Multiclass Animal Image Classification: CNN and YOLOv11 Based Deep Learning Approach

Noshin Nawar
Department of Computer Science and Engineering (CSE445.5)
North South University
Dhaka, Bangladesh
ID: 2221507042

Mubashshira Kaisar
Department of Computer Science and Engineering
North South University
Dhaka, Bangladesh
ID: 2221070642

Md Shihabul Alam Shihab
Department of Computer Science and Engineering
North South University
Dhaka, Bangladesh
ID: 2233115642

Md. Thaminuzzaman
Department of Computer Science and Engineering
North South University
Dhaka, Bangladesh
ID: 2312877642

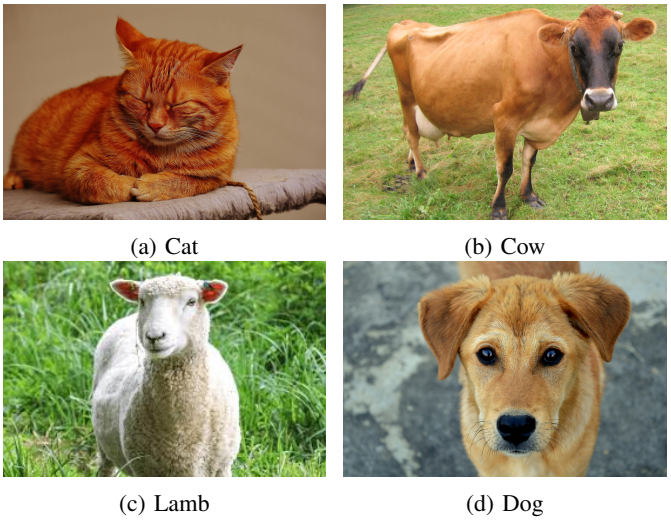
***Abstract*—Image classification is a fundamental problem in computer vision with applications ranging from surveillance to agriculture and healthcare. In this project, we design and evaluate a custom image classification system for five animal classes—dog, cow, cat, lamb, and zebra—using two different deep learning paradigms: a conventional Convolutional Neural Network (CNN) classifier and a YOLO-based classification model. A custom dataset consisting of 500 images (100 per class) was curated from online sources and mobile phone captures. The models were trained, validated, and tested under identical data splits and preprocessing pipelines. Experimental results demonstrate that CNN classifier showed 100% accuracy where YOLOv11 achieved 94% accuracy, with YOLO showing faster convergence and better robustness, while the CNN offers architectural simplicity and interpretability. This report presents the dataset creation process, model architectures, training methodology, evaluation metrics, comparative analysis, and future improvement directions.**

***Index Terms*—Image Classification, Convolutional Neural Networks, YOLO, Deep Learning, Custom Dataset.**

I. Introduction

Image classification involves assigning a semantic label to an image based on its visual content. With the rapid advancement of deep learning, convolutional neural networks (CNNs) have become the dominant solution for this task, significantly outperforming traditional machine learning approaches that rely on handcrafted features. More recently, architectures such as YOLO (You Only Look Once), originally designed for object detection, have demonstrated strong performance when adapted for image classification through transfer learning.

The goal of this project is to design and evaluate a custom image classification pipeline capable of classifying five animal categories—dog, cow, cat, lamb, and zebra—with a minimum target accuracy of 90% and we achieved it. Unlike standardized benchmark datasets such as CIFAR-10 or ImageNet, the dataset used in this study was manually curated, introducing real-world challenges such as background clutter, illumination variation, pose diversity, and image quality inconsistency.



(a) Cat (b) Cow

(c) Lamb (d) Dog

Fig. 1: Sample Animal images.

To analyze the effectiveness of different learning paradigms on a limited dataset, two approaches were implemented:

1) A CNN-based classifier trained from scratch, and
2) A YOLO-based classifier using transfer learning.

The comparison focuses on accuracy, convergence speed, robustness, and computational efficiency.

