Binary Image Classification Using Custom CNN and Transfer Learning

Comparison of Custom CNN and VGG16 Models

YAMAN IBRAHEEM
OLA MAHMOUD
DANA JAMAL
AYA ADEEB

next slide →

02

Introduction

In this project, we aim to classify images from the CIFAR-10 dataset into two categories: Airplane and Car. The CIFAR-10 dataset is widely used for image classification tasks, and we have selected these two classes to focus on a binary classification problem.

The primary objectives of this project are:

1. *To develop a custom Convolutional Neural Network (CNN) for classifying the images.*
2. *To apply transfer learning using the VGG16 model pretrained on the ImageNet dataset.*
3. *To compare the performance of the custom CNN and the transfer learning model in terms of accuracy, precision, recall, and other metrics.*

   This project explores the trade-offs between building a model from scratch and leveraging a pretrained model, providing insights into their respective advantages and limitations. By analyzing the results, we aim to determine the suitability of each approach for this binary classification task.

next slide →

03

Dataset Description

For this project, we used the CIFAR-10 dataset, a well-known image dataset consisting of 60,000 32x32 color images across 10 classes. Each class contains 6,000 images, making it a balanced and diverse dataset.

Type: Image Classification (Binary)

Classes:

- Airplane (Label: 0)
- Car (Label: 1)

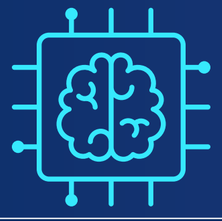
Size: 60,000 labeled images.

- *Training Set: Contains approximately 10,000 images used to train the models.*
- *Validation Set: Includes around 2,500 images for hyperparameter tuning and model validation during training.*
- *Testing Set: Comprises around 2,500 images to evaluate the final model performance.*

next slide →

04

preprocessing techniques



FILTEREDESIZING

Filtered the CIFAR-10 dataset for the "Airplane" and "Car" classes.

NORMALIZED

Normalized pixel values to scale images between 0 and 1 for efficient training.

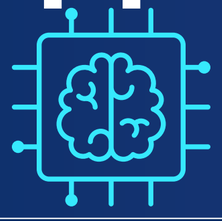
SPLIT

- *Training: 80% of the filtered data.*
- *Validation: 10% for hyperparameter tuning.*
- *Test: 10% for final evaluation.*

next slide →

05

preprocessing techniques Applied Data Augmentation



ROTATION

Randomly rotated images by up to 20 degrees.

FLIPPING

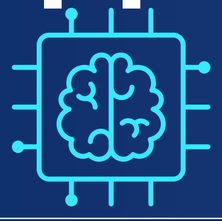
Applied horizontal flipping.

ZOOMING

Zoomed images within a range of 20%.

next slide →

preprocessing techniques Applied Data Augmentation



SHIFTING

Shifted images horizontally and vertically by 20%.

PURPOSE

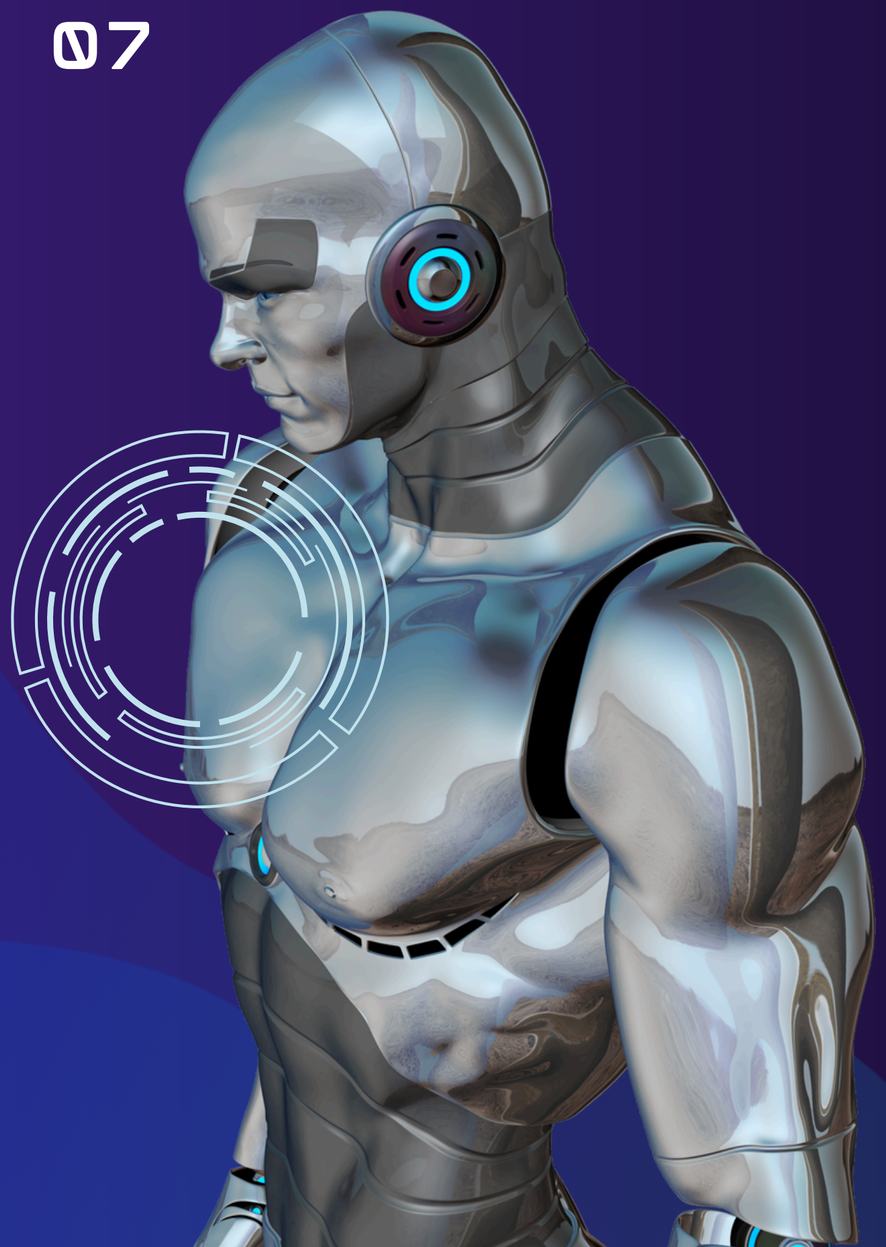
Enhance model generalization by simulating real-world variations.

