

**Report**  
**Disha (2022CS11118) and Avilasha Mandal (2022CS11631)**

**Abstract:**

The model was trained using an 80-20 split of the dataset into training and validation sets. A Convolutional Neural Network (CNN) architecture with four convolutional layers, followed by ReLU activation, pooling, and batch normalization, was implemented. To address class imbalance, class weights were assigned inversely proportional to class frequencies.

A fully connected layer (MLP) was added, with the CrossEntropyLoss function used for classification (which automatically applies softmax).

The Adam optimizer was employed for efficient parameter updates. Data augmentation techniques, including rotation and flipping, were applied.

Early stopping was implemented to prevent overfitting, and a dynamic learning rate schedule was used.

Finally, Grad-CAM was utilized to visualize the regions of the image that contributed most to the model's predictions.

**Model Architecture: BirdClassifier**

Layer Type	Output Size (Height x Width x Channels)	Number of Parameters	Description
<b>Input Image</b>	64 x 64 x 3	0	Input image of size 64x64 with 3 color channels (RGB).
<b>Conv2d</b> (Conv1)	64 x 64 x 32	896 ( $3 * 3 * 32 + 32$ )	Convolutional layer with 32 filters, 3x3 kernel, stride 1, padding 1, followed by BatchNorm and ReLU activation.
<b>MaxPool2d</b> (MaxPool1)	32 x 32 x 32	0	Max pooling layer with 2x2 kernel, stride 2.
<b>Conv2d</b> (Conv2)	32 x 32 x 64	18,496 ( $32 * 3 * 3 * 64 + 64$ )	Convolutional layer with 64 filters, 3x3 kernel, stride 1, padding 1, followed by BatchNorm and ReLU activation.
<b>MaxPool2d</b> (MaxPool2)	16 x 16 x 64	0	Max pooling layer with 2x2 kernel, stride 2.
<b>Conv2d</b> (Conv3)	16 x 16 x 128	73,856 ( $64 * 3 * 3 * 128 + 128$ )	Convolutional layer with 128 filters, 3x3 kernel, stride 1, padding 1, followed by BatchNorm and ReLU activation.
<b>MaxPool2d</b> (MaxPool3)	8 x 8 x 128	0	Max pooling layer with 2x2 kernel, stride 2.
<b>Conv2d</b> (Conv4)	8 x 8 x 256	295,168 ( $128 * 3 * 3 * 256 + 256$ )	Convolutional layer with 256 filters, 3x3 kernel, stride 1, padding 1, followed by BatchNorm and ReLU activation.

Layer Type	Output Size (Height x Width x Channels)	Number of Parameters	Description
<b>MaxPool2d</b> (MaxPool4)	4 x 4 x 256	0	Max pooling layer with 2x2 kernel, stride 2.
<b>Flatten</b>	1 x 4096	0	Flatten layer to reshape the output to a 1D vector.
<b>Linear</b> (FC1)	1 x 256	1,048,576 (4096 * 256 + 256)	Fully connected layer with 256 units and ReLU activation.
<b>Dropout</b>	1 x 256	0	Dropout layer with a dropout rate of 0.4 to prevent overfitting.
<b>Linear</b> (FC2)	1 x 10	2,570 (256 * 10 + 10)	Fully connected output layer with 10 units (for 10 classes). Softmax is applied during inference.

Code to generate the architecture

```
import torch
from torchsummary import summary
import torch.nn as nn
# Set device to GPU if available, else CPU
device = torch.device("cuda" if torch.cuda.is_available() else "cpu")
# Define the BirdClassifier model structure
class BirdClassifier(nn.Module):
    def __init__(self, num_classes=10):
        super(BirdClassifier, self).__init__()
        self.features = nn.Sequential(
            nn.Conv2d(3, 32, kernel_size=3, stride=1, padding=1),
            nn.BatchNorm2d(32),
            nn.ReLU(),
            nn.MaxPool2d(kernel_size=2, stride=2),

            nn.Conv2d(32, 64, kernel_size=3, stride=1, padding=1),
            nn.BatchNorm2d(64),
            nn.ReLU(),
            nn.MaxPool2d(kernel_size=2, stride=2),

            nn.Conv2d(64, 128, kernel_size=3, stride=1, padding=1),
            nn.BatchNorm2d(128),
            nn.ReLU(),
            nn.MaxPool2d(kernel_size=2, stride=2),

            nn.Conv2d(128, 256, kernel_size=3, stride=1, padding=1),
            nn.BatchNorm2d(256),
            nn.ReLU(),
            nn.MaxPool2d(kernel_size=2, stride=2)
        )
        self.classifier = nn.Sequential(
            nn.Flatten(),
            nn.Linear(256 * 4 * 4, 256),
            nn.ReLU(),
            nn.Dropout(0.4),
            nn.Linear(256, num_classes)
        )
    def forward(self, x):
        x = self.features(x)
        x = self.classifier(x)
        return x
# Instantiate the model and load the weights
model = BirdClassifier(num_classes=10)
model.load_state_dict(torch.load("/kaggle/working/bird.pth", map_location=device)) # Load weights to the same device
model.to(device) # Move model to the specified device (GPU or CPU)

# Print the model summary
summary(model, input_size=(3, 64, 64), device=device.type) # Specify device type for summary
```