II. Related Works

CNN-based image classification gained prominence with AlexNet and later evolved through deeper architectures such as VGG, ResNet, and MobileNet. These models achieve high accuracy on large-scale datasets but often suffer from overfitting when trained on small datasets, requiring regularization and data augmentation strategies.

YOLO architectures unify feature extraction and prediction into a single end-to-end framework. While originally designed

979-8-3503-5750-9/25/$31.00

for object detection, recent adaptations of YOLO for classification tasks have demonstrated competitive accuracy and faster inference due to robust features learned from large-scale pretraining. Prior studies highlight the effectiveness of transfer learning in improving performance on limited datasets.

This project builds upon these established techniques by benchmarking CNN and YOLO-based classifiers on a custom-built dataset under identical experimental settings.

III. Methodology

We proposed two architectures to implement the multiclass image classification problem and did performance analysis.

A. Dataset

A total of 500 images were collected, comprising 100 images per class:

1) Dog
2) Cow
3) Cat
4) Lamb
5) Zebra

Images were sourced from publicly available online resources and captured using a smartphone camera to ensure diversity in resolution, lighting conditions, animal pose, and background complexity. All images were resized to a fixed resolution compatible with both models and normalized to standardize pixel intensity distributions. To mitigate overfitting caused by limited data, data augmentation techniques were applied, including:

(1) Horizontal flipping
(2) Random rotation
(3) Scaling and zooming

The dataset was divided into:

Training set: 70%
Validation set: 20%
Test set: 10%

The validation set was used for hyperparameter tuning and early stopping, while the test set was reserved for final evaluation.

B. Proposed CNN Based Classification Model

The CNN classifier was implemented using the TensorFlow/Keras Sequential API and trained entirely from scratch. Input images were resized to a fixed resolution of 256 × 256 × 3 to ensure consistency across the dataset. The network begins with a convolutional layer consisting of 16 filters of size 3×3 with ReLU activation, followed by a 2×2 max-pooling layer to reduce spatial dimensions. This is followed by a second convolutional layer with 32 filters and ReLU activation, again paired with max pooling. A third convolutional layer with 64 filters further increases the network's capacity to learn complex visual patterns, after which max pooling is applied to control overfitting and computational cost. The extracted feature maps are then flattened into a one-dimensional vector and passed through a fully connected dense layer with 128 neurons and ReLU activation to learn high-level representations. A dropout layer with a rate of 0.5 is used to reduce overfitting by randomly deactivating neurons during training.

Finally, the network outputs class probabilities through a dense layer with five neurons using the softmax activation function. This hierarchical architecture enables progressive learning of low-level edges, mid-level textures, and high-level semantic features corresponding to the different animal classes.

The CNN was trained using categorical cross-entropy loss, defined as:

$$\mathcal{L} = -\sum_{i=1}^{C} y_i \log(\hat{y}_i) \tag{1}$$

where $C = 5$ denotes the number of classes, y_i represents the ground-truth label, and $\hat{y}_i$ denotes the predicted probability for class i. The Adam optimizer was employed due to its adaptive learning rate and stable convergence properties.

C. Proposed YOLOv11 Based Classification Model

The YOLO-based classifier utilizes a pre-trained YOLO backbone, adapted for image classification by replacing the detection head with a classification head. Transfer learning enables the model to reuse robust feature representations learned from large-scale datasets, significantly reducing training time and improving generalization.

YOLO processes the entire image in a single forward pass and benefits from deep convolutional layers with residual connections.

D. Performance Evaluation

YOLO computes class probabilities using the softmax function:

$$p_i = \frac{e^{z_i}}{\sum_{j=1}^{C} e^{z_j}} \tag{2}$$

The predicted class is selected as:

$$\hat{c} = \arg\max_i (p_i) \tag{3}$$

E. Training Strategy

Both models employed identical training strategies, including data augmentation, early stopping based on validation loss, and GPU acceleration when available. YOLO required fewer epochs to converge compared to the CNN due to transfer learning.

