

ABSTRACT

This project focuses on developing a machine learning model to identify bird species based on their vocalizations. Given the challenges posed by datasets with imbalanced class distributions, the aim is to curate a selection of bird species that have sufficient audio samples for effective model training. We utilize advanced algorithms, including Convolutional Neural Networks (CNN), Long Short-Term Memory (LSTM) networks, and WavNet, alongside comprehensive feature extraction techniques. The audio features extracted include zero-crossing rate, root mean square energy, and Mel-frequency cepstral coefficients (MFCCs), which are pivotal for distinguishing vocal characteristics among species. The model's performance is enhanced through data augmentation strategies, such as noise addition and pitch shifting, to increase the diversity of training samples. This approach allows for robust classification, even with a limited number of audio recordings per species. Ultimately, our model demonstrates the potential to accurately predict bird species based on audio input, contributing to biodiversity studies and ecological monitoring efforts.

Keywords: Bird Sound Recognition, Machine Learning, CNN, LSTM, WavNet, Feature Extraction, Zero-Crossing Rate, Root Mean Square (RMS) Energy, MFCCs, Data Augmentation, Noise Addition, Pitch Shifting, Species Classification.

INDEX

CHAPTER 1 – INTRODUCTION	0
CHAPTER 2 – SYSTEM ANALYSIS.....	2
A. Existing system	2
B. Proposed System.....	4
CHAPTER 3 – FEASIBILITY STUDY	5
A. Technical Feasibility	5
B. Operational Feasibility	5
C. Economic Feasibility	5
CHAPTER 4 – SYSTEM REQUIREMENT SPECIFICATION DOCUMENT	6
A.Overview	6
B.Modules Description.....	6
C.Process Flow	9
D.SDLC Methodology	9
E. Software Requirements	14
F. Hardware Requirements	14
CHAPTER 5 – SYSTEM DESIGN	14
A.DFD.....	14
B.ER diagram.....	16
C.UML.....	17
D.Data Dictionary:	22
CHAPTER 6 - TECHNOLOGY DESCRIPTION.....	22
CHAPTER 7 - TESTING & DEBUGGING TECHNIQUES.....	23
CHAPTER 8 – OUTPUT SCREENS	25
CHAPTER 9 –CODE.....	29
CHAPTER 10 – CONCLUSION	38
CHAPTER 11 – BIBLOGRAPHY	38

CHAPTER 1 – INTRODUCTION

Biodiversity plays a crucial role in maintaining the balance of ecosystems, with birds serving as vital indicators of environmental health. Monitoring bird populations is essential for understanding ecological changes, assessing the impact of human activities, and implementing effective conservation strategies. Traditionally, bird monitoring has relied on manual observation and identification, methods that are not only time-consuming and labor-intensive but also subject to human error and limited by the expertise required. In recent years, the advent of machine learning has opened new avenues for automating and enhancing bird species identification through the analysis of vocalizations, offering a more efficient and scalable solution to biodiversity monitoring.

Bird vocalizations, encompassing songs and calls, are unique to each species and provide rich data for identification. However, distinguishing between species based solely on audio presents significant challenges. These challenges include overlapping vocal patterns among similar species, varying recording conditions, and the inherent complexity of audio data. Moreover, datasets used for training machine learning models often suffer from imbalanced class distributions, where some bird species are underrepresented due to the scarcity of available audio samples. This imbalance can lead to biased models that perform well on frequently sampled species but poorly on others, undermining the reliability and generalizability of the identification system.

To address these challenges, this project focuses on developing a robust machine learning model capable of accurately identifying bird species based on their vocalizations. The core objective is to curate a comprehensive dataset that includes a balanced selection of bird species with sufficient audio samples, thereby mitigating the issues associated with imbalanced class distributions. By leveraging advanced algorithms such as Convolutional Neural Networks (CNN), Long Short-Term Memory (LSTM) networks, and WavNet, the project aims to harness the strengths of each architecture to enhance the model's ability to capture both spatial and temporal features inherent in bird sounds.

A pivotal aspect of this project is the extraction of meaningful audio features that are essential for distinguishing between different bird species. The feature extraction process includes calculating the zero-crossing rate, root mean square (RMS) energy, and Mel-frequency cepstral coefficients (MFCCs). The zero-crossing rate measures the rate at which the audio signal

changes sign, providing insights into the signal's frequency content. RMS energy quantifies the signal's power, offering information about the loudness and intensity of the vocalizations. MFCCs capture the short-term power spectrum of the sound, effectively representing the vocal characteristics that are crucial for species differentiation. These features collectively enable the model to discern subtle variations in bird calls, enhancing the accuracy of species classification.

