

**SWINBURNE UNIVERSITY OF TECHNOLOGY**

COS30082 – Applied Machine Learning

Assignment: Bird Species Classification

Student Name: Nguyen Cuong Nhat

Student ID: 104178590

Instructor: Mr. Duong Trung Tin

Submission Date: 20/10/2024

Table of Contents

[I. Introduction 3](#_Toc180328389)

[II. Methodology 3](#_Toc180328390)

[1. Data Exploration and Preparation 3](#_Toc180328391)

[2. Traditional Machine Learning models 6](#_Toc180328392)

[3. Deep Learning models 7](#_Toc180328393)

[Model Architectures 7](#_Toc180328394)

[Loss Function 8](#_Toc180328395)

[Hyperparameters 8](#_Toc180328396)

[Training Process 8](#_Toc180328397)

[III. Result and Discussion 9](#_Toc180328398)

[1. Evaluation Results 9](#_Toc180328399)

[2. Discussion 10](#_Toc180328400)

[Overfitting Issues: 10](#_Toc180328401)

[Performance discrepancies between the models 11](#_Toc180328402)

[IV. Conclusion 12](#_Toc180328403)

[V. References 12](#_Toc180328404)

# I. Introduction

In this assignment, we explore multi-class classification using the Caltech-UCSD Birds 200 (CUB-200) dataset, a diverse collection of bird images covering 200 species primarily from North America. With 4,829 training images and detailed annotations, this dataset is a valuable resource for developing and testing classification models.

The main goal is to accurately classify the bird species in the dataset by applying various machine learning techniques. We'll experiment with a range of models, from traditional algorithms to more sophisticated deep learning architectures. The performance of these models will be evaluated using key metrics like Top-1 accuracy, which reflects how often the correct species is identified, and Average accuracy per class, offering insights into how well the models perform across all species.

A major challenge in this task is overfitting—when a model performs well on training data but struggles with new, unseen data. To counter this, we’ll implement techniques such as data augmentation, regularization, and cross-validation to improve the model's generalization and robustness.

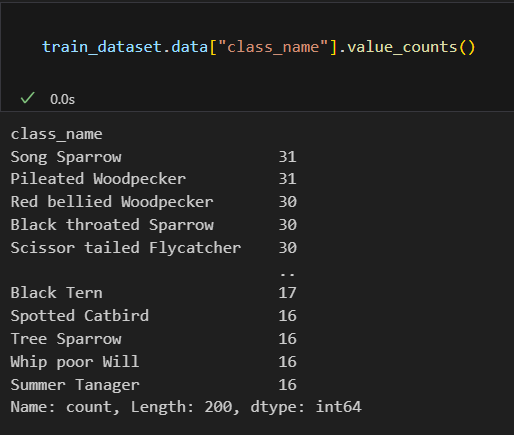
This report will also cover the preprocessing steps taken to prepare the data, the process of selecting and tuning hyperparameters, and a comparative analysis of model performance. Through this assignment, we aim to deepen our understanding of multi-class classification and evaluate the effectiveness of different machine learning approaches in solving this complex problem.

The implementation code and checkpoint of models used in the report can be found here: [*https://drive.google.com/drive/folders/1kwKk0ptwWB85edblpbP-pi4aOJJKWmfA?usp=sharing*](https://drive.google.com/drive/folders/1kwKk0ptwWB85edblpbP-pi4aOJJKWmfA?usp=sharing)

# II. Methodology

## 1. Data Exploration and Preparation

The dataset for this project is Caltech-UCSD Birds 200 (CUB-200), which contains images of 200 bird species primarily from North America. There are 4,829 images available for training, and 1204 images for testing.



*Figure 1: Distribution of training examples per class*

When logging the distribution of the number of training examples per class in the dataset, here is some finding:

* **Class Imbalance**: The dataset exhibits class imbalance. The highest number of examples for a class is 31 (e.g., Song Sparrow, Pileated Woodpecker), while the lowest visible count is 16 for several classes (e.g., Summer Tanager, Spotted Catbird). This suggests that certain classes are overrepresented while others have significantly fewer examples.
* If not addressed, this imbalance could cause bias in the model. Classes with more examples will likely have better predictions, while those with fewer examples may suffer from poor classification due to underrepresentation.

