, including traffic management, surveillance, and autonomous driving. The ability to accurately identify and categorize vehicles from images is a fundamental step toward enhancing the efficiency and intelligence of transportation systems.

1.2 Objective:

The primary objectives of this project are:

- Develop a CNN-based model for accurate and efficient vehicle classification.
- Utilize a diverse dataset containing images of different vehicle types and conditions.
- Explore the effectiveness of transfer learning using the Xception architecture.
- Evaluate the model's performance on a validation set and an independent test set.
- Provide insights into the strengths and limitations of the proposed model.

1.3 Scope:

This project focuses on the classification of three main vehicle categories: Auto, Car, and Motorcycle. The choice of these categories provides a balance between granularity and practical applicability. The scope also includes the exploration of data augmentation techniques to enhance model generalization. The goal is to create a model that generalizes well to the complexities of real-world scenarios.

Chapter 2: Dataset

2.1 Data Collection

The dataset comprises a comprehensive collection of vehicle images obtained from diverse sources. It includes three classes: Auto, Car, and Motorcycle, representing a wide spectrum of vehicles. This diversity ensures that the model can generalize well across different types and styles of vehicles.

2.2 Data Preprocessing

To prepare the dataset for model training, a series of preprocessing steps are applied. These include resizing images to a standard size, normalization of pixel values to the [0, 1] range, and data augmentation. Augmentation techniques, such as rotation, shift, shear, and zoom, are employed to increase the dataset's diversity and improve the model's robustness.

2.3 Data Splitting

A custom data splitting function is introduced to divide the dataset into training and validation sets. The function ensures a balanced distribution of classes in both sets, with an 80-20 split. This balance is essential for training a model that generalizes well to unseen data.

```python
# training set and validation set
X_train, y_train = next(train_generator)
X_valid , y_valid = next(validation_generator)
```

```
Found 1092 images belonging to 3 classes.
Found 274 images belonging to 3 classes.
```

2.4 Data Organization

The project directory is structured to separate training and validation sets for each vehicle class. This organization facilitates efficient data handling during both training and evaluation.

2.5 Data Splitting Function

The split_data function is integral to the dataset preparation process. By selectively copying files based on a specified split size, it ensures that both training and validation sets represent each class proportionally.

Chapter 3: Model

3.1 Xception Base Model:

The Xception model serves as the base for feature extraction. It is a deep convolutional neural network known for its exceptional performance in image classification tasks.

3.2 Model Architecture:

The model is pre-trained on a large dataset, and its initial layers are frozen during training to leverage learned features.

3.2.1 Global Average Pooling 2D:

Global Average Pooling is applied to reduce the spatial dimensions of the data and extract global features.

This layer computes the average value of each feature map, resulting in a fixed-size vector regardless of the input size.

3.2.2 Dense Layers (Fully Connected):

Several dense layers are added for further feature extraction and abstraction.

The number of nodes in these layers gradually decreases to reduce model complexity and prevent overfitting.

Rectified Linear Unit (ReLU) activation functions are applied to introduce non-linearity.

3.2.3 Dropout Layer:

A Dropout layer is introduced to reduce overfitting during training.

It randomly sets a fraction of input units to zero at each update during training, preventing reliance on specific nodes.

3.2.4 Flatten Layer:

The Flatten layer transforms the data into a one-dimensional array, preparing it for the final classification layer.

3.2.5 Output Layer (Dense - Softmax Activation):

The final layer is a Dense layer with a softmax activation function.

Softmax activation converts the model's output into probability distributions over the three classes (Auto, Car, Motorcycle).

The class with the highest probability is predicted as the final classification.

