Deep Learning Based Audio Classifier for Bird Species

[1]Aarti Madhavi, [2]Rajni Pamnani

[1]P.G. Student, [2]Assistant Professor
Department of Computer Engineering,
K. J. Somaiya College of Engineering, Vidyavihar, Mumbai, India

***ABSTRACT*: The effect of human activities on the environment has reached a point where it has become necessary to track the effects before it causes irreparable damage to the environment. One of the ways to track such effects is to monitor the breeding behaviour, biodiversity and population dynamics of animals. Birds are one of the best species to track as they do tend to be the most reactive ones for any change in the environment e.g., deforestation or forest fires. Till now, the tracking of the birds was done manually by experts, which is very tedious at the same time consuming and non-viable method. As a result to alleviate this issue and provide assistance to the ecologists we proposing a machine learning method to recognize the bird's species based on the audio recordings. To achieve this goal, we intend to use the state of art convolutional neural network architecture called the deep residual neural networks as compared to the traditionally used classifiers like SMACPY, SVM and other relatively less sophisticated methods. We leverage methods like data augmentation and the existing carefully crafted datasets from Neural Information Processing Scaled for Bioacoustics to showcase the effectiveness of our method.**

***KEYWORDS*: Machine Learning, Convolutional Neural Networks, Deep Residual Neural Network, Bird Species Classification**

I. INTRODUCTION

The use of acoustics to monitor and classify animals in their natural environments has received a lot of interest lately. Classification of animal species based on recorded sound data is, for example, useful when monitoring breeding behaviour, biodiversity, and population dynamics. Birds are a particularly useful ecological indicator, as they respond quickly to changes in their environment. Bird classification can be done manually by domain experts; however, with growing amounts of data, this rapidly becomes a tedious and time-consuming process. Therefore automatic tools which can aid in this process are needed. Several bird species identification challenges such as the BirdCLEF [1], the Neural Information Processing Scaled for Bioacoustics (NIPS4B) 2013, and the Machine Learning for Signal Processing (MLSP) 2013 Bird Classification Challenge [3] have been held recently, all with the goal of creating and evaluating such automatic classifiers on bird song recordings taken from the field. A promising classification technique has proven to be convolutional neural networks [1, 5]. In this work, a new convolutional neural network architecture called deep residual neural networks [2] is evaluated in this problem domain to see if this model, originally designed for image recognition, is useful in the audio domain.

II. RELATED WORK.

Simple Minded Audio Classifier in Python (SMACPY) train the set of labelled audio files and it predicts a label for other audio files [6]. SVM has some disadvantages like it requires intervention of kernel at some point which is difficult and it classifies between two classes only therefore we make use of neural network [7]. The neural network, referred to as an 'artificial' neural network (ANN) [2]. Convolutional Neural Network (CNN) based deep learning approach for bird song classification that was used in an audio record-based bird identification challenge, called BirdCLEF 2016. The training and test set contained about 24k and 8.5k recordings, belonging to 999 bird species. The recorded waveforms were very diverse in terms of length and content. In the official scores our solution reached a MAP score of over 40% for main species, and MAP score of over 33% for main species mixed with background species [5].
In MLSP 2013 the winning solution were random forests trained on probabilities derived from template matching of species-specific spectrograms [3]. The winning solution of NIPS4B 2013 by Lasseck et al. used these results as a starting point but introduced an additional set of features statistically derived from the audio files. Lasseck also used a similar method to win the BirdCLEF 2015 challenge. However, during the BirdCLEF 2016 challenge, it was shown that convolutional neural networks trained on spectral data computed from the sound recordings could outperform other state-of-the-art systems [1].

III. PROPOSED SYSTEM

Fig1 describes the used methodology for classification of audio test sample of bird's audio. The Convolution Neural Network is used for classification with the use of ResNet. The phases of architecture includes:

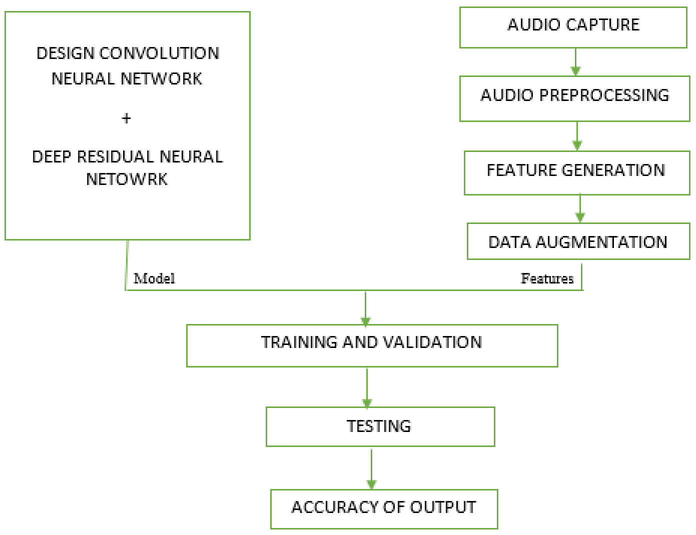


Fig 1: System Architecture of Deep Learning Based Audio Classifier of Bird Species

