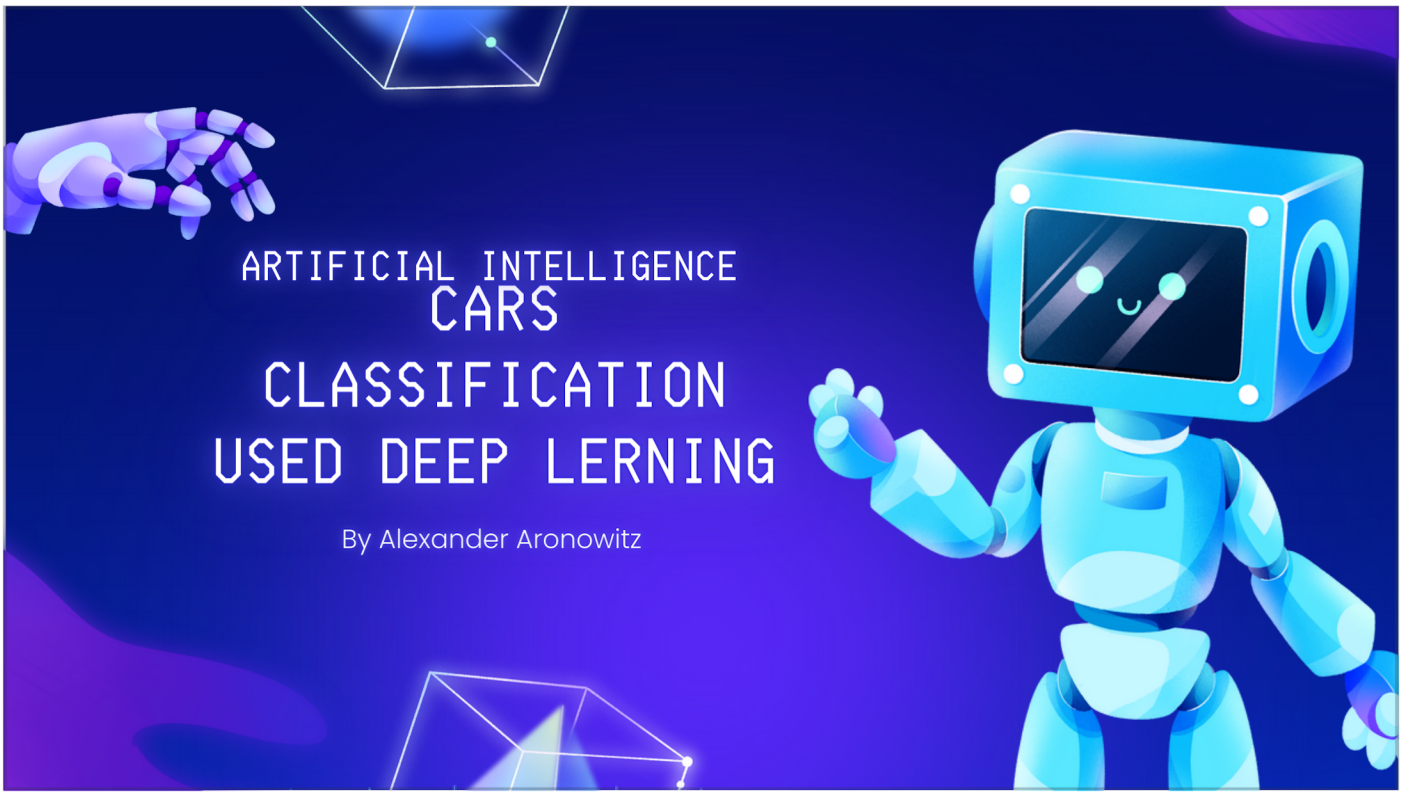

ARTIFICIAL INTELLIGENCE
CARS
CLASSIFICATION
USED DEEP LERNING
By Alexander Aronowitz

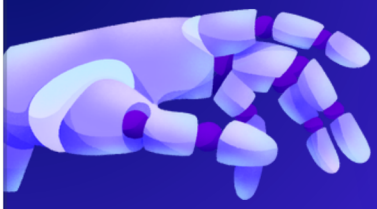
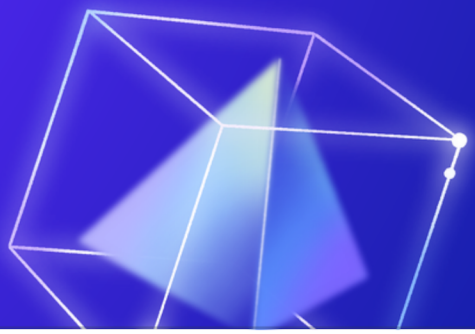


TABLE OF CONTENTS



CONTRIBUTIONS

1. Juri Babqi

Data Processing:

- Led the effort in loading and organizing the dataset of used car images.
- Implemented ImageDataGenerator for effective data augmentation to enhance model generalization.

