Mark Daza
DATA 5322
Homework 3: Deep Learning

Sounds of Seattle Birds: Identifying the sounds of birds common in the Seattle area with Deep Learning

Abstract

This report evaluates the use of neural networks to identify the sounds of birds common to the Seattle area. Utilizing the data from the Birdcall competition dataset-a Xeno-Canto crowd-sourced bird archive-recordings from 12 different species of birds were collected. Using short-time Fourier transform (STFT), bird vocalizations were transformed into mel spectrograms. From these spectrograms, a Multi-Layer Perception (MLP) and Convolutional Neural Network (CNN) were developed to perform a binary classification between two species: Bewick's Wren and Spotted Towhee. Additionally, a multi-class CNN model was trained on the full set (12 species) for spectrogram-based species classification. The final multi-class CNN model was applied to unseen spectrograms generated from three .mp3 recording containing bird calls.

Introduction

The original dataset from Xeno-Canto bird recordings contains recordings of 264 species along with its accompanying metadata[1]. The license present in the train_extended.csv metadata file was used to access the recordings for the following species: American Crow (amecro), American Robin (amerob), Bewick's Wren (bewwre), Black-capped Chickadee (bkcchi), Dark-eyed Junco (daejun), House Finch (houfin), House Sparrow (houspa), Northern Flicker (norfli), Red-winged Blackbird (rewbla), Song Sparrow (sonspa), Spotted Towhee (spotow), and White-crowned Sparrow (whcspa).

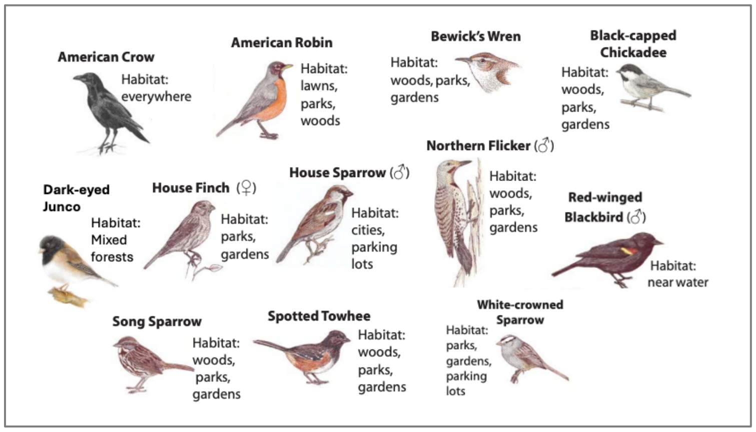

Figure 1. Common Seattle Birds (Adaptation from Seattle Audubon Society)

Recordings were filtered and prepared by selecting high-qualify samples for the 12 target species. Only clips featuring a single species, a sampling rate of 44.1kHz or 48kHz, a rating above 3 (on a

scale of 1-5), and a minimum duration of 3 seconds were retained. A mel spectrogram was generated for each clip, resulting in a 128x517 image representation of a bird sound.
This report examines the use of neural networks to identify 12 common bird species based on their vocalizations represented as spectrogram images. In the following sections we describe the process of creating and testing both binary and multi-class models, and how a final multi-class model performed on three real-world test recordings.

Theoretical Background

This report discusses an important and relevant topic in today's landscape of machine learning and artificial intelligence: deep learning, particularly the use of neural networks. Within this area, machine learning tools have been developed for specific types of high-dimensional data, such as Convolutional Neural Networks (CNN) for image and document classification. Before delving into CNNs, it's important to understand how a neural network works.

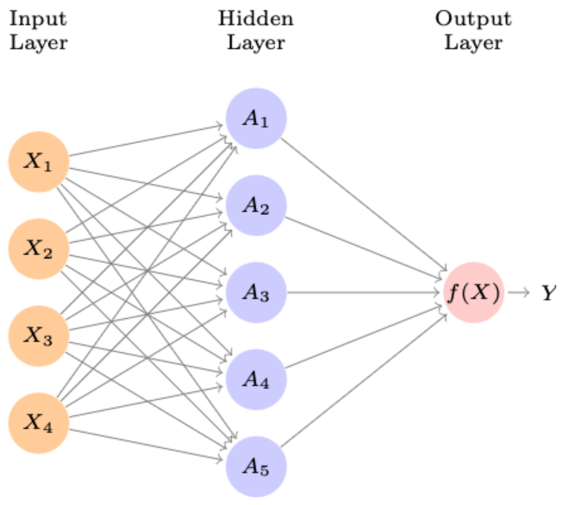

Figure 2. Neural Network with a Single Hidden Layer

A neural network is a machine learning program that makes decisions similar to those of a human[2]. Every neural network consists of layers of nodes: an input layer, one or more hidden layers, and an output layer. Figure 2 represents an example of a simple feed-forward single layer neural network. It takes ***p*** input variables, applies nonlinear transformations using activation functions, and generates a linear combination of activation functions to predict a response in the output layer.

