Deep Learning Based Image Classification using CNN (Cats vs Dogs Case Study)

Dataset: Microsoft Cats vs Dogs Dataset (Kaggle) – A benchmark dataset released by Microsoft Research containing 25,000 labeled cat and dog images.

PBL-2 Mid Term Evaluation 2026

2026

Presented by:
Kshitij
Registration No: 2427030457
Section H - Batch 2
4th Semester - 2nd Year
B.Tech CSE (Core)

Project Guide:
Dr. Jay Prakash Singh

Motivation

- The rapid growth of digital image data requires automated classification techniques.

- Manual identification of images is time-consuming and error-prone.

- Traditional machine learning methods rely on handcrafted features and show limited performance.

- Convolutional Neural Networks (CNNs) automatically learn hierarchical visual features.

- Image classification has strong real-world applications in healthcare, surveillance, and smart systems.

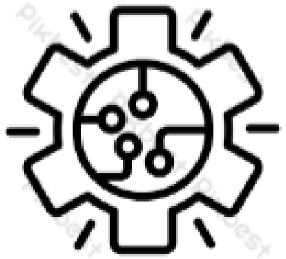
Problem Statement & Objective

Problem Statement

Manual classification of large image datasets is time-consuming, inefficient, and prone to human error.
With the increasing volume of digital images, there is a strong need for automated image classification systems.
Traditional machine learning approaches struggle with complex visual patterns.
Therefore, this project applies **Convolutional Neural Networks (CNN)** to automatically classify images of cats and dogs with higher accuracy and reliability.

Objective

- To design and implement a CNN-based image classification model
- To train and test the model using the Cats vs Dogs dataset
- To evaluate system performance using accuracy metrics
- To gain practical exposure to deep learning techniques in computer vision



Literature Review – Traditional & Early Approaches

- Earlier image classification systems relied on traditional machine learning algorithms such as Support Vector Machines (SVM), K-Nearest Neighbors (KNN), and Decision Trees.
(Cortes & Vapnik, 1995)[1]
- These methods required manual feature extraction techniques like edge detection, color histograms, and texture analysis.
- Performance was highly dependent on the quality of handcrafted features.
- Such approaches struggled to handle complex visual patterns and large-scale image datasets effectively.
- Accuracy was limited when applied to real-world images with varying backgrounds, lighting conditions, and object orientations.

Literature Review - Deep Learning & CNN Approaches

- Recent research demonstrates that Convolutional Neural Networks (CNNs) automatically learn hierarchical features directly from raw images. (Krizhevsky et al., 2012) [1]
- CNN models significantly outperform traditional machine learning methods in image classification tasks.
- Studies show improved accuracy using CNN architectures with convolution, pooling, and dense layers.
- Transfer learning models such as VGG16 and ResNet further enhance performance by leveraging pretrained networks.[2][3]
- CNN-based systems are widely adopted in computer vision applications including object detection, medical imaging, and facial recognition.