A. *Pre-processing:*

The audio files are first pre-processed into a format that can be used to train the neural network. The audio files are normalized to mono-channel 16-bit wave data re-sampled from 44,100 Hz to 22,050 Hz.

B. *Feature Generation*

The bird song recordings are separated into two different sound classes: signal (bird vocals) and noise. The separation lets us train the neural network on the most relevant data, and it gives us access to a noise class which can be used to augment training samples. After the recording has been separated into a signal wave and a noise wave these are each split into 3-second segments which are stored to disk. The noise segments can later be used to augment the training samples shown to the network which should improve generalization.
The signal part is extracted by first computing a signal mask $\bar{v}$ for the given sound wave $\bar{x}$, and then use the mask to extract the relevant part of the sound wave, where vi = 0 indicates that xi is not part of the signal, and vi = 1 indicates that it is part of the signal.

The binary image is further processed by applying a binary erosion followed by a binary dilation on the image, both using a kernel size of 4 by 4, which smooths out the regions marked as bird vocals (see Fig 2), and the signal mask $\bar{v}$ is derived from the binary image by setting vj to one if the column $\bar{b}_j$ contains one one. The mask is also smoothed by performing 2 more binary dilutions (kernel size 4), and is then re-scaled such that $|\bar{v}| = |\bar{x}|$.

The signal mask is computed by setting the threshold t = 3, and the noise mask is computed by setting t = 2.5 and then inverting the mask at the end (flipping 0s to 1s, and 1s to 0s). This may leave parts of the wave which are marked as neither signal nor noise (2.5-3). These parts are considered to not contribute with any relevant information for the network, and they are simply ignored (see Fig 3).

```
Algorithm 1: Noise/Signal Mask computation
1: procedure ComputeMask (r̄, t)
2:         P xx ← spectrogram (r̄)
3:         P xx ← normalize (P xx)
4:         BinaryImage ← medianClipping (P xx, t)
5:         BinaryImage ← erosion (BinaryImage, (4, 4))
6:         BinaryImage ← dilation (BinaryImage, (4, 4))
7:         mask ← computeMask (BinaryImage)
           Return mask
```

