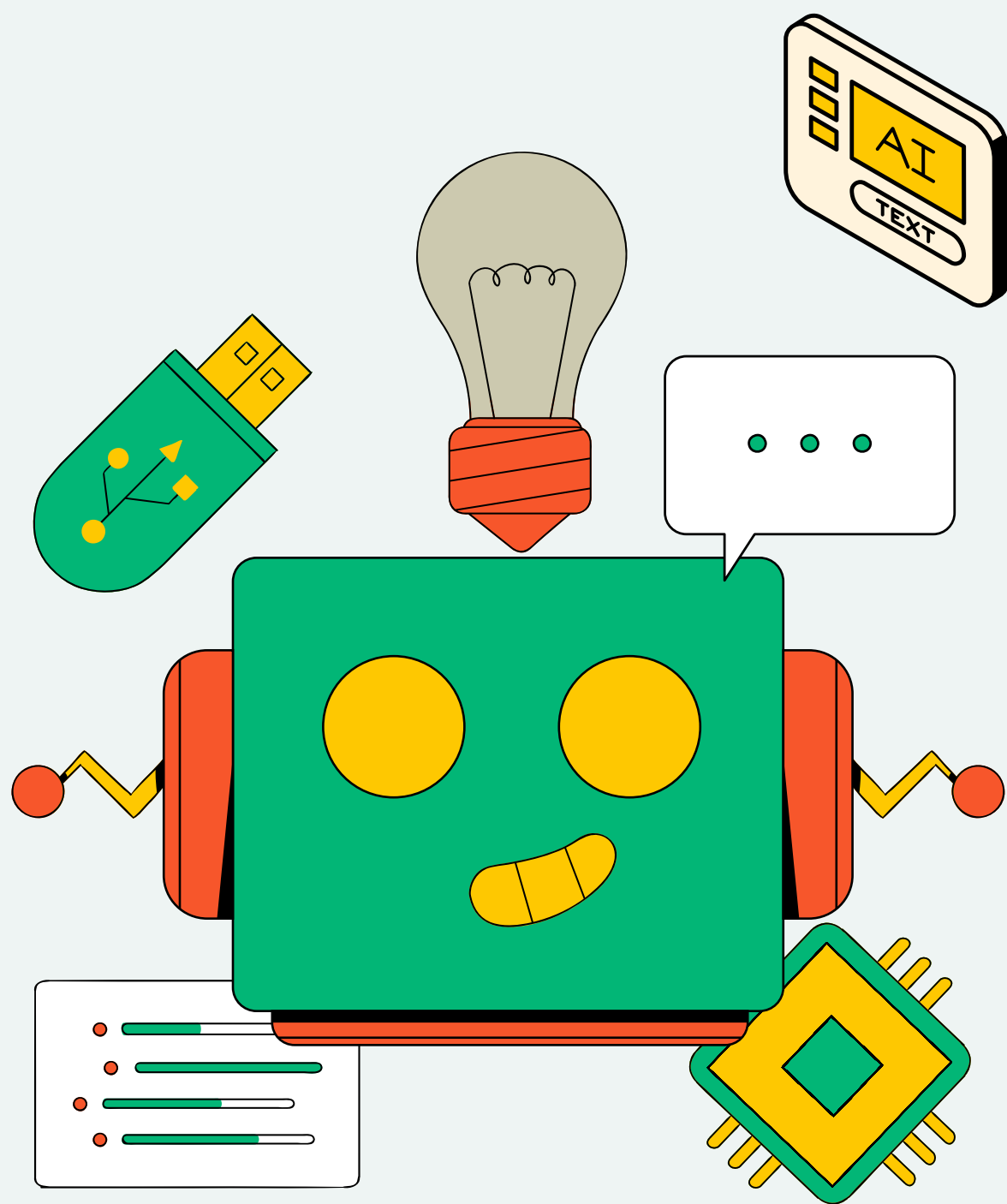


COMPUTER VISION
PROJECT



DEEP LEARNING: IMAGE CLASSIFICATION USING CNN AND TRANSFER LEARNING

PRESENTED BY:
EKATERINA
KUZNETSOVA



PROJECT TASK OVERVIEW:

- **Custom CNN Model:**

Building a Convolutional Neural Network (**CNN**) from scratch to classify images in the **CIFAR-10 dataset** into their respective categories

- **Transfer Learning:**

Using a pre-trained model to perform the same classification task, incorporating techniques such as data augmentation and fine-tuning

- **Performance Comparison:**

Evaluating and comparing performance

- **Tools used:** Python, TensorFlow/Keras, Google Colab

- **Techniques applied:**

CNN, Transfer Learning with Inception V3



LEARNING OBJECTIVES:



Understanding how Convolutional Neural Networks (CNNs) work

Learning to build and train a CNN from scratch

Applying Transfer Learning using model InceptionV3

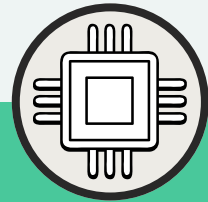
Evaluating and comparing models

Practicing with data augmentation, fine-tuning, and model optimization

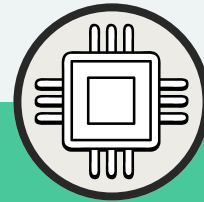
Using VS CODE and Google Colab to run experiments



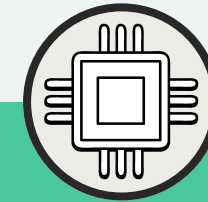
ASSESSMENT COMPONENTS



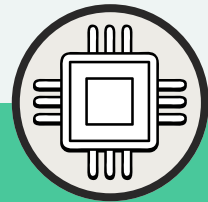
DATA PREPROCESSING



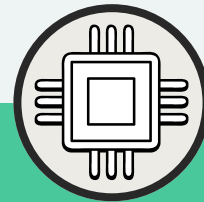
MODEL ARCHITECTURE



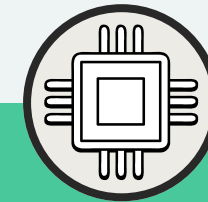
MODEL TRAINING



MODEL EVALUATION



TRANSFER LEARNING

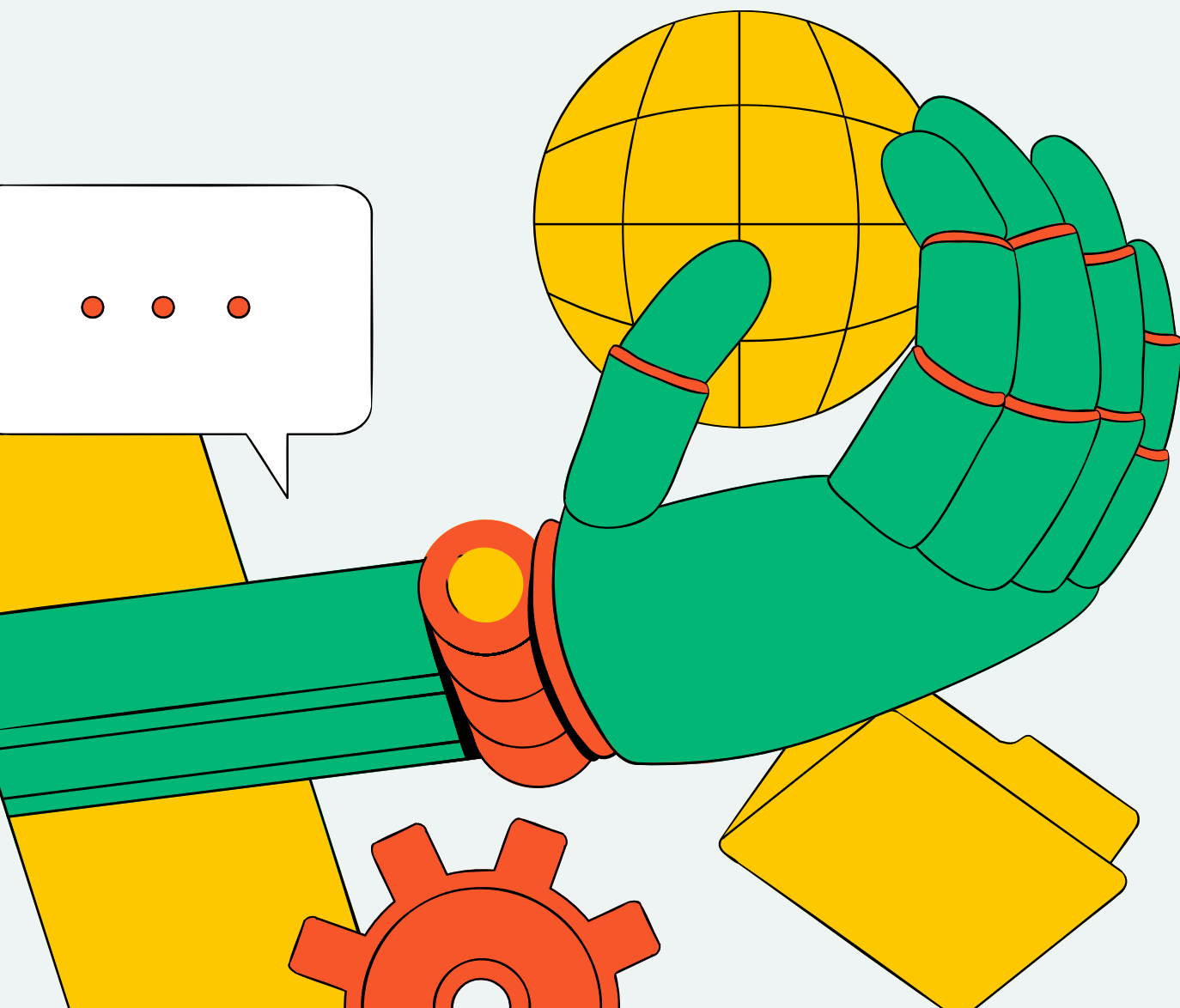


CODE QUALITY



IMPLEMENTATION:

- Preprocessing the CIFAR dataset
- Building a CNN
- Training and evaluating the CNN
- Improving the model with more layers and data augmentation, then hyperparameter tuning, and early stopping
- Moving to Transfer Learning using InceptionV3
- Applying data augmentation, tuning and adding a new training to avoid overfitting
- Comparing CNN vs Transfer Learning performance



CNN

MODEL ARCHITECTURE

Model: "sequential_1"

Layer (type)	Output Shape	Param #
conv2d_3 (Conv2D)	(None, 30, 30, 32) ✨	896
max_pooling2d_2 (MaxPooling2D)	(None, 15, 15, 32)	0
conv2d_4 (Conv2D)	(None, 13, 13, 64)	18,496
max_pooling2d_3 (MaxPooling2D)	(None, 6, 6, 64)	0
conv2d_5 (Conv2D)	(None, 4, 4, 128)	73,856
flatten_1 (Flatten)	(None, 2048) ✨	0
dense_2 (Dense)	(None, 128) ✨	262,272
dropout_1 (Dropout)	(None, 128)	0
dense_3 (Dense)	(None, 10)	1,290

Total params: 356,810 (1.36 MB)

Trainable params: 356,810 (1.36 MB)

Non-trainable params: 0 (0.00 B)

CLASIFICATION REPORT

	precision	recall	f1-score	support
airplane	0.70	0.80	0.75	1000
automobile	0.84	0.85	0.84	1000
bird	0.59	0.61	0.60	1000
cat	0.52	0.56	0.54	1000
deer	0.67	0.70	0.68	1000
dog	0.66	0.59	0.62	1000
frog	0.75	0.83	0.79	1000
horse	0.83	0.73	0.78	1000
ship	0.85	0.79	0.82	1000
truck	0.84	0.75	0.79	1000
accuracy			0.72	10000
macro avg	0.72	0.72	0.72	10000
weighted avg	0.72	0.72	0.72	10000