IMPLEMENTATION

Used ImageDataGenerator from TensorFlow/Keras.

next slide →

07

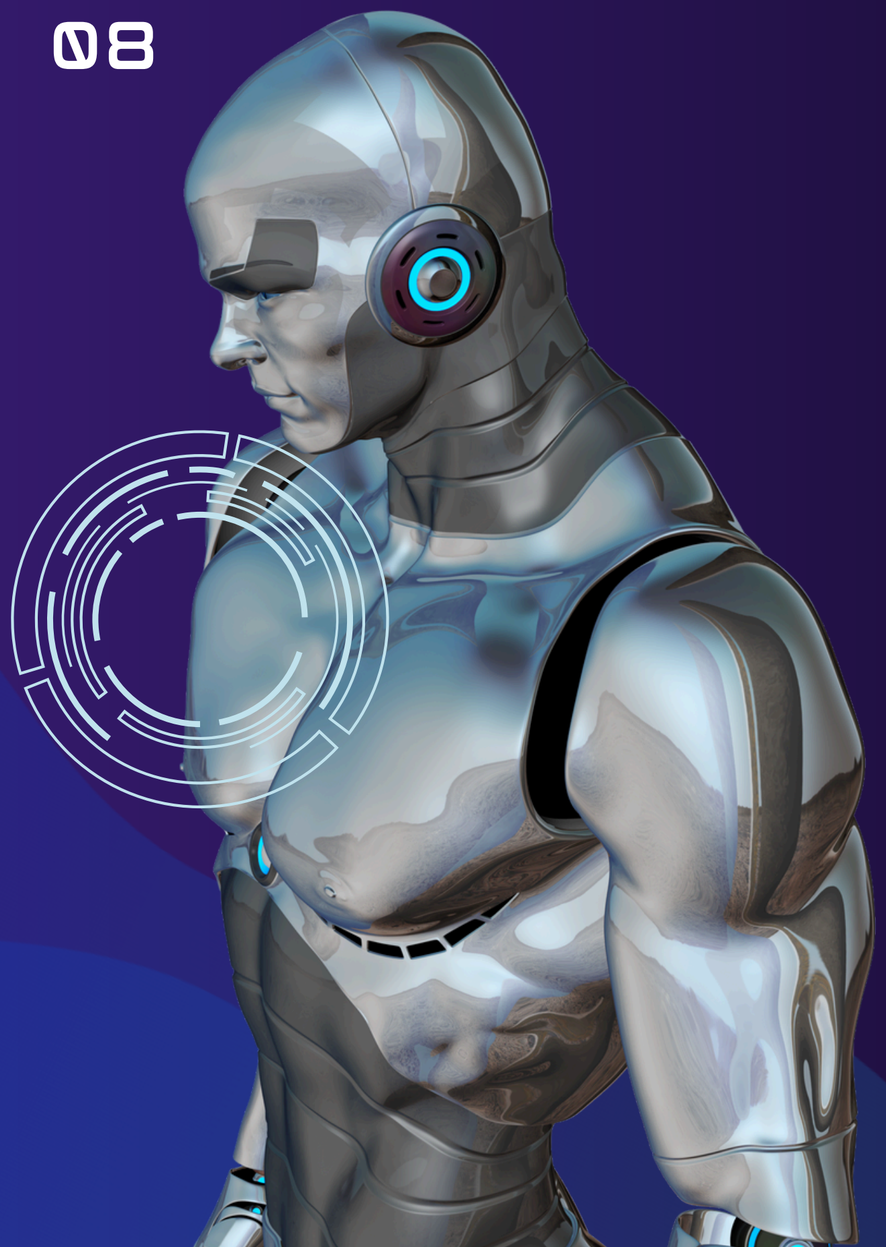


Model Architecture (Custom CNN-Layer)

- *3 convolutional layers with ReLU activation, batch normalization, and dropout.*
- *Max-pooling after each convolutional layer.*
- *Dense layers for classification.*

