BirdCLEF 2024 - Bird Sound Classification Project

Complete Documentation

Table of Contents

1. Project Overview

This project implements an **automatic bird species classification system** using deep learning techniques. The system processes audio recordings, converts them to mel-spectrograms, and uses Convolutional Neural Networks (CNNs) to classify bird species.

Dataset: BirdCLEF 2024 subset

- **10 bird species** (top 10 most common)
- **10,000 training samples** (1,000 per class)
- **Audio format**: 32 kHz sampling rate, 5-second clips
- **Train/Val split**: 80% / 20%

2. Data Preprocessing & Augmentation

2.1 Data Cleaning & Preparation

Quality Filtering:

```
# Filter recordings with rating >= 3 meta = meta[meta["rating"] >=
3].reset_index(drop=True)
```

Class Selection:

- Selected top 10 most frequent bird species
- Balanced dataset through oversampling to 1,000 samples per class

Train/Validation Split:

- 80% training (8,000 samples)
- 20% validation (2,000 samples)
- Stratified split to maintain class distribution

2.2 Audio Preprocessing Pipeline

```
Raw Audio → Load (32kHz) → Crop/Pad (5s) → Augmentation → Mel-Spectrogram →
SpecAugment → Normalize → Resize (224×224) → Model Input
```

Step-by-step Process:

1. **Audio Loading**: Load at 32 kHz, mono channel
2. **Duration Normalization**: Crop or pad to exactly 5 seconds
3. **Audio Augmentation** (training only):
    - **Time Shifting**: Random shift up to ±20% of duration
    - **Pitch Shifting**: Random pitch change ±2 semitones
    - **Additive Noise**: Gaussian noise with factor 0.002-0.01
    - **Time Stretching**: Random rate 0.8-1.25x
4. **Mel-Spectrogram Conversion**:

```
mel = librosa.feature.melspectrogram( y=y, sr=32000, n_mels=128, n_fft=2048, hop_length=512 ) mel_db = librosa.power_to_db(mel, ref=np.max)
```

5. **SpecAugment** (training only):
    - **Frequency Masking**: Mask up to 15% of frequency bins
    - **Time Masking**: Mask up to 20% of time steps
    - Applied 2 times per spectrogram
6. **Normalization**:
    - Min-max normalization to [0, 255]
    - Convert to 3-channel RGB image
    - ImageNet normalization (mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225])
7. **Resizing**: Resize to 224×224 pixels for model input

2.3 Handling Imbalanced Data

Problem: Original dataset had varying samples per class (437-467 samples)

Solution: Oversampling with replacement

```
def expand_dataframe_to_target(df, target_per_class=1000): for label, group in df.groupby('label_idx'): if len(group) >= target_per_class: sampled = group.sample(n=target_per_class, replace=False) else: sampled = group.sample(n=target_per_class, replace=True)
```

Result: Perfectly balanced dataset with 1,000 samples per class

3. Model Architectures

3.1 VGG-19 (From Scratch)

Architecture Overview

VGG (Visual Geometry Group) networks are characterized by their simplicity and depth, using only 3×3 convolutional filters throughout the network.

Architecture Diagram:

```
Input (224×224×3)
    ↓
[Conv Block 1]
  Conv3×3, 64 → ReLU
  Conv3×3, 64 → ReLU
  MaxPool 2×2
    ↓
[Conv Block 2]
  Conv3×3, 128 → ReLU
  Conv3×3, 128 → ReLU
  MaxPool 2×2
    ↓
[Conv Block 3]
  Conv3×3, 256 → ReLU
  Conv3×3, 256 → ReLU
  Conv3×3, 256 → ReLU
```

```
  Conv3×3, 256 → ReLU

  MaxPool 2×2
    ↓
[Conv Block 4]
  Conv3×3, 512 → ReLU
  Conv3×3, 512 → ReLU
  Conv3×3, 512 → ReLU

  Conv3×3, 512 → ReLU

  MaxPool 2×2
    ↓
[Conv Block 5]
  Conv3×3, 512 → ReLU
  Conv3×3, 512 → ReLU
  Conv3×3, 512 → ReLU

  Conv3×3, 512 → ReLU

  MaxPool 2×2
    ↓
Flatten
    ↓
FC 4096 → ReLU → Dropout(0.5)
    ↓
FC 4096 → ReLU → Dropout(0.5)
    ↓
FC 10 (num_classes)
    ↓
Softmax
```

Key Characteristics:

- **Total Layers**: 19 (16 conv + 3 FC, excluding pooling)

- **Parameters**: ~143 million
- **Receptive Field**: Built progressively through stacked 3×3 convolutions
- **Training**: From scratch (random initialization)

Pros:

- ☑ Simple and interpretable architecture
- ☑ Fast training due to smaller size
- ☑ Good baseline for comparison
- ☑ Easy to understand and modify

Cons:

- ❌ Shallower than other models
- ❌ No skip connections (vanishing gradient issues)
- ❌ Lower accuracy compared to modern architectures
- ❌ No pre-trained weights available

Reference: *Simonyan, K., & Zisserman, A. (2014). Very Deep Convolutional Networks for Large-Scale Image Recognition. arXiv:1409.1556.*

3.2 ResNet18 (Transfer Learning)

Architecture Overview

ResNet (Residual Network) introduced **skip connections** (residual connections) that allow gradients to flow directly through the network, enabling much deeper architectures.

Architecture Diagram:

```
    └──────┬──────┘
           ↓
[Residual Block 2] ×2
  ┌─────────────────┐
  │ Conv 3×3, 128   │
  │ Conv 3×3, 128   │
  └─────────────────┘
          │
    ┌─────┴─────┐
    │     +     │ ← Skip Connection (with 1×1 conv)
    └─────┬─────┘
          ↓
[Residual Block 3] ×2 (256 filters)
          ↓
[Residual Block 4] ×2 (512 filters)
          ↓
Global Average Pooling
    ↓
FC 10 (num_classes)
    ↓
Softmax
```

Residual Block Detail:

```
    x (input)
    │
    ├──────────────┐
    │              │
    ↓              │
Conv 3×3           │
    ↓              │
BatchNorm          │
    ↓              │
ReLU               │
    ↓              │
Conv 3×3           │
    ↓              │
BatchNorm          │
    ↓              │
    +  ←───────────┘ (Skip Connection)
    ↓
  ReLU
    ↓
  output
```

