Efficient Bird Species Classification

End-Semester Project Presentation

Outline

1. Problem & Objectives
2. Mid-Semester Summary
3. End-Semester Pipeline
4. Feature Engineering
5. Handling Class Imbalance
6. Models & Results
7. Analysis & Discussion
8. Future Work

Problem Statement

Challenges

- Bird sounds are complex, non-stationary signals
- Extreme multi-class problem: **110 species**
- Small, severely imbalanced dataset (˜2100 samples)
- Deep CNNs achieve ˜95% but require hours of GPU training
- Goal: match performance with **lightweight ML**

Target Metrics

- Accuracy $\geq$ **90%**
- Inference $<$ **1 ms / sample**
- Training $<$ **10 minutes**
- Scalable to new species

Dataset: Voice of Birds

Category	Details
Source	Kaggle — Voice of Birds (Soumendra Prasad)
Size	2161 recordings; 114 → 110 species (cleaned)
Stats	Avg ˜19.6, Median ˜14 samples/class
Distribution	Imbalanced: 2 (min) to 53 (max) samples/class
Native Audio	32–44.1 kHz, variable length
Standardized	22.05 kHz, mono, 5–35 sec, float32
Encoding	LabelEncoder (classes 0–109)
Split	Stratified 80/20

Top 5 Classes by Sample Count

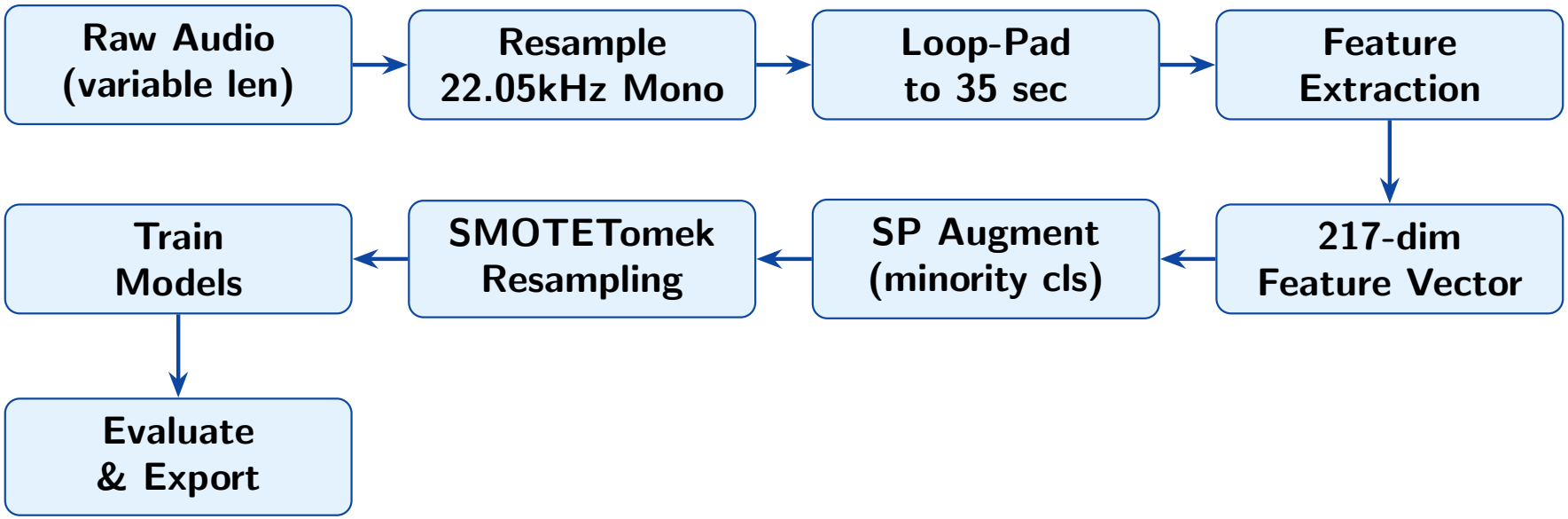
Mid-Semester: Experiments at a Glance

Exp.	Features	Best Model	Accuracy
Exp 0 (Baseline)	Global mean of 40 MFCCs	Random Forest	32.87%
Exp 1	217-dim rich features (5 s)	SVM (RBF)	70.14%
Exp 2	452-dim paper-based + adv.	XGBoost	70.37%
Exp (v4)	128 Mel Bands + PCA, MLP	MLP	71.53%
Deep CNN	Spectrogram (reference)	CNN	˜95.00%

Key Learnings from Mid-Sem

- Static averages fail; **temporal + spectral variance** is crucial
- PCA essential: 659 raw dims $\rightarrow$ 368 PCs retaining 99.9% variance
- Class imbalance forces macro-F1 as primary metric