next slide →

08

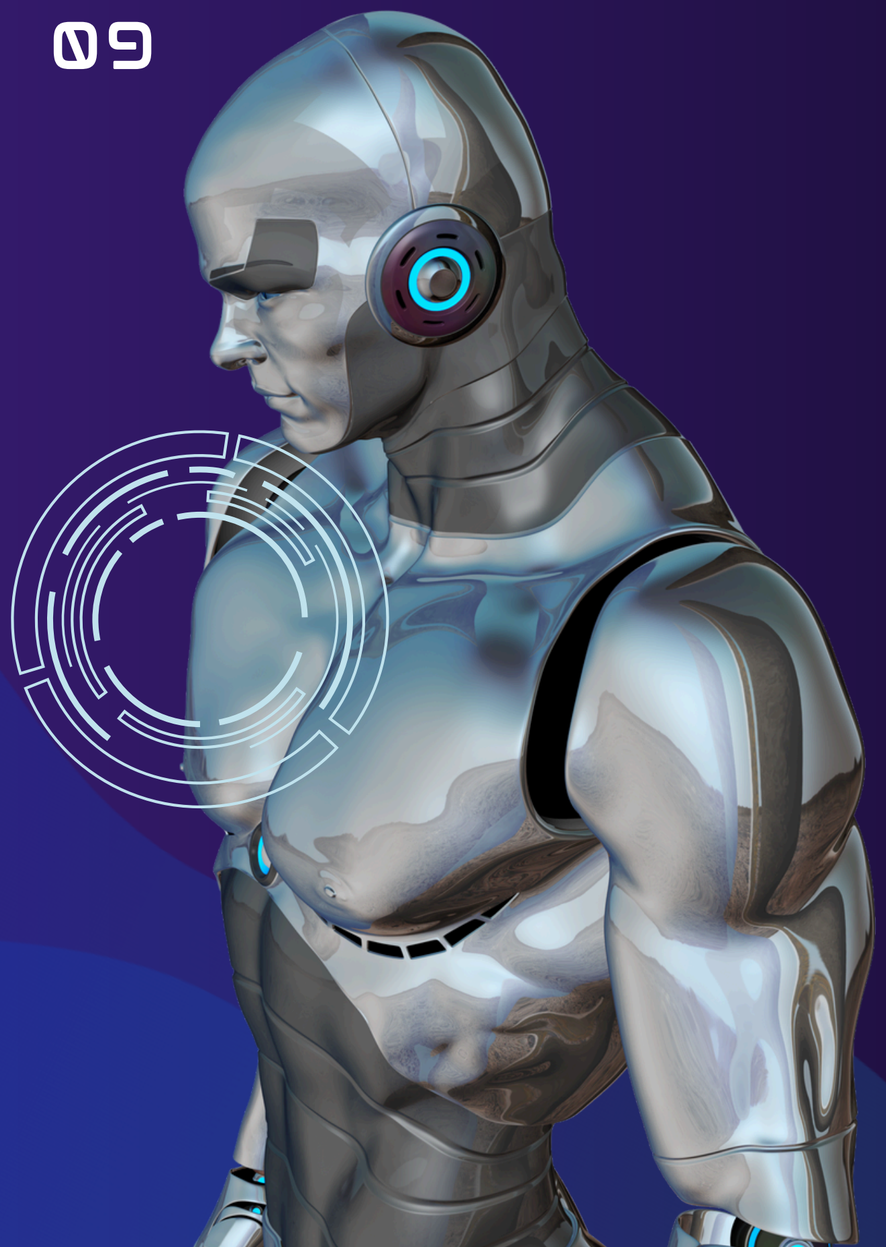


Model Architecture (Custom CNN-Layer)

- *Layers:*
  - *Conv2D (32 filters, 3x3) + ReLU + BatchNorm + MaxPooling + Dropout (20%)*
  - *Conv2D (64 filters, 3x3) + ReLU + BatchNorm + MaxPooling + Dropout (30%)*
  - *Conv2D (128 filters, 3x3) + ReLU + BatchNorm + MaxPooling + Dropout (40%)*
  - *Dense (128 units) + ReLU + Dropout (50%)*
  - *Dense (1 unit, Sigmoid activation)*

