

Deployment Link

The live deployed application can be accessed here:

<https://singh-mayank18-image-classification-app-68aeay.streamlit.app/>

1. Introduction

This project aims to build a **real-time image classification system** using deep learning, deployed through an interactive web interface. The system classifies images into five categories: **bike, bus, car, cat, and dog**.

The core of the system is a trained **Convolutional Neural Network (CNN)** built using **TensorFlow/Keras**.

A **Streamlit web application** was developed to allow users to upload images and receive instant predictions.

2. Methodology

Dataset Preparation

A custom dataset was created with five folders representing each class: **bike, bus, car, cat, and dog**.

All images were:

- Resized to 224×224
- Normalized between 0 and 1
- Split using **stratified sampling** into:
 - 60% training

- 20% validation
 - 20% testing
-

Model Architecture

The model uses **EfficientNetB0**, a pre-trained convolutional neural network known for efficiency and high accuracy.

Model layers used:

- EfficientNetB0 (without top layers) for feature extraction
- Batch Normalization
- Dense layer (256 neurons, ReLU activation)
- Dropout (0.5) to reduce overfitting
- Output Dense layer (5 neurons, Softmax activation)

Compiler settings:

- Optimizer: **Adam**

- Loss function: **categorical crossentropy**
-

Training & Evaluation

The model was trained for **10 epochs** using image data generators.

Results:

- **Training Accuracy:** ~100%
- **Validation Accuracy:** ~16%
- **Test Accuracy:** ~33%

Although results show overfitting due to a small dataset, the model produced correct predictions during testing.

Deployment with Streamlit

A fully functional **Streamlit application** was built to allow real-time prediction.

Key features:

- Upload any image (JPG/PNG)
- Automatic preprocessing:

- RGB conversion
- Image resizing
- Normalization
- Real-time prediction using the trained CNN model
- Confidence score display
- Class probability bar chart

The trained model was integrated using TensorFlow's `load_model()` function.

The live deployment is available here:

↗ <https://singh-mayank18-image-classification-app-68aeay.streamlit.app/>

3. Challenges Faced

Several challenges occurred during development:

- **Class imbalance** and few images per category caused stratification errors.
- **Shape mismatch issues** occurred when validation/test sets contained fewer classes.
-

RGB images led to input shape errors; this was fixed by converting images to RGB.

- Overfitting occurred due to a small dataset.
 - Model saving errors required using .h5 or .keras formats.
 - Streamlit-specific issues, such as deprecated parameters and missing class labels, needed refinements.
-

4. Conclusion

The project successfully demonstrates an end-to-end workflow for building, training, evaluating, and deploying a CNN-based image classification system.

Despite dataset limitations, the application performs well for real-time predictions. The Streamlit integration provides a smooth, interactive user experience and makes the model easily accessible through a browser.

The final deployed application is available at:

<https://singh-mayank18-image-classification-app-68aeay.streamlit.app/>