Mark Daza
DATA 5322
Homework 3: Deep Learning

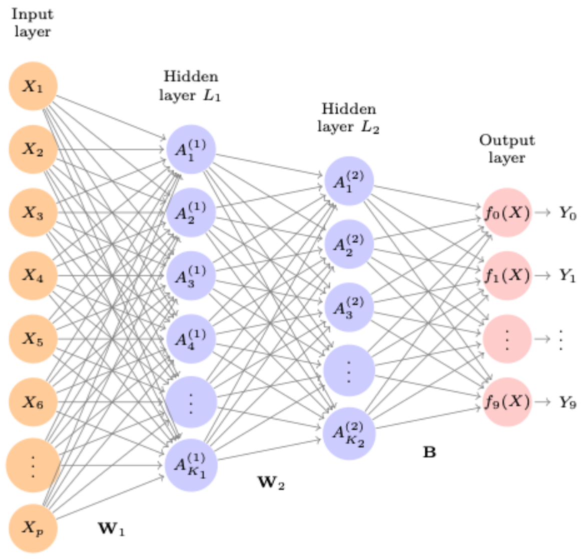

Figure 3. Neural Network with Two Hidden Layers and Multiple Outputs

Modern neural networks often use more than one hidden layer, and multiple nodes in each layer, as seen in Figure 4. This kind of structure allows the model to learn more complex patterns by capturing nonlinearities and interaction between features. For example, features like pitch or rhythmic structures in bird vocalizations — visualized in spectrograms — could be detected by combining and re-combining features in nonlinear ways.

Convolutional Neural Networks (CNN)

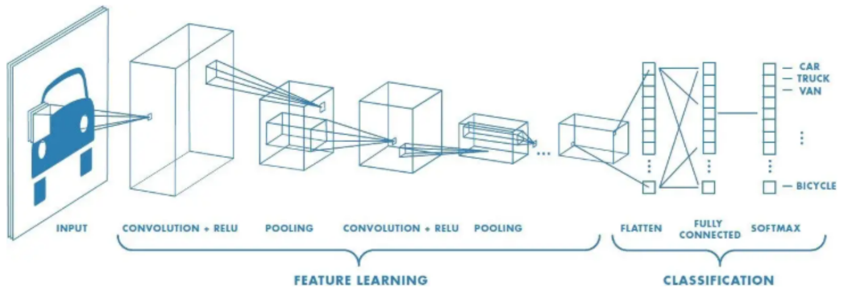

Figure 4. CNN Sequence to Classify an Image

CNNs are popular for their use in image classification, and similar to how a human would classify an image, it focuses on patterns and/or features of images to distinguish them. CNNs typically consist of two main types of hidden layers: convolutional and pooling layers.

Mark Daza
DATA 5322
Homework 3: Deep Learning

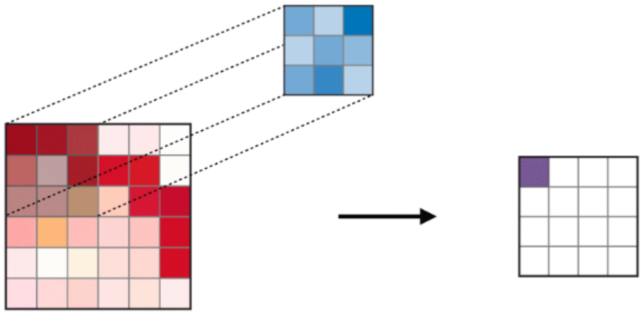
Figure 5. Visualization of a Convolutional Layer Operation

We can think of this layer as a filter that has values of a feature sliding across an image computing values for local regions and highlighting parts of the image that match the filter pattern. In a convolution operation, pixel values are repeatedly multiplied by the filter weights and added. This process allows the model to extract numerical patterns that can be learned by a neural network.

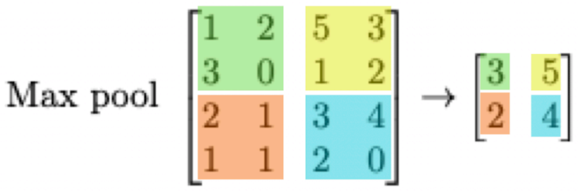

Figure 6. Visualization of a Max Pooling Layer Operation

In the pooling layer, the neural network performs downsampling of patterns picked by the convolution layer while retaining the most relevant information. You can think of this operation as generating a summary of the image scanned by the filter. In the case of max pooling, for each non-overlapping block of pixels, only the maximum value is retained and passed as an output for the next layer. This process is illustrated in Figure 6, where max pooling selects the highest value from each colored region.

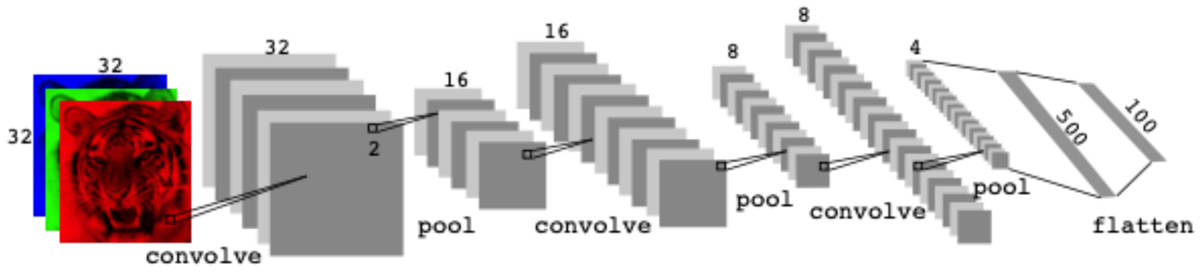

Figure 7. Architecture of a Deep CNN for the CIFAR100 Classification Task

CNNs often have multiple convolutions layers, each functioning similarly to the first. Keep in mind that with each layer, there is a reduction in spatial resolution, so the number of filters in each layer should compensate for the loss of information. Typically, these convolution and pooling layers are repeated until each image is reduced to a few pixels. At that point, the information extracted is flattened and passed to one or more fully connected layers, where the final classification of the image takes place.