next slide →

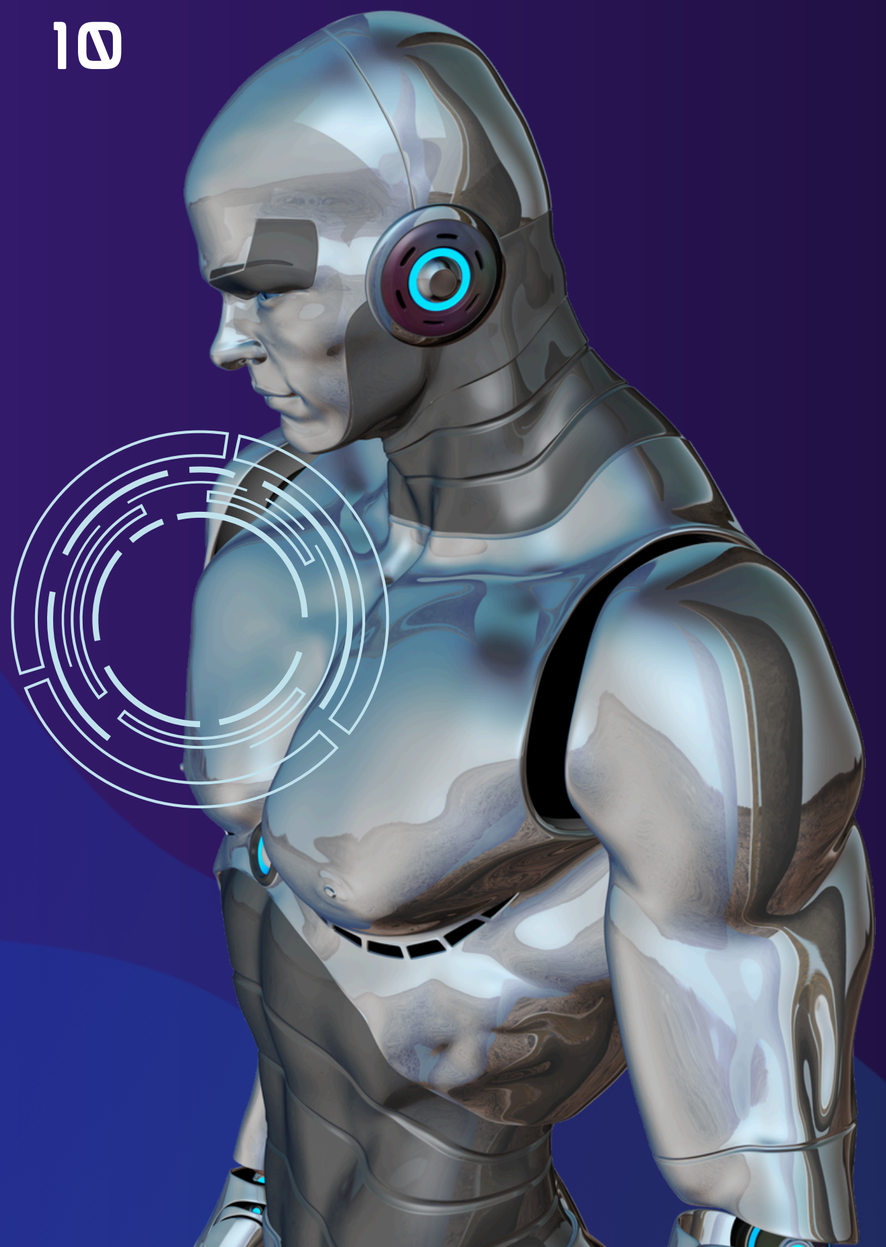
09



Model Architecture (Custom CNN-Parameter)

- Parameters:
- Activation: ReLU (hidden layers), Sigmoid (output layer).
- Optimizer: Adam.
- Loss Function: Binary Crossentropy.

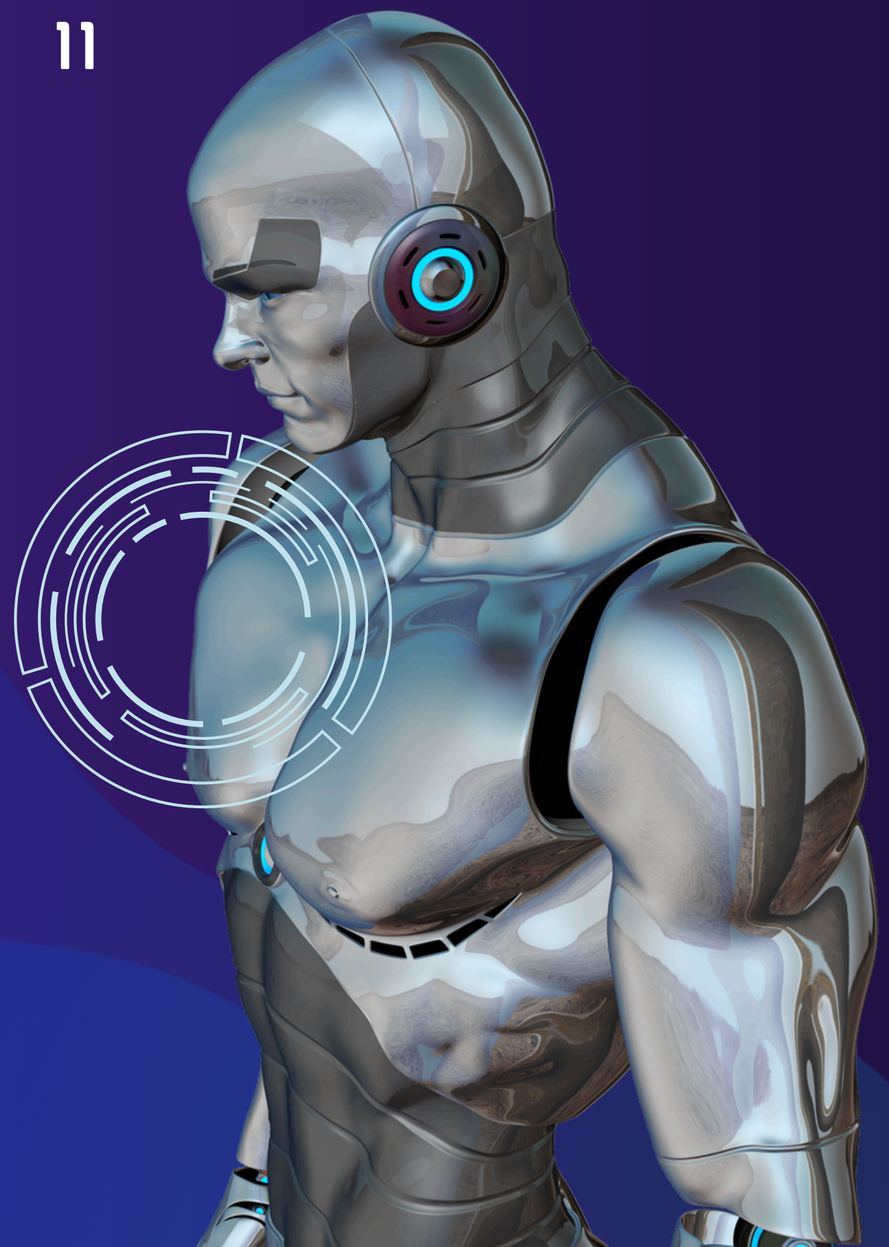
next slide →



Model Architecture (VGG16 Transfer Learning)

- *Base Model: VGG16 pretrained on ImageNet (frozen weights).*
- *Additional Layers:*
  a. *Flatten layer*
  b. *Dense (128 units) + ReLU + Dropout (50%)*
  c. *Dense (1 unit, Sigmoid activation)*