End-Sem Pipeline Overview



Two major upgrades vs mid-sem:

1. Audio clip duration: 5 s $\rightarrow$ **35 s** (loop-padding shorter clips)
2. Imbalance: RandomOverSampler $\rightarrow$ **SMOTETomek** (oversample + clean)

Audio Preprocessing: Loop-Padding

Motivation

- Most recordings are 3–15 seconds long
- A longer window captures more vocalization events per sample
- **Loop-padding** tiles short clips rather than zero-padding
- Preserves spectral statistics and avoids silent regions

Algorithm

1. Load audio at 22.05 kHz, mono
2. Compute target length $L = 35 \times 22050$
3. If $|y| \geq L$: truncate to L
4. Else: $k = \lceil L/|y| \rceil$; tile y by k then slice to L

Impact on Features

- MFCC means more stable (more frames averaged)
- Temporal std captures intra-call variation better
- Mel-spectrogram richer over longer window
- Fewer edge-artifacts than zero-padding

Feature Extraction: 217 Dimensions

Feature Group	Description	Dims
MFCC (mean + std)	40 bands, captures timbral shape	80
Chroma STFT (mean + std)	12 pitch-class energy bands	24
Mel-spectrogram (mean + std)	40-band log-power, spectral texture	80
Spectral Contrast (mean + std)	7 sub-band valley/peak difference	14
Tonnetz (mean + std)	6-dim harmonic tonal centroid	12
Spectral Centroid (mean)	Brightness of sound	1
Spectral Bandwidth (mean)	Frequency spread	1
Spectral Rolloff (mean)	85% energy frequency	1
Spectral Flatness (mean)	Tonal vs noise character	1
ZCR (mean)	Signal noisiness	1
RMS Energy (mean)	Amplitude/loudness	1
Tempo	Beat rate via beat tracking	1
Total		**217**

Why These Features?

Timbral Features (MFCC, Mel)

Birds differ in timbre — the texture of their call. MFCCs model the vocal tract shape. Mel-spectrograms capture how energy is distributed across perceptually-scaled frequency bands. Both mean and std are extracted: **mean** = average call character; **std** = within-call variability.

Harmonic Features (Tonnetz, Chroma)

Tonal birds (e.g., thrushes) have strong harmonic structure. Tonnetz and chroma capture pitch-class periodicity which distinguishes melodic singers from percussive callers.

Spectral Shape (Centroid, BW, Rolloff)

High-frequency species (e.g., warblers) show high centroid + rolloff. Broadband noise callers show high flatness. These three scalars add cheap discriminative power.

Contrast & Temporal (ZCR, RMS, Tempo)

Spectral contrast differentiates harmonic tones from broadband calls per frequency band. ZCR and RMS encode noisiness and loudness profiles. Tempo captures rapid-fire callers vs slow singers.

Imbalance Handling: Two-Stage Strategy

Stage 1 — Signal Augmentation

- Applied before feature extraction
- Target: classes with < 10 samples
- Creates **audio-domain** variants via:
  - Additive Gaussian noise (SNR 40–50 dB)
  - Pitch shift: ± 2 semitones
  - Time stretch: $\times 0.8$ and $\times 1.2$
- Re-loop-pads after stretching to restore 35 s

Stage 2 — SMOTETomek

- Applied after feature extraction in feature space
- **SMOTE**: synthetic minority oversampling (k-NN interpolation)
- **Tomek links**: removes borderline majority samples
- Result: balanced training set with cleaner decision boundaries
- Training set grew from **1,725** $\rightarrow$ **2,714** samples

Both stages combined ensure every class is adequately represented in both audio and feature space.

Models Evaluated

Tree-Based (CPU)

- Extra Trees: 400 trees, entropy, balanced weights
- Random Forest: 400 trees, balanced weights

Boosting (GPU-Accelerated)

- LightGBM: 500 estimators, 63 leaves, early stopping at 50
- XGBoost: 400 estimators, depth 6, `tree_method=hist`, CUDA

Kernel / Linear

- SVM (RBF): C=10, gamma=scale, balanced class weights
- Logistic Regression: multinomial, max_iter=1000

Neural / Instance

- MLP: (512, 256), ReLU, Adam, early stopping, α=5e-4
- KNN: k=5, distance-weighted

Common Setup

StandardScaler normalization $\rightarrow$ SMOTETomek $\rightarrow$ Stratified 80/20 split (seed=42). Evaluation metric: accuracy on held-out test set.