2. Layan Almohmadi

Model Construction:

- Collaborated on incorporating MobileNet for feature extraction in the model architecture.
- Contributed to the design and addition of dense layers for car classification.

3. Bassma Almaimani

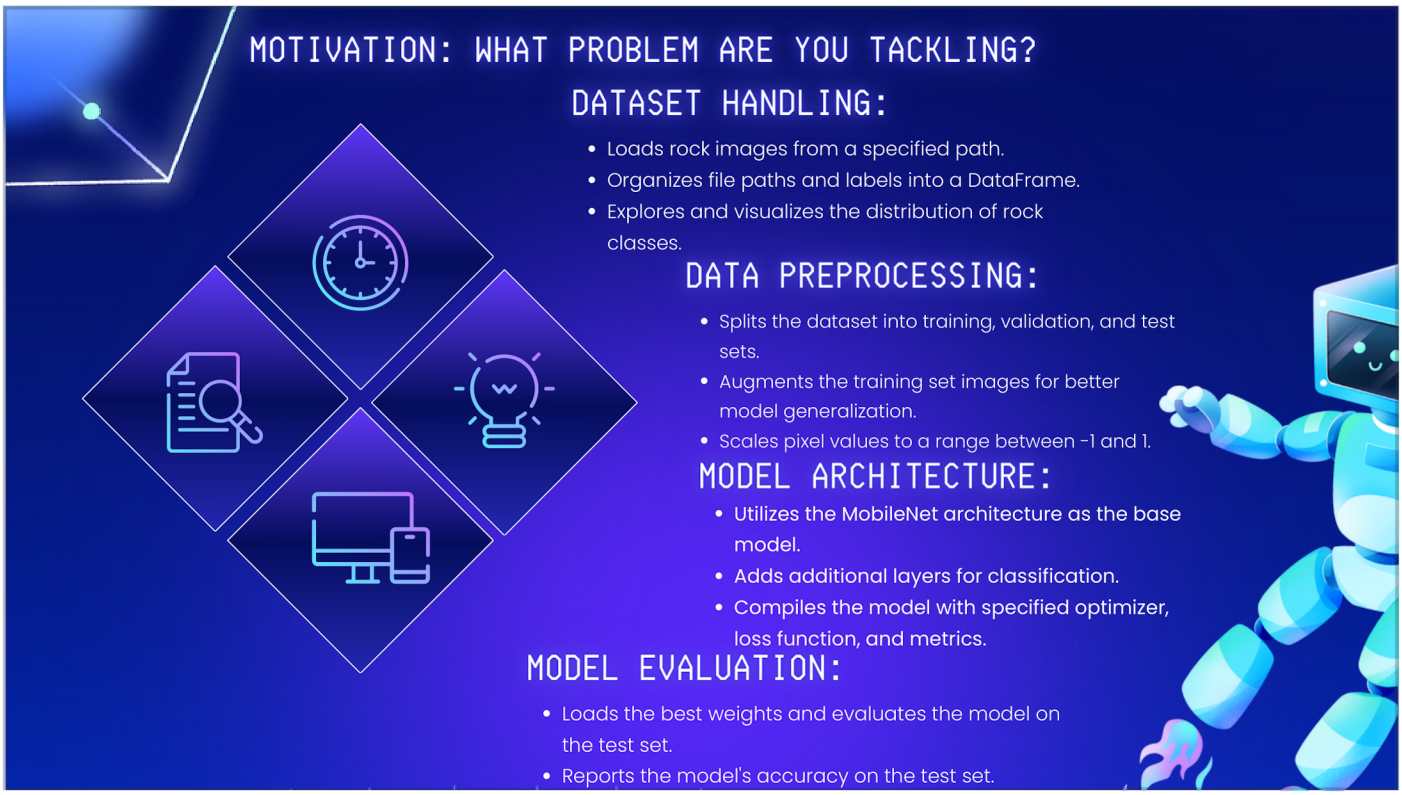
Training and Evaluation:

- Executed the training process for the model on the dataset.
- Conducted comprehensive evaluations on both validation and test sets.