Methodology

Our goals for this assignment were:

1. Build a binary model to classify two species.
2. Build a neural network to classify multiple species.
3. Test a multi-class model on external data.

We built two types of classification models: a binary classifier and a multi-class classifier to predict species from bird sounds.

Binary Model – Multi-Layered Neural Network (MLNN)
The binary model focused on distinguishing between spectrograms from two species: **Bewick's Wren** and **Spotted Towhee**. Spectrogram images were either left unscaled, pixel-scaled (divided by 255), or standardized (Z-scaled) using the mean and standard deviation from the training set. Labels were encoded as 0 (Bewick's Wren) or 1 (Spotted Towhee). A stratified train-test split was applied to maintain class balance (two-thirds training, one-third testing).
The MLNN used a flattened spectrogram input followed by two dense hidden layers with ReLU activation and an output layer with a single unit using a sigmoid activation. I tuned the units in the hidden dense layers and tested different batch sizes.

Binomial Model – Convolutional Neural Network (CNN)
A CNN was also tested as a binary classifier. It included 1-3 convolutional layers (with a ReLU) each followed by a pooling layer, and an output layer with a single unit using a sigmoid activation. All tuning used Z-scaled spectrograms.

Hyperparameters explored included:

- Pooling sizes: (2x2), (3x3), (4x4), (5x5)
- Dense units: 512, 256, 128
- Kernel sizes: (3x3), (4x4), (5x5)
- Channels: 16, 32 (doubled in each layer)
- Number of dense layers: 1, 2, 3
- Dropout rates: 0.3, 0.4, 0.5
- Batch sizes: 16, 32, 64

Each time a hyperparameter was tuned, the best-performing value was kept while the next was tested. Both MLNN and CNN models were compiled using RMSprop with binary_crossentropy loss and accuracy as the evaluation metric. An EarlyStopping callback with a patience = 3 was used to avoid overfitting. Performance was evaluated using accuracy, precision, recall, and F1-

scores. As an additional test, I reused the final tuned binary CNN model on spectrograms for **American Crow** and **House Sparrow**.

Multi-Class Model – Convolutional Neural Network
For the multi-class model, I used spectrograms for all 12 species. A CNN was again used, with 1-3 convolutional layers (ReLU activation), each followed by a pooling layer, and an output layer with 12 units using a SoftMax activation. Data was Z-scaled prior to model tuning. The model was compiled using RMSprop, categorical_crossentropy loss, and accuracy as the evaluation metric. Early stopping was used here as well.

Hyperparameter explored:

- Number of dense layers: 1, 2
- Dropout rates: 0.2, 0.3, 0.4
- Kernel sizes: (3x3), (5x5), (7x7)
- Pooling sizes: (2x2), (3x3), (4x4)

Each time a hyperparameter was tuned, the best-performing value was kept while the next was tested. Model performance was evaluated using accuracy, precision, recall, and F1-scores.
The final tuned model was used to predict the class of the species in the test dataset.

Predicting on External MP3 Files
Final multi-class CNN model was tested on three unseen .mp3 clips. I generated mel spectrograms from each clip using a similar pipeline as the training data and applied Z-scaling using the training set's mean and standard deviation. The model predicted class probabilities, and I reported the top three bird species per clip based on SoftMax scores.

Mark Daza
DATA 5322
Homework 3: Deep Learning

Computational Results

Computational results are presented here in the order described in the methodology.

Table 1. MLNN Binary Classification Performance on Unscaled Spectrograms

Units	Batch Size	Epochs Trained	True Positives Pred=bewwre Actual = bewwre	True Negatives Pred = spotow Actual= spotow	False Positives Pred=bewwre Act=spotow	False Negatives Pred = spotow Act= bewwre	Accuracy	Precision	Recall	F1-Score
16	32	61	0	47	1	45	0.51	0.00	0.00	0.00
16	64	11	45	0	48	0	0.48	0.48	1.00	0.65
16	500	146	0	48	0	45	0.52	0.00	0.00	0.00
32	32	8	0	48	0	45	0.52	0.00	0.00	0.00
32	64	6	45	0	48	0	0.48	0.48	1.00	0.65
32	500	5	0	48	0	45	0.52	0.00	0.00	0.00
64	32	24	0	48	0	45	0.52	0.00	0.00	0.00
64	64	7	9	45	3	36	0.58	0.75	0.20	0.32
64	500	10	45	0	48	0	0.48	0.48	1.00	0.65

Table 2. MLNN Binary Classification Performance on Pixel-Scaled Spectrograms

Units	Batch Size	Epochs Trained	True Positives Pred=bewwre Actual = bewwre	True Negatives Pred = spotow Actual= spotow	False Positives Pred=bewwre Act=spotow	False Negatives Pred = spotow Act= bewwre	Accuracy	Precision	Recall	F1-Score
16	32	10	1	47	1	44	0.52	0.50	0.02	0.04
16	64	19	0	48	0	45	0.52	0.00	0.00	0.00
16	500	15	0	48	0	45	0.52	0.00	0.00	0.00
32	32	5	1	48	0	44	0.53	1.00	0.02	0.04
32	64	67	0	48	0	45	0.52	0.00	0.00	0.00
32	500	6	10	43	5	35	0.57	0.67	0.22	0.33
64	32	5	0	48	0	45	0.52	0.00	0.00	0.00
64	64	9	45	0	48	0	0.48	0.48	1.00	0.65
64	500	10	6	0	48	0	0.48	0.48	1.00	0.65

