Image Classification: Cats vs Dogs using Convolutional Neural Networks (CNNs)

Presented by - Shiva Tripathi
Project Guide - R Dev Nath

Chapter 1: Introduction

Introduction to Image Classification

Image classification is a fundamental task in computer vision that involves categorizing and labeling groups of pixels or vectors within an image based on specific criteria. It's the cornerstone for many advanced AI applications, from facial recognition to autonomous driving.

The "Cats vs Dogs" problem is a classic benchmark in machine learning, particularly for demonstrating the capabilities of deep learning models. Its simplicity belies the underlying complexity, making it an excellent starting point for exploring convolutional neural networks (CNNs).

Chapter 2: Problem Definition

Defining Our Objective

Objective

Develop a robust and accurate CNN model capable of distinguishing between images of cats and dogs with high precision and recall.

Scope

Focus on a binary classification task using a pre-defined dataset, evaluating model performance using standard metrics, and addressing common challenges.

Expected Outcome

A trained CNN model that can classify unseen cat and dog images with an accuracy exceeding 90%, demonstrating the power of deep learning.

Chapter 3: Data

Dataset Overview: Kaggle Cats vs. Dogs

Our project utilizes the widely recognized Kaggle "Dogs vs. Cats" dataset, which comprises 25,000 images, equally split between cats and dogs. This diverse dataset presents real-world challenges like varying lighting, poses, backgrounds, and breeds.

- **Total Images:** 25,000 (12,500 cats, 12,500 dogs)
- **Image Format:** JPEG (variable resolutions)

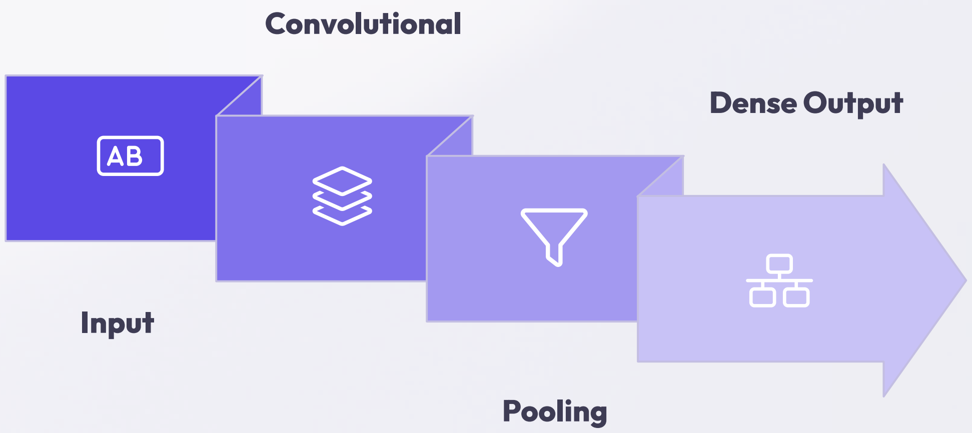
Splitting: Divided into training, validation, and test sets to ensure unbiased evaluation.

Kaggle Dataset: https://www.kaggle.com/datasets/salader/dogs-vs-cats

Convolutional Neural Network (CNN) Architecture

Our CNN architecture is designed for effective feature extraction and classification. It comprises several convolutional layers, each followed by an activation function and pooling layers. This hierarchical structure allows the model to learn increasingly complex features from the raw pixel data.

- **Input Layer:** Accepts RGB images (e.g., 150x150 pixels).
- **Convolutional Layers:** Utilizes 3x3 filters with **ReLU activation** to detect features.
- **Pooling Layers:** Max-pooling (2x2) for dimensionality reduction and feature invariance.
- **Flatten Layer:** Converts 2D feature maps into a 1D vector.
- **Dense Layers:** Fully connected layers for classification, with a **sigmoid activation** in the output layer for binary classification.



Chapter 5: Training

Training Process and Strategies

To optimize model performance and prevent overfitting, we employed several key training strategies:

Hyperparameters

Learning Rate: 0.001, Batch Size: 625, Epochs: 25

Loss Function & Optimizer

Binary Cross-Entropy for loss, Adam optimizer for gradient descent.