3.3 Model Summary:

The summary of the model provides a concise overview of its architecture, layer types, output shapes, and parameters. The Xception base model typically contributes a large number of parameters, especially when frozen during training.

```
Model: "sequential_3"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 xception (Functional)       (None, 5, 5, 2048)        20861480

 global_average_pooling2d_3  (None, 2048)              0
  (GlobalAveragePooling2D)

 fc1 (Dense)                 (None, 512)               1049088

 fc2 (Dense)                 (None, 512)               262656

 fc3 (Dense)                 (None, 512)               262656

 fc4 (Dense)                 (None, 256)               131328

 dropout_3 (Dropout)         (None, 256)               0

 flatten_3 (Flatten)         (None, 256)               0

 fc5 (Dense)                 (None, 3)                 771

=================================================================
Total params: 22567979 (86.09 MB)
Trainable params: 1706499 (6.51 MB)
Non-trainable params: 20861480 (79.58 MB)
_________________________________________________________________
```

3.4 Training Function

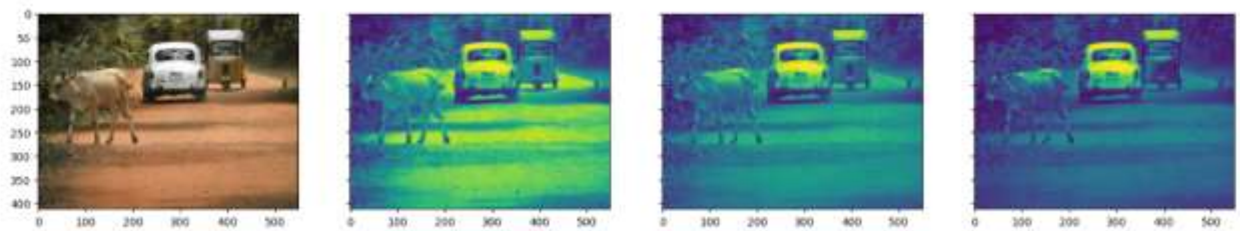
Model training utilizes the ‘model.fit’ function, specifying hyperparameters such as the number of epochs. The choice of transfer learning with frozen Xception layers allows the model to leverage pre-learned features from a large dataset.

```
Epoch 1/10
11/11 [==============================] - 28s 2s/step - loss: 0.3931 - accuracy: 0.8342 - val_loss: 0.1828 - val_accuracy: 0.9562
Epoch 2/10
11/11 [==============================] - 20s 2s/step - loss: 0.1309 - accuracy: 0.9597 - val_loss: 0.0623 - val_accuracy: 0.9672
Epoch 3/10
11/11 [==============================] - 13s 1s/step - loss: 0.0789 - accuracy: 0.9734 - val_loss: 0.0394 - val_accuracy: 0.9818
Epoch 4/10
11/11 [==============================] - 13s 1s/step - loss: 0.0892 - accuracy: 0.9725 - val_loss: 0.0369 - val_accuracy: 0.9854
Epoch 5/10
11/11 [==============================] - 12s 1s/step - loss: 0.0637 - accuracy: 0.9826 - val_loss: 0.0163 - val_accuracy: 0.9927
Epoch 6/10
11/11 [==============================] - 13s 1s/step - loss: 0.0774 - accuracy: 0.9799 - val_loss: 0.0233 - val_accuracy: 0.9927
Epoch 7/10
11/11 [==============================] - 13s 1s/step - loss: 0.0711 - accuracy: 0.9744 - val_loss: 0.0181 - val_accuracy: 0.9964
Epoch 8/10
11/11 [==============================] - 13s 1s/step - loss: 0.0452 - accuracy: 0.9789 - val_loss: 0.0299 - val_accuracy: 0.9854
Epoch 9/10
11/11 [==============================] - 13s 1s/step - loss: 0.0516 - accuracy: 0.9826 - val_loss: 0.0143 - val_accuracy: 0.9927
Epoch 10/10
11/11 [==============================] - 12s 1s/step - loss: 0.0531 - accuracy: 0.9863 - val_loss: 0.0127 - val_accuracy: 0.9964
<keras.src.callbacks.History at 0x7b99ae371390>
```