Mark Daza
DATA 5322
Homework 3: Deep Learning

Table 3. MLNN Binary Classification Performance on Z-Scaled Spectrograms

Units	**Batch Size**	**Epochs Trained**	**True Positives** Pred=bewwre Actual = bewwre	**True Negatives** Pred = spotow Actual= spotow	**False Positives** Pred=bewwre Act=spotow	**False Negatives** Pred = spotow Act= bewwre	**Accuracy**	**Precision**	**Recall**	**F1-Score**
16	32	11	45	0	48	0	0.48	0.48	1.00	0.65
16	64	6	45	0	48	0	0.48	0.48	1.00	0.65
16	500	9	11	32	16	34	0.46	0.41	0.24	0.31
32	32	8	45	0	48	0	0.48	0.48	1.00	0.65
32	64	8	26	14	34	19	0.43	0.43	0.58	0.50
32	500	7	45	0	48	0	0.48	0.48	1.00	0.65
64	32	62	0	48	0	45	0.52	0.00	0.00	0.00
64	64	14	45	1	47	0	0.49	0.49	1.00	0.66
64	500	5	0	48	0	45	0.52	0.00	0.00	0.00

Table 4. CNN Binary Classification Performance Using Pixel-Scaled vs. Z-scaled

Scaling Method	**Layers**	**Channels**	**Drop Rate**	**Epoch Trained**	**TP**	**TN**	**FP**	**FN**	**Accuracy**	**Precision**	**Recall**	**F1**
Pixel Scaled	1	16	0.5	12	22	34	14	23	0.6	0.61	0.49	0.54
Z-Scaled	1	16	0.5	18	28	32	16	17	0.65	0.64	0.62	0.63

Note: This table compares performance between two scaling methods: pixel scaling (values divided by 255), and Z-scaling (standardization using the mean and standard deviation).

Hyperparameters used:

- Convolution Layers: 1
- Channels: 16
- Kernel Size: (3,3)
- Pooling Size: (2,2)
- Dropout Rate: 0.5
- Dense Layers: 0
- Dense Units: NA
- Batch Size: 32

Mark Daza
DATA 5322
Homework 3: Deep Learning

Table 5. Effect of Convolutional Layers on CNN Performance (Z-Scaled Data)

Hyperparameter Tuned	Layers	TP	TN	FP	FN	Trainable Params	Accuracy	Precision	Recall	F1
Convolutional Layers	**1**	26	28	20	19	264,353	0.58	0.57	0.58	0.57
	2	25	34	14	20	136,897	0.63	0.64	0.56	0.60
	3	**19**	**41**	**7**	**26**	**88,833**	**0.65**	**0.73**	**0.42**	**0.54**

Note: The number of channels in the first convolutional layer was as the base (e.g., 16) and subsequent convolutional layers used double the number of filters from the previous layer (e.g., 16 → 32 →64 for a 3-layer model)

The following hyperparameters were held constant during tuning:

- Channels: 16
- Kernel Size: (3,3)
- Pooling Size: (2,2)
- Dropout Rate: 0.5
- Dense Layers: 0
- Dense Units: NA
- Batch Size: 32

Table 6. Effect of Pool Size on CNN Performance (Z-Scaled Data)

Hyperparameter Tuned	Pool Size	TP	TN	FP	FN	Trainable Params	Accuracy	Precision	Recall	F1
Pooling Size	**(2,2)**	27	31	17	18	88,833	0.62	0.61	0.6	0.61
	(3,3)	**29**	**38**	**10**	**16**	**28,161**	**0.72**	**0.74**	**0.64**	**0.69**
	(4,4)	25	39	9	20	24,321	0.69	0.74	0.56	0.63
	(5,5)	17	40	8	28	23,553	0.61	0.68	0.38	0.49

The following hyperparameters were held constant during tuning:

- Convolution Layers: 3 - Optimized
- Channels: 16 (with subsequent layers doubling to 32 and 64)
- Kernel Size: (3,3)
- Dropout Rate: 0.5
- Dense Layers: 0
- Dense Units: NA
- Batch Size: 32

Mark Daza
DATA 5322
Homework 3: Deep Learning

Table 7. Effect of Dense Layer Units on CNN Performance (Z-Scaled Data)

Hyperparameter Tuned	Units	TP	TN	FP	FN	Trainable Params	Accuracy	Precision	Recall	F1
Dense Layer Units	**512**	24	36	12	21	2,514,689	0.65	0.67	0.53	0.59
	256	**30**	**35**	**13**	**15**	**1,268,993**	**0.70**	**0.70**	**0.67**	**0.68**
	128	15	44	4	30	646,145	0.63	0.79	0.33	0.47

The following hyperparameters were held constant during tuning:

- Convolution Layers: 3 - Optimized
- Channels: 16 (with subsequent layers doubling to 32 and 64)
- Kernel Size: (3,3)
- Pooling Size: (3,3) - Optimized
- Dropout Rate: 0.5
- Dense Layers: 1
- Batch Size: 32