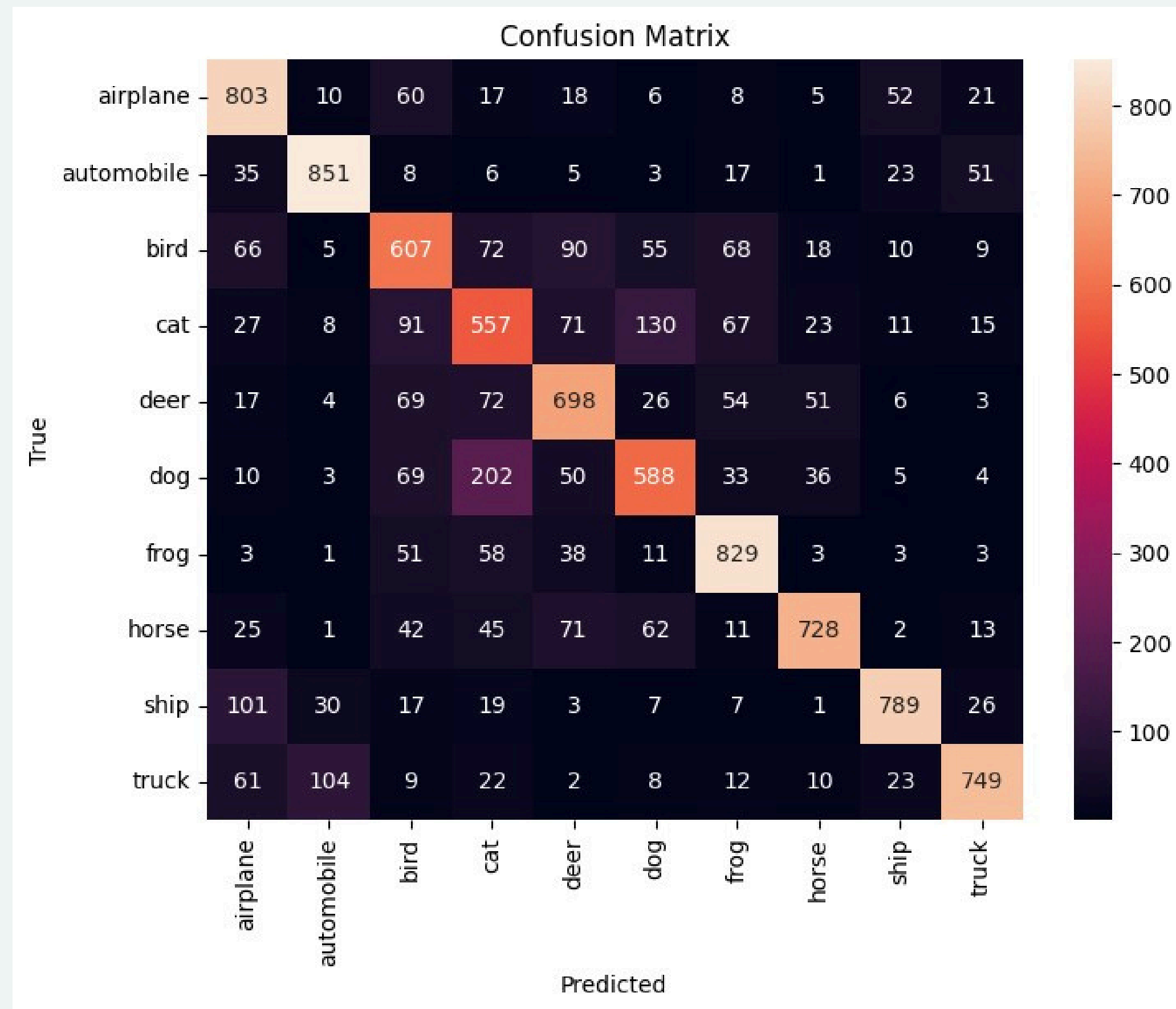
Accuracy: 0.7199

Precision: 0.7246

Recall: 0.7199

F1-score: 0.7207





IMPROVING THE MODEL WITH MORE LAYERS

```
313/313 ————— 2s 2ms/step - accuracy: 0.7718 - loss: 0.7038  
Test accuracy with more layers: 0.7673
```