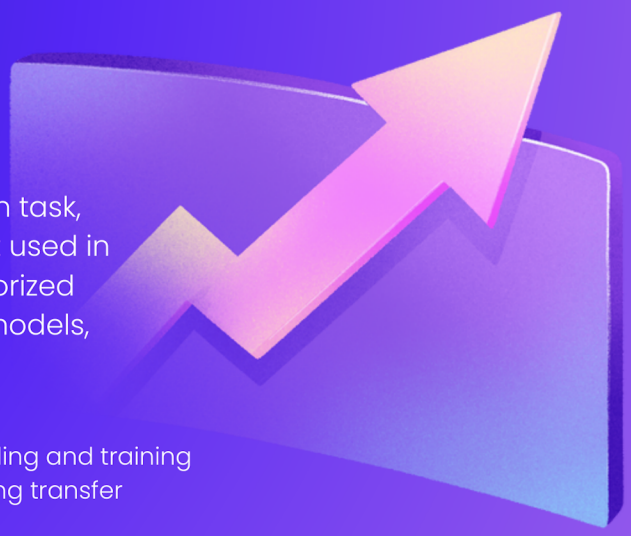
4. Omnyah Alsaedi

Results and Analysis:

- Took the lead in displaying the final test set accuracy, summarizing the model's overall performance.
- Implemented visualizations for training and validation accuracy curves to facilitate result interpretation.


MOTIVATION: WHAT PROBLEM ARE YOU TACKLING?
DATASET HANDLING:
Loads rock images from a specified path.
Organizes file paths and labels into a DataFrame.
Explores and visualizes the distribution of rock classes.
DATA PREPROCESSING:
Splits the dataset into training, validation, and test sets.
Augments the training set images for better model generalization.
Scales pixel values to a range between -1 and 1.
MODEL ARCHITECTURE:
Utilizes the MobileNet architecture as the base model.
Adds additional layers for classification.
Compiles the model with specified optimizer, loss function, and metrics.
MODEL EVALUATION:
Loads the best weights and evaluates the model on the test set.
Reports the model's accuracy on the test set.

INTRODUCTION



This project is designed for a car classification task, specifically classifying used cars. The dataset used in this project consists of images of cars, categorized into different classes based on their makes, models, or other relevant attributes.

This project aims to demonstrate the process of building and training a deep learning model for car classification, leveraging transfer learning with a pre-trained MobileNet architecture.

WORKFLOW OVERVIEW:

01

Data Processing:

- Load and organize a dataset of used car images.
- Employ ImageDataGenerator for data augmentation.

02

Model Construction:

- Use MobileNet for feature extraction.
- Add dense layers for car classification.
- Compile the model with appropriate settings.

03

Training and Evaluation:

- Train the model on the dataset.
- Evaluate performance on validation and test sets.
- Save the best weights with ModelCheckpoint.

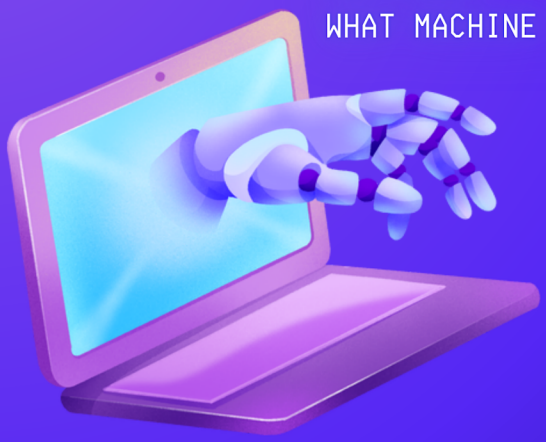
04

Results and Analysis:

- Display final test set accuracy.
- Visualize training and validation accuracy curves