To further improve the model's performance, data augmentation strategies are employed to increase the diversity of training samples. Techniques such as noise addition and pitch shifting are utilized to simulate a variety of recording conditions and vocal variations, thereby enhancing the model's ability to generalize to real-world scenarios. Noise addition introduces background sounds that birds might encounter in their natural habitats, while pitch shifting alters the frequency of the audio signals to mimic different vocal tones. These augmentation methods help in creating a more resilient model capable of maintaining high accuracy even with limited audio recordings per species.

The integration of CNNs, LSTMs, and WavNet into the model architecture allows for a comprehensive analysis of bird vocalizations. CNNs are adept at extracting spatial features from spectrogram representations of audio signals, while LSTMs excel in capturing temporal dependencies and patterns over time. WavNet, a deep generative model for raw audio, enhances the model's ability to process high-fidelity audio data, ensuring that the intricate details of bird sounds are preserved and utilized effectively for classification.

Ultimately, the developed model demonstrates significant potential in accurately predicting bird species based on audio input, thereby contributing valuable tools for biodiversity studies and ecological monitoring efforts. By automating the identification process, this project not only accelerates data collection and analysis but also supports large-scale conservation initiatives aimed at preserving avian diversity. The successful implementation of this machine learning-based bird sound recognition system underscores the transformative impact of artificial intelligence in environmental science, paving the way for more sophisticated and reliable methods in the quest to understand and protect our planet's rich biodiversity.

CHAPTER 2 – SYSTEM ANALYSIS

A. Existing system

Current bird sound recognition systems predominantly rely on manual identification through field observations, which are time-consuming and subject to human error. Automated approaches have emerged using machine learning techniques such as Support Vector Machines (SVM), Random Forest, and other classical machine learning models to classify bird species based on audio recordings. These systems typically utilize features like Mel-Frequency Cepstral Coefficients (MFCCs) and spectral analysis for sound characterization. However, they often face challenges with imbalanced datasets, limited audio samples, and variability in recording conditions, which hinder accuracy and scalability. Additionally, many existing models lack the integration of more sophisticated temporal modeling techniques, limiting their ability to capture complex vocal patterns essential for precise species identification.

Disadvantages

- **Imbalanced Datasets:** Many bird species have significantly fewer audio samples, leading to biased model training and poor generalization for underrepresented classes.
- **Limited Audio Quality:** Variability in recording conditions, such as background noise and different recording devices, can degrade the accuracy of species identification.
- **High Computational Requirements:** Advanced machine learning models like SVM, Random Forest, and Gradient Boosting require substantial computational resources, making them less accessible for smaller projects or real-time applications.
- **Feature Extraction Complexity:** Extracting meaningful audio features (e.g., MFCCs) is computationally intensive and may require specialized knowledge to implement effectively.
- **Overlapping Vocalizations:** Similar vocal patterns among different bird species can cause confusion and reduce classification accuracy.
- **Data Augmentation Limitations:** While techniques like noise addition and pitch shifting help, they may not fully capture the natural variability of bird sounds, potentially limiting model robustness.
- **Scalability Issues:** Scaling the system to include a larger number of species or integrating it into real-world monitoring systems can be challenging due to data and computational constraints.

- **Dependency on High-Quality Labels:** Accurate species identification relies on correctly labelled training data, which can be time-consuming and expensive to obtain, especially for rare species.

B. Proposed System

The proposed system involves the development of a machine learning model for bird species identification based on vocalizations. It integrates a curated dataset containing sufficient audio samples for various species, employing advanced algorithms such as Convolutional Neural Networks (CNN), Long Short-Term Memory (LSTM) networks, and WavNet for accurate classification. Key components include robust feature extraction techniques (e.g., MFCCs) and data augmentation methods to enhance model training. The system will provide a user-friendly interface, enabling researchers and enthusiasts to easily identify bird species from audio recordings, ultimately supporting biodiversity research and conservation efforts through automated analysis.

Advantages

- **Enhanced Accuracy:** Utilizes advanced machine learning architectures such as CNN, LSTM, and WavNet, significantly improving the precision of bird species classification.
- **Effective Handling of Imbalanced Data:** Implements data augmentation techniques like noise addition and pitch shifting to balance datasets, ensuring robust model training for underrepresented species.
- **Comprehensive Feature Extraction:** Employs a variety of audio features, including MFCCs, zero-crossing rate, and root mean square energy, to capture detailed vocal characteristics essential for distinguishing between similar species.
- **Scalability:** Designed to accommodate a growing number of bird species and large volumes of audio data, facilitating expansion for broader biodiversity studies.
- **Robustness to Audio Variability:** Incorporates preprocessing steps and data augmentation to mitigate issues related to background noise and varying recording conditions, enhancing the model's reliability in diverse environments.
- **Automation and Efficiency:** Automates the bird identification process, reducing the need for time-consuming and labor-intensive manual observations, and enabling large-scale ecological monitoring.