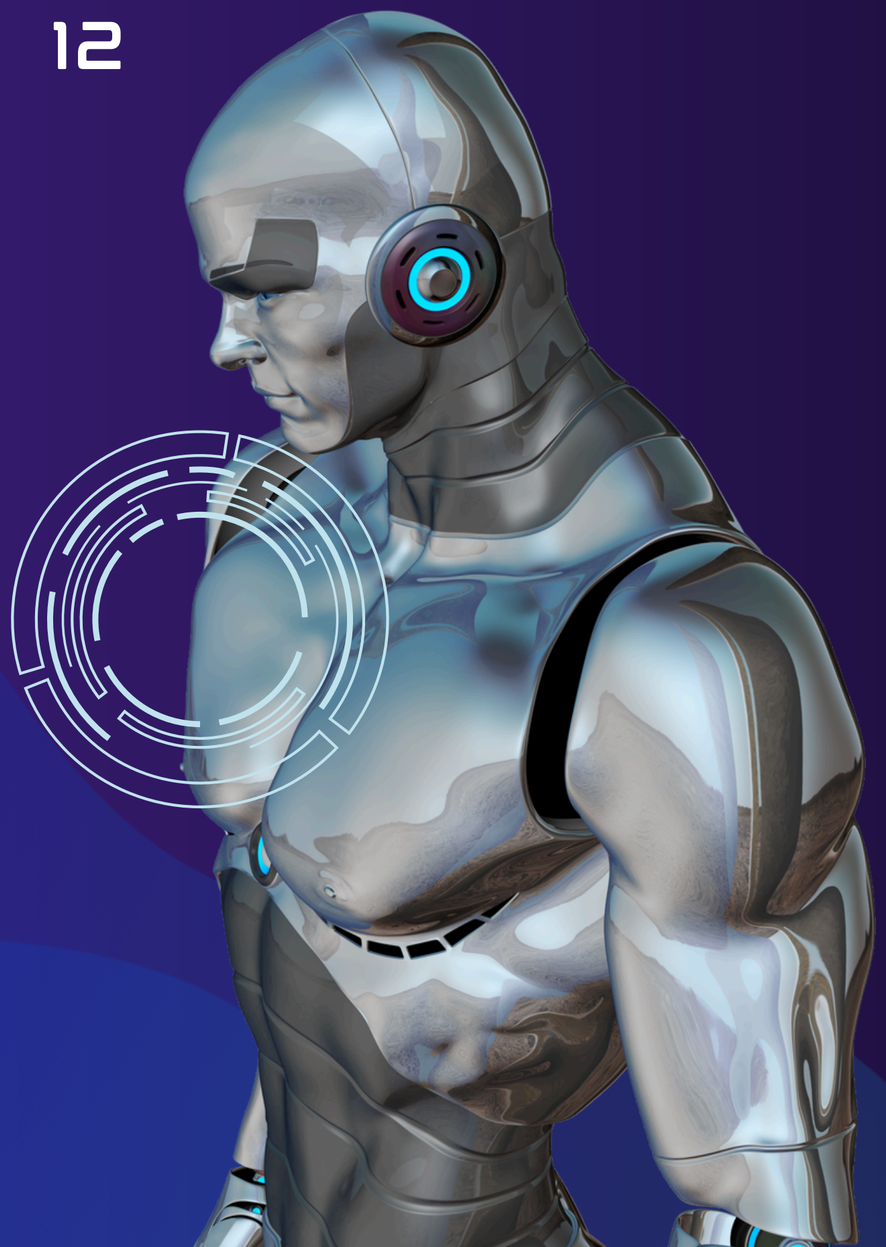
next slide →



ransfer Learning Model (VGG16)

- *Architecture:*
- *Pretrained VGG16 model as the base (ImageNet weights).*
- *Custom dense layers for classification.*
- *Modifications:*
- *Freezed VGG16 layers.*
- *Added dropout and dense layers for binary classification.*