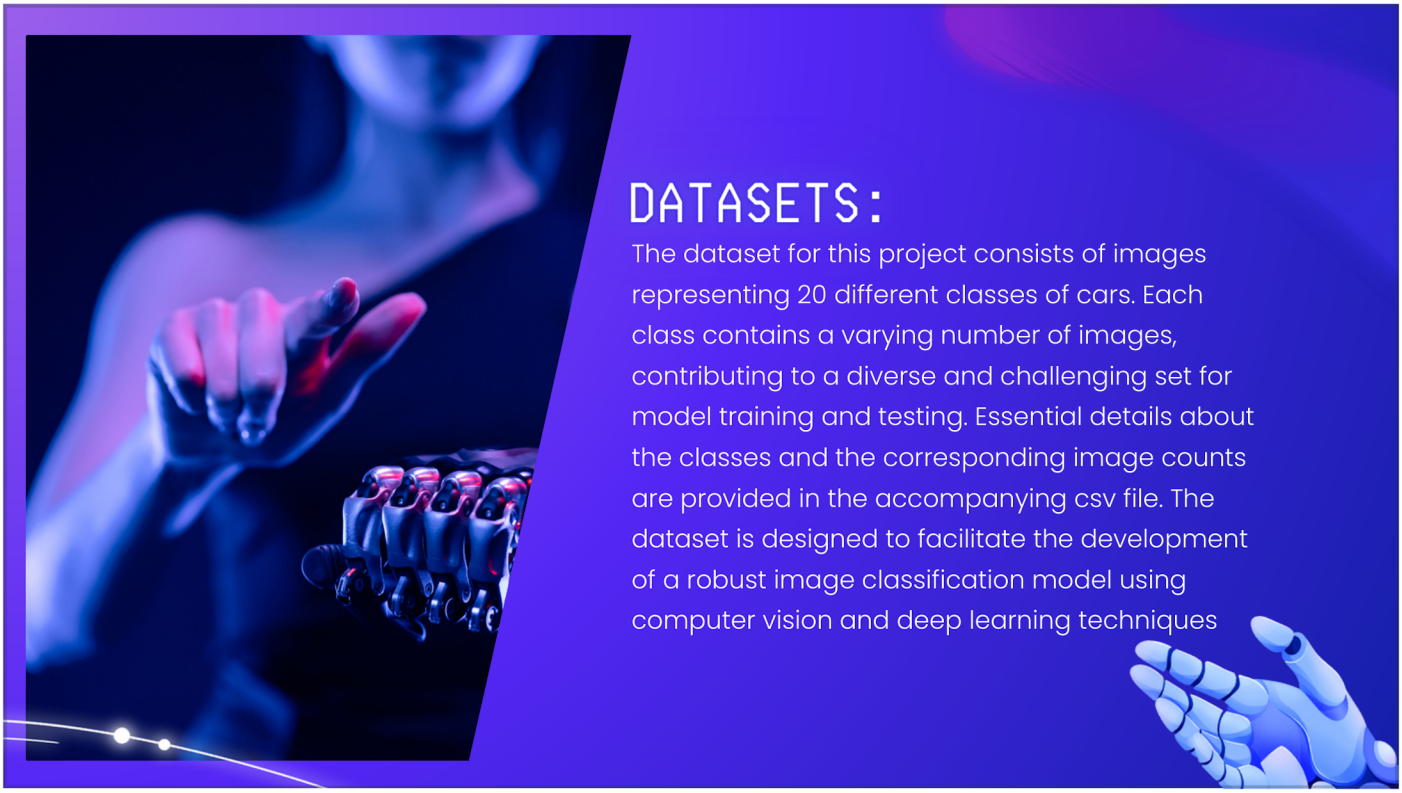
METHOD:

WHAT MACHINE LEARNING TECHNIQUES ARE APPLYING?



Transfer Learning

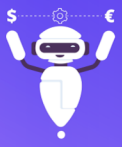
The primary machine learning technique applied in the code is Transfer Learning. It involves using a pre-trained MobileNet model as a feature extractor for image classification. The MobileNet base, pre-trained on a large dataset (ImageNet), is augmented with additional layers. The model is then fine-tuned on a specific dataset, improving its performance on the targeted image classification task.


DATASETS:
The dataset for this project consists of images representing 20 different classes of cars. Each class contains a varying number of images, contributing to a diverse and challenging set for model training and testing. Essential details about the classes and the corresponding image counts are provided in the accompanying csv file. The dataset is designed to facilitate the development of a robust image classification model using computer vision and deep learning techniques

EXPERIMENTS

BASELINE EXPERIMENT: 01

- Train the model using the basic configuration without any specific adjustments.
- Observe the accuracy and loss on the validation set.



