Classifying Bird Song from Around the World
(revised to: Europe)

Sandra Forro, Imran Naskani, Erin Rebholz

Harvard University, School of Extension Studies,
CSCI 109B Advanced Data Science
Spring 2024

Project Question:

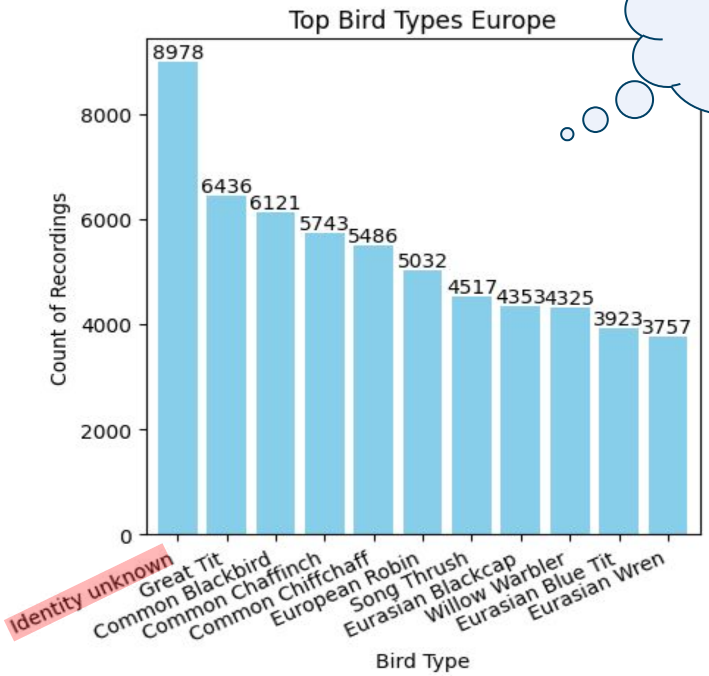
How might we use bird song audio data to **Classify Top Bird Songs in Europe**.

Success is defined as **surpassing the 85% accuracy** that the Nature and Biodiversity Conservation Union (NABU)'s model achieves.

2,901 15 second song **recordings of the top 10 European bird species** were subsetted from the Bird sound collection of Xeno-canto (XC), the Foundation for Nature Sounds in the Netherlands. There are approximately **200 samples** per bird type, with small upward or downward deviations.

Initial EDA and Data Quality: data is subsetted based on findings

Target: `vernacularName`
Initial samples: 700k



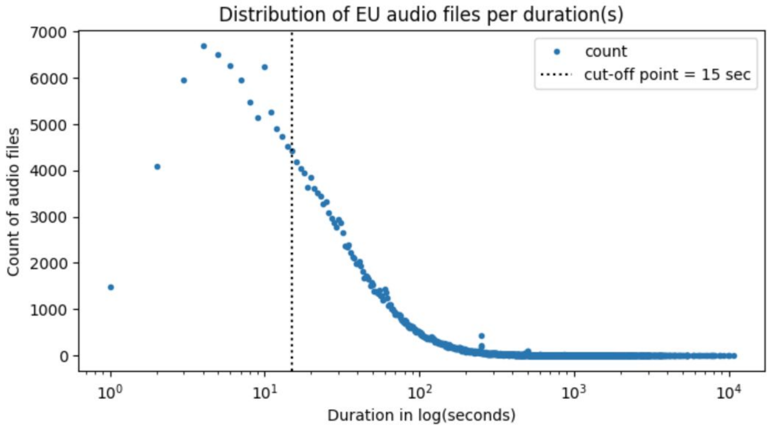
Why not these 10 birds?

`continent`

	253659
	96897
	96250
	79197
	51018
	19781
...TICA	222

	Missing Rate
behavior	0.008549
associatedTaxa	0.677127
continent	0.005642
vernacularName	0.000000
identificationRemarks	0.997811

`duration`



77,679 samples

- 2010+
- EU &
- <= 15 Sec duration
- No `identificationRemarks` but with `associatedTaxa`

`year`

min	1886
25%	2013
50%	2018
75%	2021
max	2024

Narrowing Behaviors: data is segmented into sound types. ‘Songs’ and ‘Calls’ were grouped together, other sounds and heterogeneous groups were excluded from baseline model