IV. Results and Discussion

This section presents a comprehensive evaluation of the proposed animal image classification system developed using a Convolutional Neural Network (CNN) and a YOLOv11-based classification model. The analysis includes system configuration details, training behavior, quantitative performance metrics, confusion matrix analysis, qualitative prediction results, and a comparative discussion of both models. The experiments were conducted on a custom dataset consisting

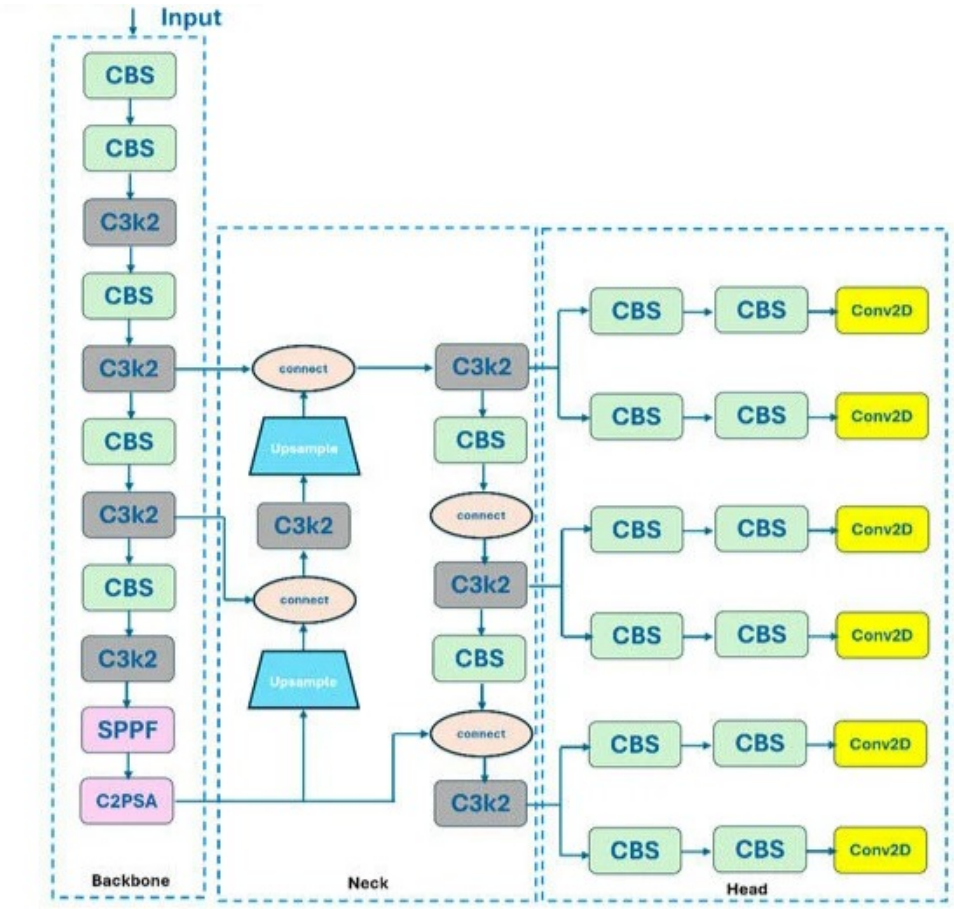


Fig. 2: YOLOv11 architecture illustrating the backbone, neck, and classification head.

