Animals-10

Deep Learning Project Report



**Group 1**

Luis

Affiong

Nicholai

**Table of Contents**

[**Project Overview 2**](#_yspy8tt3f0xe)

[**Dataset 2**](#_75rf4vta81ax)

[**1. Description of Chosen CNN Architecture 2**](#_lquiyrwpy6ke)

[**2. Explanation of the Preprocessing Steps 3**](#_kn5uvgo00ajj)

[**3. Details of the Training Process 3**](#_l0td0gtsj7z1)

[**4. Results and Analysis of the Model’s Performance 4**](#_ins8uoxsuvl7)

[**5. The Best Model Used 5**](#_1mil7spy9n8e)

[**6. Insights Gained from the Experimentation Process 6**](#_1x9ryqabo6fs)

[**7. Code Quality 6**](#_as65geht0qi1)

[**8. Visualizations and Diagrams 7**](#_qlwgay1qt0vi)

# 

# Project Overview

This project explores image classification using Convolutional Neural Networks (CNNs). The goal is to build and evaluate models capable of accurately classifying images into predefined categories using both custom CNNs and transfer learning.

# Dataset

**Chosen dataset:** Animals10

**Number of classes:** 10

**Number of images:** ~28,000

**Image dimensions:** Varying, resized to 224x224

# Description of Chosen CNN Architecture

Two models were implemented:

**Custom CNN:**

A Sequential model with 4 Conv2D layers (32–256 filters, kernel size 3×3), each followed by BatchNormalization and MaxPooling. It ends with GlobalAveragePooling, a 128-unit Dense layer with ReLU, Dropout (0.3), and a final softmax output for 10 classes.

**Transfer Learning (ResNet50):**

Used as a base with ImageNet weights. Layers below index 100 were frozen. A GlobalAveragePooling and Dropout (0.5) were added before the final Dense layer. Fine-tuning was applied in a second training phase.

# Explanation of the Preprocessing Steps

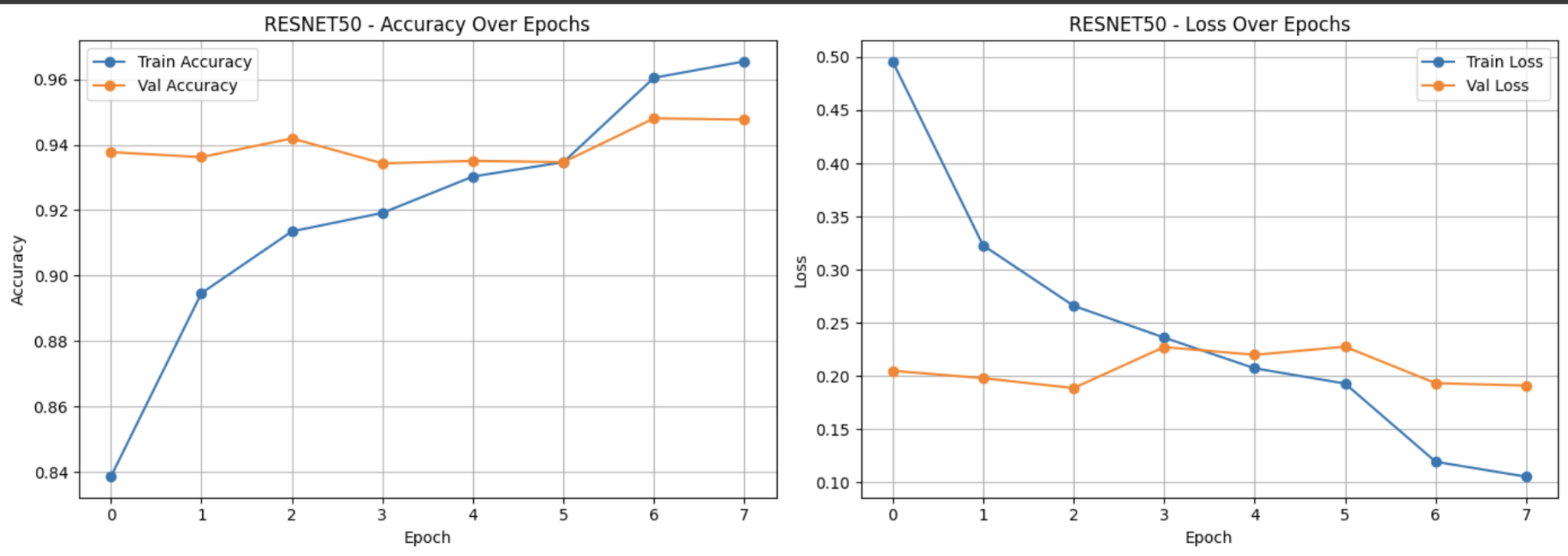
* Images resized to 224x224.
* Used ImageDataGenerator and TensorFlow augmentation (RandomFlip, RandomZoom, RandomRotation, RandomContrast).
* Dataset split into training, validation, and test sets.
* Class imbalance addressed using class weights.