DATA AUGMENTATION EXPERIMENT:02

- Introduce data augmentation techniques during training (already implemented in the provided code).
- Observe the impact on model performance, especially in scenarios where the original dataset is limited.

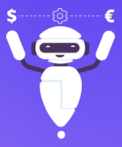
CLASS BALANCING EXPERIMENT: 03

- Analyze and address class imbalance by adjusting class weights or oversampling/undersampling techniques.
- Evaluate the model performance after addressing class imbalances.

EXPERIMENTS

TRANSFER LEARNING EXPERIMENT: 04

- Experiment with different pre-trained models as the base model.
- Explore the performance of models like VGG, ResNet, or EfficientNet.



DATA AUGMENTATION EXPERIMENT:05

- Introduce data augmentation techniques during training (already implemented in the provided code).
- Observe the impact on model performance, especially in scenarios where the original dataset is limited.

FINE-TUNING EXPERIMENT: 06

- Experiment with fine-tuning the pre-trained base model with different layers frozen or trainable.
- Observe the impact on the model's ability to generalize.

EVALUATION

01

- **Test Set Evaluation:**
- The model achieved an accuracy score of 95.7% on the test set, which is a good performance.

Evaluate the model ¶

```
# Evaluate the model on the test set
test_loss, test_accuracy = model.evaluate(test_gen)
print('\n\033[1m' + 'The model had an accuracy score of {}%!!'.format(round(100*test_accuracy,1)) +
'\033[0m')
```

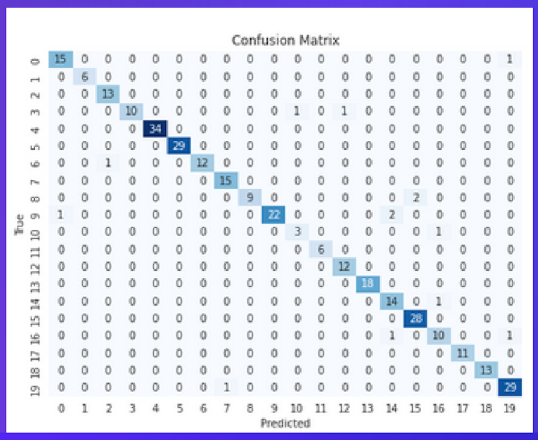
```
6/6 [==============================] - 2s 363ms/step - loss: 0.1496 - accuracy: 0.9567

The model had an accuracy score of 95.7%!!
```

EVALUATION

02

- confusion matrix



EVALUATION

03

- **Classification Report:**

Classification Report:

	precision	recall	f1-score	support
0	0.94	0.94	0.94	16
1	1.00	1.00	1.00	6
2	0.93	1.00	0.96	13
3	1.00	0.83	0.91	12
4	1.00	1.00	1.00	34
5	1.00	1.00	1.00	29
6	1.00	0.92	0.96	13
7	0.94	1.00	0.97	15
8	1.00	0.82	0.90	11
9	1.00	0.88	0.94	25
10	0.75	0.75	0.75	4
11	1.00	1.00	1.00	6
12	0.92	1.00	0.96	12
13	1.00	1.00	1.00	18
14	0.82	0.93	0.87	15
15	0.93	1.00	0.97	28
16	0.83	0.83	0.83	12
17	1.00	1.00	1.00	11
18	1.00	1.00	1.00	13
19	0.94	0.97	0.95	30
accuracy			0.96	323
macro avg	0.95	0.94	0.95	323
weighted avg	0.96	0.96	0.96	323


DISCUSSION: DISCUSS YOUR RESULTS
Accuracy: Achieved an impressive accuracy of 95.7% on the test set.
Classification Report:
Many classes exhibit high precision, recall, and F1-scores.
Some classes show perfect scores, indicating robust performance.
Macro and Weighted Averages:
Macro average: 95%, indicating overall strong performance.
Weighted average: 96%, considering variations in class sizes.

Conclusion

- The project involved building and training a deep learning model for car image classification using TensorFlow and Keras.
- Data augmentation was employed to enhance the model's ability to generalize.
- The model demonstrated good accuracy on the test set, showcasing its effectiveness in classifying car images.


THANK YOU!