- **Resource Optimization:** Optimizes computational resources through efficient model architectures, making the system accessible for both large-scale projects and smaller, resource-constrained applications.

CHAPTER 3 – FEASIBILITY STUDY

A. Technical Feasibility

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

B. Operational Feasibility

The proposed bird sound recognition system is operationally feasible and well-suited for real-world deployment in ecological monitoring and biodiversity studies. The system's architecture, built using established machine learning models like CNN, LSTM, and WavNet, ensures reliable performance with minimal manual intervention. The user interaction is straightforward audio recordings are input into the system, which then processes and classifies them automatically using pre-trained models. The data augmentation techniques implemented during training enhance the system's ability to generalize across varied environmental conditions and background noise, ensuring consistent performance in field applications. Furthermore, the system is lightweight enough to be integrated into mobile or edge devices for on-site species identification. With proper training for field researchers and conservationists, the system can be seamlessly incorporated into existing wildlife monitoring workflows, making it both practical and impactful for long-term ecological use.

C. Economic Feasibility

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus, the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

CHAPTER 4 – SYSTEM REQUIREMENT SPECIFICATION DOCUMENT

A. Overview

This project aims to develop a machine learning-based system capable of identifying bird species solely from their vocalizations. Bird sounds offer a rich and unique dataset for classification tasks, as each species possesses distinct acoustic features. However, real-world bird sound datasets often suffer from class imbalances and environmental noise, which pose significant challenges for accurate classification. To address this, the project focuses on curating a dataset with sufficient and balanced audio samples of selected bird species and enhancing model performance through effective feature extraction and data augmentation techniques.

The system utilizes deep learning architectures such as Convolutional Neural Networks (CNN), Long Short-Term Memory (LSTM) networks, and WavNet to learn and recognize patterns in bird calls. Key audio features such as zero-crossing rate, root mean square (RMS) energy, and Mel-frequency cepstral coefficients (MFCCs) are extracted and used to train the model. Additionally, techniques like noise addition and pitch shifting are applied to diversify the dataset and improve generalization.

The end goal is to develop a reliable, accurate, and scalable tool for automatic bird species classification using audio recordings. This can significantly aid ecological research, biodiversity tracking, and wildlife conservation by enabling passive and non-invasive monitoring of bird populations in various habitats.

B. Modules Description

The bird species classification system is composed of several integrated modules that work together to ensure accurate and efficient audio-based prediction. The **Audio Input and Preprocessing Module** is responsible for receiving raw bird sound recordings and applying preprocessing techniques such as noise removal, normalization, and segmentation. The **Feature Extraction Module** converts the cleaned audio into meaningful numerical representations using features like zero-crossing rate, root mean square (RMS) energy, and Mel-frequency cepstral coefficients (MFCCs), which are essential for distinguishing species-specific vocal traits. The **Data Augmentation Module** enhances the diversity of training data by applying transformations such as pitch shifting, time stretching, and background noise

addition to improve the model's robustness. The **Model Training and Classification Module** utilizes deep learning algorithms including CNN, LSTM, and WavNet to learn temporal and spectral patterns from the extracted features, enabling accurate classification of bird species. Finally, the **Evaluation and Prediction Module** tests the trained models on unseen data, provides performance metrics, and allows for real-time prediction of bird species from new audio samples. These modules collectively support a scalable and accurate solution for avian sound recognition and monitoring.

1. System:

Create Dataset:

The dataset comprises audio recordings of various bird species with sufficient samples for each class. These recordings are organized and divided into training and testing sets, typically allocating 20-30% of the data for testing. This partitioning ensures effective model evaluation and helps in assessing the system's generalization capabilities across different bird species.

Pre-processing:

Audio files are standardized by resampling to a uniform frequency and trimming or padding to ensure consistent duration. Comprehensive feature extraction techniques are applied, including Mel-Frequency Cepstral Coefficients (MFCCs), zero-crossing rate, and root mean square (RMS) energy. Data augmentation methods, such as noise addition and pitch shifting, are employed to enhance the diversity of the training samples, thereby improving the model's ability to generalize to varied input conditions.

Training:

The pre-processed training dataset is utilized to train advanced machine learning models, including Convolutional Neural Networks (CNN), Long Short-Term Memory (LSTM) networks, and WavNet. Transfer learning techniques are leveraged by fine-tuning pre-trained models, which accelerates the training process and enhances overall model performance. Hyperparameter tuning is conducted to optimize the models for accurate bird species classification.

Evaluation:

Trained models are evaluated using the testing dataset to assess their performance. Key performance metrics such as accuracy, precision, recall, and F1-score are computed to determine the effectiveness of the models in correctly identifying bird species. Confusion matrices and ROC curves are also analyzed to understand the models' strengths and areas for improvement.