HYPERTUNING AND EARLY STOPPING

```
313/313 ————— 1s 2ms/step - accuracy: 0.7482 - loss: 0.7276  
Test accuracy with early stopping: 0.7467
```

DATA AGMENTATION

```
313/313 ————— 1s 2ms/step - accuracy: 0.7884 - loss: 0.6657  
Test accuracy with data augmentation: 0.7807
```

Base CNN ~ 72%
More layers (deeper CNN) ~ 77%
Dropout & Earlystopping ~ 74.7%
Data Agmentation ~ 78%



INCEPTION V3

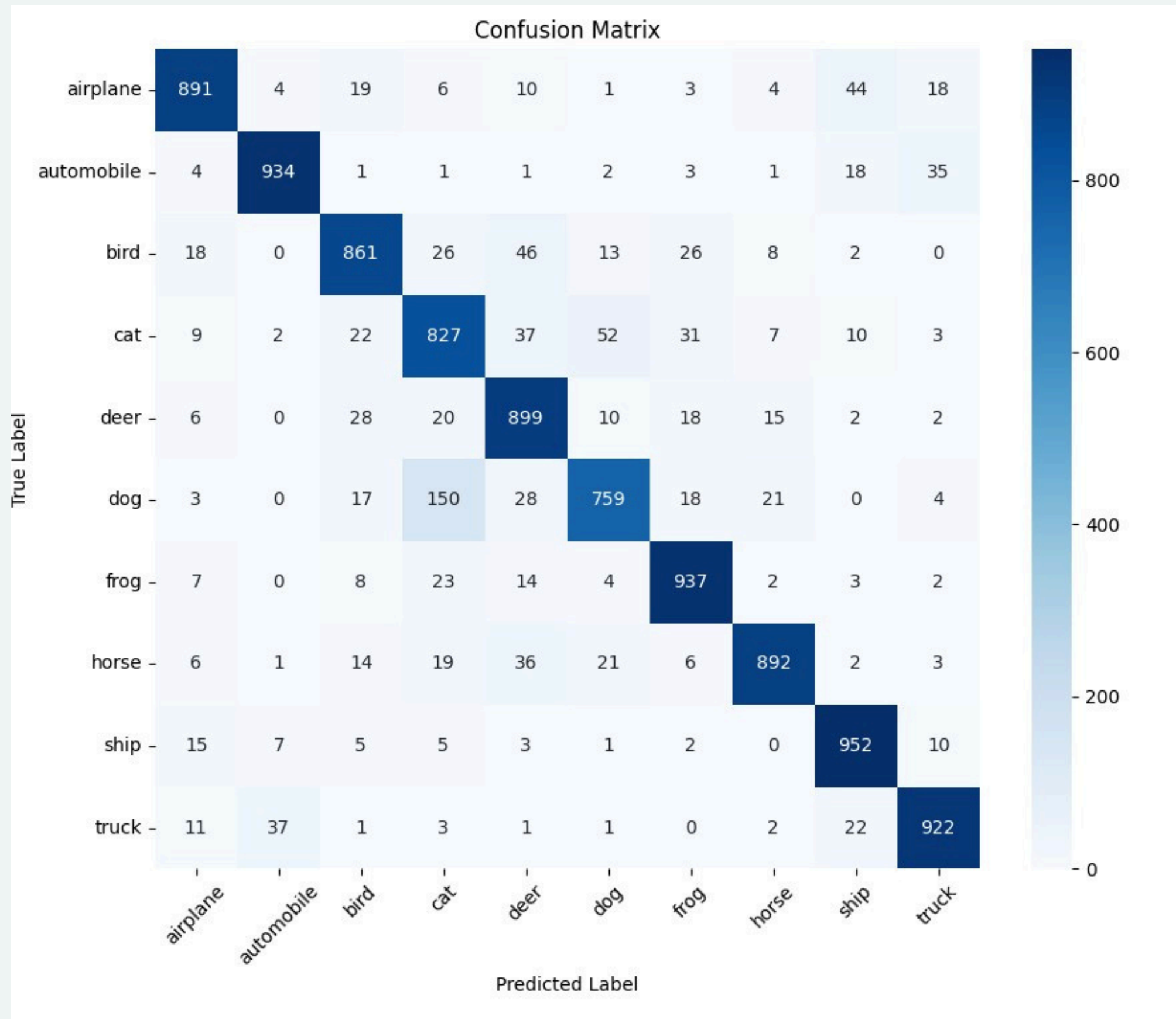
EVALUATION AFTER PRETRAINING

313/313 5s 8ms/step - accuracy: 0.8881 - loss: 0.4687
Test Accuracy: 0.8874