Table 8. Effect of Kernel Size on CNN Performance (Z-Scaled Data)

Hyperparameter Tuned	Kernel Size	TP	TN	FP	FN	Trainable Params	Accuracy	Precision	Recall	F1
Kernel Size	(3, 3)	27	35	13	18	1,268,993	0.67	0.68	0.60	0.64
	(4, 4)	18	42	6	27	1,287,025	0.65	0.75	0.40	0.52
	(5, 5)	**29**	**34**	**14**	**16**	**1,310,209**	**0.68**	**0.67**	**0.64**	**0.66**

The following hyperparameters were held constant during tuning:

- Convolution Layers: 3 - Optimized
- Channels: 16 (with subsequent layers doubling to 32 and 64)
- Pooling Size: (3,3) - Optimized
- Dropout Rate: 0.5
- Dense Layers: 1
- Dense Layer Units: 256 - Optimized
- Batch Size: 32

Mark Daza
DATA 5322
Homework 3: Deep Learning

Table 9. Effect of Batch Sizes on CNN Performance (Z-Scaled Data)

Hyperparameter Tuned	Batch Size	TP	TN	FP	FN	Trainable Params	Accuracy	Precision	Recall	F1
Batch Size	16	18	39	9	27	1,310,209	0.61	0.67	0.40	0.50
	32	18	39	9	27	1,310,209	0.61	0.67	0.40	0.50
	64	**36**	**28**	**20**	**9**	**1,310,209**	**0.69**	**0.64**	**0.80**	**0.71**

The following hyperparameters were held constant during tuning:

- Convolution Layers: 3 - Optimized
- Channels: 16 (with subsequent layers doubling to 32 and 64)
- Kernel Size: (5,5) - Optimized
- Pooling Size: (3,3) - Optimized
- Dropout Rate: 0.5
- Dense Layers: 1
- Dense Layer Units: 256 - Optimized

Table 10. Effect of Starting Channels on CNN Performance (Z-Scaled Data)

Hyperparameter Tuned	Channels	TP	TN	FP	FN	Trainable Params	Accuracy	Precision	Recall	F1
Starting Channel	**16**	**16**	**43**	**5**	**29**	**1,310,209**	**0.63**	**0.76**	**0.36**	**0.48**
	32	16	41	7	29	2,747,905	0.61	0.70	0.36	0.47

The following hyperparameters were held constant during tuning:

- Convolution Layers: 3 - Optimized
- Kernel Size: (5,5) - Optimized
- Pooling Size: (3,3) - Optimized
- Dropout Rate: 0.5
- Dense Layers: 1
- Dense Layer Units: 256 – Optimized
- Batch Size: 64 - Optimized

Mark Daza
DATA 5322
Homework 3: Deep Learning

Table 11. Effect of Dense Layers on CNN Performance (Z-Scaled Data)

Hyperparameter Tuned	Batch Size	TP	TN	FP	FN	Trainable Params	Accuracy	Precision	Recall	F1
Dense Layers	1	21	38	10	24	1,310,209	0.63	0.68	0.47	0.55
	2	24	37	11	21	1,310,209	0.66	0.69	0.53	0.60
	3	18	38	10	27	1,310,209	0.60	0.64	0.40	0.49

Note: When using more than one dense layer, each subsequent layer had half the number of units as the previous (e.g., 256 →128 →64). Only the number of dense layers was varied.

The following hyperparameters were held constant during tuning:

- Convolution Layers: 3 - Optimized
- Channels: 16 (with subsequent layers doubling to 32 and 64) - Optimized
- Kernel Size: (5,5) - Optimized
- Pooling Size: (3,3) - Optimized
- Dropout Rate: 0.5
- Dense Layer Units: 256 – Optimized
- Batch Size: 64 – Optimized

Table 12. Effect of Dropout Rate on CNN Performance (Z-Scaled Data)

Hyperparameter Tuned	Batch Size	TP	TN	FP	FN	Trainable Params	Accuracy	Precision	Recall	F1
Dropout Rate	0.5	26	32	16	19	1,310,209	0.620	0.620	0.580	0.600
	0.4	28	27	21	17	1,310,209	0.590	0.570	0.620	0.600
	0.3	**33**	**32**	**16**	**12**	**1,310,209**	**0.700**	**0.670**	**0.730**	**0.000**

The following hyperparameters were held constant during tuning:

- Convolution Layers: 3 - Optimized
- Channels: 16 (with subsequent layers doubling to 32 and 64) - Optimized
- Kernel Size: (5,5) - Optimized
- Pooling Size: (3,3) – Optimized
- Dense Layers: 1 - Suboptimal
- Dense Layer Units: 256 – Optimized
- Batch Size: 64 - Optimized

Mark Daza
DATA 5322
Homework 3: Deep Learning

Table 13. Final CNN Binary Classification Model Performance on Two Species Paris (Z-Scaled Data)

Species Pair	Epochs Trained	TP	TN	FP	FN	Trainable Params	Accuracy	Precision	Recall	F1
Bewick's Wren vs. **Spotted Towhee**	16	29	28	20	16	1,310,209	0.61	0.59	0.64	0.62
American Crow vs. **House Sparrow**	19	207	18	4	1	1,310,209	0.98	0.98	1.00	0.99

Final hyperparameters:

- Convolution Layers: 3 - Optimized
- Channels: 16 (with subsequent layers doubling to 32 and 64) - Optimized
- Kernel Size: (5,5) - Optimized
- Pooling Size: (3,3) – Optimized
- Dropout Rate: 0.3 - Optimized
- Dense Layers: 1 - Suboptimal
- Dense Layer Units: 256 – Optimized
- Batch Size: 64 - Optimized

Figure 8. Training and Validation Accuracy and Loss for CNN Model on Bewick's Wren vs. Spotted Towhee

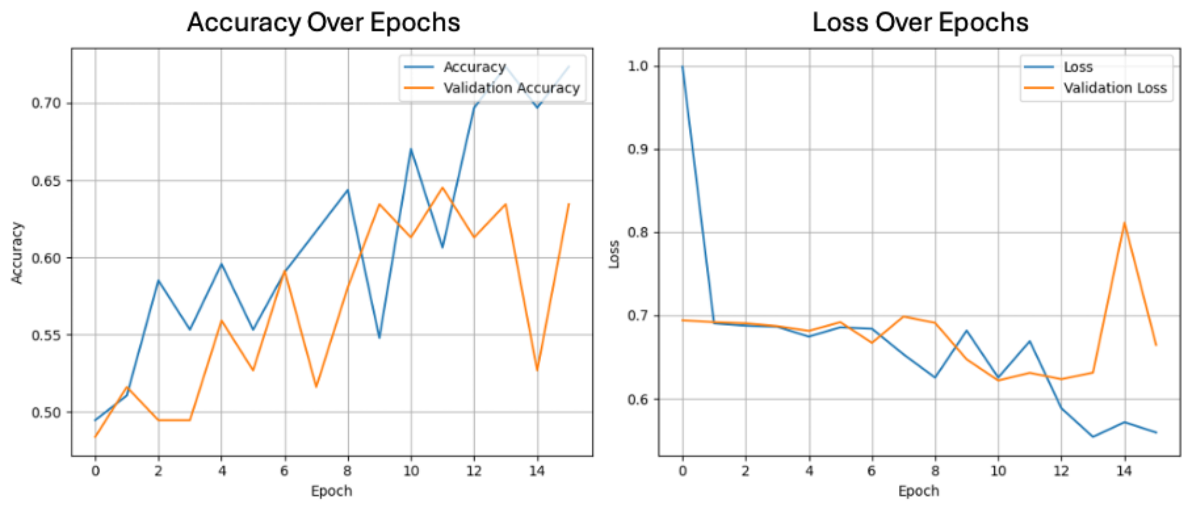

Figure 9. Training and Validation Accuracy and Loss for CNN Model on American Crow vs. House Sparrow

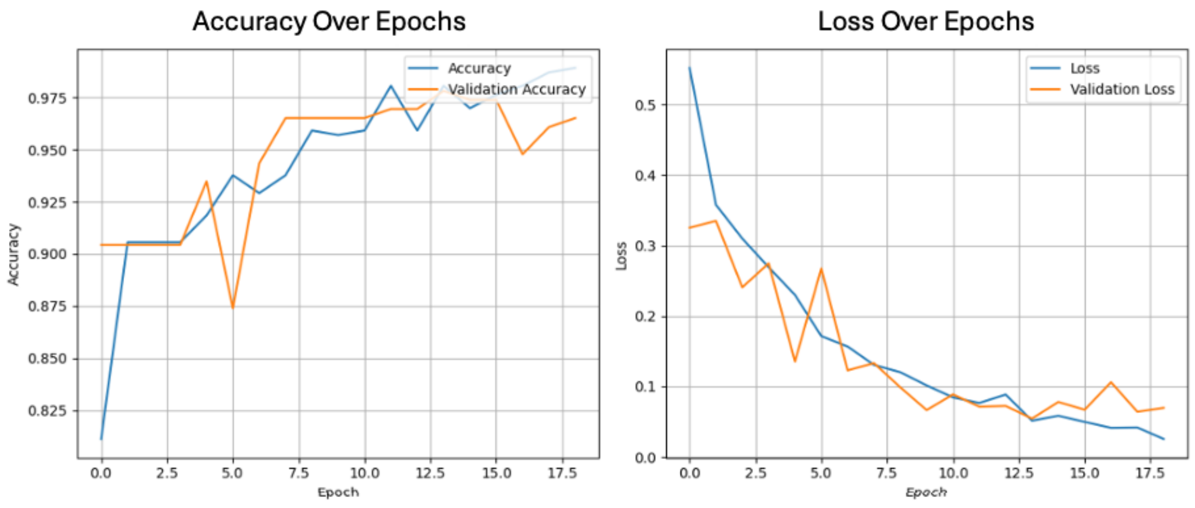

Table 14. Effect of Convolutional Layers on Multi-Class CNN Performance (Z-Scaled Data)

Hyperparameter Tuned	Layers	Trainable Params	Accuracy	Precision	Recall	F1
Convolutional Layers	1	59,183,180	0.32	0.03	0.08	0.04
	2	13,129,868	0.47	0.36	0.28	0.28
	3	2,750,732	0.52	0.41	0.34	0.34

Note: The number of channels in the first convolutional layer was as the base (e.g., 32) and subsequent convolutional layers used double the number of filters from the previous layer (e.g., 32 → 64 →128 for a 3-layer model)

The following hyperparameters were held constant during tuning:

- Channels: 32 (with subsequent layers doubling to 64 and 128)
- Kernel Size: (5,5)
- Pooling Size: (3,3)
- Dropout Rate: 0.3
- Dense Layers: 1
- Dense Layer Units: 256
- Batch Size: 64

Mark Daza
DATA 5322
Homework 3: Deep Learning

Table 15. Effect of Number of Dense Layers on Multi-Class CNN Performance (Z-Scaled Data)

Hyperparameter Tuned	Layers	Trainable Params	Accuracy	Precision	Recall	F1
Dense Layers	1	2,750,732	0.52	0.42	0.36	0.35
	2	2,782,092	0.53	0.34	0.35	0.32

Note: When using more than one dense layer, each subsequent layer had half the number of units as the previous (e.g., 256 →128). Only the number of dense layers was varied.