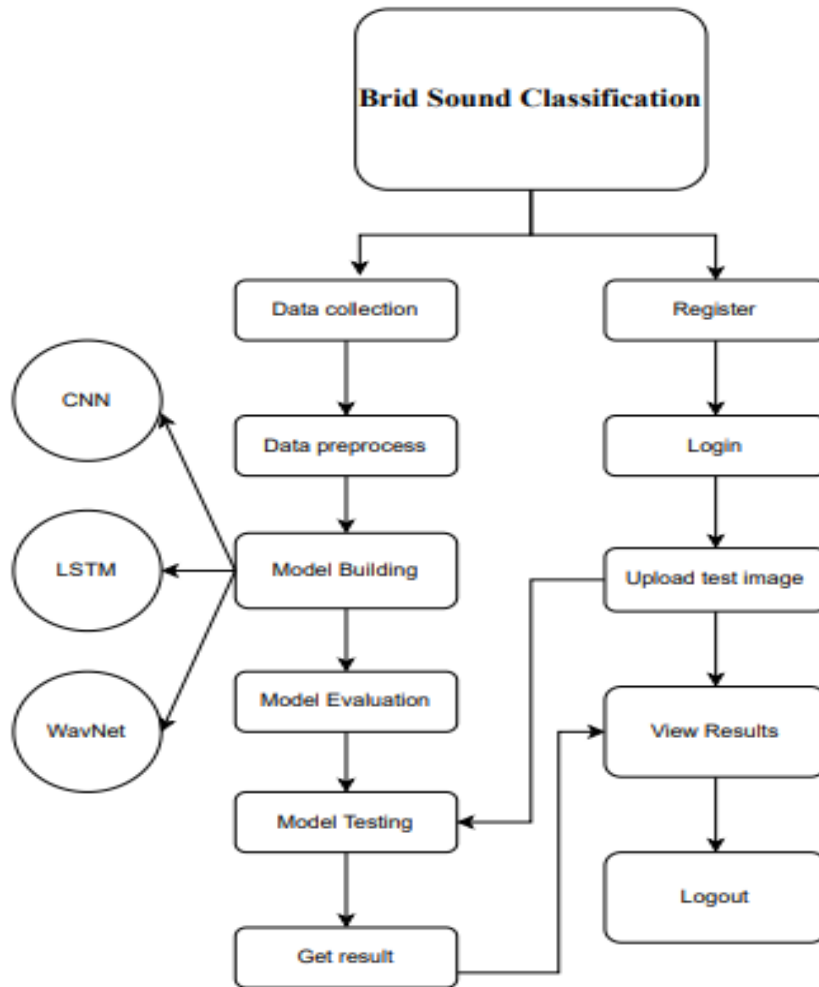
Classification:

The finalized models are deployed to classify new audio inputs. Upon receiving an audio file, the system processes it through the trained models to predict the corresponding bird species. The results include the identified species along with confidence scores, providing users with interpretable and reliable classification outcomes.

User

- **Registration:** Users can create an account by providing necessary information such as name, email, and password. Registration is essential for accessing personalized features, saving user-specific data, and tracking analysis history.
- **Login:** Registered users can securely log in using their credentials (email and password). The authentication process ensures that user data and system access are protected, maintaining privacy and security.
- **Upload Audio:** Users can upload audio files of bird vocalizations through a user-friendly web or mobile interface. The system supports various audio formats and automatically prepares the files for classification by performing necessary pre-processing steps.
- **View Results:** After processing, users receive and view the classification results, which include the identified bird species and corresponding confidence levels. Results are displayed in an intuitive format, often accompanied by additional information such as species descriptions and images to enhance user understanding and engagement.

C. Process Flow



D.SDLC Methodology

SOFTWARE DEVELOPMENT LIFE CYCLE

The meaning of Agile is swift or versatile. "Agile process model" refers to a software development approach based on iterative development. Agile methods break tasks into smaller iterations, or parts do not directly involve long term planning. The project scope and requirements are laid down at the beginning of the development process. Plans regarding the number of iterations, the duration and the scope of each iteration are clearly defined in advance. Each iteration is considered as a short time "frame" in the Agile process model, which typically lasts from one to four weeks. The division of the entire project into smaller parts helps to minimize the project risk and to reduce the overall project delivery time requirements. Each

iteration involves a team working through a full software development life cycle including planning, requirements analysis, design, coding, and testing before a working product is demonstrated to the client.