Also, when plotting some sample images, there are some issues that can affect the performance of the model:

* **Background Clutter:** Some images may have a dense background with many branches or camouflage effect, making the birds harder to distinguish. This can make it challenging for the model to focus on the bird features, especially if the background noise is the same across other classes. The model may mistakenly learn to associate certain background elements (like trees or branches) with a specific class.
* **Multiple Birds**: There are two birds in the image, and the model might struggle if it expects a single bird per image. Depending on how the dataset is curated and annotated, this can confuse the classification.
* **Small Object Size**: The birds appear relatively small in the image, occupying only a small portion of the frame. Models often perform worse when the object of interest is small because the model has fewer pixels to analyze the key features of the birds, which could lead to misclassification.



*Figure 2: Examples of images with Background Clutter*

*Figure 3: Examples of images contains multiple birds*

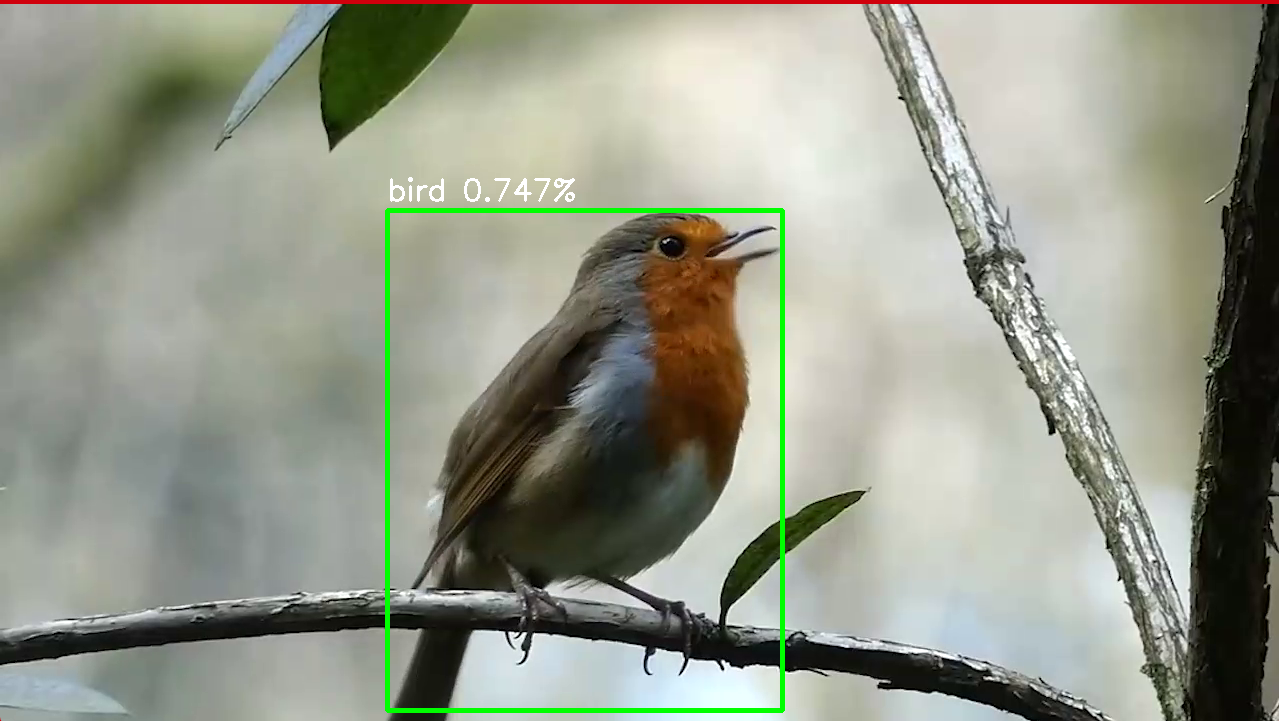
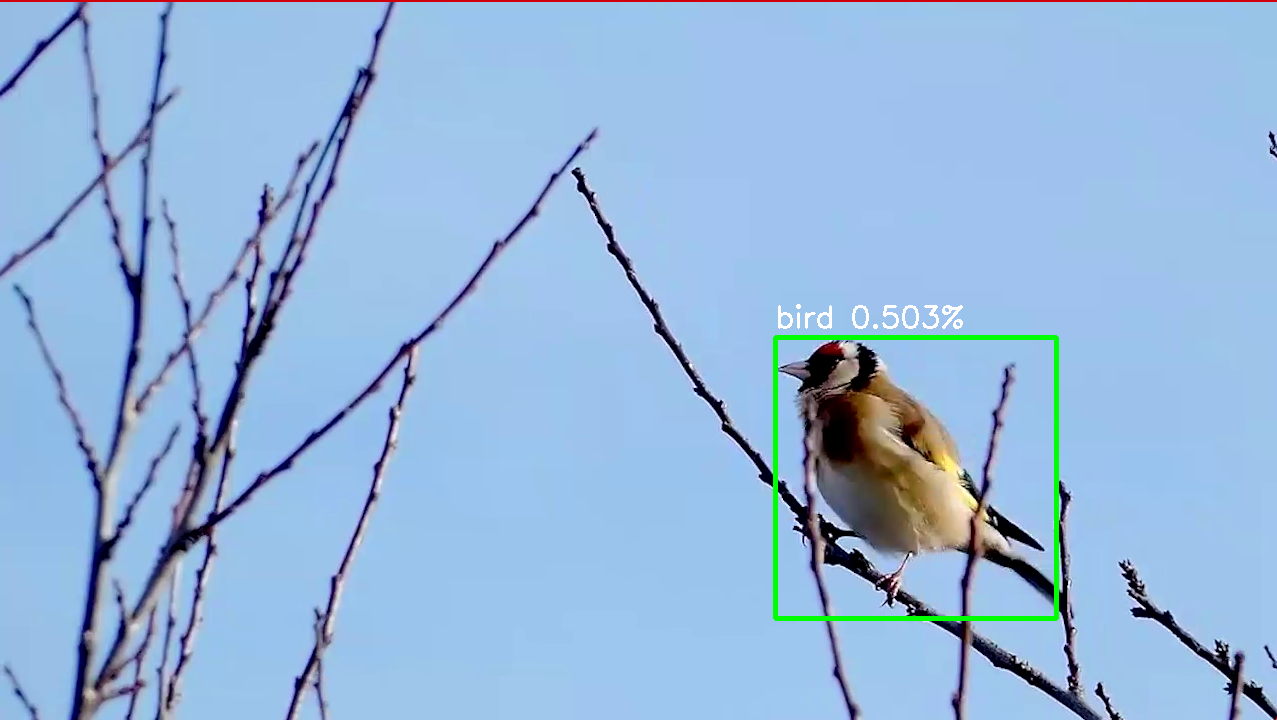
 