The following hyperparameters were held constant during tuning:

- Convolutional Layers: 3 - Optimized
- Channels: 32 (with subsequent layers doubling to 64 and 128)
- Kernel Size: (5,5)
- Pooling Size: (3,3)
- Dropout Rate: 0.3
- Dense Layer Units: 256
- Batch Size: 64

Table 16. Effect of Dropout Rate on Multi-Class CNN Performance (Z-Scaled Data)

Hyperparameter Tuned	Layers	Trainable Params	Accuracy	Precision	Recall	F1
Dropout Rate	0.2	2,750,732	0.5	0.37	0.34	0.34
	0.3	2,750,732	0.5	0.33	0.32	0.3
	0.4	2,750,732	0.52	0.4	0.34	0.35

The following hyperparameters were held constant during tuning:

- Convolutional Layers: 3 - Optimized
- Channels: 32 (with subsequent layers doubling to 64 and 128)
- Kernel Size: (5,5)
- Pooling Size: (3,3)
- Dense Layers: 1 - Optimized
- Dense Layer Units: 256
- Batch Size: 64

Mark Daza
DATA 5322
Homework 3: Deep Learning

Table 17. Effect of Kernel Size on Multi-Class CNN Performance (Z-Scaled Data)

Hyperparameter Tuned	Size	Trainable Params	Accuracy	Precision	Recall	F1
Kernel Size	(3, 3)	2,586,380	0.51	0.42	0.35	0.35
	(5, 5)	2,750,732	0.52	0.41	0.35	0.35
	(7, 7)	2,997,260	0.54	0.39	0.4	0.39

The following hyperparameters were held constant during tuning:

- Convolutional Layers: 3 - Optimized
- Channels: 32 (with subsequent layers doubling to 64 and 128)
- Pooling Size: (3,3)
- Dropout Rate: 0.4 - Optimized
- Dense Layers: 1 - Optimized
- Dense Layer Units: 256
- Batch Size: 64

Table 18. Effect of Pooling Size on Multi-Class CNN Performance (Z-Scaled Data)

Hyperparameter Tuned	Size	Trainable Params	Accuracy	Precision	Recall	F1
Pooling Size	(2, 2)	34,061,324	0.47	0.32	0.29	0.29
	(3, 3)	2,997,260	0.47	0.28	0.25	0.25
	(4, 4)	1,031,180	0.49	0.41	0.28	0.29

The following hyperparameters were held constant during tuning:

- Convolutional Layers: 3 - Optimized
- Channels: 32 (with subsequent layers doubling to 64 and 128)
- Kernel Size: (7,7) - Optimized
- Dropout Rate: 0.4 - Optimized
- Dense Layers: 1 - Optimized
- Dense Layer Units: 256
- Batch Size: 64

Mark Daza
DATA 5322
Homework 3: Deep Learning

Table 19. Final Multi-Class CNN Performance (Z-Scaled Data)

Species	Epochs Trained	Trainable Params	Accuracy	Precision	Recall	F1
All	22	1,031,180	0.59	0.44	0.43	0.43

Note: This final model was selected after tuning all major hyperparameteres. The final configuration included:

- Convolutional Layers: 3 - Optimized
- Channels: 32 (with subsequent layers doubling to 64 and 128)
- Kernel Size: (7,7) – Optimized
- Pooling Size: (4,4) - Optimized
- Dropout Rate: 0.4 - Optimized
- Dense Layers: 1 - Optimized
- Dense Layer Units: 256
- Batch Size: 64
- Scaling: Z-scaled spectrograms

Figure 10. Confusion Matrix - Multi-Class CNN Predictions on Test Set

Rows=Actual Species, Columns=Predicted Species

	American Crow	**American Robin**	**Bewick's Wren**	**Black-capped Chickadee**	**Dark-eyed Junco**	**House Finch**	**House Sparrow**	**Northern Flicker**	**Red-winged Blackbird**	**Song Sparrow**	**Spotted Towhee**	**White-crowned Sparrow**
American Crow	17	0	0	0	1	0	0	0	2	1	1	0
American Robin	0	36	4	1	1	0	8	0	3	0	3	1
Bewick's Wren	0	0	21	0	2	3	5	0	3	10	2	1
Black-capped Chickadee	0	1	1	0	1	0	2	0	2	3	1	4
Dark-eyed Junco	0	0	8	0	25	0	2	0	1	2	2	1
House Finch	1	3	1	0	1	8	11	0	0	2	1	0
House Sparrow	1	3	4	0	4	0	189	0	2	1	3	1
Northern Flicker	1	4	0	0	1	1	2	0	0	1	2	0
Red-winged Blackbird	0	16	2	0	1	1	14	0	15	11	2	0
Song Sparrow	2	10	10	0	3	2	8	0	3	41	5	3
Spotted Towhee	2	3	10	0	0	0	5	0	3	2	20	0
White-crowned Sparrow	0	1	0	0	3	0	2	0	5	7	1	11