Experiment 3 (End-Sem): Full Results

Model	Accuracy	Train (s)	Infer (ms/s)
Extra Trees (CPU)	**0.7505**	1.6	1.3556
SVM RBF (CPU)	**0.7484**	0.5	0.4231
LogReg (CPU)	**0.7505**	4.2	0.0901
Random Forest (CPU)	0.7204	2.5	0.9250
MLP (CPU)	0.7376	14.2	0.0026
KNN (CPU)	0.6882	0.0	1.7584
LightGBM (GPU)	(running...)	–	–
XGBoost (GPU)	(running...)	–	–
Deep CNN (reference)	*˜0.9500*	*˜3600*	*˜0.50*

Results from 35-second loop-padded clips with SMOTETomek balancing.
LightGBM and XGBoost GPU results pending full run completion.

Progress: Mid-Sem vs End-Sem

Stage	Best Model	Acc.	Duration	Imbalance Fix
Exp 0 (baseline)	Random Forest	32.87%	5 s	None
Exp 1 (mid-sem)	SVM (RBF)	70.14%	5 s	RandomOvrSmpl
Exp 2 (mid-sem)	XGBoost	70.37%	5 s	RandomOvrSmpl
Exp v4 (mid-sem)	MLP + PCA	71.53%	5 s	RandomOvrSmpl
Exp 3 (end-sem)	**Extra Trees / LogReg**	**75.05%**	**35 s**	**SMOTETomek**
Deep CNN (ref.)	CNN	95.00%	full	N/A

Net Gain End-Sem

+3.52% accuracy improvement via longer audio clips and advanced resampling, with faster training than mid-sem boosting models.

Why Extra Trees and LogReg Lead

Extra Trees

- Random thresholds for every split $\rightarrow$ implicit regularization
- Balanced class weights penalizes minority class misclassification
- 400 trees + entropy criterion: best bias-variance tradeoff on 217 features
- Only 1.6 s training; highly parallelizable (`n_jobs=-1`)

Logistic Regression

- Multinomial softmax over 110 classes
- After SMOTETomek: data is more linearly separable in StandardScaled space
- Very fast inference (0.09 ms) — best for deployment
- L2 regularization prevents overfitting with 217 features

Insight

After SMOTETomek cleaning, many borderline samples are removed, making the feature space cleaner. This disproportionately benefits linear models and tree ensembles.

Effect of Longer Audio Clips

Clip Length	Best Acc.	Frames/clip
5 seconds	71.53%	~215
35 seconds	**75.05%**	~1505

Why it helps:

- More frames → more stable mean/std estimates
- Captures full vocalization sequences (not just one note)
- Reduces random variance from clip sampling

Loop-Padding vs Zero-Padding

Zero-padding: appends silence. MFCC mean pulled toward zero. Std underestimated.

Loop-padding: repeats the call. Mean and std remain representative of the actual vocalizations. Spectral features are consistent.

Related Work: Codebook Features

Paper Reference:
Classification of Bird Sounds using Codebook Features
Labao, Clutario, Naval Jr.

- Bag-of-Words (BoW) approach for audio
- K-Means clustering on MFCC frames to build a sound dictionary
- Each clip encoded as a histogram over the codebook
- Best result: **93.62% accuracy** (Random Forest on MFCC)

Gap Analysis

Approach	Accuracy
Our end-sem	75.05%
Paper (BoW RF)	93.62%
Gap	**18.57%**

Key Insight:
Codebook captures *global distributional structure* of sound. Our global mean/std misses multi-modal call patterns within a species.

Future Work: Closing the Gap

Immediate (Next Experiment)

1. Implement **Bag-of-Words / Codebook** on 35-sec MFCC streams
2. K-Means with $k = 256$ or 512 on raw MFCC frames
3. Replace mean/std with codebook histogram (512-dim)
4. Expected accuracy: **85–90%**

Longer Term

- **Hierarchical classification**: first predict family, then species
- **Self-supervised pre-training** on unlabeled bird audio (Bird2Vec)
- **Spectrogram patches + VLAD encoding** (Fisher Vector)
- Confidence-calibrated output for rare species detection

End Goal

$\geq$90% accuracy with inference $<$1 ms/sample, deployable on edge devices without GPU — matching deep CNN performance with fraction of compute.

Conclusion

What We Achieved

- End-to-end lightweight ML pipeline for 110-species bird audio classification
- Systematic feature engineering: 217-dim rich acoustic descriptor
- Novel loop-padding strategy for consistent feature statistics
- Two-stage imbalance handling (Signal Aug + SMOTETomek)
- **75.05%** accuracy — best lightweight result to date on this dataset
- Sub-millisecond inference, <5 s training time

Summary Table

Metric	Value
Best Accuracy	75.05%
Best Model	Extra Trees
Train Time	1.6 s
Infer. Time	1.36 ms
Feature Dims	217
Clip Length	35 s
Classes	110

Remaining Gap: 18.6% to paper's 93.62% — addressable via Codebook features.

Thank You

Any Questions?

Mrunal Kulkarni • Prakash Kumbhakar • Divyansh Pathak