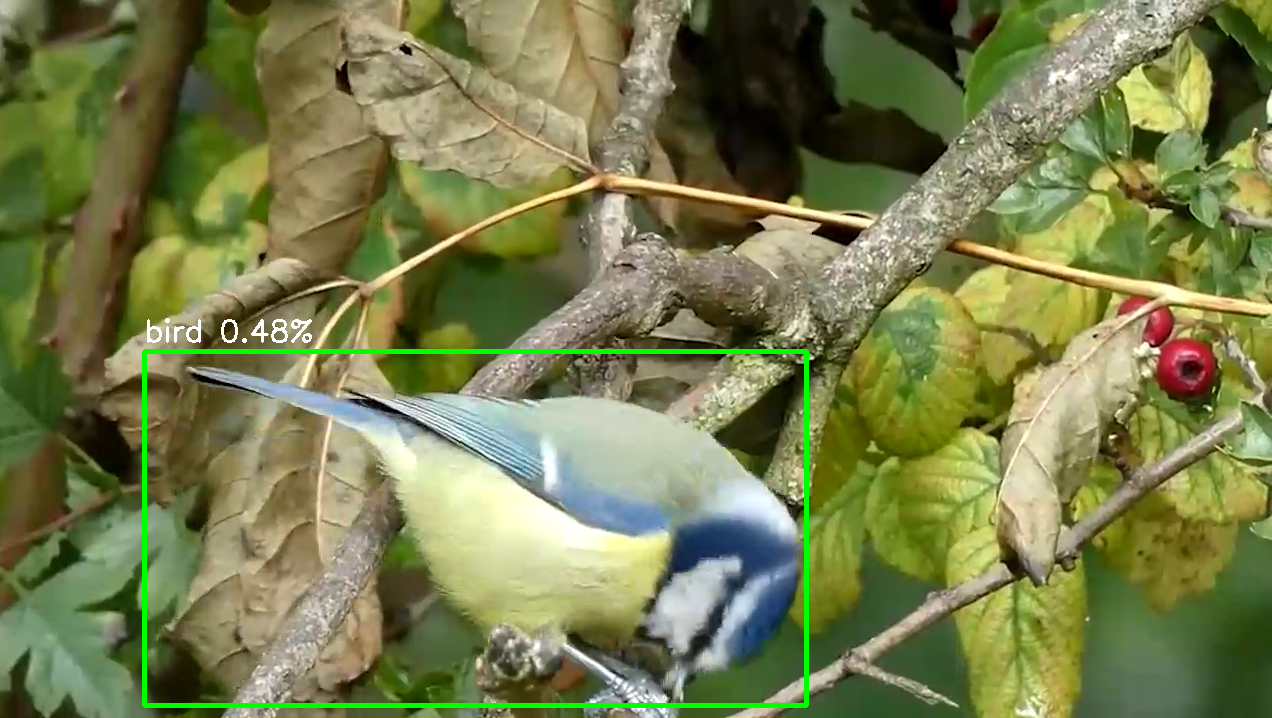
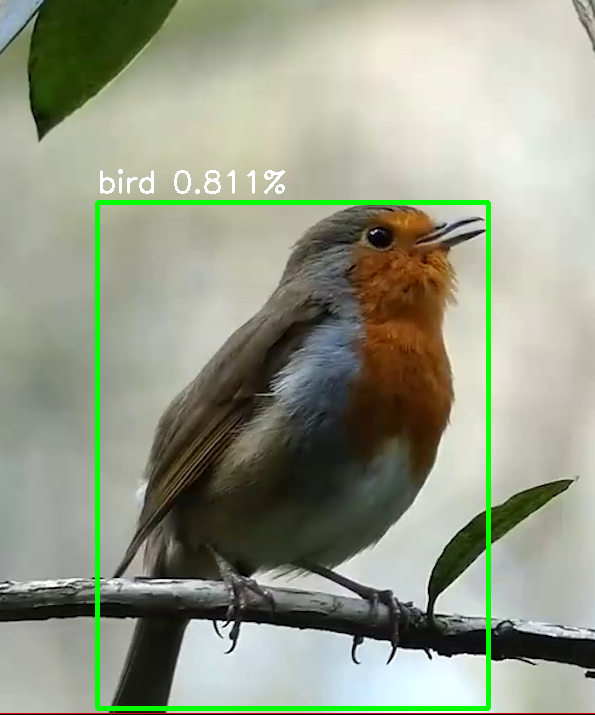
*Figure 4: Examples of images with small object(bird) size*