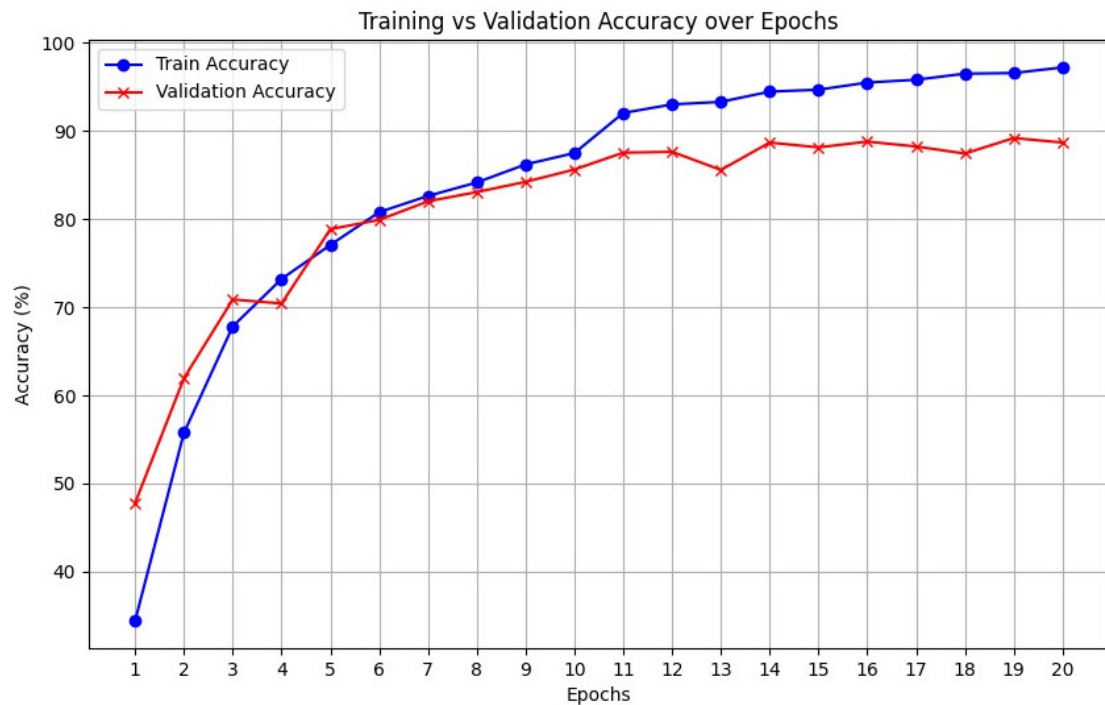
## Architecture generated via code

Layer (type)	Output Shape	Param #
Conv2d-1	[-1, 32, 64, 64]	896
BatchNorm2d-2	[-1, 32, 64, 64]	64
ReLU-3	[-1, 32, 64, 64]	0
MaxPool2d-4	[-1, 32, 32, 32]	0
Conv2d-5	[-1, 64, 32, 32]	18,496
BatchNorm2d-6	[-1, 64, 32, 32]	128
ReLU-7	[-1, 64, 32, 32]	0
MaxPool2d-8	[-1, 64, 16, 16]	0
Conv2d-9	[-1, 128, 16, 16]	73,856
BatchNorm2d-10	[-1, 128, 16, 16]	256
ReLU-11	[-1, 128, 16, 16]	0
MaxPool2d-12	[-1, 128, 8, 8]	0
Conv2d-13	[-1, 256, 8, 8]	295,168
BatchNorm2d-14	[-1, 256, 8, 8]	512
ReLU-15	[-1, 256, 8, 8]	0
MaxPool2d-16	[-1, 256, 4, 4]	0
Flatten-17	[-1, 4096]	0
Linear-18	[-1, 256]	1,048,832
ReLU-19	[-1, 256]	0
Dropout-20	[-1, 256]	0
Linear-21	[-1, 10]	2,570
Total params: 1,440,778		
Trainable params: 1,440,778		
Non-trainable params: 0		