CLASIFICATION REPORT

	precision	recall	f1-score	support
airplane	0.92	0.89	0.90	1000
automobile	0.95	0.93	0.94	1000
bird	0.88	0.86	0.87	1000
cat	0.77	0.83	0.80	1000
deer	0.84	0.90	0.87	1000
dog	0.88	0.76	0.81	1000
frog	0.90	0.94	0.92	1000
horse	0.94	0.89	0.91	1000
ship	0.90	0.95	0.93	1000
truck	0.92	0.92	0.92	1000
accuracy			0.89	10000
macro avg	0.89	0.89	0.89	10000
weighted avg	0.89	0.89	0.89	10000





FINAL EVALUATION AFTER DATA AGUMENTATION

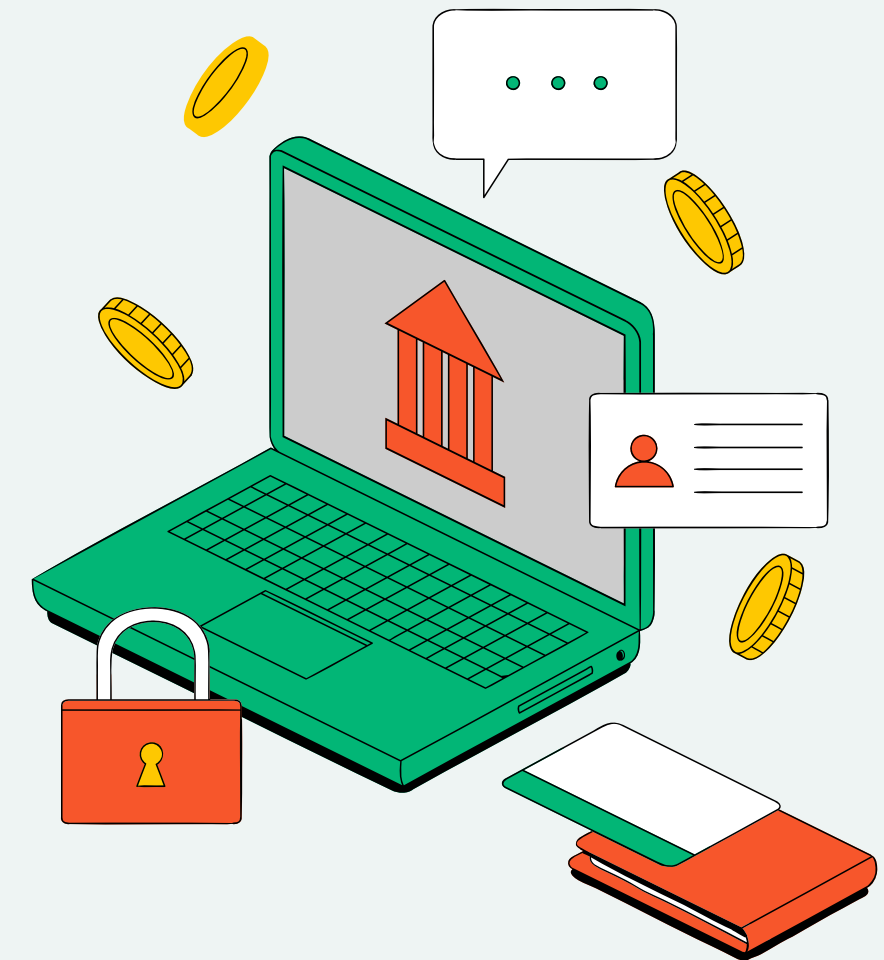
```
Test Accuracy: 0.9214  
Test Loss: 0.2858
```