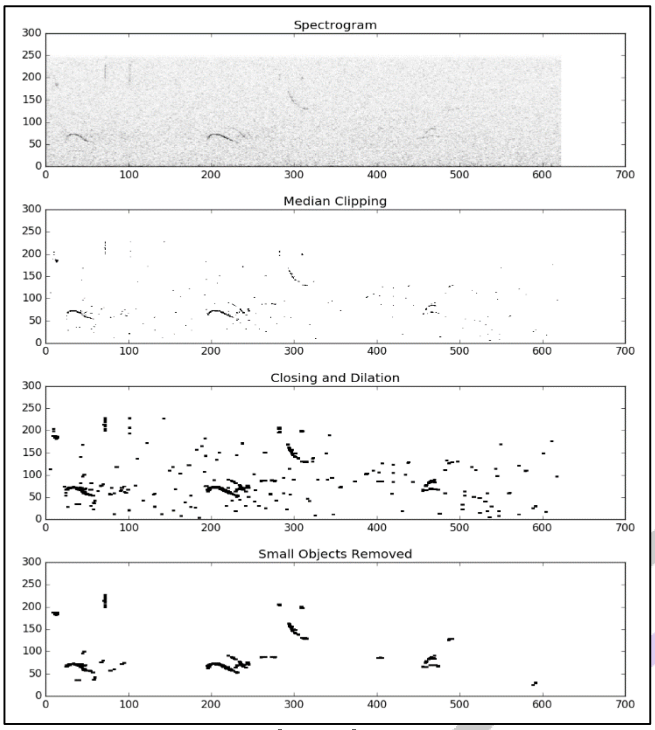


Fig 2: Noise and Signal part computation

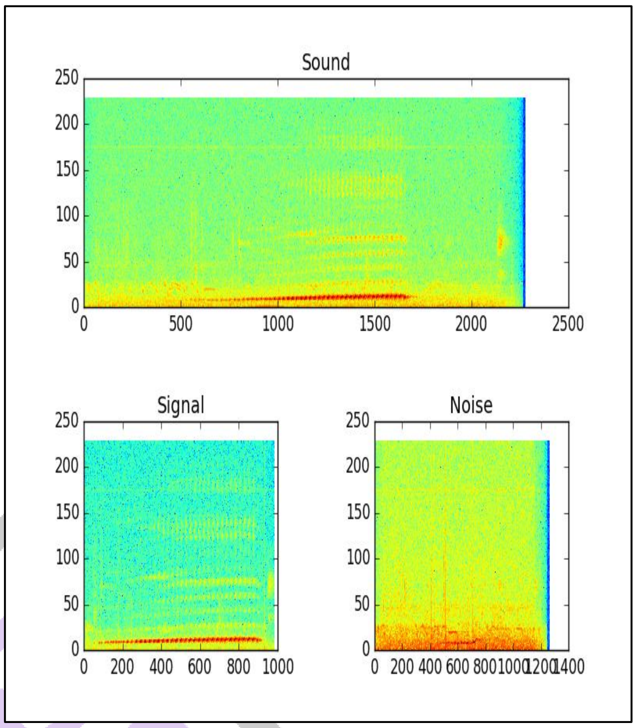


Fig 3: Noise and Signal part separation

C. *Multiple-Width Frequency-Delta Data Augmentation*

Mel-frequency cepstral coefficients (MFCCs) are commonly used features when training audio classifiers, data augmentation technique which is used to improve the usefulness of such features. The idea is to compute the delta features of the MFCCs, which contain additional information about the trajectories of the MFCCs. This gives the network not only local static information about the signal, but also dynamic information about how the signal will proceed in the (near) future.
The multiple-width frequency-delta (MWFD) data augmentation is computed as such:

$$d_t = \frac{\sum_{k=1}^{k} = k(x_{t+k} - x_{t-k})}{2\sum_{k=1}^{k} k^2}$$

where dt is the delta feature computed from frame t in terms of the static Mel frequency cepstrum coefficients x_{t-k} to x_{t+k}. The delta width refers to 2K+1, widths Δ3, Δ11 and Δ19 (i.e., K = 1, K = 5, K = 9). The input to the CNN is then the vector, or four channel image, (static, Δ3, Δ11,Δ19), where static refers to the MFCCs. We use the first 32 MFCCs following the results of [4], which show that 32 MFCCs are a sweet spot for identifying frog calls, and insects. These are extracted using the library librosa which computes a Mel-spectrogram using a 2048 length FFT window and a hop size of 512.

D. *Training*

To train the input our model we use Convolution Neural Network's – ResNet architecture i.e. Residual Neural Network. A typical CNN consists of convolutional layers and pooling layers followed by a fully connected neural network. The convolutional layers and the pooling layers are supposed to learn how to extract relevant, locally distortion invariant, features from the input [3], and the fully connected neural network is supposed to learn how to classify these features. The features learned by the convolutional layers could be, e.g., edges of objects if considering an image.

Deep residual neural networks use "shortcut connections" in order to improve the convergence rate and classification accuracy of very deep CNNs [8]. Deep networks are usually harder to train because of vanishing or exploding gradients, which has mostly been mitigated by using normalized weight initialization, ReLU activation and intermediate normalization layers. The reason behind the shortcuts is that by letting the signal flow more easily through the network the problem of exploding or vanishing gradients can be reduced, which in turn makes the deep network easier to train. This is implemented in the network by a shortcut, where the input, x, of a stacked layer is connected to the output of the stacked layer, using a simple additive unit. This building block is what we call the residual unit (see Fig 4. for a simplified overview of a residual unit). In the simplest case, where the dimensions of the output of the residual function and the input are the same, the residual unit can be defined as:

$$y_l = h(x_l) + f(x_l, W_l) \quad (1)$$

$$x_{l+1} = f(y_l) \quad (2)$$

where x_l and x_{l+1} are the input and output of the l-th residual unit, f is the activation function, f is the learned residual function, h is the mapping applied on the data flowing through the shortcut, in Equation (1) above it would be an identity mapping, and W_l is the parameters associated with the l-th residual unit. If the dimensions of $f(x_l, W_l)$ and x_l are not the same then h would need to adjust the dimension of x_l by, e.g., subsampling.