Input size (MB): 0.05		
Forward/backward pass size (MB): 6.13		
Params size (MB): 5.50		
Estimated Total Size (MB): 11.67		

We wrote a code that shows the model optimization for the first 4 epochs

	Configuration	Validation Accuracy (%)
0	No Augmentation, No Regularization	86.244980
1	Horizontal Flip, L2 Regularization	86.897590
2	Horizontal Flip, Dropout (0.4)	84.889558
3	Flip + Resize Crop, L2 + Dropout	62.901606

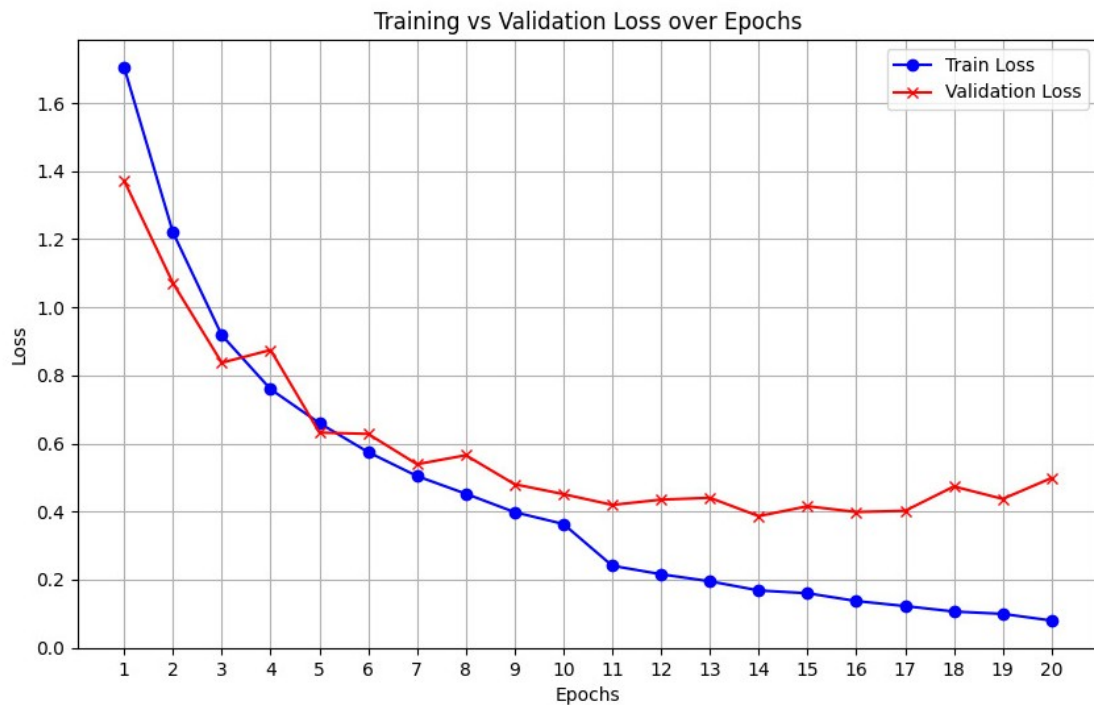
[+ Code](#)
[+ Markdown](#)



**Training accuracy** steadily increases from **34.44%** to **97.23%**. This is a typical pattern in model training, showing that the model is learning and adapting to the training data over time.

**Validation accuracy** increases consistently from **47.79%** to **89.21%** by epoch 19, with a slight drop to **88.70%** in epoch 20.