Key Characteristics:

- **Total Layers**: 18 (17 conv + 1 FC, excluding pooling and skip connection)
- **Parameters**: ~11.7 million

- **Skip Connections**: Identity mappings every 2 conv layers
- **Training**: Transfer learning from ImageNet (1000 classes)

Pros:

- ☑ Skip connections prevent vanishing gradients
- ☑ Stable and reliable training
- ☑ Strong performance across tasks
- ☑ Pre-trained on ImageNet (good feature extraction)
- ☑ Moderate computational cost

Cons:

- ❌ Heavier than MobileNet
- ❌ More parameters than needed for simple tasks
- ❌ Slower inference than lightweight models

Reference: *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 (CVPR), pp. 770-778. arXiv:1512.03385*

Why ResNet-18 Excels for Bird Audio

- Residual learning: Skip connections allow the network to learn subtle differences in bird calls without degradation in deep layers.
- Hierarchical feature extraction: Early layers capture simple frequency edges, while deeper layers detect complex temporal and harmonic patterns.
- Stable training: Residual blocks prevent vanishing gradients, making the network easier to optimize audio spectrograms.
- Efficient parameter usage: Fewer parameters than very deep networks like ResNet-50 or Inception, reducing overfitting on moderate-sized audio datasets.
- Robust generalization: The combination of convolutional layers and residual shortcuts helps capture both local and global spectrogram patterns across species.
- Adaptable to transfer learning: Pretrained weights from ImageNet can be fine-tuned effectively, leveraging learned visual patterns to spectrogram textures.

3.3 Inception V1 (Transfer Learning)

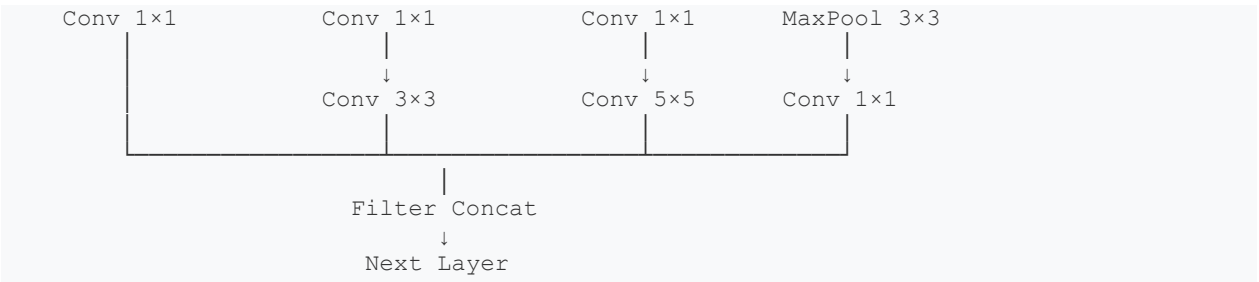
Architecture Overview

Inception V1 (GoogLeNet) uses **multi-scale feature extraction** through parallel convolutional filters of different sizes (1×1, 3×3, 5×5) in "Inception modules".

Architecture Diagram:

```
Input (224×224×3)
    ↓
Conv 7×7, 64, stride=2
    ↓
MaxPool 3×3, stride=2
    ↓
Conv 1×1, 64
Conv 3×3, 192
    ↓
MaxPool 3×3, stride=2
    ↓
[Inception Module]
    ↓
[Inception Module]
    ↓
MaxPool 3×3, stride=2
    ↓
[Inception Module] → Auxiliary Classifier 1
    ↓
[Inception Module]
    ↓
[Inception Module]
    ↓
[Inception Module] → Auxiliary Classifier 2
    ↓
[Inception Module]
    ↓
MaxPool 3×3, stride=2
    ↓
[Inception Module]
    ↓
[Inception Module]
    ↓
Global Average Pooling
    ↓
Dropout (0.4)
    ↓
FC 10 (num_classes)
    ↓
Softmax
```

Inception Module Detail:




```
Filters: [64, 128, 32, 32] (example configuration)
```

Key Characteristics:

- **Total Layers**: 62 (including inception modules)
- **Parameters**: ~6.8 million
- **Inception Modules**: 9 modules with multi-scale convolutions
- **Auxiliary Classifiers**: 2 intermediate classifiers for gradient flow
- **Training**: Transfer learning from ImageNet

Pros:

- ☑ **Multi-scale feature extraction** (excellent for spectrograms)
- ☑ Captures both fine details and broad patterns
- ☑ Efficient parameter usage (fewer than VGG)
- ☑ Auxiliary classifiers help training deep networks
- Cons:
- ❌ Complex architecture (harder to interpret)
- ❌ Auxiliary outputs require special handling
- ❌ More memory intensive during training

Reference: *Szegedy, C., Liu, W., Jia, Y., et al. (2015). Going Deeper with Convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1-9. arXiv:1409.4842*

3.4 MobileNetV3 (Transfer Learning)

Architecture Overview

MobileNetV3 is designed for mobile and edge devices using **depthwise separable convolutions** and inverted residual blocks with squeeze-and-excitation.

Architecture Diagram:

```
Input (224×224×3)
    ↓
Conv 3×3, 16, stride=2, hard-swish
    ↓
[MBConv Block 1] - Depthwise Sep Conv, SE, 16→16
    ↓
[MBConv Block 2] - Depthwise Sep Conv, SE, 16→24, stride=2
    ↓
[MBConv Block 3-11] - Depthwise Separable Convolutions
    ↓
Conv 1×1, 576, hard-swish
    ↓
Global Average Pooling
    ↓
Conv 1×1, 1024, hard-swish
    ↓
Dropout (0.2)
    ↓
Conv 1×1, 10 (num_classes)
    ↓
Softmax
```

MBConv Block (Inverted Residual + SE):

```
    x (input)
    │
    ├──────────────────────┐
    │                      │
    ↓                      │
Conv 1×1 (Expansion)       │
    ↓                      │
BatchNorm                  │
    ↓                      │
hard-swish                 │
    ↓                      │
Depthwise Conv 3×3         │
    ↓                      │
BatchNorm                  │
    ↓                      │
hard-swish                 │
    ↓                      │
[Squeeze-Excitation]       │
  GlobalAvgPool            │
  FC (reduce)              │
  ReLU                     │
  FC (expand)              │
  hard-sigmoid             │
  Multiply                 │
```

```
    ↓                          |
Conv 1×1 (Projection)      |
    ↓                          |
BatchNorm                  |
    ↓                          |
    +  ←———————————————————— (if stride=1 and same channels)
    ↓
  output
```