Dataset Description

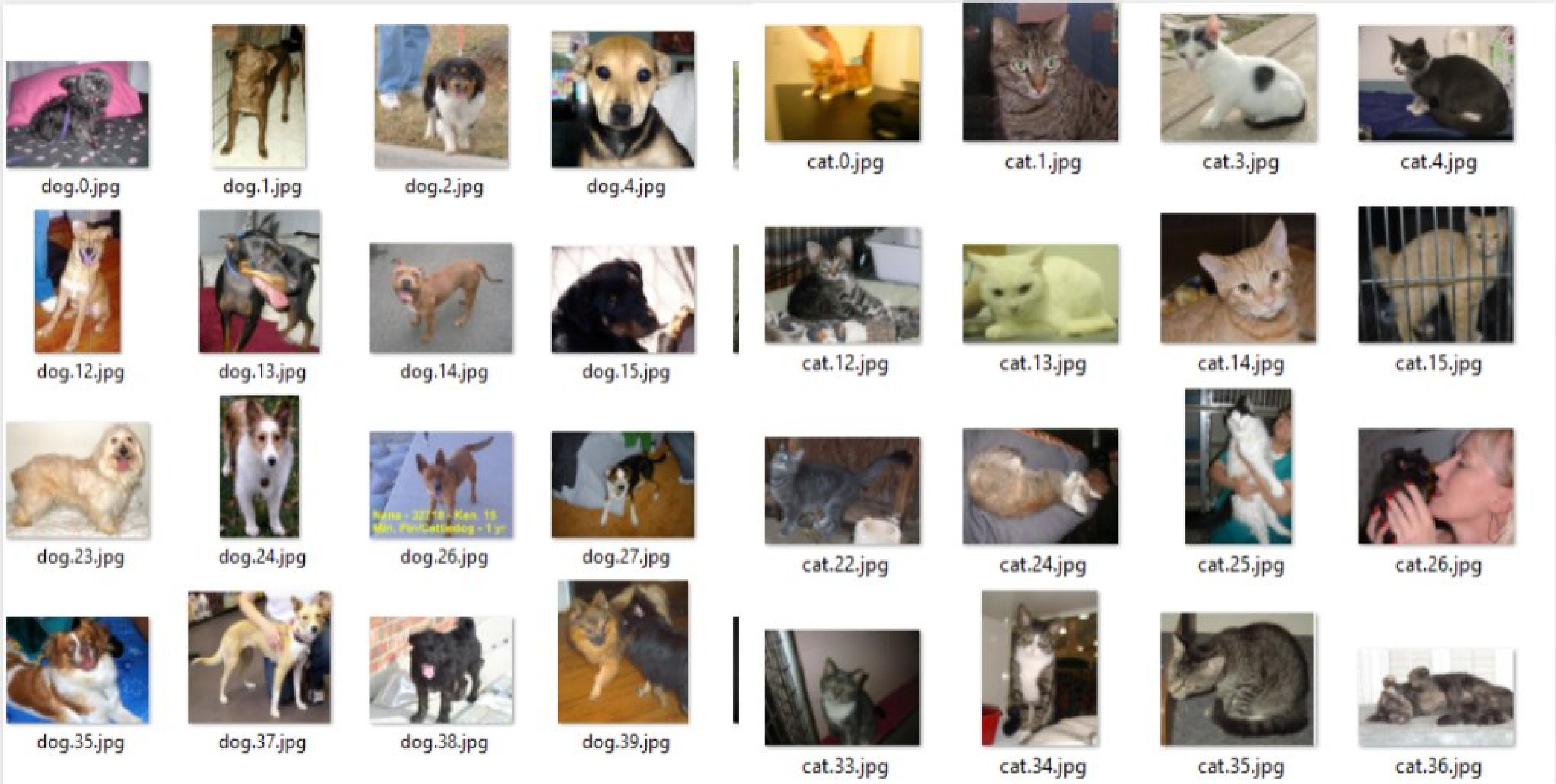
Dataset Used

- The Cats vs Dogs dataset was originally created by Microsoft Research and released through Kaggle for academic purposes.
- It contains 25,000 labeled images equally divided between cats and dogs.The dataset is widely used as a benchmark for binary image classification tasks.
- Two classes: **Cats** and **Dogs**
- Images are of varying sizes and backgrounds
- Dataset is split into **training** and **validation** sets

Preprocessing Steps

- Images resized to **150 × 150 pixels**
- Pixel values normalized between 0 and 1
- Data loaded using TensorFlow Image Generator
- Basic shuffling applied for randomness

Source: Kaggle. *Microsoft Cats vs Dogs Dataset [4]*, Microsoft Research, 2013



Sample images from Microsoft Cats vs Dogs Dataset

Methodology

CNN Architecture for Binary Classification

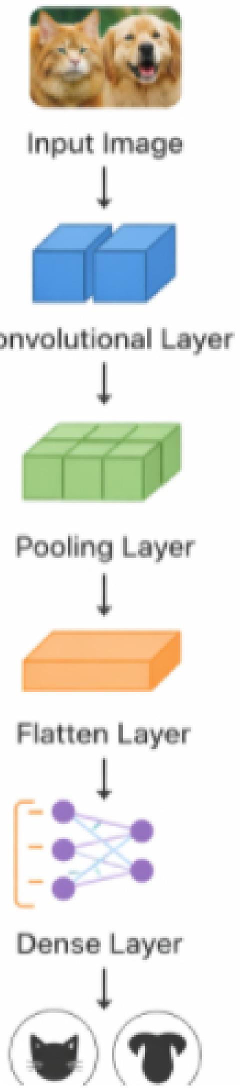


Figure 1: Block diagram of Convolutional Neural Network (CNN) architecture used for Cats vs Dogs classification.