The overall increase in validation accuracy, without sharp fluctuations, suggests that the model is **generalizing well** and is not overfitting. The minor dip in the last epoch could be due to minor fluctuations in the learning process (or perhaps a slight instability from the training process or due to randomness).



**Training loss** decreases steadily from **1.7046** to **0.0797**, indicating that the model is continuously getting better at predicting the training data.

Since the training loss is decreasing and training accuracy is increasing, this means the model is learning effectively and fitting well to the training set.

**Validation loss** decreases from **1.3719** to **0.4984**, though with some fluctuations around epoch 12. However, the general downward trend still shows that the model is improving its generalization on the validation set.

The **validation loss fluctuations** (around epochs 12–14) are small and not a sign of significant overfitting. Instead, they may reflect minor variations as the model's learning process adapts to unseen data. The model is still managing to reduce the loss over time.

Grad-CAM Images Table

**Bird-Specific Features:** For accurate classification, it’s essential for the model to focus on unique bird characteristics such as head shape, beak, wings, and feather patterns. In most of these images, the model correctly identifies and highlights these areas, especially in images where the head and body are well-defined.

**Background Influence:** The model sometimes focuses on the background, especially in images where the bird blends with the environment or when the background has distinct textures. This suggests that the model might be learning some non-essential background features, potentially leading to misclassifications. Techniques like tighter cropping or background masking could help reduce this effect.

**Ecological Context-Based Bias:** In some images, the model appears to consider surrounding vegetation or habitat, which might reflect an ecological bias. This could be beneficial in natural scenes where certain species are linked to specific habitats, but it may not generalize well if backgrounds vary widely across test data.

**Improving Model Focus:** To enhance the model's discriminative power on bird-specific features, additional data augmentation (such as random cropping or varying backgrounds) might help. Training the model with a broader variety of backgrounds could also reduce reliance on environmental cues, leading to better generalization.

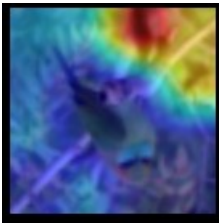
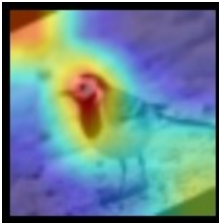
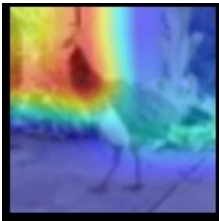
Image Index	CAM Image	Explanation
0		<p><b>CAM Interpretation:</b> The CAM shows significant activation in the background, with some focus on branches rather than any specific features of the bird.</p> <p><b>Insights:</b> This could indicate that the model is confused by background textures, possibly over-fitting to environmental patterns rather than bird-specific characteristics. This might lead to incorrect classifications if other classes share similar background elements. The model could benefit from more explicit bird-centric training to shift focus away from background textures.</p> <p>The model is recognizing bird feature inspire of it being inverted due to data augmentation</p>
1		<p><b>CAM Interpretation:</b> The CAM highlights the bird’s head and upper body, specifically the contrasting colors and textures around the face.</p> <p><b>Insights:</b> This focus is promising, as the head and facial markings are often unique identifiers for bird species. The model’s attention on these areas suggests it is learning relevant class-specific features, which helps improve accuracy for birds with distinctive head features. Minimal background influence here indicates the model is effectively isolating the bird for classification.</p>
2		<p><b>CAM Interpretation:</b> The CAM strongly highlights the intricate feather patterns on the bird's body.</p> <p><b>Insights:</b> This focus on feather texture is ideal for identifying species</p>



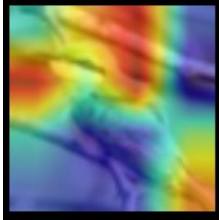

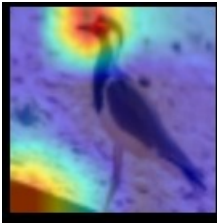
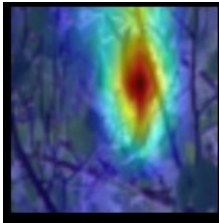
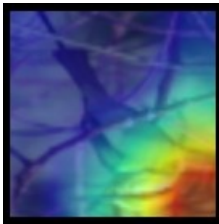
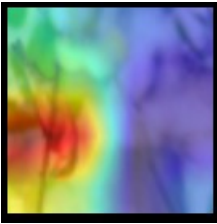