Samples Grouped by `behavior`

behavior	
call	17985
flight call	12119
nocturnal flight call	11647
song	10404
flight call, nocturnal flight call	2781
call, flight call	2576
alarm call	1761
call, flight call, nocturnal flight call	992
song, call	952
call, alarm call	707
uncertain	500
begging call	355
alarm call, flight call	292
call, nocturnal flight call	271
drumming	269
song, flight call	263
call, alarm call, flight call	164
subsong	149
song, nocturnal flight call	133
call, begging call	133
wing beats	94
song, imitation, mimicry/imitation	88
flight call, nocturnal flightcall	83
song, subsong	74
song, call, flight call	73
call, wailing call	65
call, wing beats	65
song, aberrant	62
call, aberrant	54
nocturnal flight call, aberrant	53

Behaviors Were Grouped into Similar Types:

Song = ‘song, ‘subsong’, ‘song, subsong’

Call = 'call', 'flight call', 'nocturnal flight call', 'flight call, nocturnal flight call', 'call, flight call'

Excludes 6,189, ‘ Identity Unknown’ for Vernacular Name

Top 10 Vernacular Names by Behaviour Group (sample cts)

Initial Model

Cetti's Warbler	699
Eurasian Wren	334
Common Cuckoo	307
Great Tit	256
Eurasian Blackcap	244
Willow Warbler	236
Common Chiffchaff	227
Common Quail	215
Common Chaffinch	209
European Green Woodpecker	174

9 samples with calls >200

2901 samples Total for Top 10

Common Moorhen	972
Red Crossbill	965
Water Rail	959
Redwing	830
Eurasian Coot	822
Common Sandpiper	717
Common Blackbird	671
Song Thrush	654
Tree Pipit	600
European Robin	583

77 birds with call samples >200

Evaluation Pipeline: Parameters within our evaluation pipeline were iterated to optimize accuracy results in baseline CNN model, then further models were tested to improve accuracy

Evaluation Pipeline

Pre-Processing And Oversampling

Narrow sample of interest to 10 categories of bird song

Oversample to create synthetic samples to improve class balance

Mel-Spectrogram Sample Generation

Trims, normalize and loop sampled audio

Generate mel-spec images and save to folder for multiple uses

Image Data Generator

Parses into Test/Train/Validation generators that are used within models.

Light transformation to 1/255 ranges and image size normalized to 128 x 128 x 3

CNN/ Autoencoder Model Evaluation

Baseline CNN

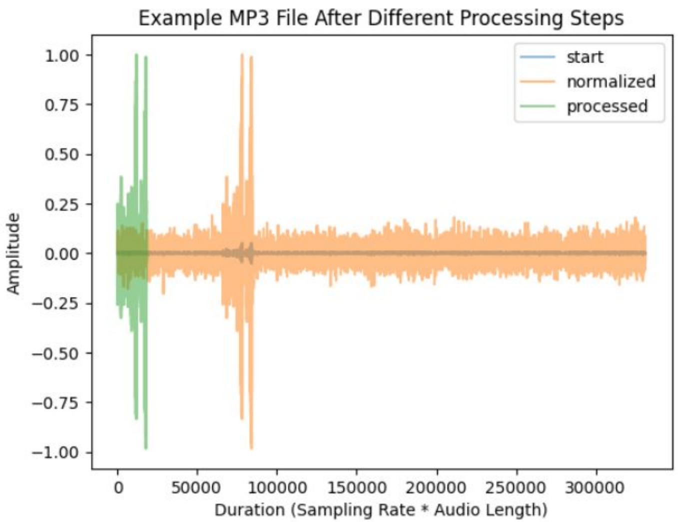
Optimization of CNN with regularization / drop out for consistency

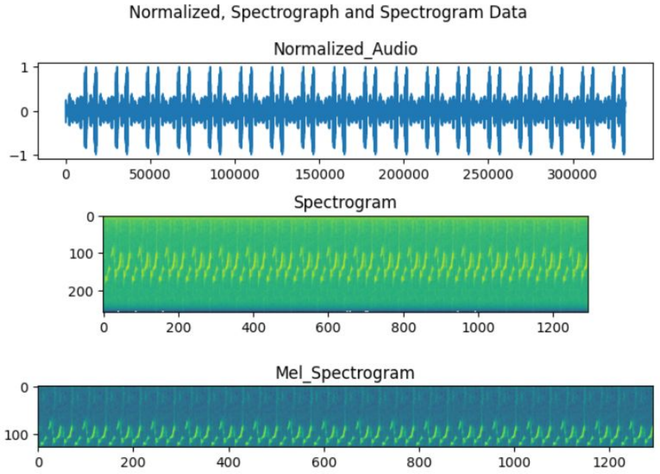
Experimentation with autoencoders

Refined Approach: TensorFlow I/O was used for GPU-enabled normalization, noise reduction, and mel spectrogram image generation

Audio was normalized to amplitudes between 1 & -1, then files were trimmed to exclude baseline noise that wasn't relevant.