**Potential Solutions:**

* **Data Augmentation:** Techniques such as random cropping and zooming can encourage the model to focus on birds, reducing the influence of background noise.
* **Object Detection:** Implementing an object detection model to first identify the bird within an image, followed by cropping the detected region, ensures that the classifier receives inputs with minimized background clutter. This method enhances the model's ability to focus on the bird and improve classification accuracy.

For this assignment, a pretrained YOLOv8 model was utilized to detect birdswithin each image. Based on the predicted bounding boxes, the bird regions were cropped and saved, generating a new dataset devoid of background clutter, multiple birds, or small object sizes. This refined dataset was then used to evaluate the model's performance. During inference on the original dataset, each image is processed through the bird detection model to isolate the bird region, followed by the classification model to produce the final classification result.

*Figure 5: The bird detected by YOLOv8 and will be cropped from the bounding box*

## 2. Traditional Machine Learning models

Since the size of the dataset is quite small (about 6000 samples), we will try some machine learning model for classification. The selected models are:

* **Random Forest**: An ensemble learning method, was used for classification based on the features extracted by ResNet50. The model consists of 100 decision trees, each trained on random subsets of the data. The final decision is made by averaging the predictions of all the trees. This model was evaluated using Top-1 accuracy and per-class accuracy. The model's random nature ensures robustness and reduces overfitting.
* **Decision Tree:** The Decision Tree classifier is a simple, non-parametric model that recursively splits the data based on feature values, forming a tree structure for decision-making. It was applied to classify the extracted features. Despite being prone to overfitting in some cases, Decision Trees offer interpretability and relatively fast computation. The results were evaluated for both Top-1 accuracy and per-class accuracy.
* **Support Vector Machine:** The SVM classifier was applied using a linear kernel to classify the feature vectors. SVM works by finding the optimal hyperplane that maximizes the margin between different classes. It minimizes hinge loss, which helps separate classes effectively. The model is well-suited for high-dimensional data and provides strong classification performance with controlled regularization.

## 3. Deep Learning models

In the deep learning phase of the project, several state-of-the-art convolutional neural networks (CNNs) were implemented for the bird species classification task. The models included **ResNet50**, **EfficientNet (B0, B3, B5)**, and **MobileNetV2**. Each model's performance was assessed based on accuracy, generalization, and computational efficiency.

### Model Architectures

* **ResNet50**: A widely-used CNN architecture with residual connections to mitigate the vanishing gradient problem. It is known for its deep structure and ability to handle complex feature learning.
  + **Residual Blocks**: Each block contains a stack of convolutional layers where the output of the block is added to the input, creating a shortcut connection. This helps the network learn identity mappings, making deeper architectures feasible.
  + **50 Layers**: ResNet-50 has 50 layers, including convolutional layers, batch normalization, ReLU activation, and fully connected layers
  + **Activation Function**: ReLU is used after every convolutional layer.
  + **Global Average Pooling**: This layer reduces the spatial dimensions, feeding the output to the final softmax layer for classification.
* **EfficientNet**: EfficientNet family models (B0, B3, and B5) balance model accuracy and computational efficiency by scaling depth, width, and resolution. EfficientNet B5 offers the highest capacity for feature extraction among the tested variants. It has the following components:
  + **Convolutional Layers**: EfficientNet’s backbone consists of convolutional layers with depth-wise separable convolutions, which reduce the computational complexity.
  + **Squeeze-and-Excitation Blocks**: These blocks help the network recalibrate the feature maps by explicitly modeling channel-wise dependencies, allowing the network to focus on more important features.
  + **Swish Activation Function**: This non-linear activation function improves the performance over traditional activation functions like ReLU.
  + **Global Average Pooling Layer**: A global average pooling layer at the end reduces each feature map to a single value, avoiding overfitting.
  + **Fully Connected Layer**: The model outputs a probability distribution over 200 bird species (CUB-200-2011 dataset) through a fully connected layer followed by a softmax.
* **MobileNetV2**: A lightweight model optimized for mobile devices, offering a good trade-off between accuracy and efficiency.
  + **Inverted Residuals**: MobileNetV2 introduces an inverted residual structure where each residual block expands the input features to a higher dimension, applies depthwise separable convolutions, and then projects the features back to a smaller dimension.
  + **Bottleneck Layers**: These layers consist of pointwise convolutions that expand and contract the number of channels, which reduces the computation cost.
  + **Depth-wise Separable Convolutions**: These convolutions decompose the traditional convolution into two parts: depth-wise convolution (applied to each channel separately) and pointwise convolution (to combine the channels).
  + **Low Computational Cost**: MobileNetV2 is designed to be efficient in terms of parameters and FLOPs (Floating Point Operations), making it ideal for deployment on resource-constrained devices.
  + **ReLU6 Activation**: This activation function is used in MobileNetV2 to improve performance on low-precision hardware.

### Loss Function

The **CrossEntropyLoss** is used as the loss function. This is standard for multi-class classification tasks and computes the difference between the predicted class probabilities and the true class labels. It encourages the model to output high probabilities for the correct class while minimizing the probabilities for incorrect ones.