Table 20. Top 3 Predicted Bird Species for External Audio Clips

Audio Clip	1st Prediction	Probability	2nd Prediction	Probability	3rd Prediction	Probability
Test1.mp3	House Sparrow	1.0	Red-winged Blackbird	0.0	White-crowned Sparrow	0.0
Test2.mp3	House Sparrow	0.9	Spotted Towhee	0.1	Red-winged Blackbird	0.0
Test3.mp3	House Sparrow	1.0	Spotted Towhee	0.0	White-crowned Sparrow	0.0

Discussion

Limitations and Training Time

One of the mean laminations encountered during this homework was the computational demand required to train my models. Training the multi-class neural network on the full set of spectrograms required more processing time than expected. Depending on the hyperparameter being tuned, training time varied from a few minutes to over 20 minutes.

Additionally, it was straightforward balancing the dataset used for binary classification, but balancing species representation for my multi-class model training was challenging. This class imbalance likely introduced biases and led to uneven learning.

Difficult-to-Predict Species

As observed in Figure 10, my final multi-class model shows there were species that were harder to classify. For example, species like Red-winged Blackbird were misclassified as an American Robin and House Sparrow.

One thing to note is that my multi-class model failed to correctly identify Black-capped Chickadee and Northern Flicker. Looking back at the training dataset, these two species were underrepresented, which likely prevented my model from learning enough vocal patterns to recognize them reliably.

Alternative Models

The use of neural networks, particularly CNNs, made sense for this task because mel spectrograms resemble images - they encode time and frequency patters, which CNNs are known to be the best suited for this kind of interpretation. The advantage of neural networks over alternative models like SVMs or tree-based models is that they handle nonlinearity and high dimensionality more effectively. I have a hard time thinking of another model that would have the adaptability to interpret these complex spectrogram patterns as well as a neural network.

Mark Daza
DATA 5322
Homework 3: Deep Learning

Conclusion

This report demonstrates the effectiveness of neural networks—particularly convolutional neural networks—in classifying bird species based on audio spectrograms.

My binary classification model trained quickly and reached 61% accuracy for distinguishing Bewick's Wren vs. Spotted Towhee, and 98% accuracy with American Crow vs. House Sparrow. While the model was not originally trained on this pair, it generalized better than expected. This likely reflects the benefits of a balanced training dataset and the more distinct audio patterns between American Crow and House Sparrow.

As for the multi-class classification model, it achieved a reasonable 59% accuracy, but accuracy alone does not capture the full picture. The model struggled to identify species that were underrepresented in the validation set, with a recall and F1-score around 43%. Again, this is likely due to insufficient training data for certain species.

When we tested on three external .mp3 clips, the model consistently predicted House Sparrow as the top species, with a probability score of near zero for other classes. This prediction suggests that the multi-class model became biased toward species with more training examples. The tested clips may not have contained clean or isolated bird vocalizations, or they contained background noise that was not present in the training set.

Despite the results, convolutional neural networks felt like the right approach for the classification of bird species based on spectrograms. To improve future performance, especially for imbalanced classes, techniques incorporating class weights or data augmentation could be explored. This assignment was a great introduction to working with neural networks, and their relevance to real-world problems like species identification.

Mark Daza
DATA 5322
Homework 3: Deep Learning

References

The Cornell Lab of Ornithology. (2021, April 28). What's that bird song? Merlin Bird ID can tell you. All About Birds. https://www.allaboutbirds.org/news/whats-that-bird-song-merlin-bird-id-can-tell-you/

Rao, R. (2020). Xeno-Canto bird recordings (extended A-M) [Dataset]. Kaggle. https://www.kaggle.com/datasets/rohanrao/xeno-canto-bird-recordings-extended-a-m

Xeno-Canto Foundation. (n.d.). xeno-canto: Bird sounds from around the world. Retrieved from https://www.xeno-canto.org

Seattle Audubon Society. (n.d.). Common Seattle Birds. Retrieved from https://seattleaudubon.org

IBM. (n.d.). What are neural networks? IBM. Retrieved May 13, 2025, from https://www.ibm.com/think/topics/neural-networks#:~:text=A%20neural%20network%20is%20a,options%20and%20arrive%20at%20conclusions

GeeksforGeeks. (2022, March 30). Neural networks – A beginner's guide. https://www.geeksforgeeks.org/neural-networks-a-beginners-guide/

Amazon Web Services. (n.d.). What is a neural network? AWS. https://aws.amazon.com/what-is/neural-network/

Ng, A. Y. (n.d.). Multi-layer neural networks. Stanford University. http://deeplearning.stanford.edu/tutorial/supervised/MultiLayerNeuralNetworks/

IBM. (n.d.). What is a convolutional neural network (CNN)? IBM. https://www.ibm.com/think/topics/convolutional-neural-networks

Saturn Cloud. (n.d.). A comprehensive guide to convolutional neural networks (the ELI5 way). Saturn Cloud. https://saturncloud.io/blog/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way/

Mark Daza
DATA 5322
Homework 3: Deep Learning

Appendix A: Link to Code

https://github.com/mdazab/DATA533-Homework-3