Car Make and Model Prediction Using Machine Learning

Team 06 | Big Data Analytics

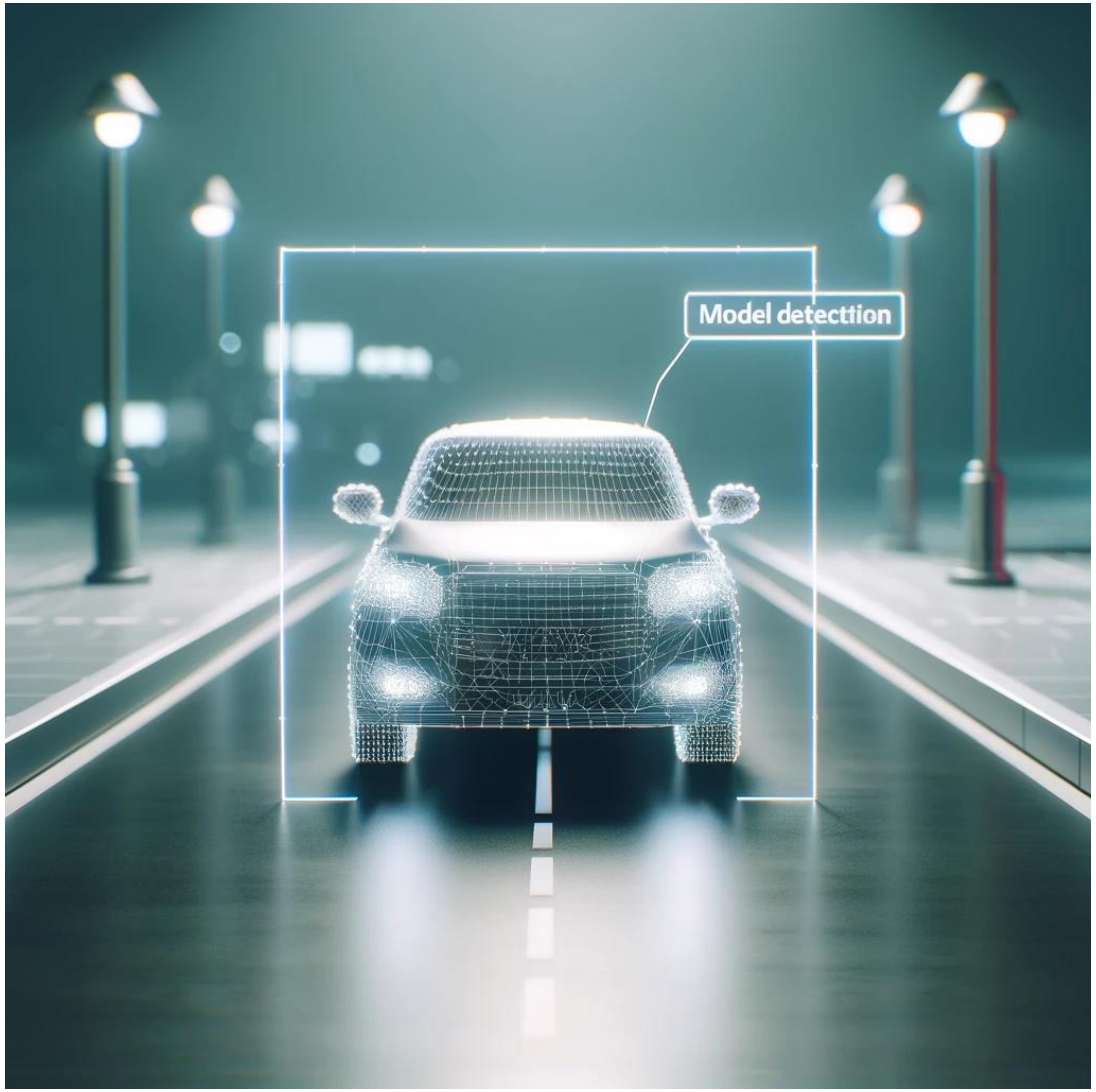
Team Members:

Sai Kiran Belana

Ummea Salma

Ariannah Black

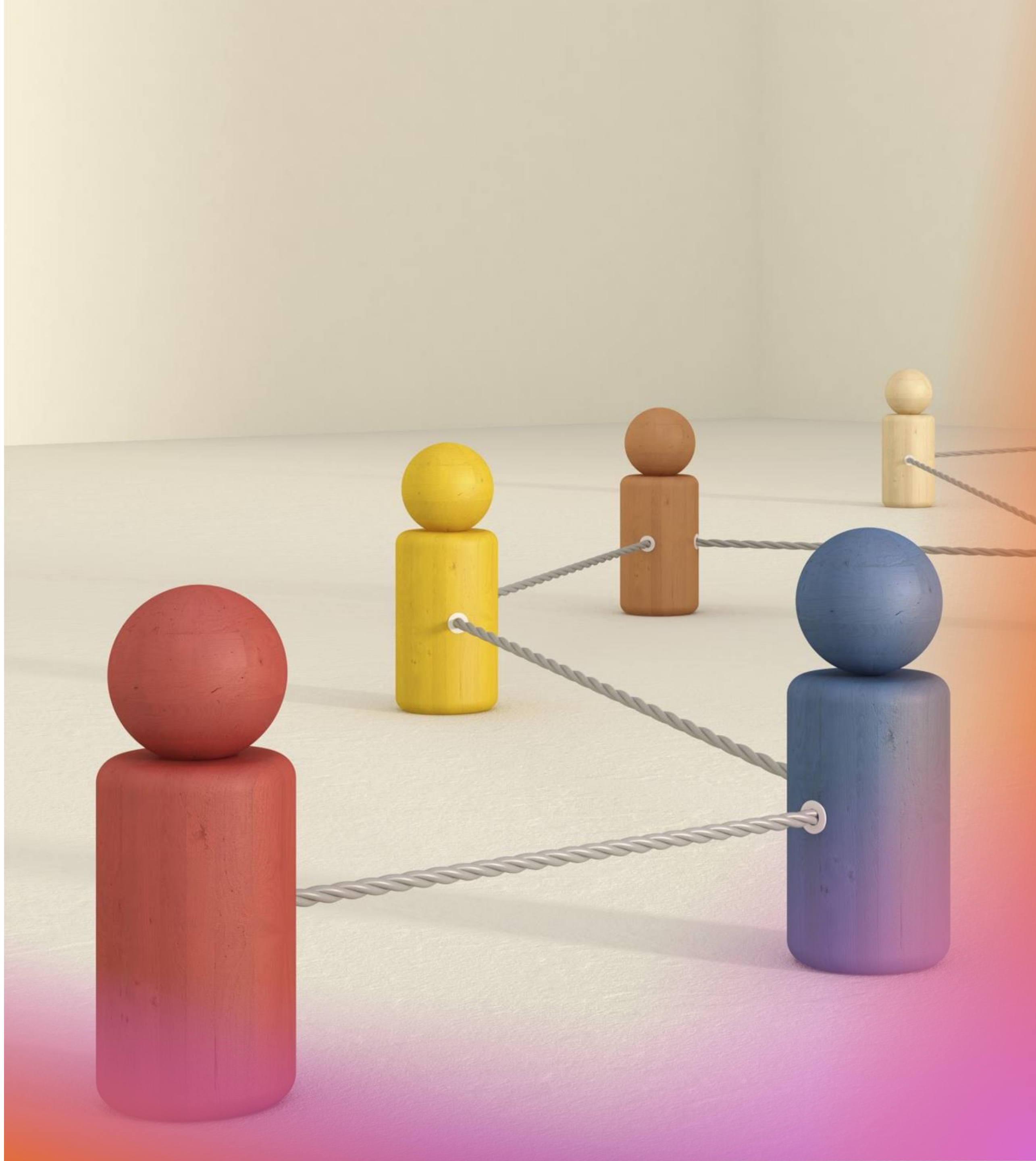
Md Ishtyaq Mahmud



Department of Computer Science and Engineering
University of Connecticut, Storrs

Outline:

- Problem Statement
- Dataset
- Dataset Preparation
- Model and Technique Selection
- Initial Model Evaluation
- Error Analysis
- Methods of Improving the Model
- Conclusion and Future Works



Problem Statement

Developing a usable machine learning model to quickly and accurately identify the car's make and model from an Image.