### Hyperparameters

* **Learning Rate**: Initially set to 0.001, the learning rate is adjusted dynamically using the ReduceLROnPlateau scheduler. This scheduler reduces the learning rate by a factor of 0.1 if the validation loss does not improve for 3 consecutive epochs.
* **Batch Size**: A batch size of 16 is used for both the training and validation data loaders.
* **Optimizer**: **AdamW** optimizer is employed, which is an extension of the Adam optimizer that decouples weight decay from the gradient update process. This allows for better regularization by applying weight decay only to the weights and not to the gradient-based parameter updates.
* **Weight Initialization**: All the deep learning models use pre-trained weights from models trained on ImageNet, allowing it to generalize better on the bird classification task.

### Training Process

The models were trained on both original dataset and preprocessed dataset (contains cropped images of birds), with each architecture fine-tuned to maximize accuracy.

* **Data Preprocessing**:
  + **Image Resizing**: To maintain consistency across the dataset, all images were resized to a fixed resolution of 256x256 pixels. This resizing ensures uniform input dimensions for the model, particularly in handling the varying sizes present in the original dataset, which ranged from smaller images to larger ones.
  + **Normalization**: The images were normalized based on the standard mean and standard deviation values used for the ImageNet dataset (mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225]). This process standardizes the pixel values, enhancing the model's stability and facilitating faster convergence during training.
* **Model Initialization**: The model is initialized and transferred to the GPU if available. The loss function and optimizer are initialized with the configuration discussed above.
* **Training Loop**: The model is trained for 25 epochs, with each epoch consisting of two phases:
  + **Training Phase**: The model is set to training mode, and input data is passed through the network. Loss is computed, and backpropagation is performed to update the model parameters. The accuracy is calculated at each epoch.
  + **Validation Phase**: After each training epoch, the model is evaluated on the validation set. No gradients are calculated during this phase. The validation loss and accuracy are recorded.
* **Learning Rate Scheduler**: Learning rate scheduler is used to reduce the learning rate, helping the model converge more effectively.
* **Saving the Best Model**: The model with the lowest validation loss is saved as the best model, ensuring that the final saved model performs optimally on unseen data.
* **Logging**: **TensorBoard** is used to log the training and validation losses, accuracies, and learning rate throughout the training process, enabling performance tracking and analysis.

# III. Result and Discussion

## 1. Evaluation Results

The performance of both traditional machine learning models and deep learning models was evaluated using Top-1 accuracy and Average accuracy per class. As seen from the table, the results show a clear performance discrepancy between traditional algorithms and deep learning models. The summary of the results is as follows:

|  |  |  |
| --- | --- | --- |
| **Machine Learning Models** | **Top-1 Accuracy (%)** | **Average Accuracy per Class (%)** |
| Random Forest | 47.64 | 46.38 |
| Decision Tree | 23.79 | 22.24 |
| Support Vector Machine | **67.57** | **66.42** |

*Table 1: Results of ML models*

|  |  |  |
| --- | --- | --- |
| **Model** | **Top-1 Accuracy (%)** | **Average Accuracy per Class (%)** |
| ResNet50 | 66.86 | 66.13 |
| MobileNet V2 | 59.22 | 58.57 |
| EfficientNet B0 | 67.61 | 66.83 |
| EfficientNet B3 | 74.67 | 74.24 |
| EfficientNet B5 | **77.82** | **77.38** |

*Table 2: Results of DL models on the original dataset*

|  |  |  |
| --- | --- | --- |
| **Model** | **Top-1 Accuracy** | **Average Accuracy per Class** |
| ResNet50 | 71.76 | 71.08 |
| MobileNet V2 | 74.83 | 74.37 |
| EfficientNet B0 | 81.64 | 81.41 |
| EfficientNet B3 | 81.73 | 81.51 |
| EfficientNet B5 | **81.79** | **81.66** |

