**Dataset**

The dataset was created by collating audio of Indian birds from the Indian Bird Song website (https://indianbirdsong.org/). The database is sourced from Xeno-Canto avian acoustic database. There are around 1,321 avian species in India, we have recordings of 967 of them available. Out of these, we selected species with a larger base of recordings (30-35 and above per specie) to be able to train our model well with a good variety of recordings. Alphabetically birds were accessed and audio was manually downloaded. A total of 67 species’ audio was collected. Around 4,101 audios were put together.

The audios obtained are from different places in India, different environments, and recorded by different people. The length of the audio, amount of noise, and disturbance all differ. Thus giving us a dataset very close to the data we will be getting in real-time when the application is put to use.

**PreProcessing - 1**

The audio files put together were denoised. Both the noisy and noiseless audio are used to train, test and validate the model. Each file is considered a separate instance, so initially, a single file is denoised, and then both the noiseless and the noisy file are used as 2 separate instances of the same class. So a total of 8,202 audios are used.

**Feature Extraction**

Following denoising, feature extraction was performed. Multiple features were extracted over various domains. Spectral Centroids, RMS, Chromagram, and MFCCS features are extracted.

Spectral Features describe the vibration amount at every frequency. Each individual bird has a unique combination of frequencies for their tune, recognizing this sequence can be used to identify a bird thus making it a valid feature for training.

The first spectral feature that was extracted was the Spectral Centroid. It aims to find the center of mass of the audio, it is connected to the timbre or tone quality. For each frame, the mean is calculated after normalizing the magnitude spectrum. The frequency bins generally considered use Fourier Frequencies. Using the librosa library we were able to find the spectral centroids with a sampling rate of 44,100 and Short-time Fourier Transform (STFT) hop length of 8,192 for the returned feature array length to be kept less than 1,000.

The RMS (Root Mean Square) is the next spectral feature that was calculated. It computes the average loudness for the audio in other words for a waveform it is the net effective value. For each specie, this value differs thus making it a good parameter to base our classification. For each frame the RMS is found from the spectrogram, as even though this takes a bit more time for computation of the STFT it is more accurate in representing the energy in time over frames as it can use windows. Again, a hop length of 8,192 and a sampling rate of 44,100 was used.

The Chroma feature is another spectral feature we used. It converts the values of the magnitude spectrum into 12 chroma or semitones i.e., 12 bins. It is basically the pitch-class profile. The melodic and harmonic characters of the audio clip are identified by this feature. Species have their own melody to their bird songs, identifying what is unique to it can contribute to identifying the specie. This feature is robust to change in the timbre, and harmony and can thus identify the difference in octaves of pitches. STFT is again made use of to calculate the chromograms, we used a hop length of 8,192 and a sampling rate of 44,100.

The final spectral feature taken was the Mel-Frequency Cepstral Coefficients (MFCCs). We used MFCCs as it shows the information in the signal that mimics human hearing perception, low level auditory, letting us thus add that small human hearing aspect. Technically, the MFCC is a representation on the Mel scale, as human perception of frequencies is not linear. On this scale the pitch that sounds to have equal distance to humans is equally distant. A STFT spectrogram is again used here, a hop length of 8,192 and a sampling rate of 44,100 was used by us. The spectrogram is then represented in the Mel scale by applying mel scale filter banks, this spectrogram is called Mel spectrogram. A Discrete Cosine Transform (DCT) is then applied on this to get the MFCCs. Normally the first 2 to 13 coefficients are taken of the signal after the transform, and 26 features that include coefficients of derivatives of first and second order giving a total of 39 features, but for this project we used 125 features, the finer details of the spectrum are represented by the higher coefficients.

**PreProcessing - 2**

The extracted features are then padded with zeros or trimmed depending on their size. A maximum length of 1000 is applied on all features, to equalize the dataset. Around **64** instances had their features trimmed, rest had them padded with zeroes.

Following this the labels are encoded using LabelEncoder, that is here the labels are normalized. The non-numerical classes here ( the specie names) are converted into numerical classes to transform the data to be able to pass it to the model. Here now each class is assigned a number from 0 to 67.

The dataset is now split into train, test and validation with 64% , 16%, 20% of the data giving 5,250 instances to train, 1,641 to validate and 1,313 to test the model on.

**Model**

We initially tried a variety of models on a smaller dataset with just 38 classes to find the one best suited for this task. A simple Neural Network , simple Convolutional Neural Network (CNN) , pre-trained CNN models like Inception, ResNet50, VGG16, and EfficientNet were used with Stochastic Gradient Descent, Adagrad, and RMS as optimizers. The accuracy was very marginal, the highest being 35.8% for the Neural Network with the Stochastic Gradient Descent optimizer.

An Artificial Neural Network (ANN) works well with textual or tabular data, when compared to a CNN or an RNN it is less powerful, and also has the problem of vanishing gradient. CNN works well with image data here using a sequence is easier, and more features are extracted, also it is spatially invariant i.e., when we shift/move the signal at input there is an equal output signal shift. When following this, RNN seemed to be the better solution. To work on the long-term dependence and the vanishing gradient problem, we used Long Short-Term Memory (LSTM), Gated Recurrent Unit (GRU), and Bidirectional LSTM (BiLSTM), instead to improve the memory.

An LSTM has both feedforward and unlike other standard feedforward neural networks’ feedback connections too. This allows us to use a chain of data instead of single points. LSTM passes important information through the network and ignores unnecessary information. It makes use of its three different gates to do so, the input, the forget, and the output gates. Although an LSTM does solve the vanishing gradient problem to some existent, it still exists. The sequence length that it can remember is still short and the path traveled is a long one.

A GRU makes use of reset and update gates to store information. They can keep very old information without diluting it over time while adding only necessary information and removing unnecessary information.

A BiLSTM has two LSTMs, one to take the input forward and one backward. The quantity of information available to the network is increased, thus the context given to the algorithm is improved drastically. Sequential dependencies are thus modelled well. Using mean, sum, concatenation or multiplication the outputs are combined from both the layers. When compared to GRUs and LSTMs, the BiLSTM can store more data and can use both past and future information in a particular time step.

When trained on our dataset a GRU based model did not give us satisfying results, so we tried a combination of LSTM and Bidirectional LSTM layers for our model.