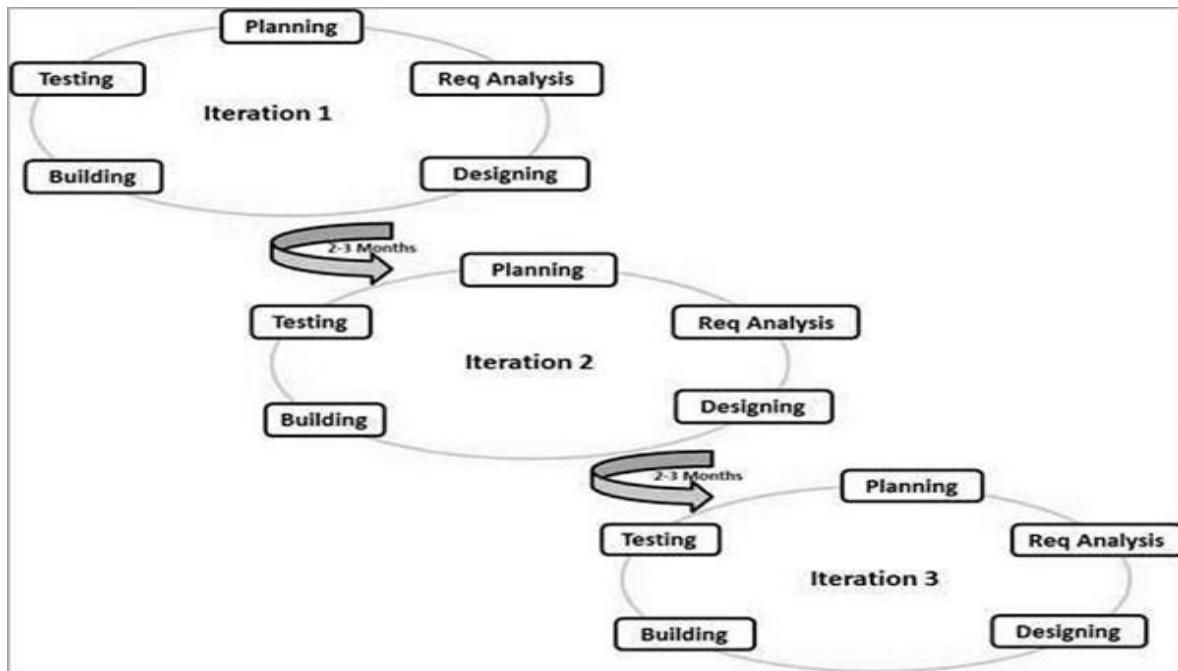
Actually, Agile model refers to a group of development processes. These processes share some basic characteristics but do have certain subtle differences among themselves. A few Agile SDLC models are given below: Crystal A tern Feature-driven development Scrum Extreme programming (XP) Lean development Unified process In the Agile model, the requirements are decomposed into many small parts that can be incrementally developed.

The Agile model adopts Iterative development. Each incremental part is developed over an iteration. Each iteration is intended to be small and easily manageable and that can be completed within a couple of weeks only. At a time one iteration is planned, developed and deployed to the customers. Long-term plans are not made.

Agile model is the combination of iterative and incremental process models. Steps involve in agile SDLC models are:

- Requirement gathering
- Requirement Analysis
- Design Coding
- Unit testing
- Acceptance testing

The time to complete an iteration is known as a Time Box. Time-box refers to the maximum amount of time needed to deliver an iteration to customers. So, the end date for an iteration does not change. Though the development team can decide to reduce the delivered functionality during a Time-box if necessary to deliver it on time. The central principle of the Agile model is the delivery of an increment to the customer after each Time-box.



Principles of Agile model:

- To establish close contact with the customer during development and to gain a clear understanding of various requirements, each Agile project usually includes a customer representative on the team. At the end of each iteration stakeholders and the customer representative review, the progress made and re-evaluate the requirements.
- Agile model relies on working software deployment rather than comprehensive documentation.
- Frequent delivery of incremental versions of the software to the customer representative in intervals of few weeks.
- Requirement change requests from the customer are encouraged and efficiently incorporated.
- It emphasizes on having efficient team members and enhancing communications among them is given more importance. It is realized that enhanced communication among the development team members can be achieved through face-to-face communication rather than through the exchange of formal documents.
- It is recommended that the development team size should be kept small (5 to 9 people) to help the team members meaningfully engage in face-to-face communication and have collaborative work environment.

DETERMINING SPECIES OF BIRD USING THEIR VOICE

- Agile development process usually deploys Pair Programming. In Pair programming, two programmers work together at one work-station. One does code while the other reviews the code as it is typed in. The two programmers switch their roles every hour or so.

Advantages:

- Working through Pair programming produce well written compact programs which has fewer errors as compared to programmers working alone.
- It reduces total development time of the whole project. Customer representatives get the idea of updated software products after each iteration. So, it is easy for him to change any requirement if needed.