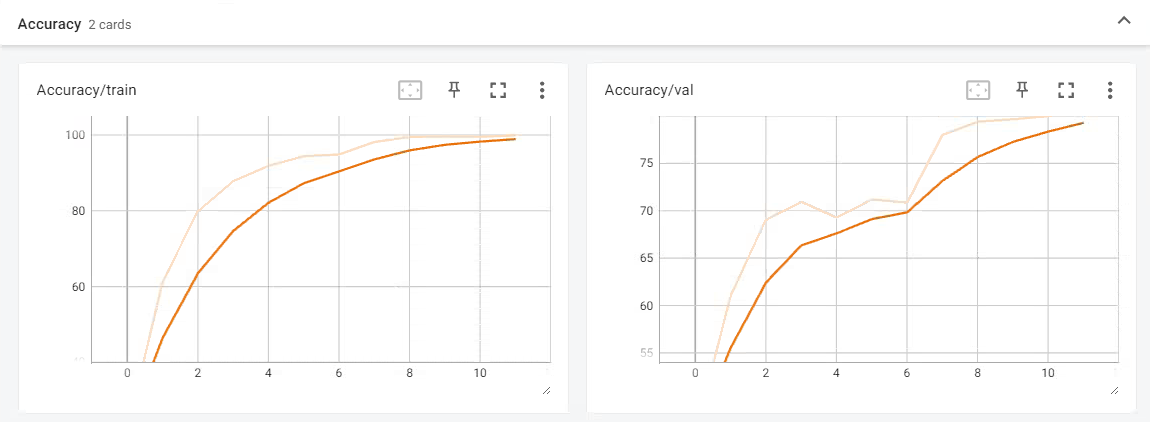
*Table 3: Results of DL models on the preprocessed dataset*

## 2. Discussion

### Overfitting Issues:

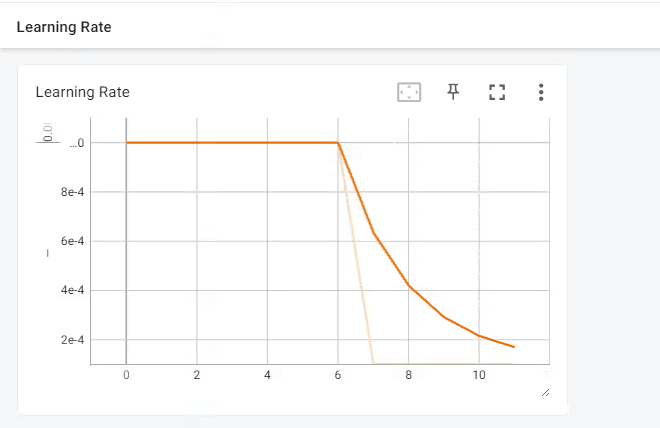
During the development and training of the model, overfitting emerged as a significant issue, where the model performed well on the training data but showed degraded performance on unseen data. Several strategies were implemented to mitigate overfitting, including:

* **Advance Data Processing:** To mitigate the problems related to background clutter, multiple birds, and small object size in bird classification, YOLOv8 was used to detect and isolate individual birds before classification. By cropping only the bird region and removing irrelevant or distracting background elements, these techniques ensure the model focuses on the bird itself rather than overfitting to specific background patterns that may not generalize well to new data.
* **Validation Set:** A validation set was used during training to monitor the model's performance on unseen data and adjust hyperparameters. Regular validation allowed for early detection of overfitting and provided insights into when to stop training before the model began to memorize the training data. The model with the best validation accuracy was selected as the final model. It was then tested on the test set, and the results were recorded for final evaluation.



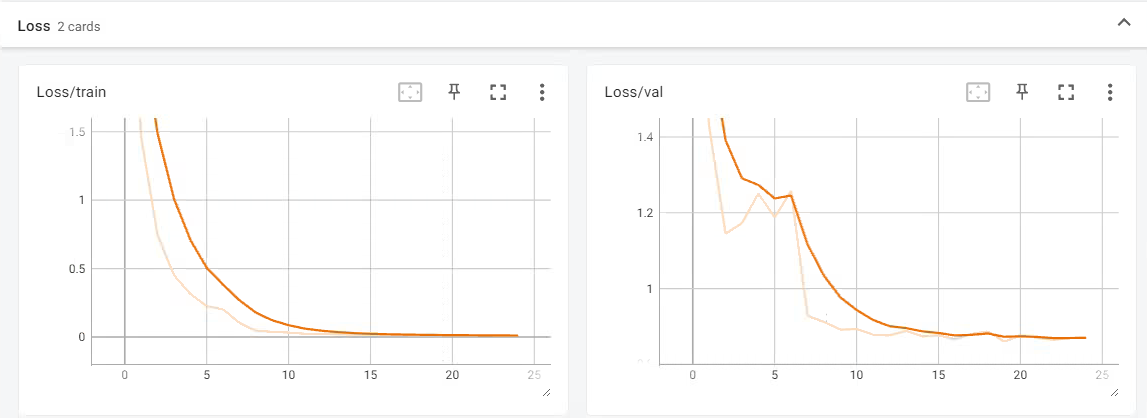
*Figure 6: Training and validation accuracy logged by TensorBoard*