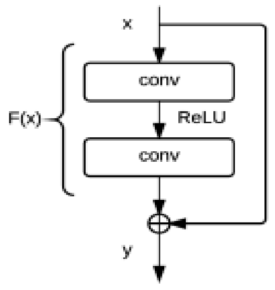


Fig 4. A simple residual unit.

E. Architecture

The architecture of an 18-layer deep residual neural network consists of an initial convolving and max pooling layer followed by eight basic blocks (with different configurations), and at the end the result is pooled, flattened and used as input to the final dense layer which has one neuron for each class label, activated by a softmax function. A basic block consists of two convolving layers, each preceded by batch normalization and ReLU activation, and the input of the block is allowed to flow directly to the output of the stacked layers via a shortcut realized using an additive merge layer (see Table 1).

Table 1
The architecture of the 18-layer deep residual neural.

Layer (type)	Configuration	Output shape
InputLayer		(256, 512, 1)
Convolution2D	64 7x7 kernels, 2x2 stride	(128, 256, 64)
MaxPooling2D	3x3 kernel, 2x2 stride	(64, 128, 64)
BasicBlock	64 3x3 kernel, 1x1 stride	(64, 128, 64)
BasicBlock	64 3x3 kernel, 1x1 stride	(64, 128, 64)
BasicBlock	128 3x3 kernel, 2x2 stride	(32, 64, 128)
BasicBlock	128 3x3 kernel, 1x1 stride	(32, 64, 128)
BasicBlock	256 3x3 kernel, 2x2 stride	(16, 32, 256)
BasicBlock	256 3x3 kernel, 1x1 stride	(16, 32, 256)
BasicBlock	512 3x3 kernel, 2x2 stride	(8, 16, 512)
BasicBlock	512 3x3 kernel, 1x1 stride	(8, 16, 512)
AveragePooling2D	8x16 pool size, 1x1 stride	(1, 1, 512)
Flatten		(512)
Dense	He normal, softmax	(999)
Total Params	11,691,751	

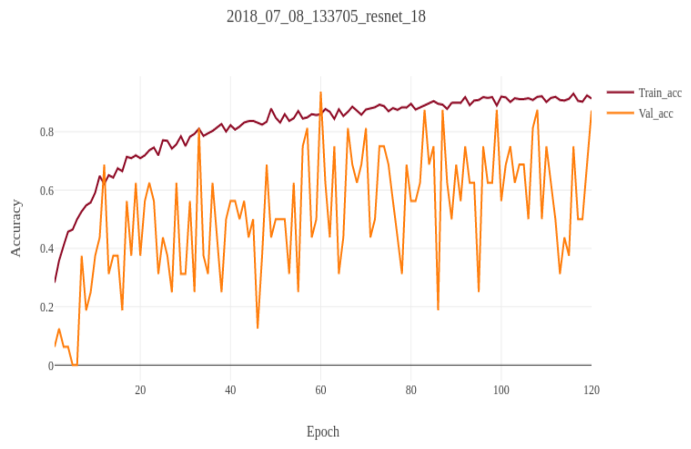
The **loss function** is a measure of how well the network model performs on a set of labelled data points. The most commonly used loss functions are mean squared error and cross entropy both of which are continuous, differentiable, and can be computed efficiently, which is needed during the optimization to compute the gradient. In this work we use the cross entropy loss function. The single-label **categorical cross entropy** function can be defined as:

$$C = -\frac{1}{N}\sum_{N=1}^{N}[y_n \log y_n + (1 - y_n) + (1 - y_n)]$$

where N is the total number of training samples, $y_n = f(\overline{x_n})$ is the desired output for data point $\overline{x_n}$ and $\hat{y}$n = $fw(\overline{x_n})$ is the estimated output.

IV. Experimental results

We present the results for the deep residual neural network. The residual network learns to classify bird species based on bird song. The classification accuracy is comparable to that of the baseline – SMACPY [6]. The training history for the deep residual network can be seen in Figure 5. The figures show the loss for the models when evaluated on the training data (red) and the validation data (orange) with respect to the training epoch. They also show the top-1 accuracy for the models when evaluated on the training data (red) and the validation data (orange) with respect to the training epoch. The top-1 accuracy shown in the figures is computed with respect to each individual segment in the training and validation data i.e., It is not the average of the segments in each recording. Each network was trained for 120 epochs, which resulted in 3 days on the GPU for the residual neural network. The MWFD data augmentation was only tested while training on the smaller dataset as we are working on 30 classes.