Project Goals:

Enable enhanced traffic safety, aid in recovery of stolen vehicles, and support authorities in enforcing traffic laws.

Significance:

Improves efficiency and accuracy in vehicle identification across various sectors, including law enforcement and insurance.

DataSet Overview

Dataset:
Stanford Car Dataset

Description:
- Contains 16,185 images.
- Represents 196 different vehicle classes.
- Used for car make and model identification.

Annotations:
- Includes metadata:
  - Bounding box coordinates (x1, y1, x2, y2).
  - Class names and numbers for each image.
- Annotations aid in precise model training.

Dataset Preparation

1. Dividing the Dataset

- Training and testing set with the ratio of 80/20
- Splits the resulting training set into training and validation with ratio of 75/25
- No missing or incomplete data in the dataset

2. Cropping Images:

Images were cropped according to their bounding boxes to focus on the vehicles and reduce noise.

3. Resizing Images:

Images resized to 299x299 pixels to maintain consistency and detail necessary for accurate classification.

4. Data Augmentation:

Applied techniques like random erasing and image transformations to increase dataset diversity and reduce overfitting.

Fig: A number of images produced by a single instance after data augmentation, random eraser not visualized.

Using Transfer Learning Model

Model and Technique Selection:

- Initial Approach: Started with **basic CNN models**; explored feature extraction and augmentation techniques.
- **VGG16 Model**: Started with VGG16 since the dataset is smaller.
- **Xception Model**: Considering the Xception model, after facing challenges with other models like VGG16.
    - Chosen for its efficiency and effectiveness, utilizing depthwise separable convolutions for enhanced learning.
- **Fine-tuning Strategy**: To prevent overfitting and ensure stable performance and increase accurate predictions, the models included additional layers for regularization.

Initial Model Evaluation:

Evaluation of Previous Model: VGG16

- Utilized VGG16 model for feature extraction, excluding top layers, and designed a new model for fast feature extraction without data augmentation.
- **Model Compilation:**
  - Loss function: Sparse categorical cross-entropy
  - Optimizer: RMSprop
  - Metrics: Accuracy for training
- **Training Results:**
  - Higher training and validation accuracies observed.
  - However, both training and validation losses are significantly higher.
  - VGG16 model couldn't predict images properly.
- Fine tuning the model by freezing last four layers increases the accuracy but we got higher losses leading to an unusable model.

```
Model: "vgg16"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_2 (InputLayer)        [(None, 224, 224, 3)]     0

 block1_conv1 (Conv2D)       (None, 224, 224, 64)      1792

 block1_conv2 (Conv2D)       (None, 224, 224, 64)      36928

 block1_pool (MaxPooling2D)  (None, 112, 112, 64)      0

 block2_conv1 (Conv2D)       (None, 112, 112, 128)     73856

 block2_conv2 (Conv2D)       (None, 112, 112, 128)     147584

 block2_pool (MaxPooling2D)  (None, 56, 56, 128)       0

 block3_conv1 (Conv2D)       (None, 56, 56, 256)       295168

 block3_conv2 (Conv2D)       (None, 56, 56, 256)       590080

 block3_conv3 (Conv2D)       (None, 56, 56, 256)       590080

 block3_pool (MaxPooling2D)  (None, 28, 28, 256)       0

 block4_conv1 (Conv2D)       (None, 28, 28, 512)       1180160

 block4_conv2 (Conv2D)       (None, 28, 28, 512)       2359808

 block4_conv3 (Conv2D)       (None, 28, 28, 512)       2359808

 block4_pool (MaxPooling2D)  (None, 14, 14, 512)       0

 block5_conv1 (Conv2D)       (None, 14, 14, 512)       2359808

 block5_conv2 (Conv2D)       (None, 14, 14, 512)       2359808

 block5_conv3 (Conv2D)       (None, 14, 14, 512)       2359808

 block5_pool (MaxPooling2D)  (None, 7, 7, 512)         0
```

Xception Model Evaluation:

Chosen for its efficiency and effectiveness, utilizing depthwise separable convolutions for enhanced learning.