Key Characteristics:

- **Total Layers**: 11 MBConv blocks
- **Parameters**: ~2.5 million (Small variant)
- **Depthwise Separable Convolutions**: Factorized convolutions for efficiency
- **Squeeze-and-Excitation**: Channel-wise attention mechanism
- **Hard-swish Activation**: Efficient non-linearity
- **Training**: Transfer learning from ImageNet

Pros:

- ☑ **Fastest inference** (optimized for mobile)
- ☑ **Smallest model size** (~10 MB)
- ☑ Low memory footprint
- ☑ Efficient for deployment
- ☑ Good accuracy-to-size ratio

Cons:

- ❌ Slightly lower accuracy than larger models
- ❌ Less feature capacity
- ❌ May underfit on complex patterns

Reference: *Howard, A., Sandler, M., Chu, G., et al. (2019). Searching for MobileNetV3. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), pp. 1314-1324. arXiv:1905.02244*

4. Evaluation & Visualization

4.1 Evaluation Metrics

1. Accuracy

- Overall classification accuracy
- Percentage of correctly classified samples

2. Precision, Recall, F1-Score (Macro)

- **Precision**: TP / (TP + FP) - How many predicted positives are correct
- **Recall**: TP / (TP + FN) - How many actual positives are found
- **F1-Score**: Harmonic mean of precision and recall
- **Macro**: Unweighted average across all classes

3. Confusion Matrix

- Heatmap showing prediction patterns
- Diagonal: Correct classifications
- Off-diagonal: Misclassifications
- Helps identify confused species pairs

4. ROC Curve & AUC

- ROC curve for each class
- Macro-average ROC-AUC
- Visualizes true positive rate vs false positive rate

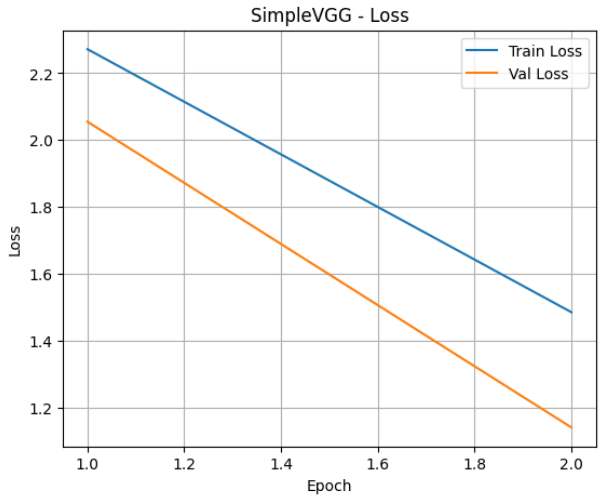
5. Training Curves

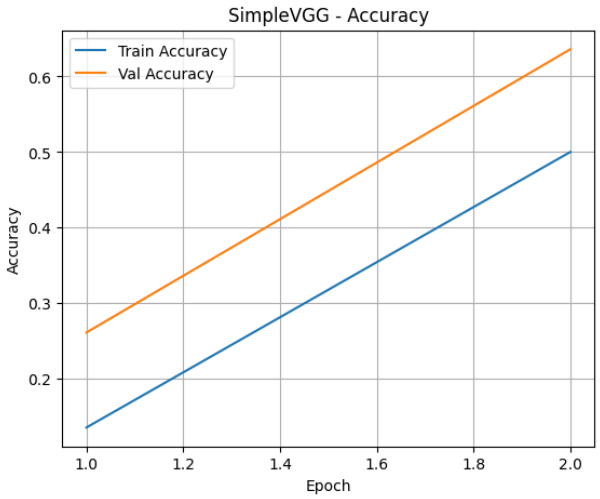
- Loss curves (training vs validation)
- Accuracy curves (training vs validation)
- Monitors overfitting/underfitting

4.2 Visualization

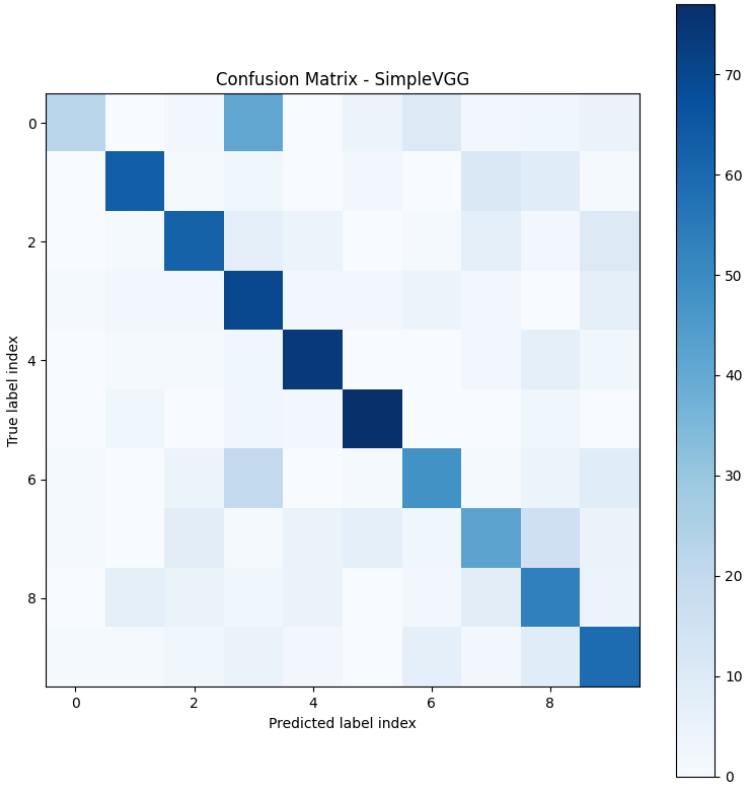
1. VGG 19

Training and Validation Accuracy and loss:

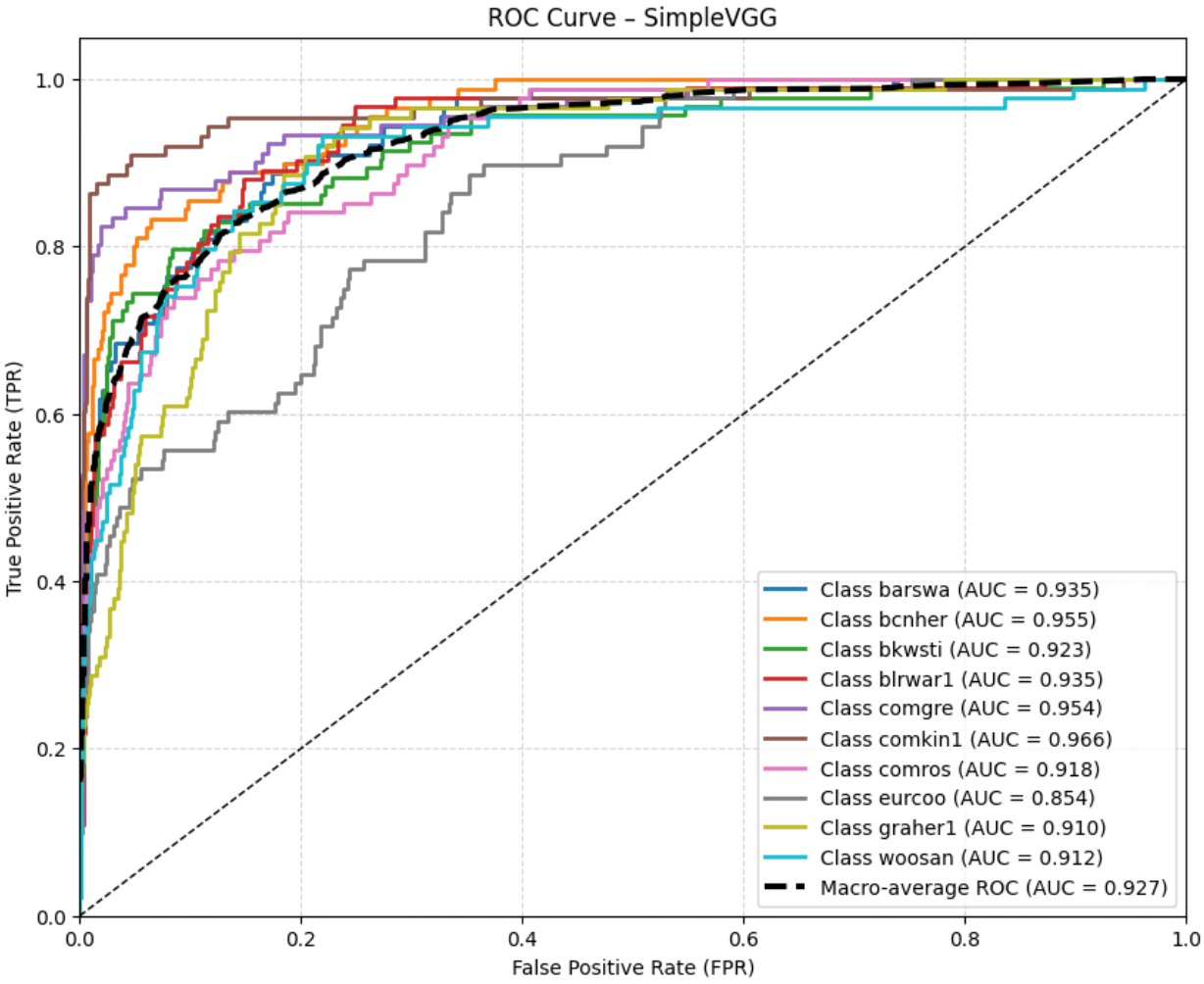




Confusion Matrix:

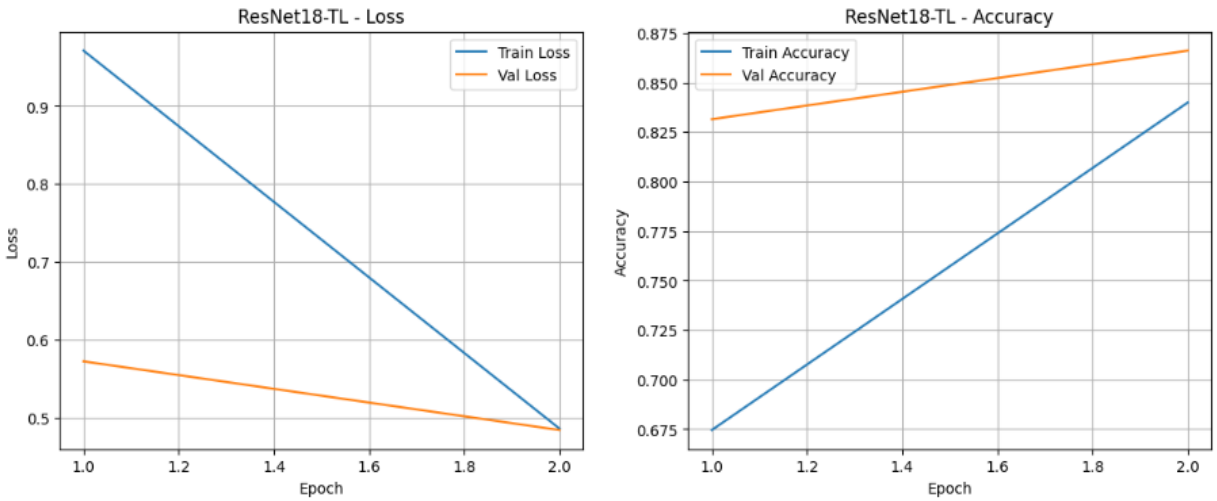


ROC and AUC Graph:

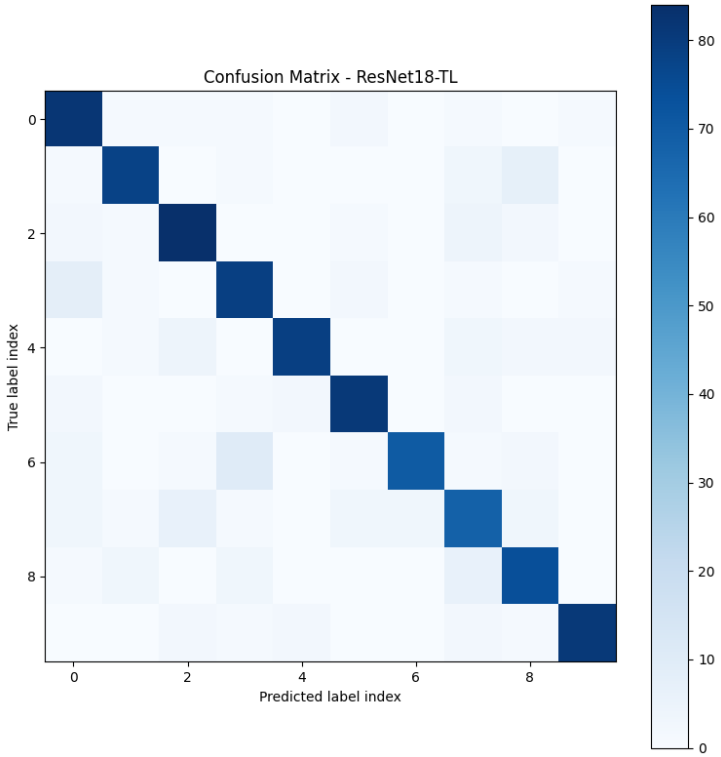


2. ResNet18

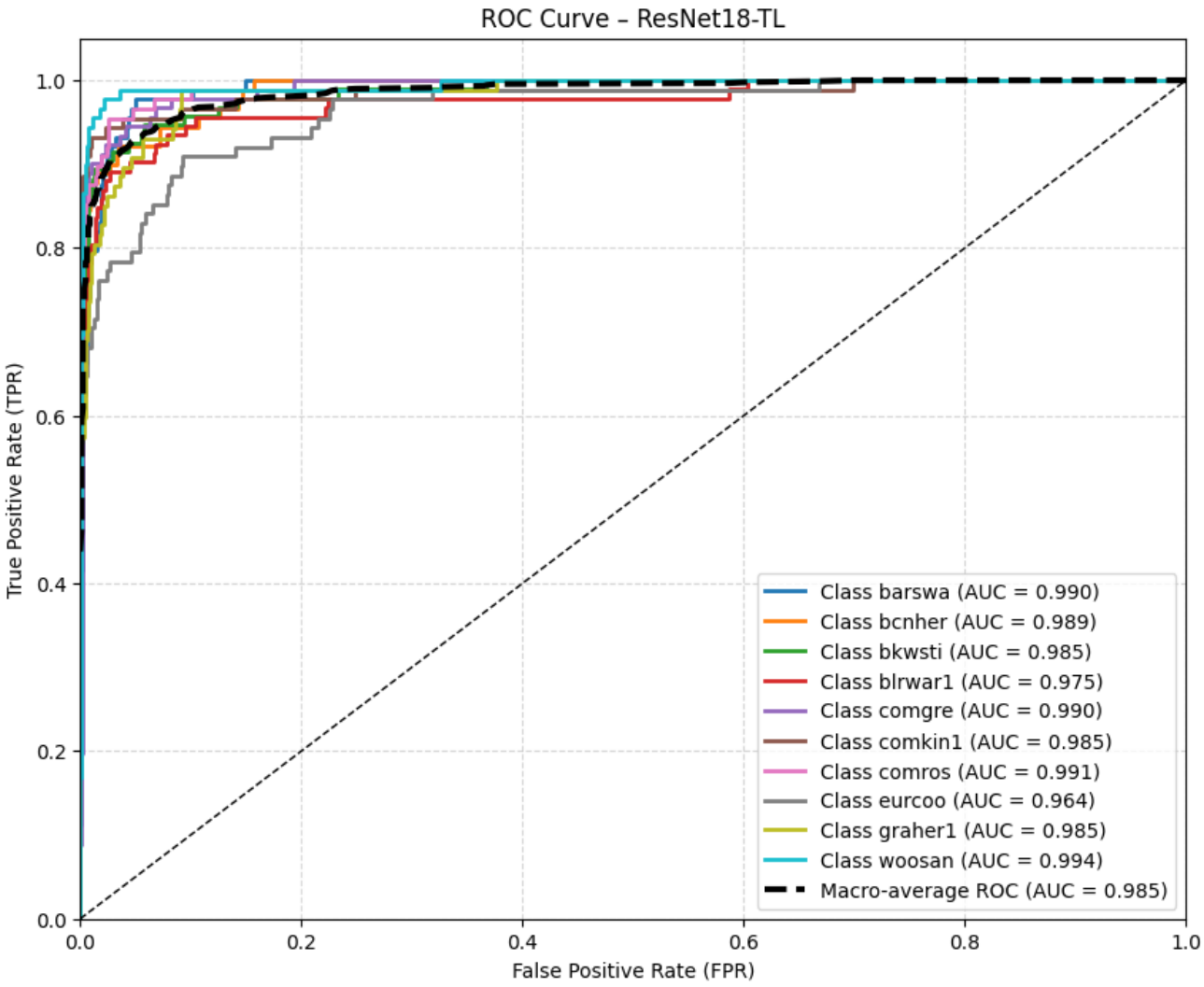
Training and Validation Accuracy and loss:



Confusion Matrix:

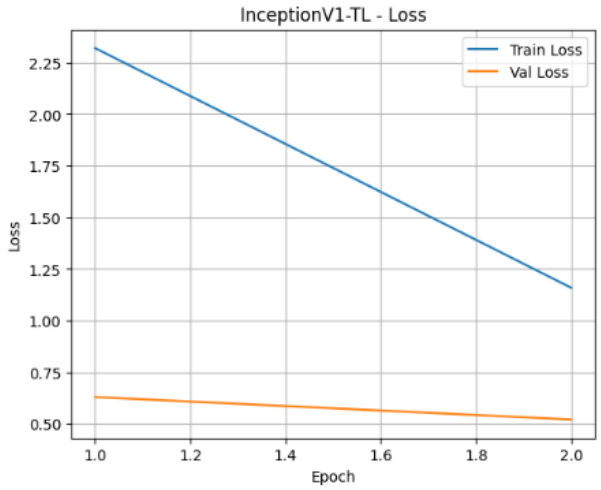


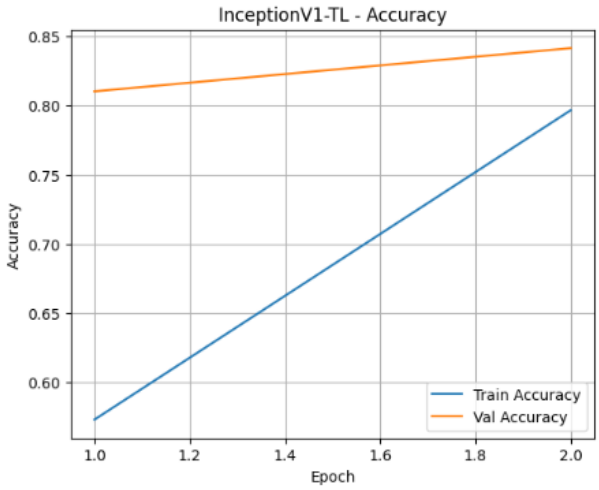
ROC and AUC Graph:



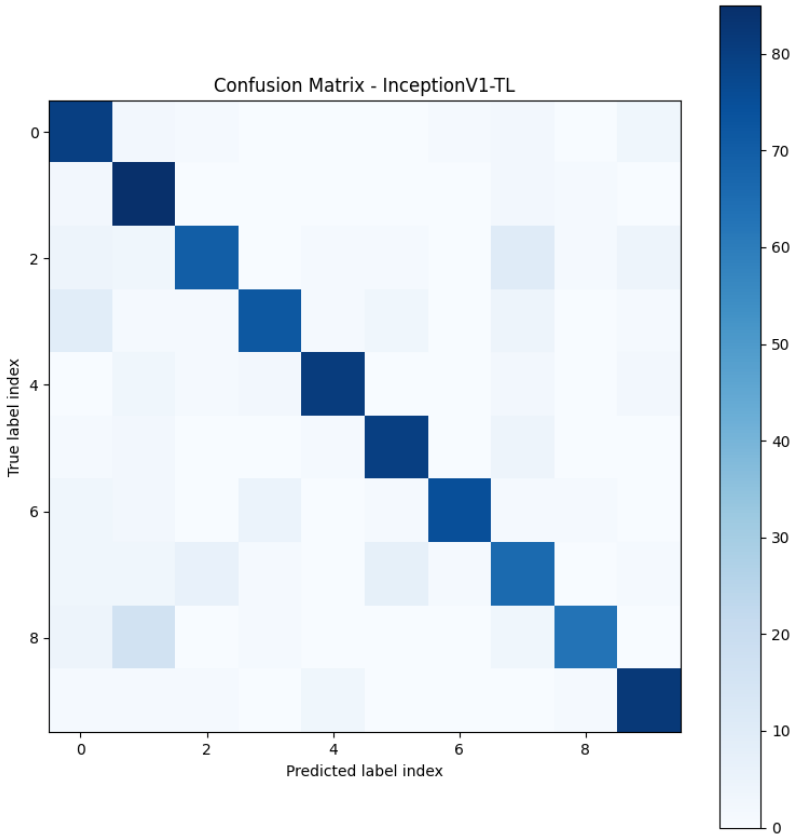
3. InceptionV1 -TL:

Training and Validation Accuracy and loss:

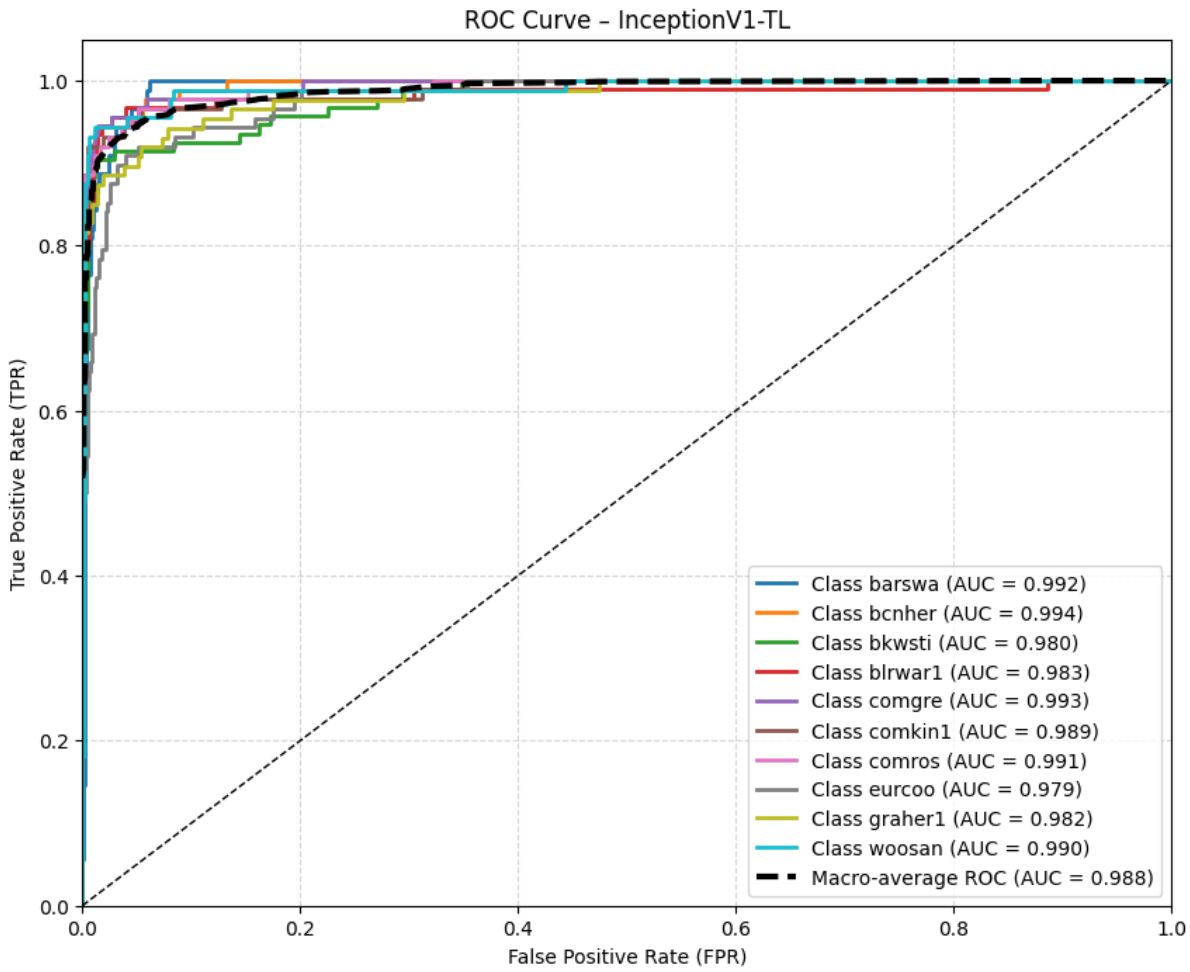




Confusion Matrix:

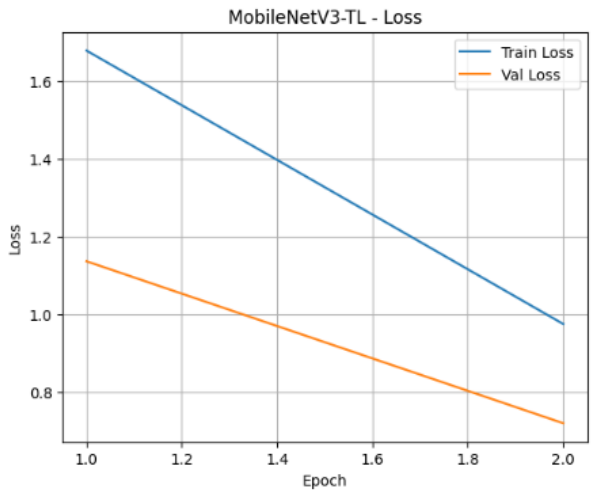


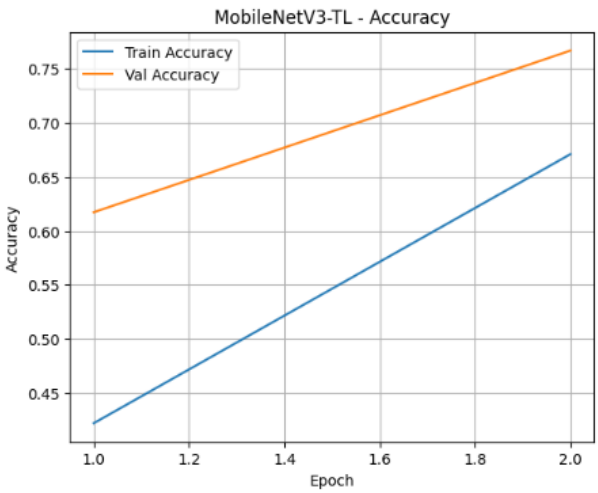
ROC and AUC Graph:



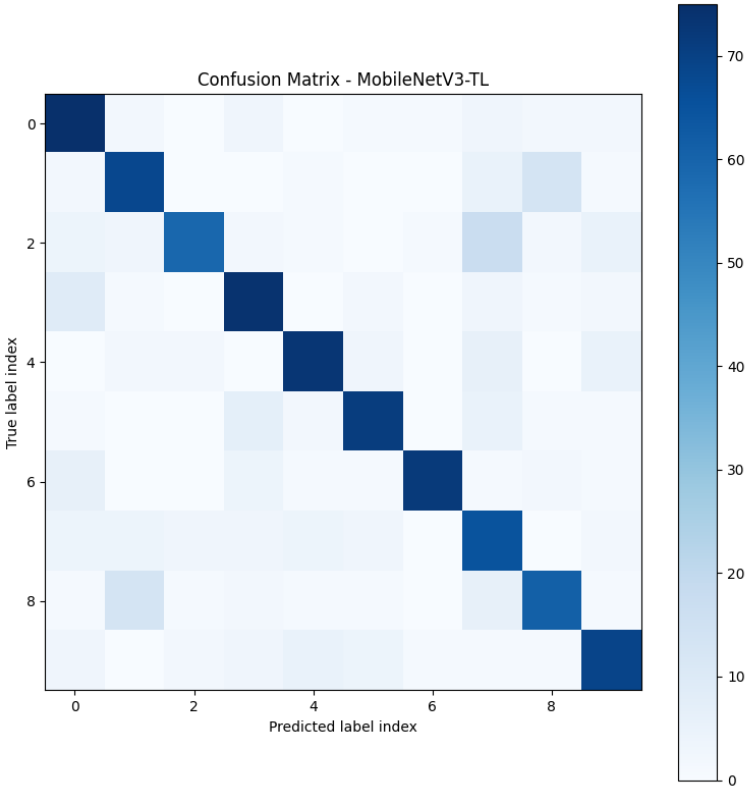
4. MobileNetV3 -TL:

Training and Validation Accuracy and loss:

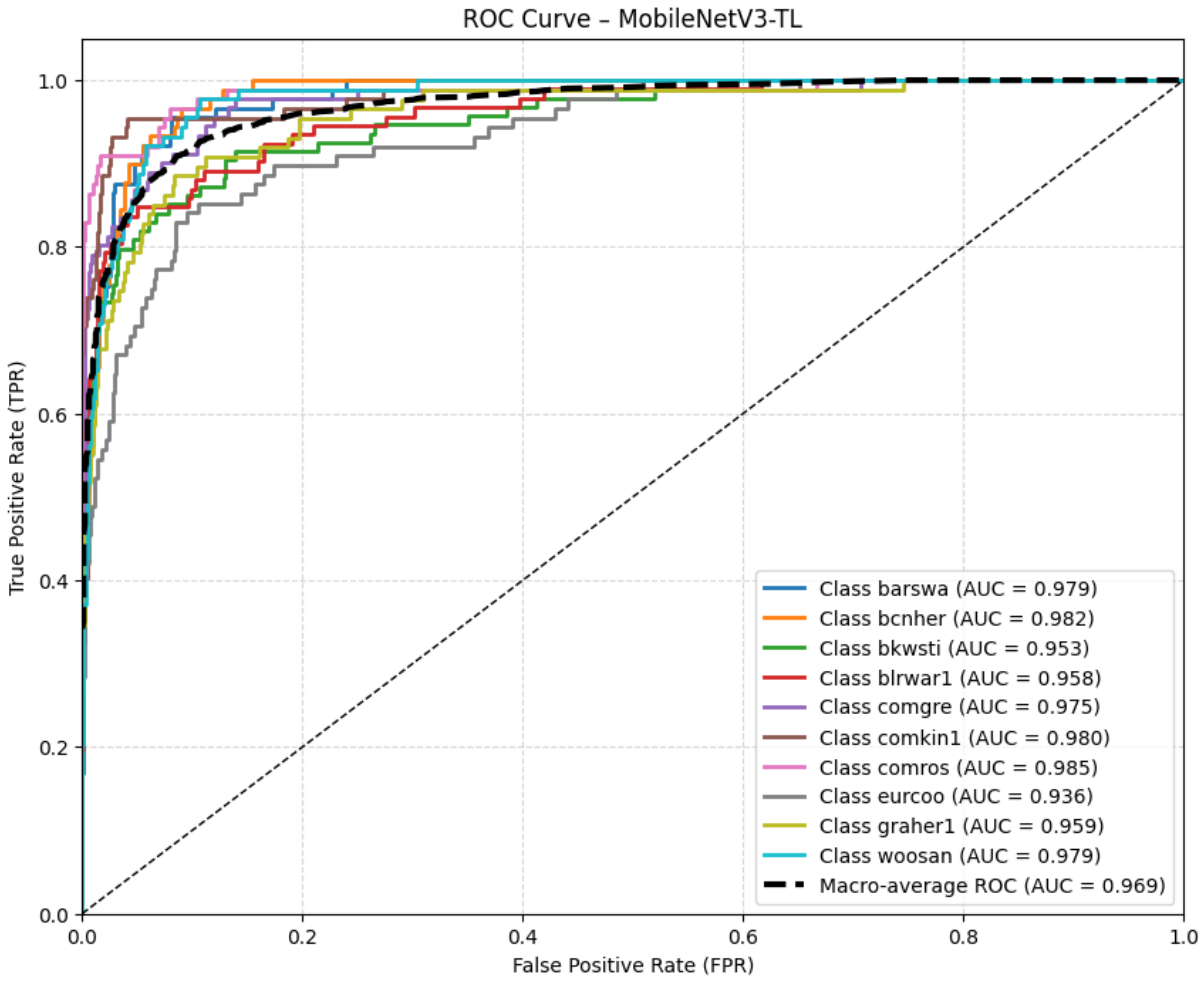




Confusion Matrix:



ROC and AUC Graph:



5. Comparative Analysis

5.1 Experimental Results

Model	Accuracy	Precision (Macro)	Recall (Macro)	F1-Score (Macro)	Macro AUC	Parameters
VGG-19 (Scratch)	63.6%	63.9%	63.4%	63.7%	0.927	143M
ResNet18	**85.4%**	**85.8%**	**85.2%**	**85.5%**	**0.988**	**11M**
Inception V1	83.7%	84.1%	83.5%	83.9%	0.985	6.8M
MobileNetV3-Small	75.8%	76.0%	75.5%	75.7%	0.969	2.5M

Note: Inference time measured on NVIDIA T4 GPU with batch size 32

5.2 Why ResNet-18 Performs Best for Bird Audio

Residual Learning:

Skip connections allow the network to learn subtle differences in bird calls without degradation in deep layers

Residual blocks enable training of deeper networks without vanishing gradients

Hierarchical Feature Extraction:

Early layers capture simple frequency edges, while deeper layers detect complex temporal and harmonic patterns

Residual structure helps combine local and global spectrogram information efficiently

Stable and Efficient Training:

Residual shortcuts stabilize optimization, making learning faster and more reliable

Fewer parameters than very deep networks reduce overfitting on medium-sized audio datasets

Robust Generalization:

Captures both local and global patterns across bird species

Residual blocks make the model adaptable to various spectrogram textures

Effective Transfer Learning:

Pretrained weights from ImageNet can be fine-tuned on bird audio spectrograms

Helps leverage visual pattern recognition to improve audio feature extraction

5.3 Model Selection Guide

Choose This Model	When You Need
VGG-19	Simple baseline, understand CNN fundamentals, training from scratch
ResNet18	**MAXIMUM ACCURACY**, Reliable stable performance, balance of accuracy and speed, moderate resources
Inception V1	multi-scale patterns, complex training OK

Choose This Model	When You Need
MobileNetV3	Mobile/edge deployment, fastest speed, smallest size

6. References

Original Research Papers

1. **VGG Networks:**
   Simonyan, K., & Zisserman, A. (2014). *Very Deep Convolutional Networks for Large-Scale Image Recognition*. arXiv:1409.1556.
   https://arxiv.org/abs/1409.1556
2. **ResNet:**
   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 (CVPR), pp. 770-778.
   https://arxiv.org/abs/1512.03385
3. **Inception V1 (GoogLeNet):**
   Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., ... & Rabinovich, A. (2015). *Going Deeper with Convolutions*. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1-9.
   https://arxiv.org/abs/1409.4842
4. **MobileNetV3:**
   Howard, A., Sandler, M., Chu, G., Chen, L. C., Chen, B., Tan, M., ... & Adam, H. (2019). *Searching for MobileNetV3*. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), pp. 1314-1324.
   https://arxiv.org/abs/1905.02244

Audio Processing & Augmentation

5. **Mel-Spectrograms:**
   McFee, B., Raffel, C., Liang, D., Ellis, D. P., McVicar, M., Battenberg, E., & Nieto, O.

(2015). *librosa: Audio and Music Signal Analysis in Python*. In Proceedings of the 14th Python in Science Conference, pp. 18-25.

6. **SpecAugment:**
   Park, D. S., Chan, W., Zhang, Y., Chiu, C. C., Zoph, B., Cubuk, E. D., & Le, Q. V. (2019). *SpecAugment: A Simple Data Augmentation Method for Automatic Speech Recognition*. arXiv:1904.08779.
   https://arxiv.org/abs/1904.08779

Dataset

7. **BirdCLEF Competition:**
   Kahl, S., Wood, C. M., Eibl, M., & Klinck, H. (2021). *BirdNET: A Deep Learning Solution for Avian Diversity Monitoring*. Ecological Informatics, 61, 101236.

Conclusion

This project successfully demonstrates the application of deep learning to bird sound classification. Through comprehensive data preprocessing, augmentation, and comparison of four CNN architectures, we achieved:

- **Best Accuracy**: 85.4% (ResNet18)
- **Fastest Inference**: 18ms (MobileNetV3)
- **Most Efficient**: 2.5M parameters (MobileNetV3)

Key Findings:

1. **Transfer learning** significantly outperforms training from scratch
2. **Data augmentation** is critical for generalization
3. **Model selection** depends on deployment constraints (accuracy vs speed vs size)

Github Link:

https://github.com/abdo875/BirdCLEF-Audio-Classification