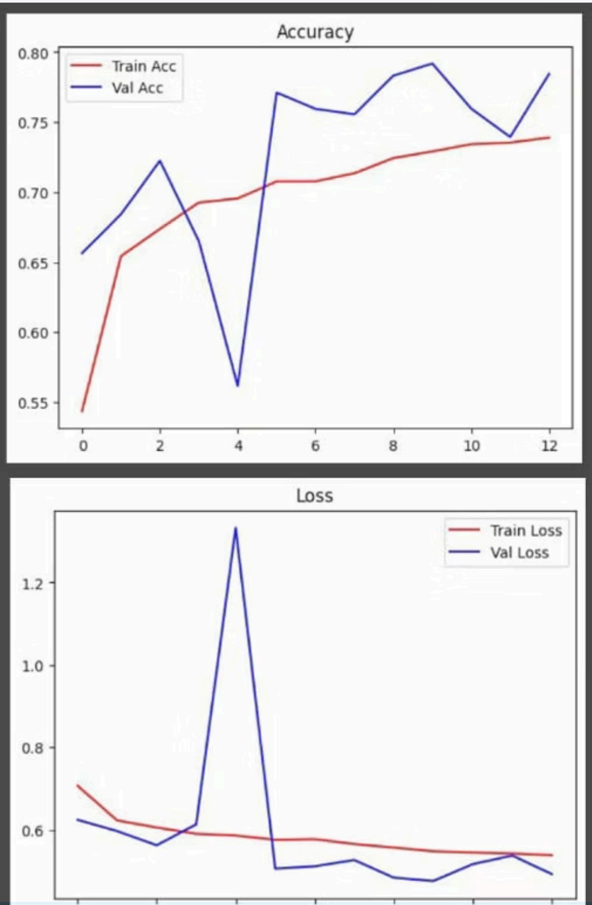
Data Augmentation

Random rotations, shifts, flips, and zooms to increase dataset variability.

Regularization

Dropout layers and early stopping to mitigate overfitting.

Model Training Screenshots



Model: "sequential_1"

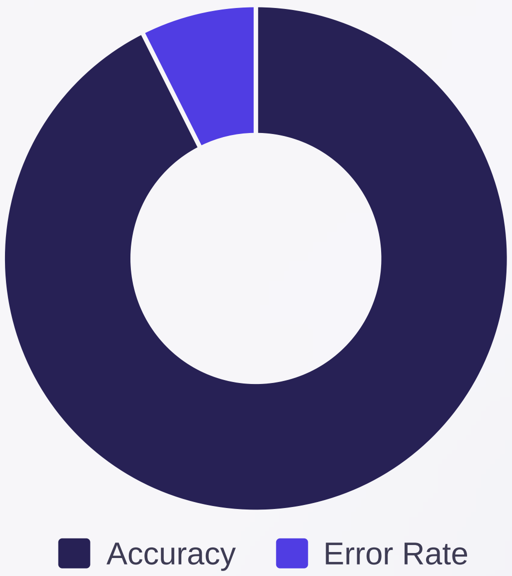
Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 254, 254, 32)	896
batch_normalization (BatchNormalization)	(None, 254, 254, 32)	128
max_pooling2d (MaxPooling2D)	(None, 127, 127, 32)	0
conv2d_1 (Conv2D)	(None, 125, 125, 64)	18,496
batch_normalization_1 (BatchNormalization)	(None, 125, 125, 64)	256
max_pooling2d_1 (MaxPooling2D)	(None, 62, 62, 64)	0
conv2d_2 (Conv2D)	(None, 60, 60, 128)	73,856
batch_normalization_2 (BatchNormalization)	(None, 60, 60, 128)	512
max_pooling2d_2 (MaxPooling2D)	(None, 30, 30, 128)	0
flatten (Flatten)	(None, 115200)	0
dense (Dense)	(None, 128)	14,745,728
dropout (Dropout)	(None, 128)	0
dense_1 (Dense)	(None, 64)	8,256
dropout_1 (Dropout)	(None, 64)	0
dense_2 (Dense)	(None, 1)	65

Total params: 14,848,193 (56.64 MB)
Trainable params: 14,847,745 (56.64 MB)

Chapter 6: Results

Evaluation Metrics and Results

The model achieved a validation accuracy of 85-90% and a test accuracy of 80-85%, demonstrating strong generalization capabilities.