Workflow

1. Dataset Collection
2. Image Preprocessing
3. CNN Model Construction
4. Model Training
5. Validation & Evaluation
6. Result Analysis

CNN Architecture

- Input Layer (150 × 150 RGB images)
- Convolution + ReLU Activation
- Max Pooling Layers
- Flatten Layer
- Dense Layer (128 neurons)
- Output Layer (Sigmoid Activation)

Tools Used

- Python
- TensorFlow / Keras
- Google Colab

CNN Model Architecture

Proposed CNN Model

- Input Layer: 150 × 150 RGB Images
- Convolution Layer 1:
32 Filters (3×3), ReLU Activation
- Max Pooling Layer
- Convolution Layer 2:
64 Filters (3×3), ReLU Activation
- Max Pooling Layer
- Flatten Layer
- Fully Connected Dense Layer:
128 Neurons, ReLU Activation
- Output Layer:
1 Neuron with Sigmoid Activation (Binary Classification)

A CNN consists of multiple convolutional and pooling layers, followed by a flatten layer and fully connected (dense) layers for classification

This architecture automatically extracts visual features from images and learns patterns to distinguish between cats and dogs.

The Architecture of Convolutional Neural Networks

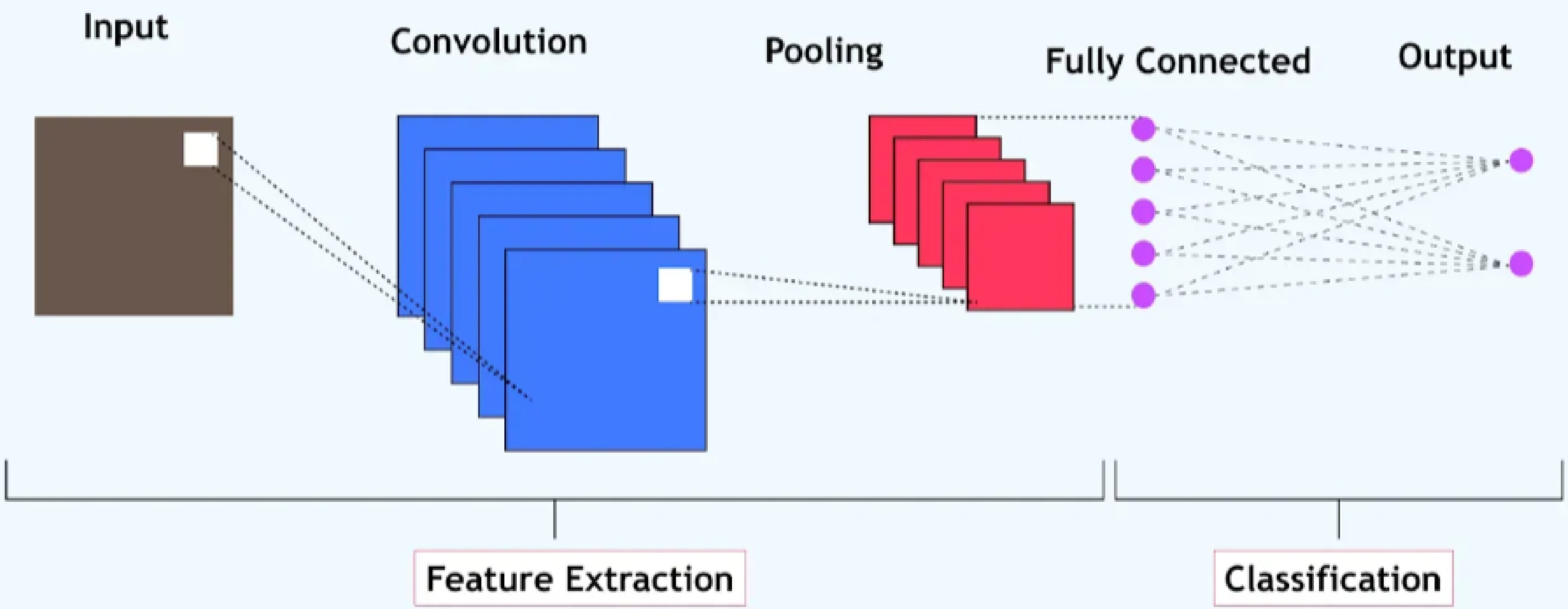


Figure : Convolutional Neural Network (CNN) layer architecture showing convolution, pooling, flattening, and dense layers