Disadvantages:

- Due to lack of formal documents, it creates confusion and important decisions taken during different phases can be misinterpreted at any time by different team members.
- Due to the absence of proper documentation, when the project completes and the developers are assigned to another project, maintenance of the developed project can become a problem.

SOFTWARE DEVELOPMENT LIFE CYCLE – SDLC:

In our project we use waterfall model as our software development cycle because of its step-by-step procedure while implementing.

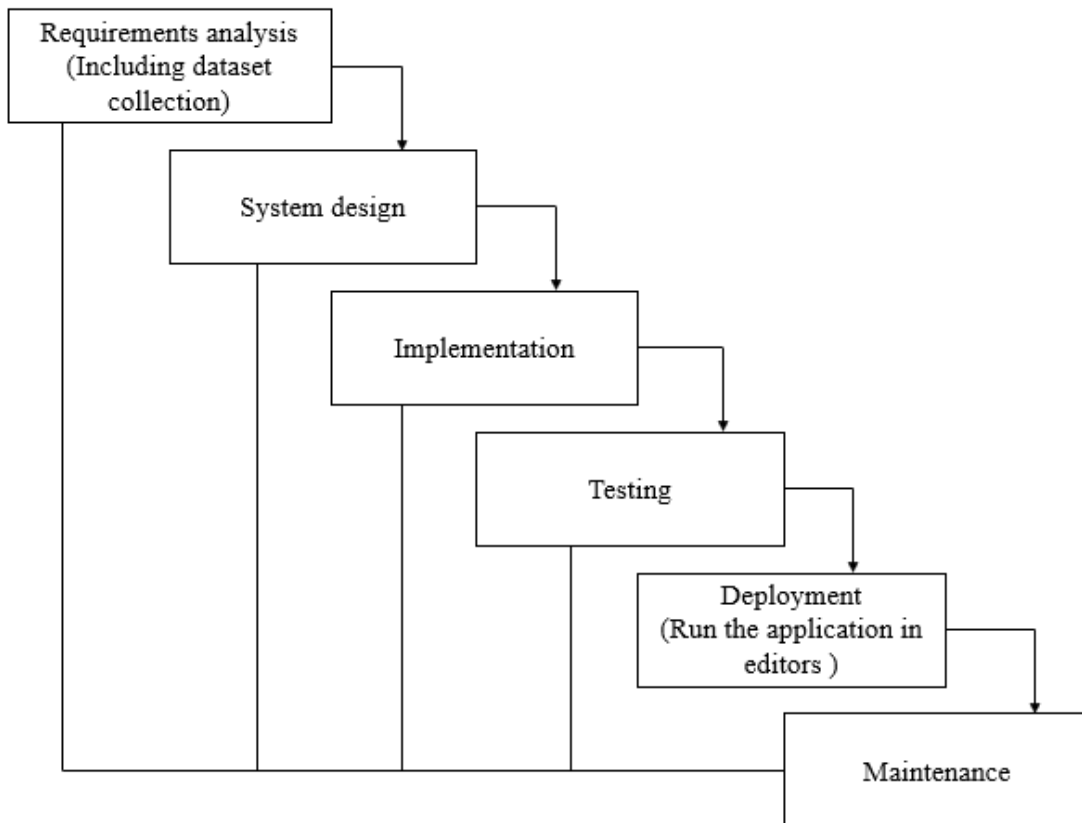


Fig1: Waterfall Model

- **Requirement Gathering and analysis** – All possible requirements of the system to be developed are captured in this phase and documented in a requirement specification document.
- **System Design** – The requirement specifications from first phase are studied in this phase and the system design is prepared. This system design helps in specifying hardware and system requirements and helps in defining the overall system architecture.
- **Implementation** – With inputs from the system design, the system is first developed in small programs called units, which are integrated in the next phase. Each unit is developed and tested for its functionality, which is referred to as Unit Testing.
- **Integration and Testing** – All the units developed in the implementation phase are integrated into a system after testing of each unit. Post integration the entire system is tested for any faults and failures.

- **Deployment of system** – Once the functional and non-functional testing is done; the product is deployed in the customer environment or released into the market.
- **Maintenance** – There are some issues which come up in the client environment. To fix those issues, patches are released. Also, to enhance the product some better versions are released. Maintenance is done to deliver these changes in the customer environment.