- **Techniques applied for Data Augmentation:**
  - Image rotations, Width and height shifts
  - Horizontal flip
  - Zoom range, Shear range, Brightness ranges
- **K-Fold Cross-Validation:**
  - Used to assess the performance and generalization ability of the machine learning model.
  - Split the dataset into K = 2 subsets (or folds).
  - Trained the model K times, each time using a different subset as the validation set and the remaining subsets for training.
- **Training:**
  - Achieved high validation accuracy of 99.50% and low validation loss of 0.0150
- **Testing:**
  - Achieved approximately **94.6%** test accuracy.

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 xception (Functional)       (None, 10, 10, 2048)      20861480

 global_average_pooling2d (  (None, 2048)              0
 GlobalAveragePooling2D)

 dense (Dense)               (None, 2048)              4196352

 dropout (Dropout)           (None, 2048)              0

 dense_1 (Dense)             (None, 196)               401604

=================================================================
Total params: 25459436 (97.12 MB)
Trainable params: 25404908 (96.91 MB)
Non-trainable params: 54528 (213.00 KB)
_________________________________________________________________
```

Model Evaluation:

1. **VGG16 with Fast Feature Extraction**
   - Achieved a high training accuracy of 97.02%.
   - However, validation accuracy was significantly lower at 66.02%.
   - Higher validation loss compared to training loss indicated overfitting.

2. VGG16 with Feature Extraction and Data Augmentation