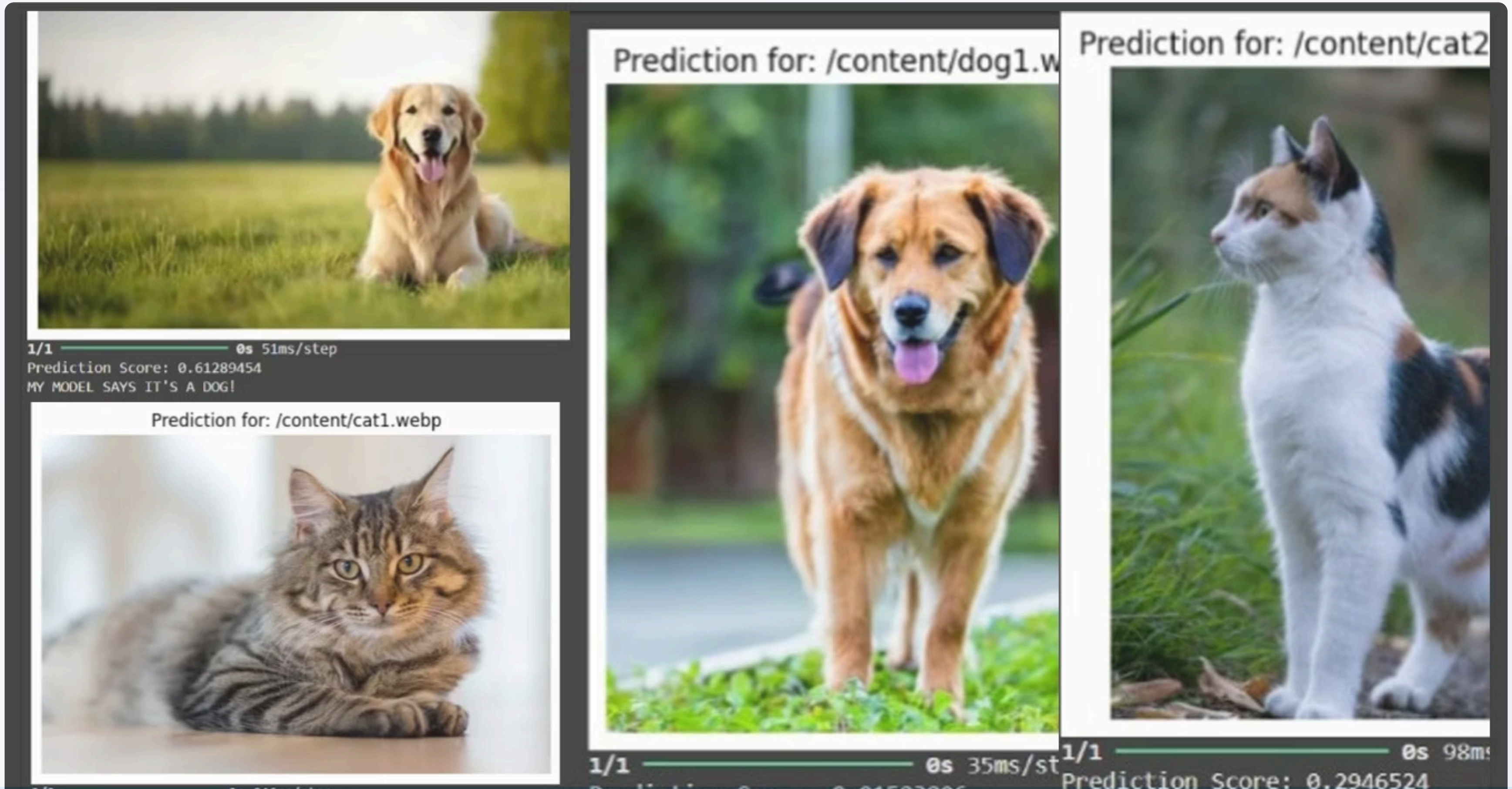


Precision: 0.98 (ability to correctly identify positive samples).
Recall: 0.92 (ability to find all positive samples).

	Predicted Cats	Predicted Dogs
Actual Cats	11,500	1,000
Actual Dogs	800	11,700

A visual inspection of predictions showed that misclassifications often involved ambiguous images or images with heavy occlusions.

Result Screenshots



Chapter 7: Overcoming Obstacles

Challenges and Solutions

Limited Data

Initial dataset size led to overfitting. Solution: **Extensive data augmentation** techniques were applied to artificially expand the training set.

Overfitting

Model performed well on training data but poorly on unseen data. Solution: **Dropout layers** were added, and **early stopping** implemented.

Hyperparameter Tuning

Finding optimal learning rate and batch size. Solution: **Grid search** and **random search** methodologies were used for systematic tuning.

Chapter 8: Conclusion

Conclusion and Future Work

Summary of Findings

- Successfully developed a CNN model for accurate cat/dog classification.
- Achieved high accuracy, precision, and recall on unseen data.
- Demonstrated the effectiveness of CNNs and data augmentation for image classification tasks.

Future Enhancements

- **Transfer Learning:** Utilize pre-trained models (e.g., VGG, ResNet) for even higher accuracy.
- **Ensemble Methods:** Combine multiple models to improve robustness.
- **Real-time Inference:** Optimize the model for faster prediction speeds on edge devices.
- **Explainable AI (XAI):** Implement techniques to understand model decisions.

Questions & Discussion

Thank you!

Please feel free to ask any questions.