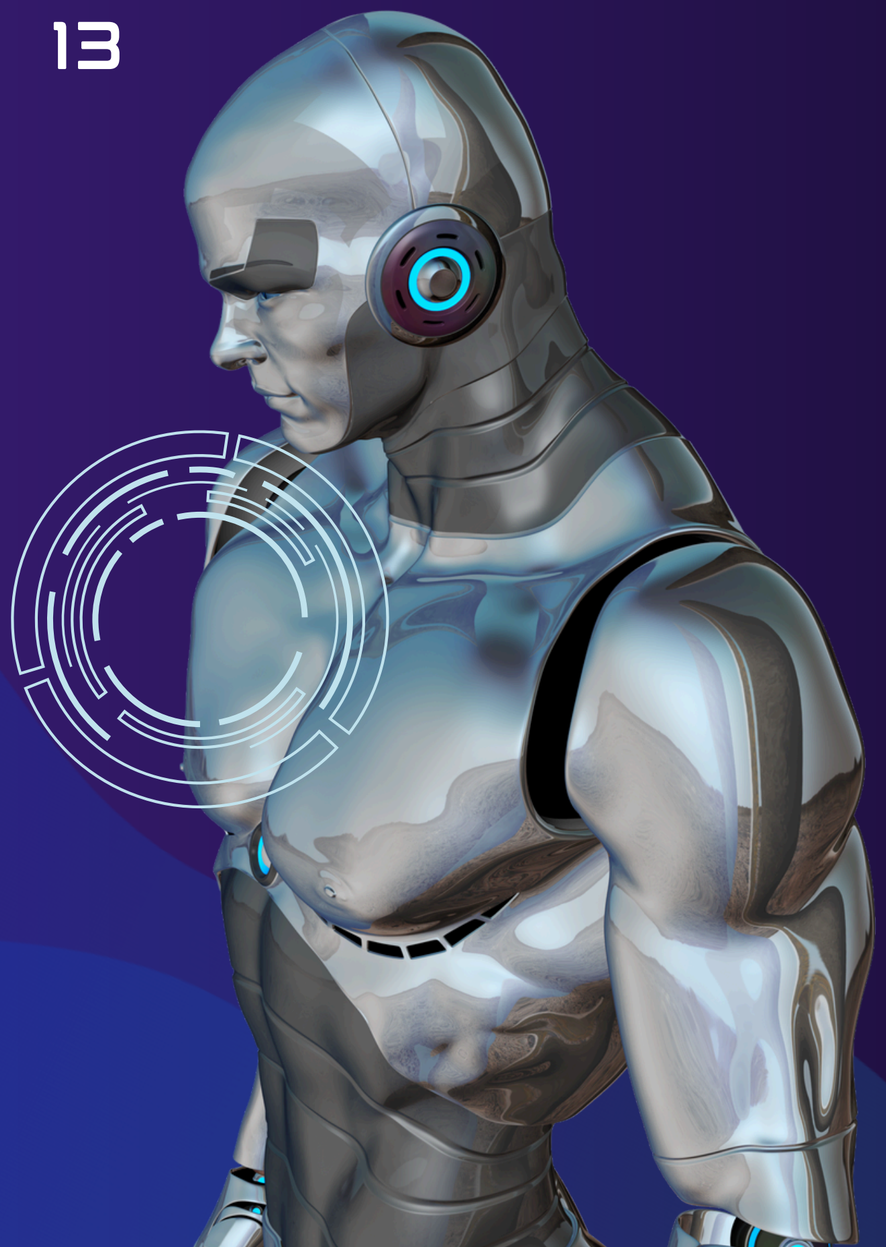
next slide →



Data Augmentation

- 3Rotation: 20 degrees
- Width/Height shift: 20%
- Horizontal flip: Enabled
- Zoom range: 20%

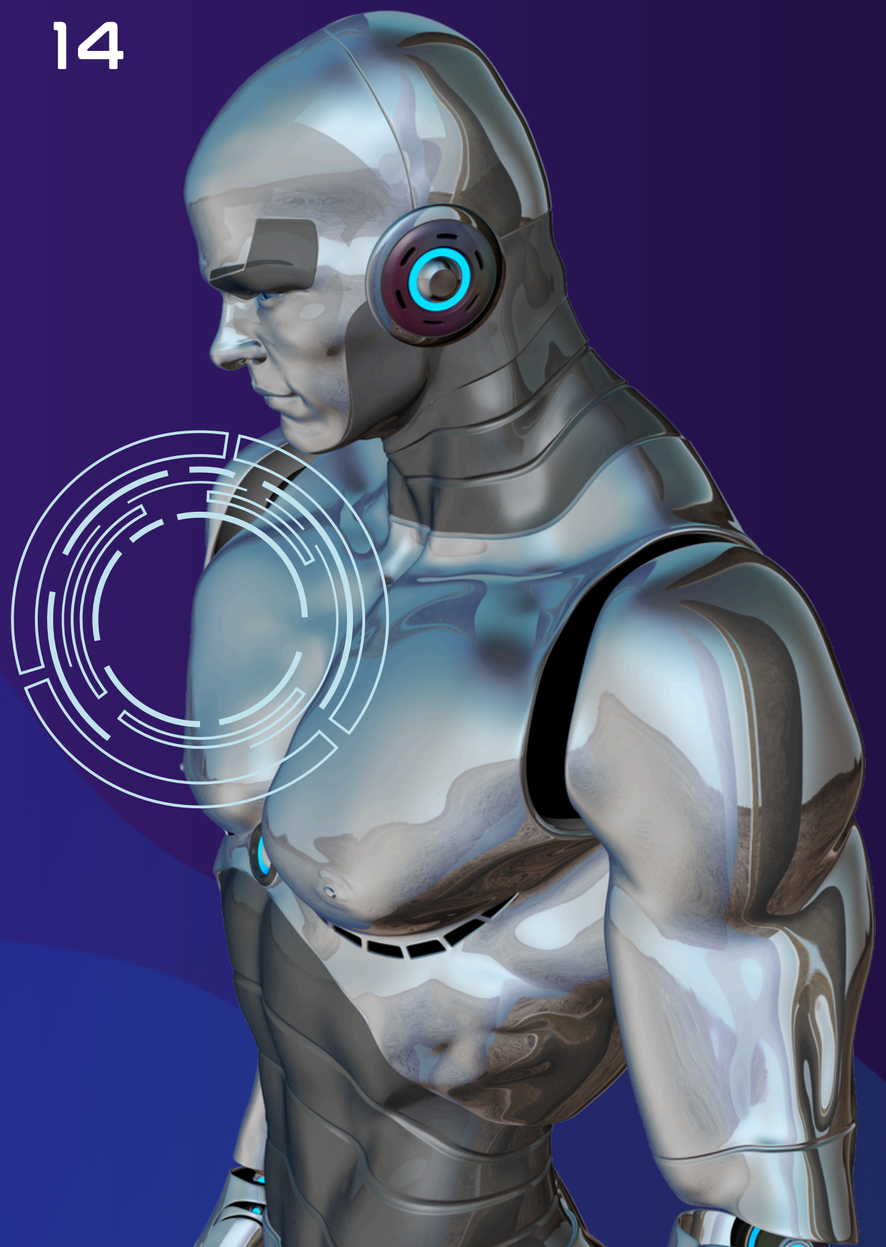
next slide →



Training Configuration

- Optimizer: Adam
- Loss Function: Binary Crossentropy
- Metrics: Accuracy
- Learning Rate Decay: Exponential (0.95 per epoch)
- Early Stopping: Patience of 10 epochs
- Model Checkpoint: Save best model based on validation loss.