The CNN approach requires a standard size image, so sound was looped to full 15 seconds and used to generate mel-spectrogram images:

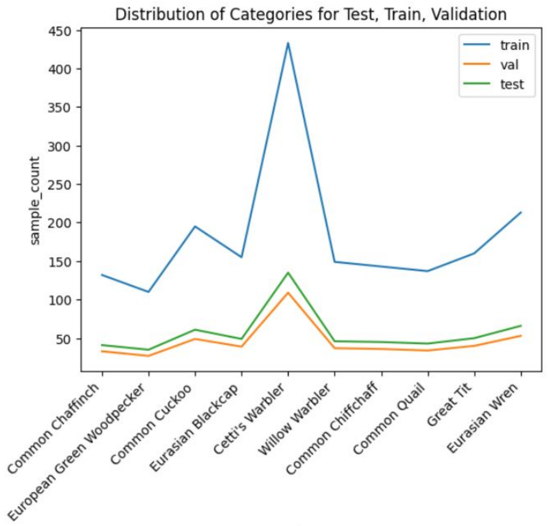




https://www.tensorflow.org/io/api_docs/python/tfio/audio

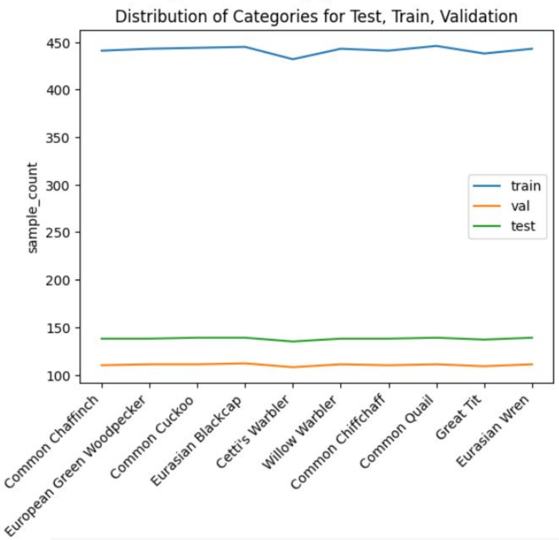
Correcting for class imbalance: Oversampling was performed to generate synthetic image samples by varying parameters

Sample with No Oversampling



Initial data had sample variation with a majority class of 690 samples and the others ranged from 174-334

Sample with Oversampling



Oversampling generated additional mel-spectrographs by varying frequency filters, max decibel ranges & baseline noise exclusion

Finding:

Generic tweaks to photos available in **image-data generator decreased prediction accuracy.**

Instead,**permutations in parameters used** within **audio processing** were used **to supplement image samples**.

Model Architectures:

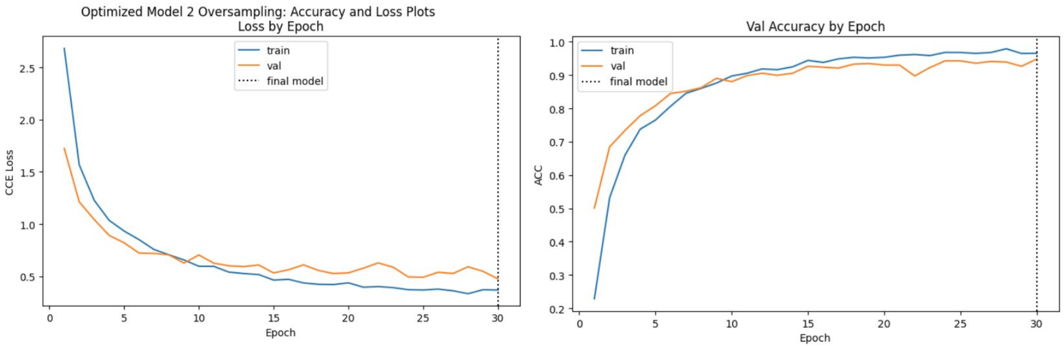
Baseline ‘Vanilla’ CNN	Optimized Model 1	Optimized Model 2	Optimized Model 2 Autoencoder
3 conv + 3 max pool layers 1 Dense layer No regularization	4 conv2D + 4 max pool layer 2 Dense layers No regularization	Optimized 1 + Optimized kernel sizes 2 Additional drop-out layers (0.5) + L1 Regularization Optimizer learning rate adjustments	Experimentation with autoencoders Basic construction
Trainable Parameters: 2.2 MM	0.73 MM	0.88 MM	0.13 MM *autoencoder* + 0.88 MM *model*