Image Index	CAM Image	Explanation
3		<p>with unique color patterns, as it suggests the model is learning to rely on class-specific features like color and pattern, rather than background cues. The model's attention to feather details over environmental features is a positive indicator of robust learning.</p> <p><b>CAM Interpretation:</b> The CAM shows attention on both the bird's head and surrounding foliage.</p> <p><b>Insights:</b> Although the head focus is beneficial, the model's attention on the foliage suggests it might be associating certain backgrounds with this class. This could cause confusion if similar foliage is present in other classes. Training the model to focus more strictly on bird-specific features rather than background elements may enhance its performance.</p>
4		<p><b>CAM Interpretation:</b> The CAM emphasizes the bird's head and beak area, with minimal background attention.</p> <p><b>Insights:</b> This is an ideal focus since the beak and head are often defining features for bird classification. The lack of background influence indicates that the model is isolating the bird well and is likely robust to environmental variations, which is beneficial for accurate classification.</p>
5		<p><b>CAM Interpretation:</b> The CAM predominantly highlights the bird's head, particularly around the eye and beak area.</p> <p><b>Insights:</b> This attention on the head, especially around the beak and eye, suggests the model is learning to identify unique facial features for this class. This is effective for birds with distinct head characteristics. Slight background attention could be minimized further, but overall, the focus is appropriate and contributes to accurate classification.</p>
6		<p><b>CAM Interpretation:</b> The CAM primarily focuses on the bird's wings and body, highlighting feather patterns while ignoring most of the background.</p> <p><b>Insights:</b> This is beneficial because the wing and feather patterns are likely distinguishing features for this species. The model's ability to focus on the body structure and feather details suggests it can effectively capture unique physical characteristics, which aids in accurate classification.</p> <p>The model is recognizing bird feature inspire of it being inverted due to data augmentation</p>
7		<p><b>CAM Interpretation:</b> The CAM emphasizes the bird's head and upper body, especially the coloration and eye area.</p> <p><b>Insights:</b> The model's focus on the head and color is advantageous, as these are often key differentiators in bird classification. The lack of background influence shows that the model is effectively concentrating on relevant features, which improves accuracy for species with distinct head and body colors.</p> <p>The model is recognizing bird feature inspire of it being inverted due to data augmentation</p>

Image Index	CAM Image	Explanation
8		<p><b>CAM Interpretation:</b> The CAM highlights the bird's beak and vibrant feather coloration on the body, with minimal background activation.</p> <p><b>Insights:</b> This focused attention on the beak and body coloration is highly effective, as these are distinguishing characteristics of this class. The model's lack of reliance on the background here indicates strong feature isolation, allowing it to accurately classify birds based on unique colors and shapes.</p>
9		<p><b>CAM Interpretation:</b> The CAM does not highlight, minimal background activation.</p> <p><b>Insights:</b> The model's lack of reliance on the background here indicates strong feature isolation, allowing it to accurately classify birds based on unique colors and shapes. The model is recognizing bird feature inspire of it being inverted due to data augmentation</p>

## Model Optimization

Optimization Technique	Description	Validation Accuracy	Impact on Performance
<b>Data Augmentation</b>	Horizontal/Vertical flips, Resizing, Normalization	Improved up to 47.79% in early epochs	Significantly increased generalization by introducing varied data, especially in the first few epochs (initial Val Accuracy reached 47.79%).
<b>L2 Regularization</b>	Weight decay with $1e-4$	Helped reach 88.70% by Epoch 20	Controlled overfitting, making the model more robust and aiding in achieving high final accuracy. Improved stability with minor accuracy fluctuations.
<b>Batch Normalization</b>	Applied to all convolution layers	Contributed to peak Val Accuracy of 89.21%	Improved convergence and stability, allowing higher learning rates and smoother training. Stabilized intermediate accuracy jumps across epochs.
<b>Dropout (0.4)</b>	Used in fully connected layers	Helped maintain Val Accuracy >87% consistently	Prevented overfitting by introducing noise, which resulted in higher validation consistency (Val Accuracy > 87% from Epoch 11).
<b>Learning Rate Scheduler</b>	Step decay every 10 epochs, $\gamma=0.5$	Final Val Accuracy stabilized at 88.70%	Allowed controlled training by reducing the learning rate after Epoch 10, facilitating final convergence and consistency in validation metrics.