# Details of the Training Process

* Batch size: 32
* Epochs:
  + Base training: 30
  + Fine-tuning: 10
* Optimizer: Adam
  + Base learning rate: 1e-3
  + Fine-tuning learning rate: 1e-5
* Loss Function: sparse\_categorical\_crossentropy
* Callbacks: EarlyStopping (patience=5) and ReduceLROnPlateau

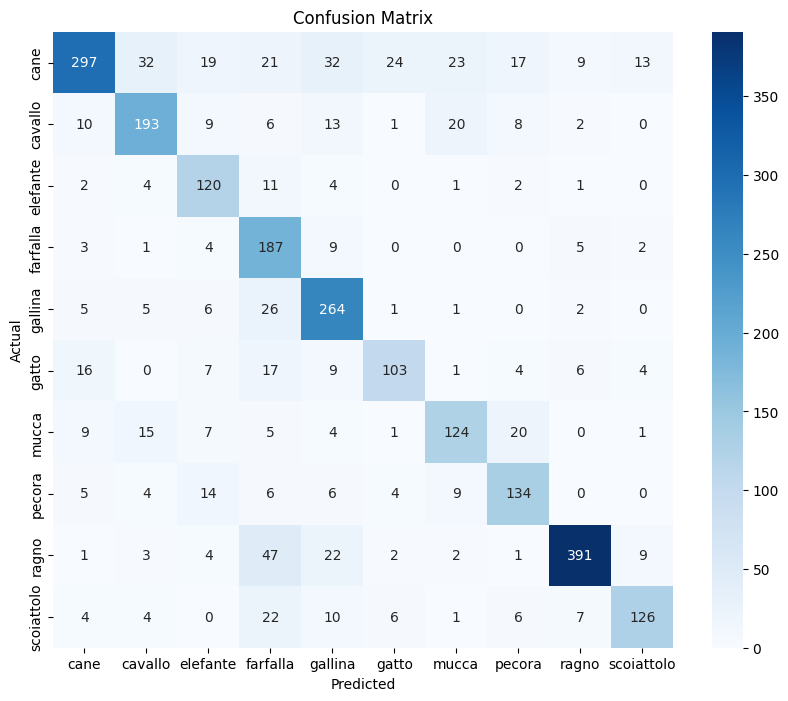
# Results and Analysis of the Model’s Performance

**Accuracy and loss improved** steadily with **early stopping** and learning rate reduction.



A slight discrepancy (or jump) was detected along the accuracy and loss curves during early testing. We believe this was due to the learning rate affecting the validation loss. It was corrected accordingly, as is reflected in the graph above.

The **Confusion Matrix** revealed that some classes were harder to distinguish (likely visually similar) than others, as can be seen below.



*Below you can find our* ***Classification Report.*** *It indicates the following:*

* F1 Score: 0.96
* Precision: 0.96
* Recall: 0.96



The classification report for the animal evaluation indicates strong performance across all categories. The model achieved an overall accuracy of 98%, demonstrating its effectiveness in identifying various animals.

The macro and weighted averages of precision (97% and 98% respectively), recall (98% for both), and f1-scores (97% and 98% respectively) further underscore the model's robustness across different categories. Overall, the results demonstrate a highly effective classification system for these animals.

# The Best Model Used

The ResNet50 model's application of transfer learning, coupled with the benefits of pretrained weights and fine-tuning, resulted in higher accuracy and better generalization in animal classification tasks. This approach not only improves model performance but also reduces the time and resources required for training, making it a highly effective strategy in the field of deep learning.

# Insights Gained from the Experimentation Process

The experimentation process yielded several key insights:

1. A custom **Convolutional Neural Network (CNN)** served as a baseline but **underperformed** compared to pretrained networks, **emphasizing the benefits** of using **established models** (i.e. ResNet50).
2. Data augmentation proved essential in preventing overfitting by increasing training dataset diversity, which helped the model generalize better to unseen data.
3. Fine-tuning the ResNet model significantly improved performance by adapting the pretrained network to the specific characteristics of the dataset.
4. Additionally, addressing class imbalance early on was crucial, as it impacts model learning and ensures more accurate predictions across all categories.