*values may change on next execution of same model due to randomness in tensorflow

Best Model based on performance*

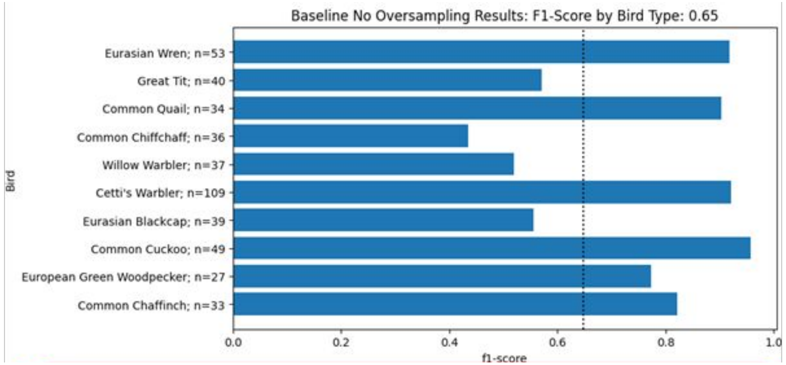
Model Training Results:

CNN Models	Baseline Model	Baseline Model	Optimized Model 1	Optimized Model 2	Optimized Model 2 with Autoencoders
Data	Imbalanced (No oversampling)	Balanced (with oversampling)	Balanced (with oversampling)	Balanced (with oversampling)	Balanced (with oversampling)
Trainable Parameters	2,212,522	2,212,522	726,634	878,698	1,004,013 [125,315 (autoencoder) + 878,698 (model)]
Epochs	30	30	30	30	30
Patience	5	5	5	5	5
Best Epoch	4	8	8	30	15
Train Accuracy	0.8002	0.9878	0.9701	0.9955	0.9837
Validation Accuracy	0.6674	0.9158	0.9112	0.9484	0.9212