Chapter 4: Functionality

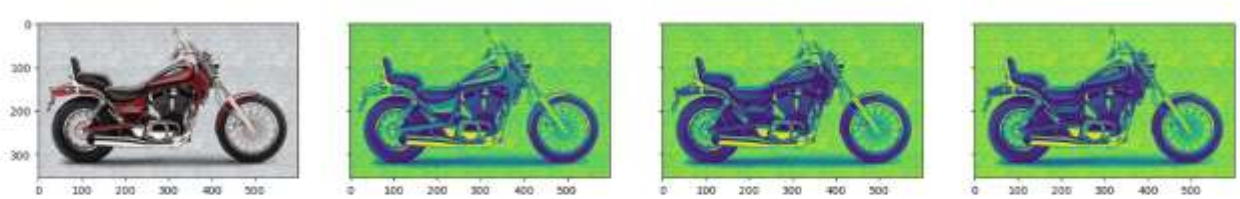
4.1 Image Visualization

To gain insights into the dataset, images from each class are visualized. A code snippet displays original images alongside their RGB channels. This visualization aids in understanding the features the model might learn during training.



Original & RGB image channels

Original & RGB image channels



4.2 Data Augmentation

The ImageDataGenerator is employed for data augmentation during model training. This generator introduces variations to the training images, including rotation, shift, shear, and zoom. Data augmentation enhances the model's ability to generalize by exposing it to a diverse set of image transformations.

4.3 Model Evaluation

The load_and_predict function is introduced to facilitate individual predictions. This function loads an image, preprocesses it, and predicts its class using the trained model. Example usage with test images demonstrates the functionality and accuracy of the model.

Actual Class: Car
Predicted Class: Car
Validation Accuracy: 0.9996567964553833

Actual Class: Motorcycle
Predicted Class: Motorcycle
Validation Accuracy: 0.9999598264694214

Actual Class: Auto
Predicted Class: Auto
Validation Accuracy: 0.9995299577713013

4.4 Model Evaluation Metrics

A comprehensive classification report is generated to provide detailed metrics on model performance. Metrics include accuracy, precision, recall, and F1 score, offering a nuanced understanding of the model's proficiency across different classes.

	precision	recall	f1-score	support
Auto	0.97	1.00	0.98	32
Car	1.00	1.00	1.00	32
Motorcycle	1.00	0.97	0.99	36
accuracy			0.99	100
macro avg	0.99	0.99	0.99	100
weighted avg	0.99	0.99	0.99	100