- Implemented feature extraction with data augmentations to address overfitting.
- Training accuracy improved to 93.71% and validation accuracy to 81.74%.
- However, overfitting persisted, leading to higher losses for validation data (more than 100%) and incorrect class predictions.

3. Xception with K-Fold Cross-Validation

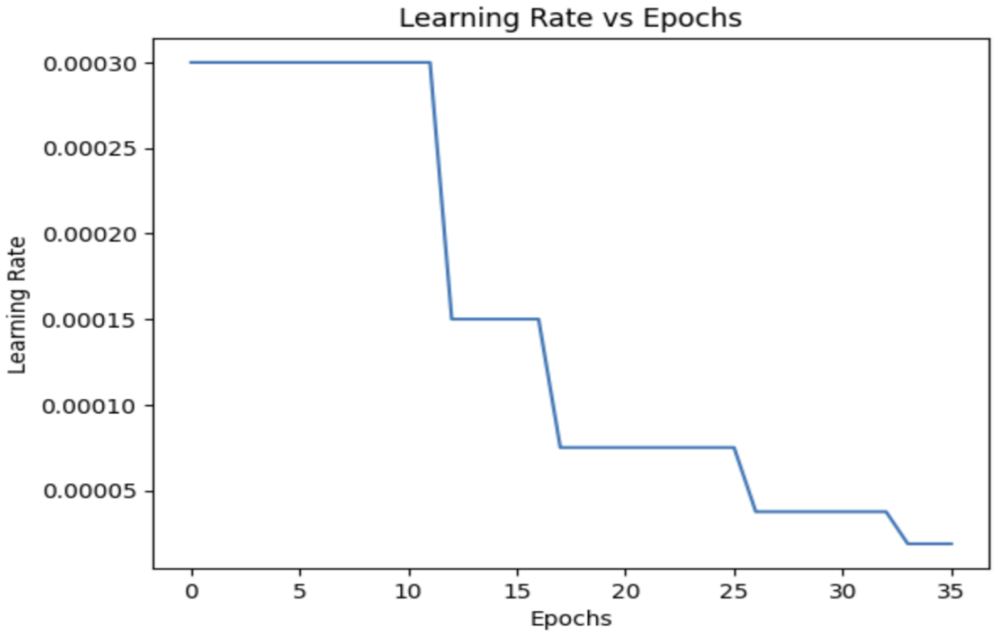
- Utilized Xception architecture with K-Fold Cross-Validation.
- Achieved significantly higher validation accuracy compared to VGG16.
- Demonstrated the best performance among the models.

Test Performance:

- Predicted classes of test images with an average accuracy of approximately **94.60%**.
- Out of **3027** testing images, **2853** were correctly predicted (unseen data).

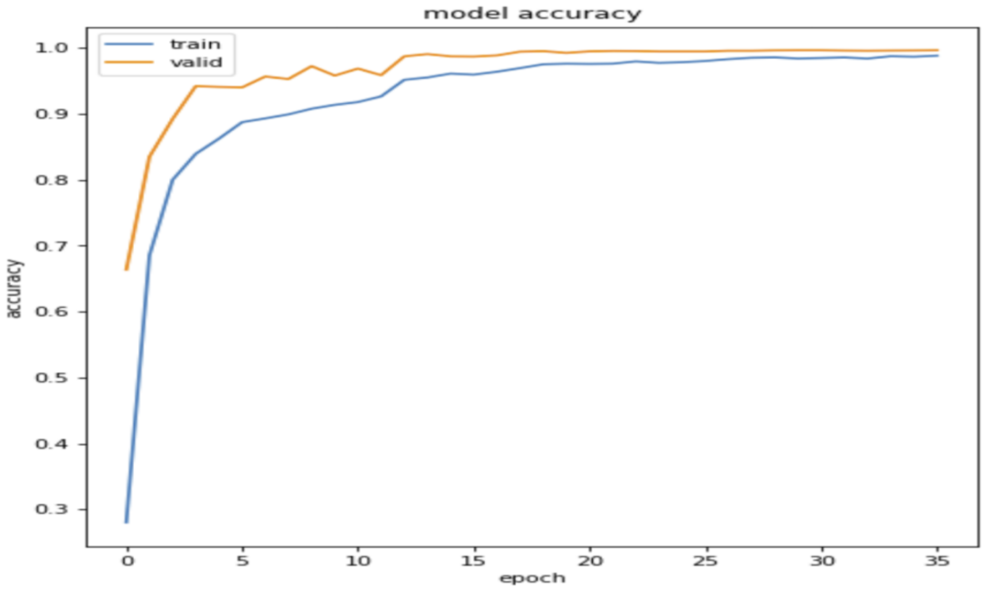
Methods of Improving the Model: Xception and VGG16

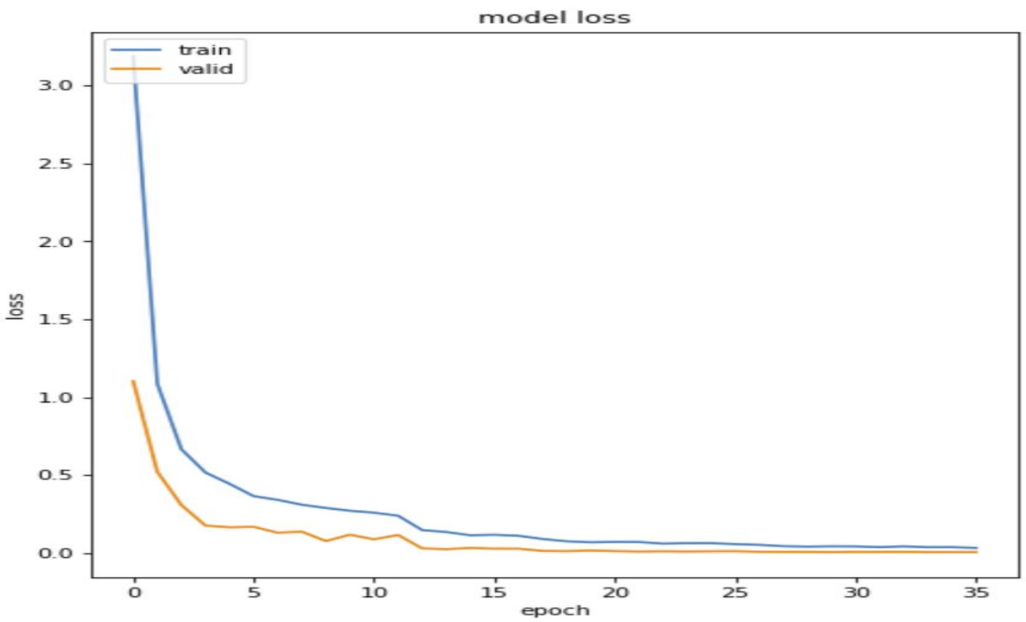
1. For reducing overfitting, **Early Stopping** was introduced to halt training once model performance stops improving on the validation dataset.
2. Modified the **Number of Epochs** to adjust overfitting and underfitting.
3. Added **Custom Layers** such as GlobalAveragePooling, Dropouts, and extra dense layers converging to # of classes using Softmax.
4. Added **Optimizers** such as Nadam optimization function helped to reduce the overall loss and improve the accuracy throughout training and validation.
5. Adjusted **Learning Rate** to decrease epoch by epoch.



Model Evaluation - Xception

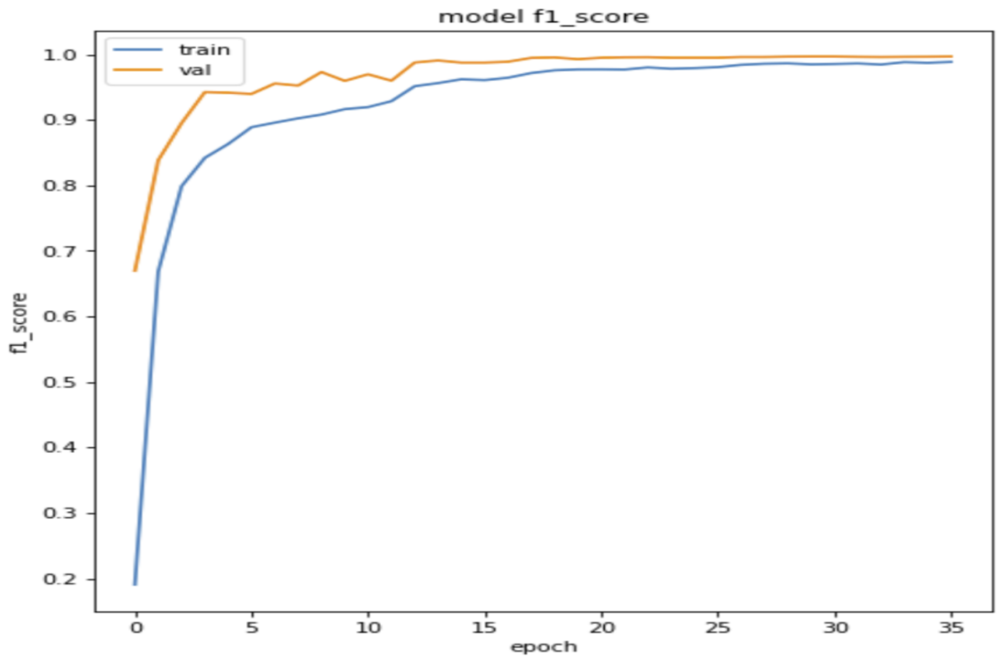
- Model accuracy indicates the percentage of correct predictions made by the model, while loss represents how well the model is performing based on the difference between predicted and actual values.