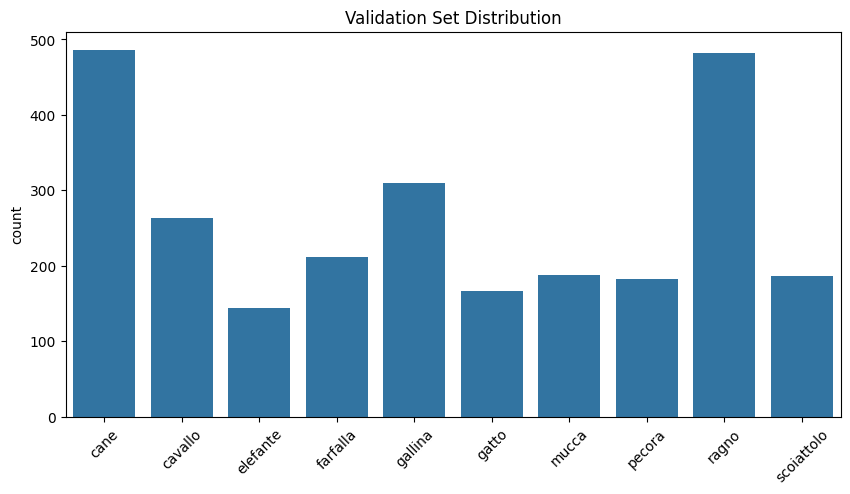
# Code Quality

Pylint was used to assess and improve the code quality. The rating improved from 1.85 7.29, indicating a focus on continuous improvement. This shows that the team actively worked on addressing issues and refining the code.

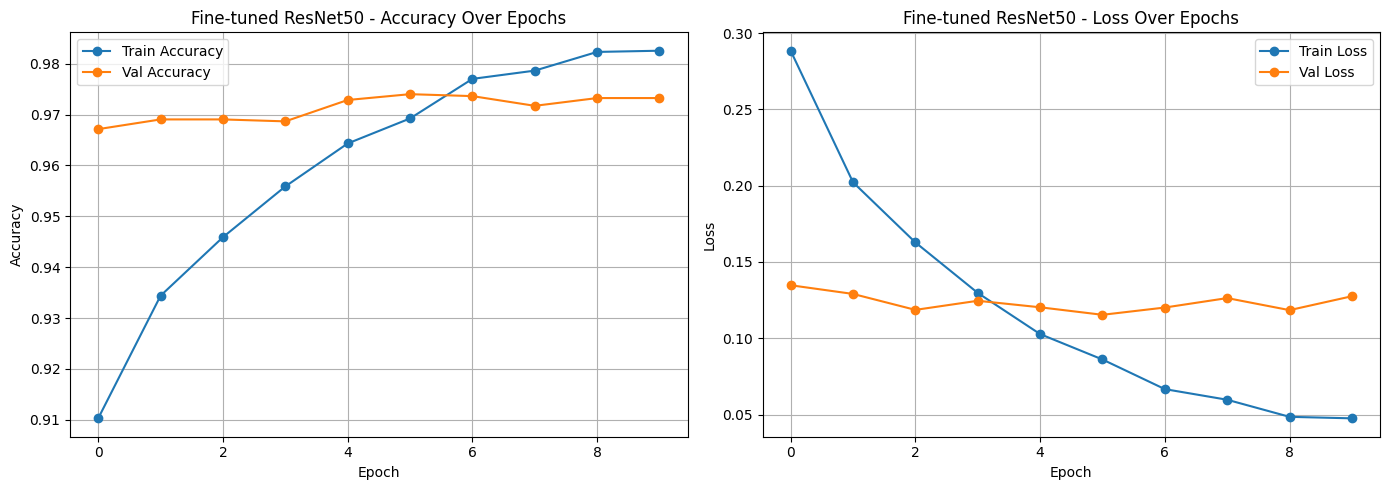
Regularly running Pylint and addressing feedback ensured that the code remained robust and maintainable.

# Visualizations and Diagrams

* **Class Distribution:**  
  + *Visualized to detect imbalance across datasets.*



* **Training Curves:**  
  + *Accuracy and loss plotted across epochs.*



* **Confusion Matrix:**  
  + *Showed class-wise performance.*