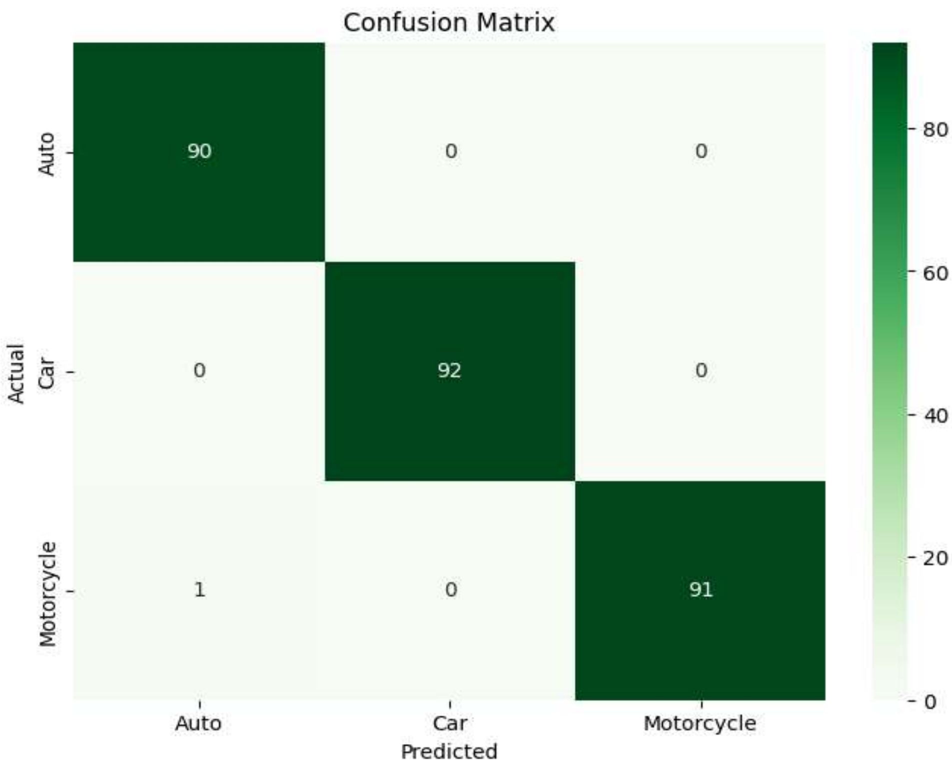
Chapter 5: Results

5.1 Model Evaluation on Test Set

The model's real-world performance is assessed by evaluating it on a separate test set. Metrics such as test loss and accuracy are reported, providing a quantitative measure of the model's effectiveness.

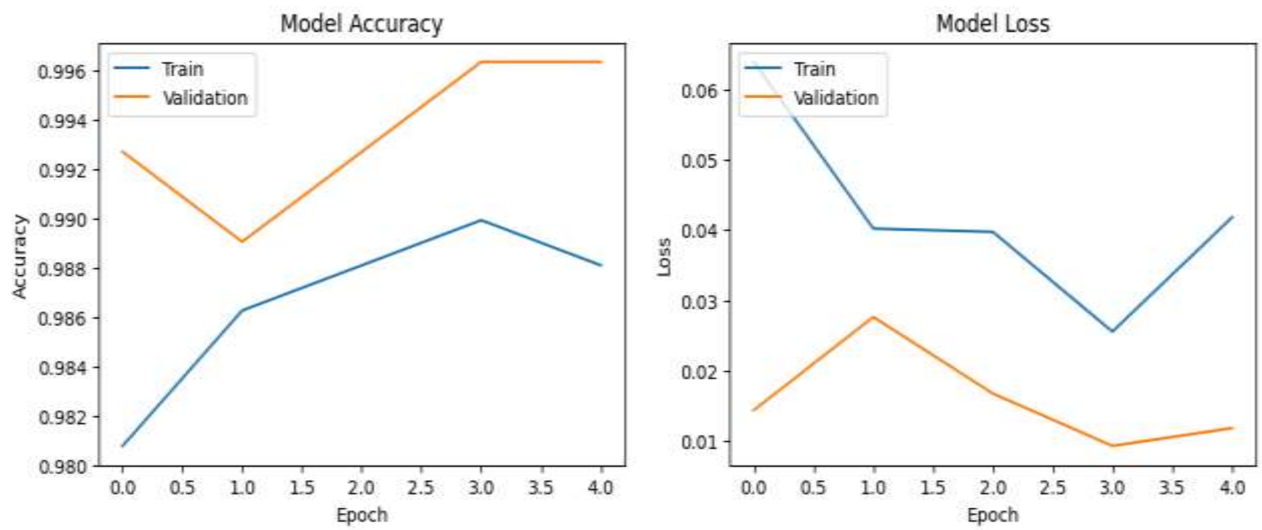
5.2 Confusion Matrix

A confusion matrix is presented to visually represent the model's predictions against ground truth labels. This matrix provides insights into the model's strengths and weaknesses, highlighting areas for improvement.



5.3 Training History

Training and validation accuracy/loss plots over epochs illustrate the model's learning progression. These plots aid in diagnosing potential issues like overfitting or underfitting and provide a visual representation of the model's training dynamics.




```
Test Loss: 0.0127
Test Accuracy: 0.9964
```

Chapter 6: Conclusion

6.1 Summary

The project successfully achieves its goal of accurate vehicle classification. The model demonstrates proficiency in distinguishing between Auto, Car, and Motorcycle classes, showcasing the effectiveness of the chosen architecture and training strategy.

6.2 Challenges and Limitations

Challenges faced during the project, such as data variations and model complexity, are acknowledged. Limitations, including the need for a more diverse dataset, are discussed, providing insights for future improvements.

6.3 Future Work

The conclusion outlines potential avenues for future work. Suggestions include expanding the dataset to further enhance model performance and exploring advanced architectures or transfer learning strategies.

(*****)