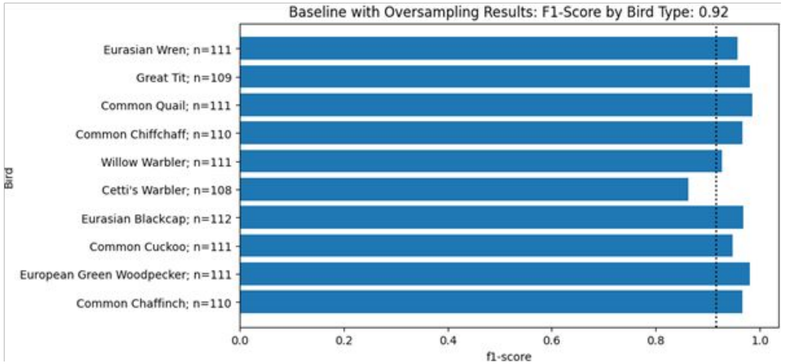


F1 Score by Bird Type: Validation Results

Baseline no Oversampling: F1-Score: 0.65

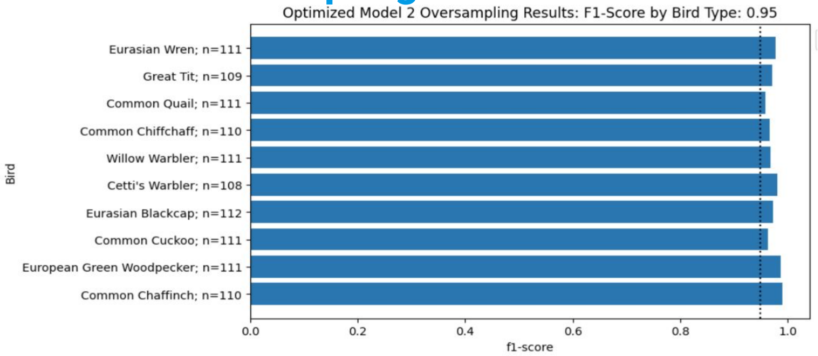


Baseline w/ Oversampling: F1-Score: 0.92

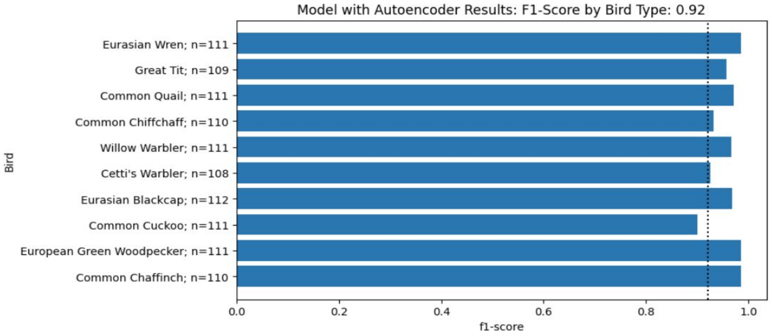


Optimized 2 w/ Oversampling: F1-Score: 0.95

Best F1-score



Auto Encoder: F1-Score: 0.92



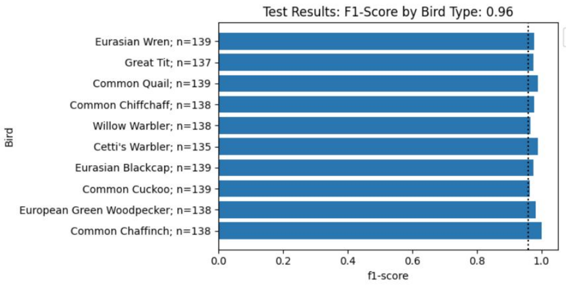
Final Inference and Discussion of Results

Test Accuracy: 95.9%

Val Accuracy: 94.8%

F1-score (weighted): .96:

Balanced across bird types



Incorrect Predictions:

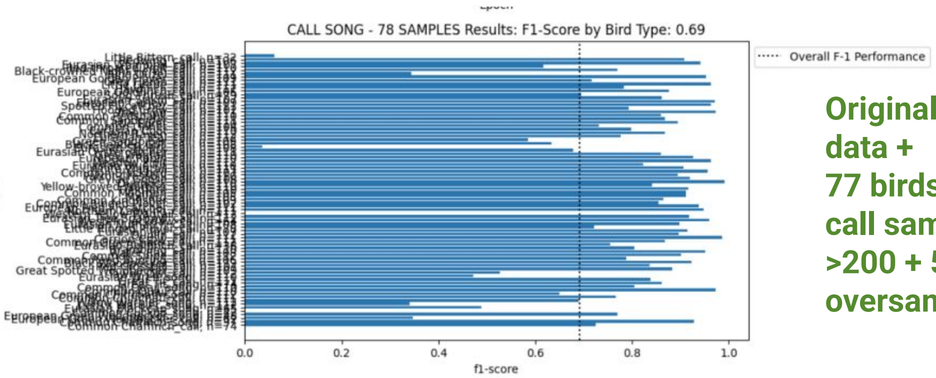
Correct Predictions:

Areas for Future Expansion

Scenario 1: Classifying bird types for top songs and top calls

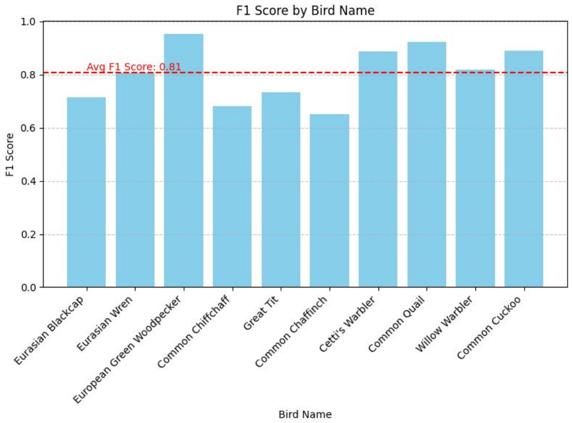
Top 10 Call Birds

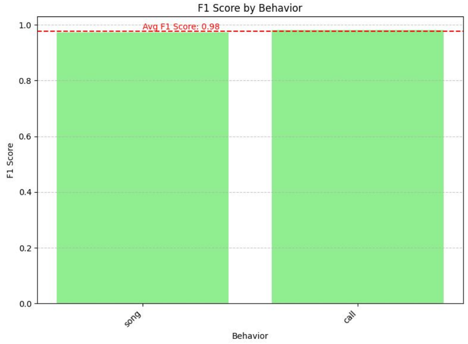
Common Moorhen	972
Red Crossbill	965
Water Rail	959
Redwing	830
Eurasian Coot	822
Common Sandpiper	717
Common Blackbird	671
Song Thrush	654
Tree Pipit	600
European Robin	583



Original song data + 77 birds with call samples >200 + 50k oversampling

Scenario 2: Classifying both call and song data for the top song birds (multi classification)





Original song data + new song and call data (8k)+ oversampling (2k)

Appendix

MODEL ARCHITECTURES

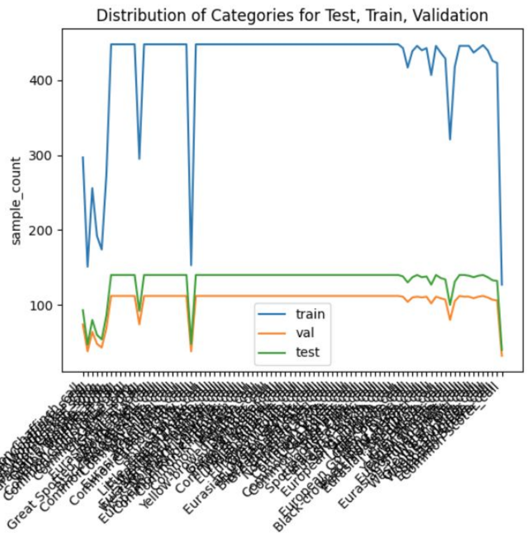
CNN Models	Baseline Model	Optimized Model 1	Optimized Model 2	Optimized Model 3 with Autoencoders
Input	128 x 128 x 3	128 x 128 x 3	128 x 128 x 3	128 x 128 x 3
Conv2D layers	3	4	4	4
Conv2D channels	64, 128, 32	64, 128, 64, 32	64, 128, 64, 32	64, 128, 64, 32
Kernel size	3x3, 3x3, 3x3	3x3, 3x3, 3x3, 3x3	11x3, 9x3, 3x3, 3x3	11x3, 9x3, 3x3, 3x3
Padding	Same	same	same	same
MaxPooling2D	(2,2), (2,2), (2,2)	(2,2), (2,2), (2,2), (2,2)	(2,2), (2,2), (2,2), (2,2)	(2,2), (2,2), (2,2), (2,2)
Stride	1	1	1	1
Dense layers	1	2	2	2
Dense units	256	256, 128	256, 128	256, 128
Dropouts	-	-	0.5, 0.5	0.5, 0.5
L1 regularization	-	-	0.0001, 0.0001	0.0001, 0.0001
Output layer	1	1	1	1
Output units	10	10	10	10
learning rate (Adam)	default	default	0.0005	0.0005
Trainable Parameters	2,212,522	726,634	878,698	125,315 (autoencoder) + 878,698 (model)

Autoencoder Architecture		
Encoder	Input	128 x 128 x 3
	Conv2D layers	3
	Conv2D channels	128, 64, 32
	Kernel size	3x3, 3x3, 3x3
	Padding	same
	MaxPooling2D	(2,2), - , (2,2)
	Stride	1
Decoder	Conv2D layers	3
	Conv2D channels	32, 64, 3
	Kernel size	3x3, 3x3, 3x3
	Padding	same
	UpSampling2D	(2,2), (2,2), -
	Stride	1
Trainable Parameters	125,315	

Bird Song Results Additional Call Data

Early Results: 10 bird_song and 80 bird_call types

Sample Generation Using Oversampling:
78 of the 90 bird_behavior pairs produce results:



<u>**Finding:**</u> Parameter tuning required to account for different frequencies and decibel levels of call data & reduce omits

<u>70% Val accuracy</u> using Optimized Model 2