FINAL EVALUATION AFTER FINE-TUNING

```
Training Accuracy: 0.9982, Loss: 0.0068  
Test Accuracy: 0.9278, Loss: 0.2761
```

AVOIDING OVERFITTING - NEW TRAINING

```
313/313 ————— 5s 7ms/step - accuracy: 0.9256 - loss: 0.5327  
Test Accuracy with Dropout: 0.9261
```



KEY INSIGHTS:



CNN learns from scratch so struggles with limited data

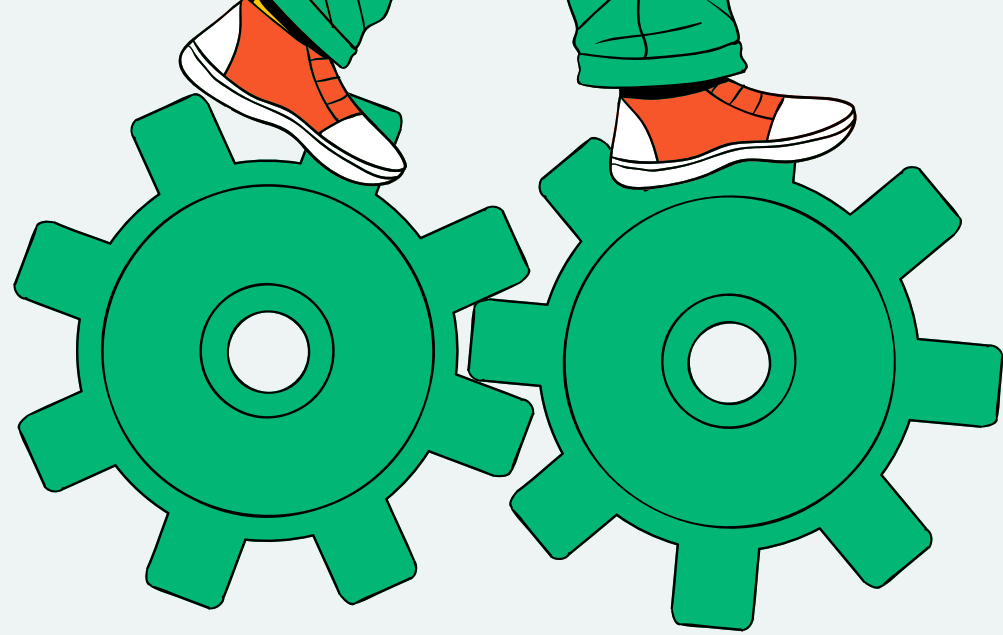
CNN is good for learning but will probably underperform on complex tasks unless it will be trained or optimized

InceptionV3 achieves higher accuracy on the test set than CNN

InceptionV3 with data augmentation and fine-tuning proves to be more effective model for CIFAR-10

Fine-tuning & data augmentation allow InceptionV3 to adapt to CIFAR





MAJOR OBSTACLES

Before getting extra GPU – constant problems with crushing

Re-running CNNs data processing was increasing y_train shape

VGG16 – the picture size/resizing/errors. I had to drop this model

Keeping both models in one notebook – models were split



THANK YOU