next slide →

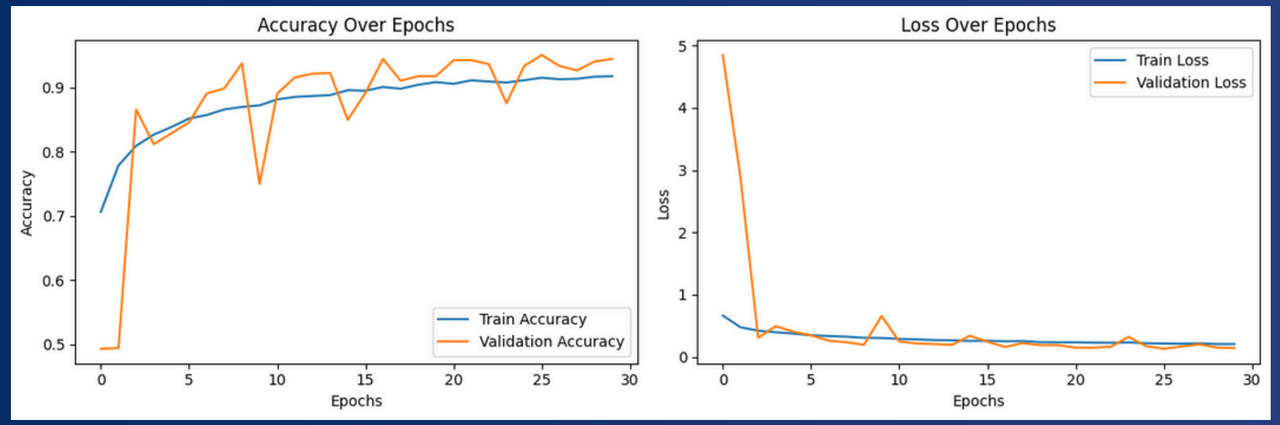


Training Configuration

- Training Parameters:
- Batch Size: 64
- Epochs: 30
- Learning Rate Scheduler: Exponentially decaying.
- Callbacks: EarlyStopping, ModelCheckpoint.

next slide →

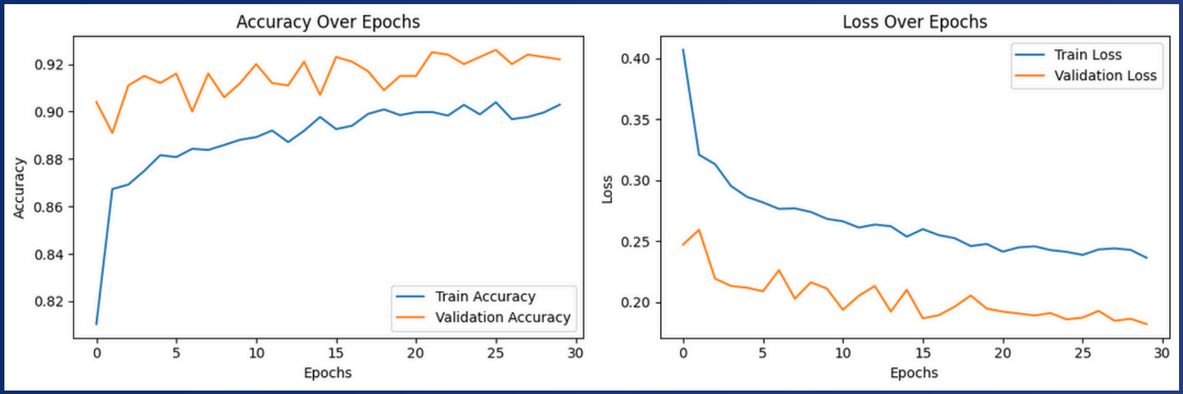
Training Results (Custom CNN)



- *Observations:*
  - *Gradual improvement in accuracy.*
  - *Early stopping to prevent overfitting*

next slide →

Training Results (VGG16 Transfer Learning)



- Observations:
- Faster convergence due to pretrained weights.
- Higher validation accuracy compared to custom CNN.

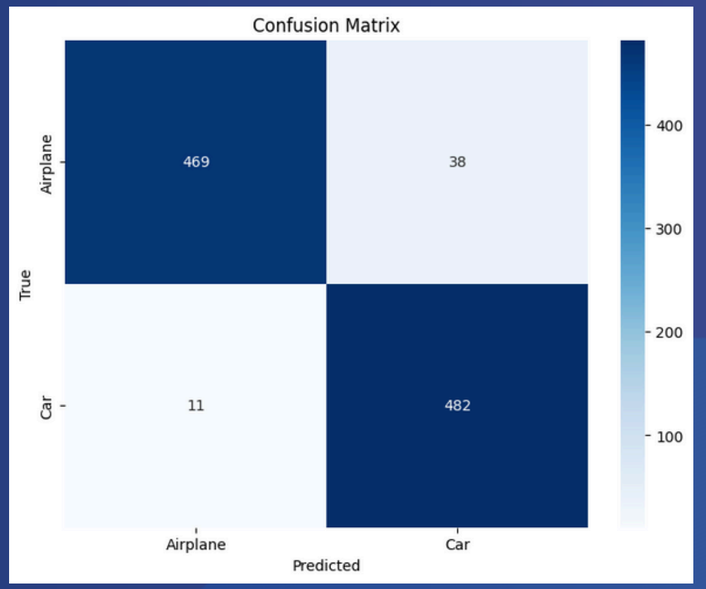
next slide →

Final Evaluation Metrics

- *Custom CNN:*
  - *Test Loss: 0.1234*
  - *Test Accuracy: 0.9510*
- *VGG16 Transfer Learning:*
  - *Test Loss: 0.1882*
  - *Test Accuracy: 0.9260*

Confusion Matrix (Custom CNN)