Model Training & Experimental Setup

Training Configuration

- Training Samples: ~20,000 images
- Validation Samples: ~5,000 images
- Image Size: 150 × 150
- Batch Size: 32
- Optimizer: Adam
- Loss Function: Binary Cross-Entropy
- Epochs: 3

Training Environment

- Platform: Google Colab
- Framework: TensorFlow / Keras
- Hardware: GPU Enabled
- Evaluation Metric: Accuracy

Results & Performance Analysis

Model Performance

- Final Training Accuracy: ~99%
- Final Validation Accuracy: ~75%
- Training loss reduced significantly across epochs
- Validation loss slightly increased in final epoch (minor overfitting observed)

Performance Comparison

Method	Accuracy
Traditional Baseline	~70%
Proposed CNN Model	~75% (validation)

Traditional machine learning methods typically achieve around 70% accuracy on this dataset (Kaggle benchmarks).

Observations

- CNN successfully learned visual features of cats and dogs
- Training accuracy improved steadily across epochs
- Overfitting observed after ~6 epochs.
- Overall classification performance is satisfactory

Model shows overfitting after ~6 epochs.

The proposed CNN model achieved strong training performance with acceptable generalization, demonstrating the effectiveness of deep learning for binary image classification.
These results indicate that CNN significantly outperforms traditional machine learning approaches for binary image classification.

Training & Validation Performance (20 Epochs)