* **Learning Rate Scheduler:** A learning rate scheduler was employed to reduce the learning rate when the model's performance plateaued. This helps the model converge more smoothly and prevents abrupt changes that could lead to overfitting.



*Figure 7: Learning rate reduced logged by TensorBoard*

* **Regularization:** L2 regularization (also known as weight decay) have been applied to penalize large weights in the model, promoting simpler and more generalizable patterns.



*Figure 8: Loss during training/validation process logged by TensorBoard*

### Performance discrepancies between the models

As seen from the table, the results show a clear performance discrepancy between traditional algorithms and deep learning models. The deep learning models significantly outperform traditional machine learning models due to their ability to automatically learn and extract hierarchical features from images. The discrepancies between the models can be attributed to several factors:

* **Model Architecture**: EfficientNet B5, due to its higher capacity and superior feature extraction, exhibited the highest accuracy. It balances depth, width, and resolution effectively, allowing it to capture finer details in the bird images. On the other hand, ResNet50 and MobileNet V2, while still competent, were less effective at handling the complexity of the dataset.
* **Dataset Preprocessing**: The application of YOLOv8 for bird detection and cropping significantly improved model performance across all deep learning architectures. By isolating the bird from the background clutter, the models were able to focus on the relevant features, reducing the noise that could otherwise mislead the classifier. This resulted in a marked improvement in accuracy for all models on the preprocessed dataset compared to the original dataset, and mitigate the overfitting issue.
* **Computational Efficiency**: While EfficientNet B5 achieved the highest accuracy, it comes at the cost of increased computational demand due to its larger architecture. In contrast, MobileNet V2, optimized for low-resource environments, offered a favorable trade-off between accuracy and computational efficiency, performing exceptionally well on the preprocessed dataset despite its smaller size.
* Among the models, **EfficientNet B5** yielded the best results due to its sophisticated scaling mechanism, which adjusts depth, width, and resolution based on the available computational resources. This allowed the model to extract more detailed features from the bird images, particularly after preprocessing. The incorporation of squeeze-and-excitation blocks further enhanced its ability to focus on the most relevant features of each bird species.

# IV. Conclusion

In conclusion, this report has presented a comprehensive exploration of bird species classification using the CUB-200 dataset. Various machine learning models, including both traditional algorithms and deep learning architectures, were employed to tackle the classification challenge. Through the comparative analysis of these models, it was observed that deep learning models, particularly EfficientNet B5, outperformed traditional approaches in terms of accuracy and robustness. The preprocessing techniques, such as using the YOLOv8 model to isolate bird regions, significantly improved classification performance by reducing background noise and addressing overfitting issues. These findings highlight the importance of advanced deep learning models and effective data preprocessing in solving complex classification tasks. Future work may involve exploring additional model optimization strategies and further refinement of preprocessing techniques to enhance the system’s performance and scalability.

# V. References

Tan, M., & Le, Q. (2019, May). Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning* (pp. 6105-6114). PMLR.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 770-778).

Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., ... & Adam, H. (2017). MobileNets: efficient convolutional neural networks for mobile vision applications (2017). *arXiv preprint arXiv:1704.04861*, *126*.

Roslan, R., Nazery, N. A., Jamil, N., & Hamzah, R. (2017, October). Color-based bird image classification using Support Vector Machine. In *2017 IEEE 6th Global Conference on Consumer Electronics (GCCE)* (pp. 1-5). IEEE.