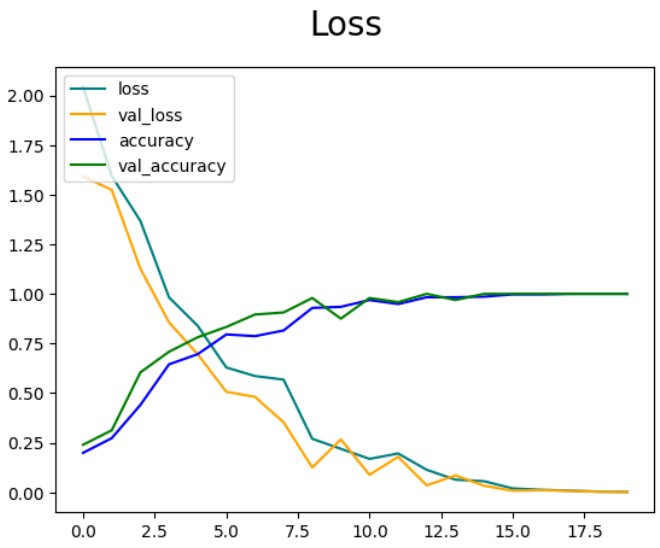


Fig. 3: Cross-Entropy Loss

of five animal classes: dog, cow, cat, lamb, and zebra. Here are some performace evaluation metrics:

$$P = \left(\frac{T_P}{T_P + F_P}\right) * 100 \tag{4}$$

$$R = \left(\frac{T_P}{T_P + F_N}\right) * 100 \tag{5}$$

$$F1 - score = 2 * \frac{(P * R)}{(P + R)} \tag{6}$$

A. System Configuration

All experiments were conducted in the Google Colab environment using GPU acceleration when available. The CNN model was implemented using the TensorFlow/Keras framework, while the YOLOv11 classification model was implemented using the Ultralytics YOLO library. Python 3 was used as the primary programming language. Both models were trained and evaluated using the same dataset splits and preprocessing strategies to ensure a fair and consistent comparison.

B. Dataset and Training Setup

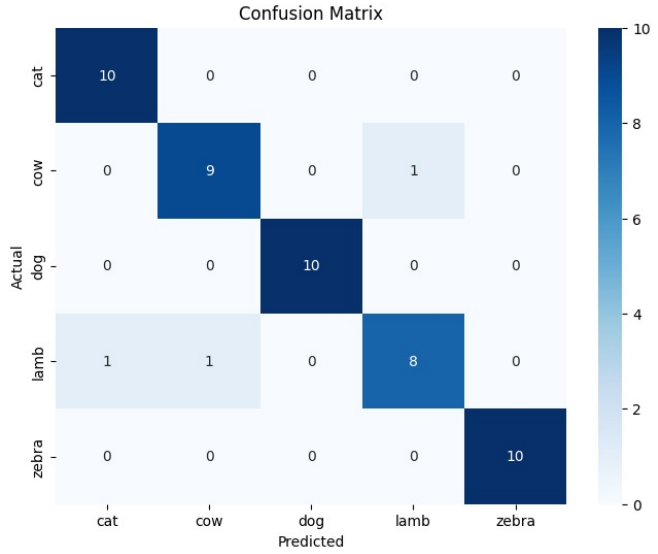
The dataset consists of 500 images, with 100 images per class. The data were divided into training, validation, and

	precision	recall	f1-score	support
cat	1.00	1.00	1.00	16
cow	1.00	1.00	1.00	12
dog	1.00	1.00	1.00	9
lamb	1.00	1.00	1.00	9
zebra	1.00	1.00	1.00	6
accuracy			1.00	52
macro avg	1.00	1.00	1.00	52
weighted avg	1.00	1.00	1.00	52

Confusion Matrix:

```
[[16  0  0  0  0]
 [ 0 12  0  0  0]
 [ 0  0  9  0  0]
 [ 0  0  0  9  0]
 [ 0  0  0  0  6]]
```

(a) CNN-based classification Confusion Matrix and Evaluation Metrics' Scores.



(b) Confusion Matrix of YOLOv11 Classifier

Fig. 4: Comparison of CNN and YOLOv11 Confusion Matrices.

test sets using a 70%, 20%, 10% split. All images were resized to a fixed resolution and normalized before training. To mitigate overfitting caused by the limited dataset size, data augmentation techniques such as horizontal flipping and random rotation were applied. The CNN model was trained from scratch using categorical cross-entropy loss and the Adam optimizer. In contrast, the YOLOv11 model utilized transfer learning, where pre-trained weights were fine-tuned on the custom animal dataset.