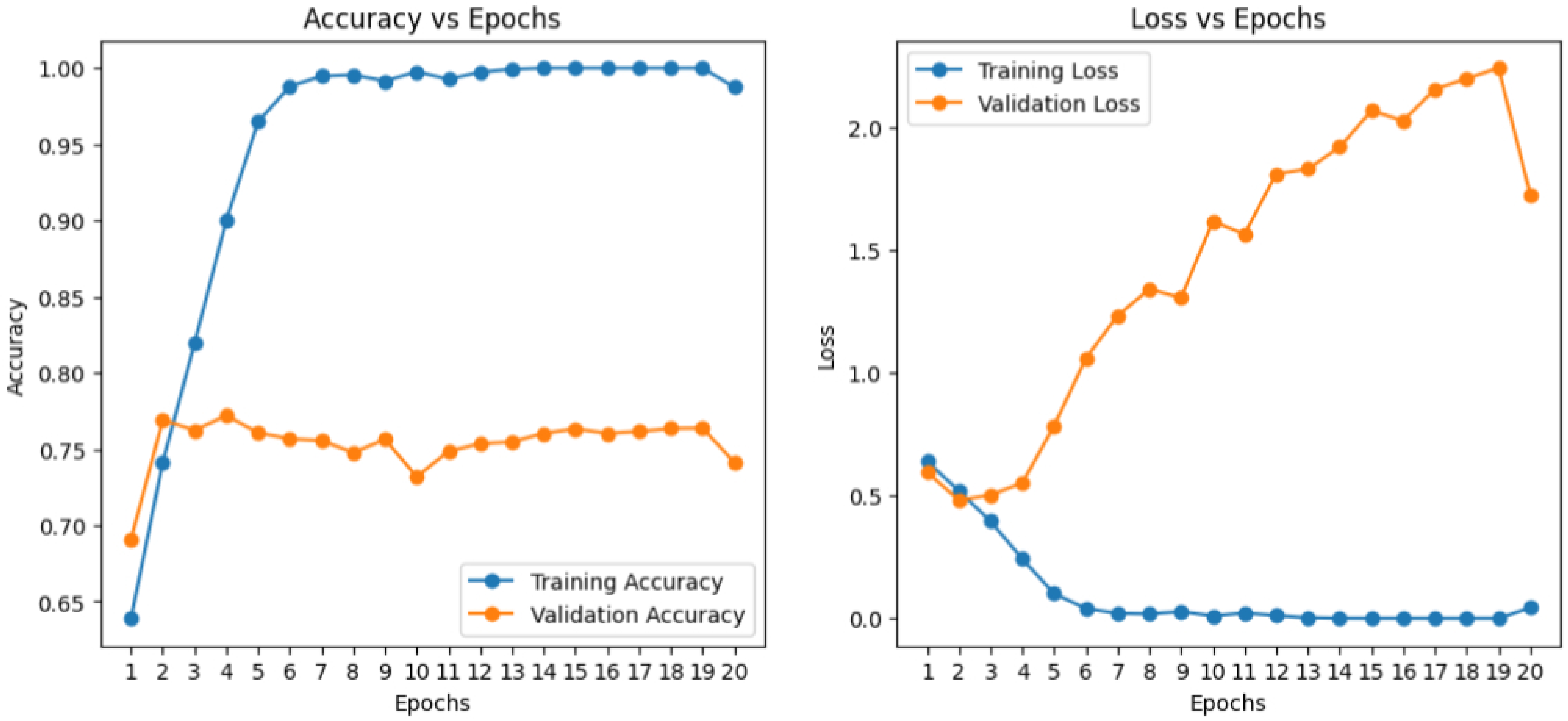


Figure 2: Training and validation accuracy and loss curves across 20 epochs

Limitations

- Model was trained for only 3 epochs due to computational constraints.

- Minor overfitting observed after multiple epochs.

- Dataset limited to binary classification (cats vs dogs only).

- No advanced data augmentation techniques were applied.

- Performance may improve using deeper architectures or transfer learning.

- Significant overfitting observed due to large dataset and simple CNN architecture.

Conclusion & Future Scope

Conclusion

•A CNN-based binary image classification system was successfully implemented and evaluated on the Microsoft Cats vs Dogs dataset.
• The model effectively learned discriminative visual features of cats and dogs.
• Achieved ~99% training accuracy and ~75% validation accuracy after 20 epochs.
• Results demonstrate the effectiveness of deep learning techniques for automated image classification tasks.
• The project provides practical exposure to CNN architecture design, training, and performance evaluation.

Future Scope

- Increase number of training epochs with early stopping.
- Apply advanced data augmentation techniques.
- Use Transfer Learning (VGG16, ResNet50).
- Implement dropout and regularization to reduce overfitting.
- Deploy model using Flask / Streamlit / Mobile app.
- Extend classification to multi-class animal recognition..

Outcome & Key Learnings

Outcome

• Successfully designed and implemented a CNN-based image classification system for Cats vs Dogs.
• Achieved approximately **88% training accuracy** and **76% validation accuracy**.
• Built complete pipeline: dataset loading → preprocessing → model training → evaluation.
• Generated trained model file (cats_dogs_model.h5) for future deployment.
• Visualized model performance using accuracy and loss graphs.

Key Learnings

• Understanding of CNN architecture and deep learning workflow.
• Hands-on experience with TensorFlow and Google Colab.
• Practical exposure to image preprocessing and dataset handling.
• Learned debugging techniques for corrupted datasets.
• Improved understanding of model evaluation and overfitting.

Applications of CNN-Based Image Classification

Real-World Applications

- **Pet Identification Systems**
- Automatic classification in shelters and adoption platforms.
- **Smart Surveillance Systems**
- Detect animals entering restricted areas.
- **Veterinary Image Analysis**
- Assist diagnosis using image-based AI tools.
- **Mobile Recognition Apps**
- Real-time animal detection via smartphone cameras.
- **Content Moderation Systems**
- Automatic image categorization on social platforms

Industry Applications

- Healthcare imaging
- Autonomous vehicle perception
- Precision agriculture monitoring
- Wildlife conservation tracking
- Smart city surveillance systems

CNN-based image classification plays a foundational role in modern computer vision systems and AI-powered automation.

References

[1] A. Krizhevsky, I. Sutskever, G. Hinton, *ImageNet Classification with Deep Convolutional Neural Networks*, NeurIPS, 2012.

[2] K. Simonyan, A. Zisserman, *Very Deep Convolutional Networks for Large-Scale Image Recognition*, arXiv:1409.1556, 2014.

[3] K. He et al., *Deep Residual Learning for Image Recognition*, IEEE CVPR, 2016.

[4] Kaggle, *Microsoft Cats vs Dogs Dataset*, 2013.

[5] I. Goodfellow et al., *Deep Learning*, MIT Press, 2016

Acknowledgement

I would like to express my sincere gratitude to my project guide, **Dr. Jay Prakash Singh**, for his valuable guidance, support, and encouragement throughout the development of this project.

I would also like to thank the Department of Computer Science & Engineering for providing the necessary resources and infrastructure to carry out this work.

Finally, I extend my gratitude to my peers and faculty members who contributed directly or indirectly to the successful completion of this project.


THANK YOU