E. Software Requirements

- Operating System : Windows 7/8/10
- Server side Script : HTML, CSS, Bootstrap & JS
- Programming Language : Python
- Libraries : Flask, Torch, Tensorflow, Pandas, Mysql.connector
- IDE/Workbench : VSCode
- Server Deployment : Xampp Server
- Database : MySQL

F. Hardware Requirements

- Processor - I3/Intel Processor
- RAM - 8GB (min)
- Hard Disk - 128 GB
- Key Board - Standard Windows Keyboard
- Mouse - Two or Three Button Mouse
- Monitor - Any

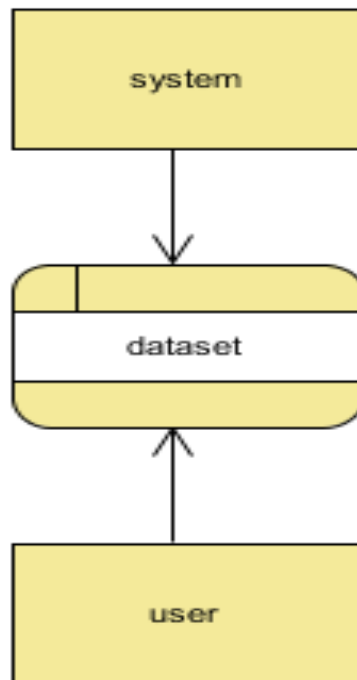
CHAPTER 5 – SYSTEM DESIGN

A.DFD

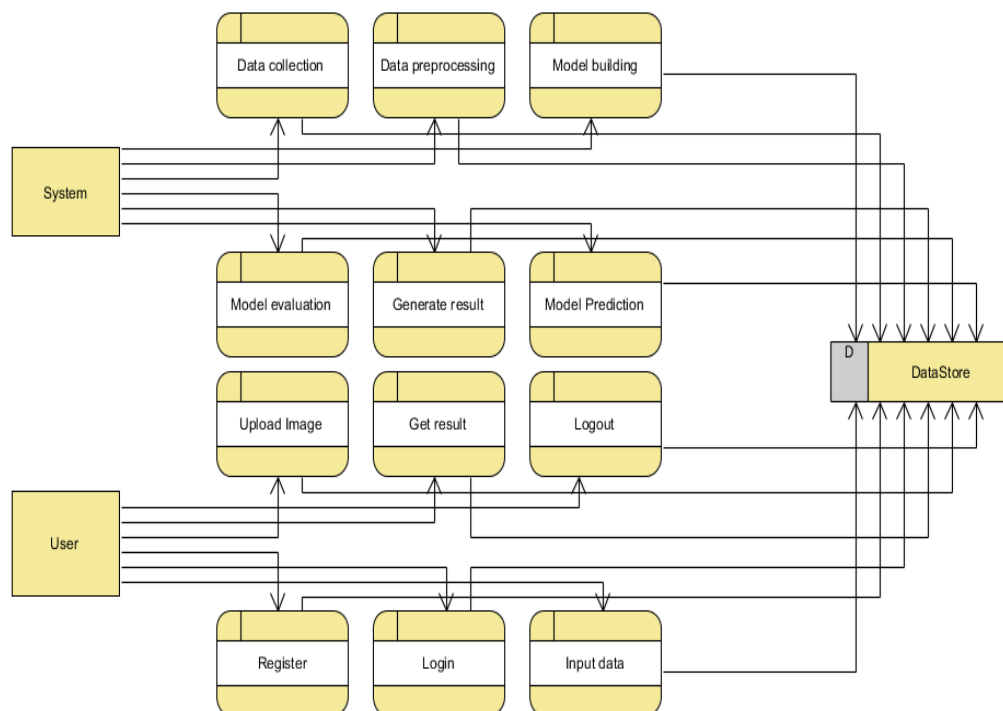
A Data Flow Diagram (DFD) is a traditional way to visualize the information flows within a system. A neat and clear DFD can depict a good amount of the system requirements graphically. It can be manual, automated, or a combination of both. It shows how information enters and leaves the system, what changes the information and where information is stored. The purpose of a DFD is to show the scope and boundaries of a system as a whole. It may be used as a