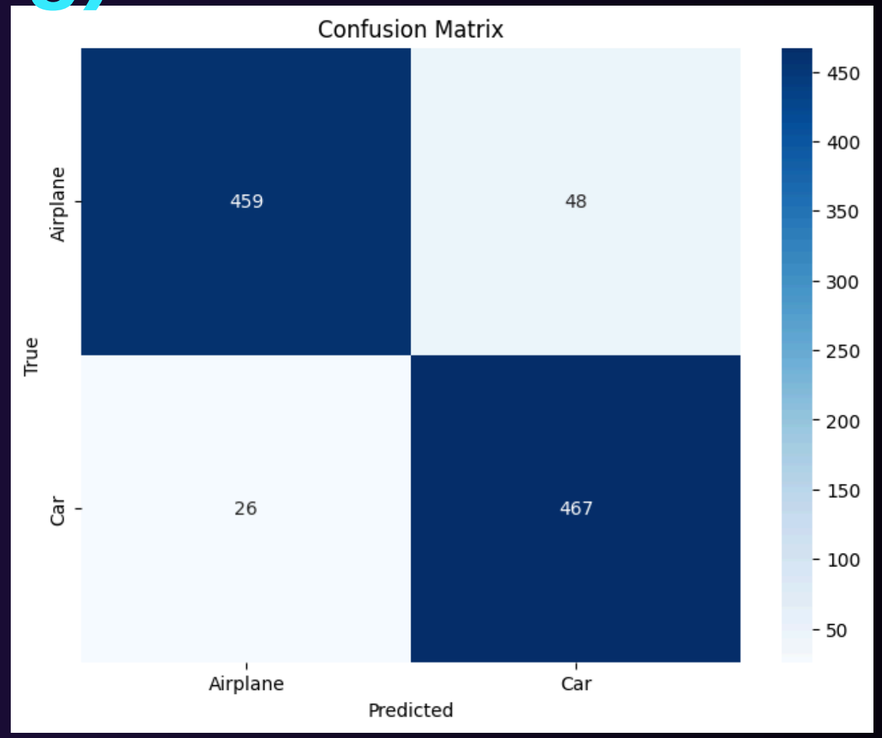
Metrics: Precision, Recall, F1-score.



next slide →

Confusion Matrix (VGG16 Transfer Learning)

Metrics: Precision, Recall, F1-score.



next slide →

Results

Classification Report:

	precision	recall	f1-score	support
Airplane	0.98	0.93	0.95	507
Car	0.93	0.98	0.95	493
accuracy			0.95	1000
macro avg	0.95	0.95	0.95	1000
weighted avg	0.95	0.95	0.95	1000

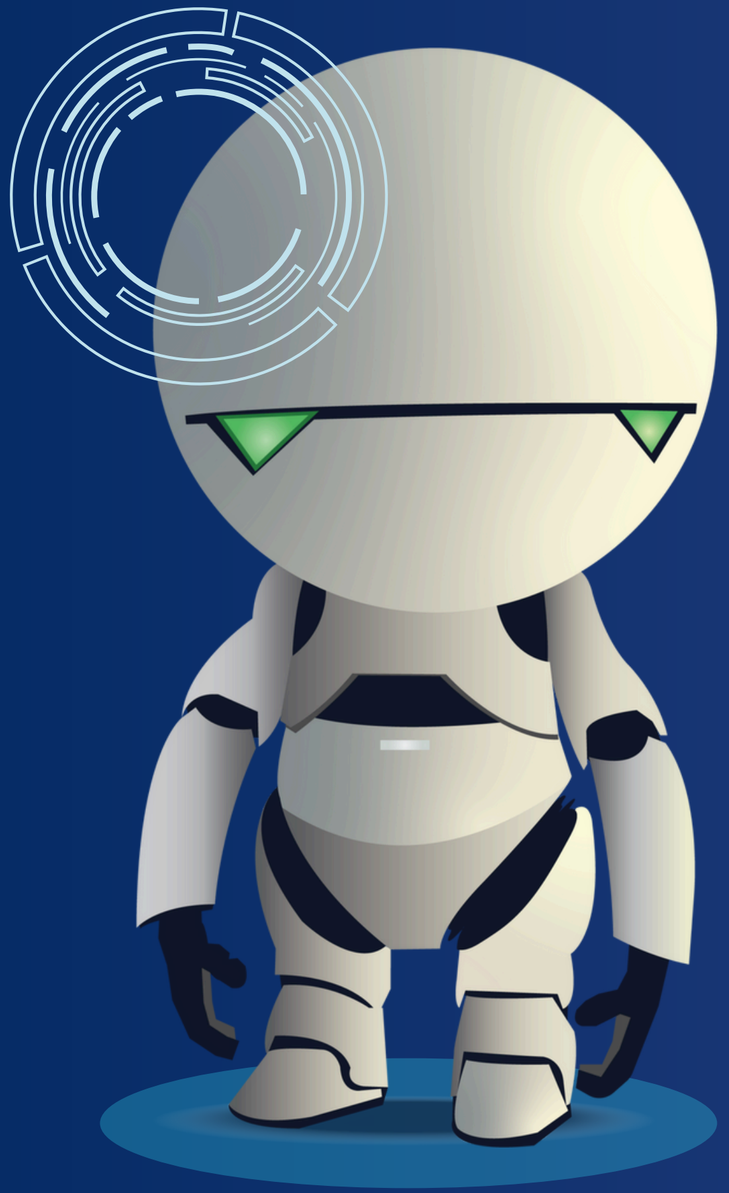
Custom CNN

Classification Report:

	precision	recall	f1-score	support
Airplane	0.95	0.91	0.93	507
Car	0.91	0.95	0.93	493
accuracy			0.93	1000
macro avg	0.93	0.93	0.93	1000
weighted avg	0.93	0.93	0.93	1000

VGG16 Transfer Learning

next slide →



Observations

- *Custom CNN:*
  - *Lightweight model.*
  - *Lower accuracy compared to VGG16.*
- *VGG16 Transfer Learning:*
  - *Higher accuracy due to pretrained features.*
  - *Computationally intensive.*

next slide →

Challenges

- *Imbalanced dataset.*
- *Limited image size and resolution.*
- *Computational cost for transfer learning.*

next slide →