- Finally got **94.60%** test accuracy using Xception.
- Training loss: **3.16%**
- Training Accuracy: **98.80%**

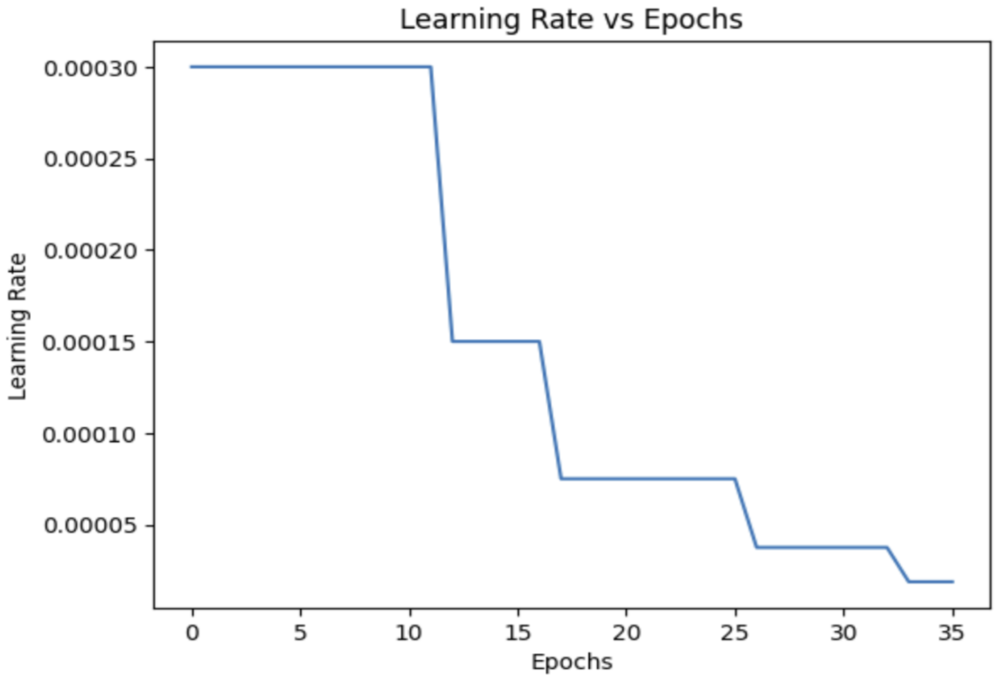




Model Evaluation - Xception

- The F1 score, a metric that combines precision and recall into a single value, increased substantially after around 10 or 15 epochs. A high F1 score indicates a balance between precision and recall, showcasing the model's ability to make accurate and comprehensive predictions.
- f1_m: 98.83
- Validation Loss: 0.58%
- Validation Accuracy: 99.69%





Results on Test DataSet

2853 images are correct out of 3027 testing images - unseen

What went wrong ?

Upon Manual Inspection, some of the images are having wrong class names and some are useless.

Proof from Internet:

Cleanlab
12,562 followers
11mo •

+ Follow

❗Stanford Cars (cars196) Dataset contains Many Errors❗

Here we consider the Stanford Cars dataset (cars196), originally used in a research paper with over 1000 citations. This dataset contains labeled images of 196 types of cars such as “BMW 3 Series Sedan 2012”, and “Ford F-150 Regular Cab 2012”. We discovered tons of issues and outliers in this famous computer vision dataset just by quickly running it through Cleanlab Studio.

Stanford Cars (cars196) contains many Fine-Grained Errors

discussion

Hey Redditors,

I know the cars196 dataset is nothing new, but I wanted to share some label errors and outliers that I found within it.