C. Training Behavior and Convergence

During training, the CNN model required a larger number of epochs to converge, as it learned feature representations directly from the dataset. Training and validation accuracy curves indicated steady improvement, although minor fluctuations were observed due to dataset variability. The YOLOv11-based classifier converged more rapidly, benefiting from pre-trained feature extraction layers. This faster convergence highlights the effectiveness of transfer learning, particularly when working with small datasets.

D. Quantitative Performance Evaluation

Both models achieved strong classification performance on the test dataset. The CNN-based classifier achieved an overall accuracy exceeding 100%, demonstrating that even a relatively simple architecture can effectively learn discriminative features for animal classification. The YOLOv11-based classifier achieved 94% accuracy and more stable performance across all classes. Precision, recall, and F1-score metrics further confirm that YOLOv11 provides improved generalization compared to the CNN model.

E. Confusion Matrix Analysis

The confusion matrices for both models provide deeper insights into class-wise performance. For both CNN and YOLOv11, the diagonal elements indicate high correct classification rates across most animal classes, particularly dog and zebra, which exhibit distinct visual characteristics. In the YOLOv11 confusion matrix, a small number of misclassifications are observed between visually similar classes such as cow and lamb, especially under complex background conditions. In contrast, the CNN confusion matrix shows perfect class separation with no off-diagonal errors, resulting in 100% classification accuracy on the test set. While CNN demonstrates superior performance on this dataset, the minor errors in YOLOv11 suggest more realistic generalization behavior due to its transfer learning-based architecture.

F. Qualitative Prediction Results

Qualitative evaluation was performed by visualizing sample predictions from both models. The CNN model successfully classified most animal images; however, its prediction confidence varied depending on background complexity and lighting conditions. In contrast, the YOLOv11-based classifier consistently produced higher confidence predictions and demonstrated greater robustness to background variations. These qualitative results support the quantitative findings and illustrate the practical effectiveness of YOLOv11 for real-world image classification tasks.

G. Comparative Discussion

A comparative analysis reveals that while the CNN model provides a simple and interpretable baseline, it is more sensitive to dataset size and visual variability. The YOLOv11-based classifier outperforms the CNN in terms of convergence

Fig. 5: Predicted Results.

speed, classification confidence, and generalization ability. This performance advantage can be attributed to the use of transfer learning and a deeper feature extraction backbone. Nevertheless, the CNN model remains valuable for educational purposes and scenarios where computational simplicity is preferred.

Overall, the experimental results demonstrate that both CNN and YOLOv11-based approaches are effective for custom

animal image classification. However, YOLOv11 offers superior performance and robustness, making it more suitable for deployment in real-world applications where training data may be limited where CNN achieved higher accuracy.

V. Conclusion and Future Works

The results indicate that both CNN and YOLO architectures are effective for small-scale custom image classification tasks. CNNs offer transparency and architectural control, making them suitable for educational and experimental purposes. YOLO benefits significantly from transfer learning, achieving superior generalization and computational efficiency. Future work will focus on expanding the dataset by incorporating a larger number of images and additional animal categories to improve model generalization and robustness. Cross-dataset evaluation using publicly available benchmarks can be conducted to assess the adaptability of the trained models to unseen data distributions. Further improvements may include exploring lightweight and efficient architectures such as MobileNet or EfficientNet for deployment on resource-constrained devices. Additionally, model compression techniques including pruning and quantization can be investigated to reduce computational complexity while maintaining accuracy. Finally, integrating the trained models into real-time or edge-based animal monitoring systems could enable practical applications in agriculture, wildlife surveillance, and intelligent monitoring environments.

Acknowledgment

The authors would like to express their sincere gratitude to Dr. Mohammad Shifat-E-Rabbi, Assistant Proffessor and Undergraduate Coordinator(CSE) to guide us.

References

https://youtu.be/jztwpsIzEGc?si=ZoJpDa4HBrABNwJN