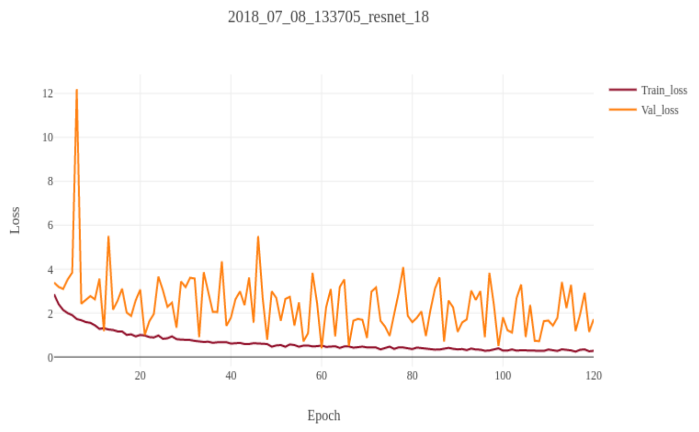


Fig 5. The training history for the 18-layer residual neural network. The figure shows the change in training and validation accuracy (Top image), and the change in training and validation loss (Bottom image) with respect to the training epoch

As a result it considers identification and predicted output as a result of this model. The accuracy of model diverge for different classes of birds sound. The model has a decent accuracy for 30 sound classes on which it gives accuracy of 80%. As out of 30 birds classes it identifies 24 accurate classes. A perfect accuracy would result in identification of accurate 30 bird classes. The comparative analysis based on Accuracy of the model (baseline and ResNet) is shown below (see Table 2).

Table 2
the top-1 accuracy of the baseline, and the residual neural network. Each method has been trained on the different 4 data set three times, and then the average of the evaluation scores and their standard deviation has been computed.

Dataset	Class_5		Class_15		Class_25		Class_30	
Methods	Train_acc	Val_acc	Train_acc	Val_acc	Train_acc	Val_acc	Train_acc	Val_acc
Non-Deep	0.7930	0.9300	0.6073	0.6100	0.5438	0.5500	0.4706	0.4400
Deep (ResNet)	0.9930	1.0000	0.9573	0.5000	0.9338	0.7500	0.9306	0.7500

V. Conclusion and Future Work

RNN is an effective approach for audio classification. It Outperforms existing open source implementation i.e. SMACPY therefore it can be an effective alternative to other existing methods. Proposed architecture experimentally seemed to be more scalable as with the increase in the number of classes. In this work, we set out to improve the classification accuracy of the state-of-the art bird species classifier. From 30 bird classes we properly classify the 24 species. That means we got about 80% of accuracy. Therefore we can say RNN able to comfortably beat the existing open source implementation. With the availability of more computing resources, the effectiveness of the presented method can be further improved. Since, deep learning method is effective but at the same time it's computational very expensive.

We could tackle the harder **single-instance multi-label** problem and modify the loss function such that it considers the background species. In the pre-processing step it could be used to actually classify which parts of the recordings are noise and which are bird song. This could result in a more accurate representation of the training data.

References

[1] E. Sprengel, M. Jaggi, Y. Kilcher and T. Hofmann, "Audio bird species identification using deep learning techniques", In: CLFF Working Notes (Springer, Cham, Switzerland), 2016.
[2] Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun, "Deep Residual Learning for Image Recognition", In: CVPR, arXiv: 1512.03385v1 [CS], 2015.
[3] S. Ioffe, C. Szegedy, "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift", In: ICML, 2015.
[4] Ruben Gonzalez, "Better than MFCC audio classification features", In:The Era of Interactive Media, pages 291–301, 2013.
[5] Dan Stowell, Yannis Stylianou, Mike Wood, "Automatic acoustic detection of birds through deep learning: the first Bird Audio Detection challenge", arXiv: 1807.05812[cs.SD], 2017.
[6] Stefan Kahl, Thomas Wilhelm-Stein, Hussein Hussein, and Maximilian Eibl, "Large-Scale Bird Sound Classification using Convolution Neural Networks", In: CLEF 2017, At Dublin, Ireland, 2017.
[7] K. Uma Rani, M. S. Holi, "A Comparative study of Neural Networks and Support Vector Machines for neurologist disorder voice classification", In: IJERT, ISSN:2278-0181, 2014.
[8] https://blog.waya.ai/deep-residual-learning-9610bb62c355
[9] https://towardsdatascience.com/an-overview-of-resnet-and-its-variants-5281e2f56035