It’s interesting to note that the primary goal of the original paper that curated/used this dataset was “fine-grained categorization” meaning discerning the differences between something like a Chevrolet Cargo Van and a GMC Cargo Van. I found numerous examples of images that exhibit **very nuanced** mislabelling which is directly counterintuitive to the task they sought to research.

Here are a few examples of nuanced label errors that I found:

- Audi TT RS Coupe labeled as an Audi TT Hatchback
- Audi S5 Convertible labeled as an Audi RS4
- Jeep Grand Cherokee labeled as a Dodge Durango

I also found examples of outliers and generally ambiguous images:

- multiple cars in one image
- top-down style images
- vehicles that didn't belong to any classes.

I found these issues to be pretty interesting, yet I wasn't surprised. It's pretty well known that many common ML datasets exhibit thousands of errors.

Conclusion and Future Work

Issues with Stanford Cars:

- Some images with classes mislabeled, belonging to classes other than their assigned ones.
- Ambiguous images, which include multiple cars within a single image.

Limitations:

- Nuanced, but significant issues with the dataset.
- A lack of time for model development, training, and research.
- A lack of computing resources and funding.

Future Work and Project Extensions:

- Re-clean the data.
- Add more images to each class.
- Need to do a detailed analysis on VGG16 to improve its accuracy and reduce the loss occurring due to overfitting.
- Create a usable API for predicting the results on new unseen data using the Xception Model.
- Run with real-time video feed and predict the results on the go.

THANKS!!

QUESTIONS?