DETERMINING SPECIES OF BIRD USING THEIR VOICE

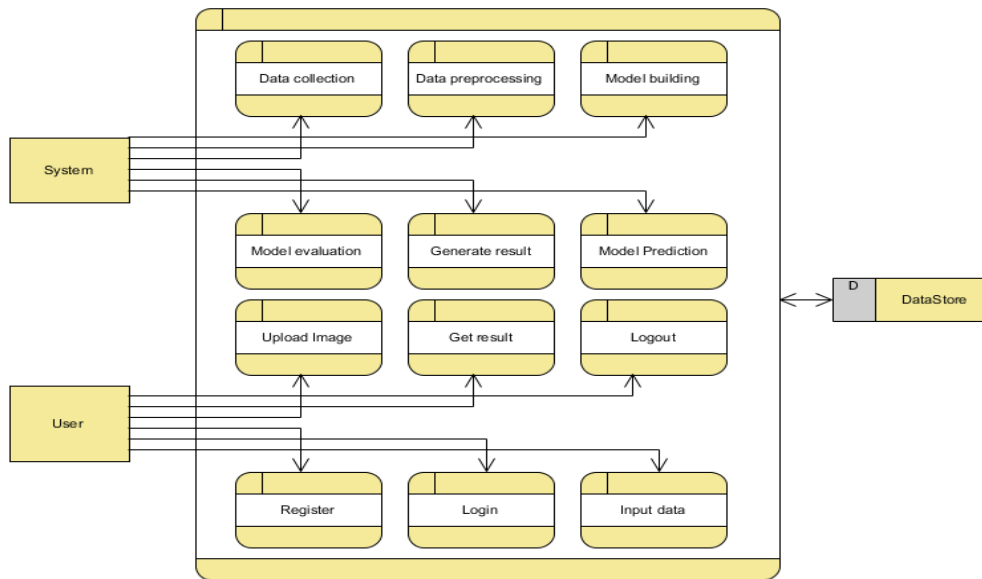
communications tool between a systems analyst and any person who plays a part in the system that acts as the starting point for redesigning a system.



DFD 1 DIAGRAM:



DFD 2 DIAGRAM:

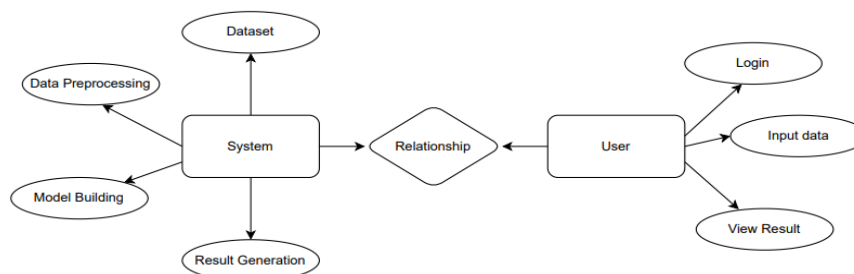


B.ER diagram

An Entity-relationship model (ER model) describes the structure of a database with the help of a diagram, which is known as Entity Relationship Diagram (ER Diagram). An ER model is a design or blueprint of a database that can later be implemented as a database. The main components of E-R model are: entity set and relationship set.

An ER diagram shows the relationship among entity sets. An entity set is a group of similar entities and these entities can have attributes. In terms of DBMS, an entity is a table or attribute of a table in database, so by showing relationship among tables and their attributes, ER diagram shows the complete logical structure of a database. Let's have a look at a simple ER diagram to understand this concept.

ER Diagram

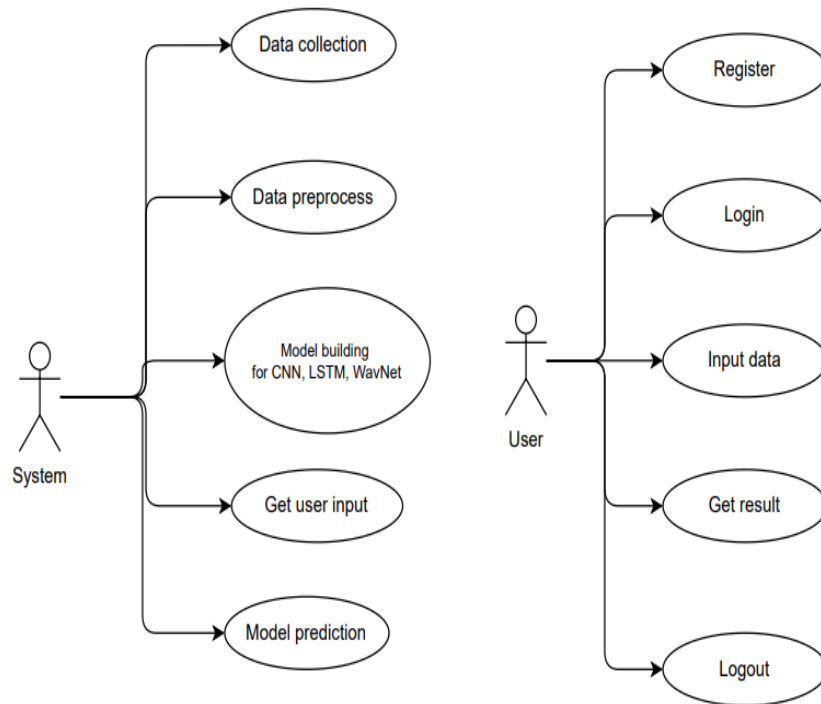


C.UML

- ✓ Uml stands for unified modelling language. Uml is a standardized general-purpose modelling language in the field of object-oriented software engineering. The standard is managed, and was created by, the object management group.
- ✓ The goal is for uml to become a common language for creating models of object-oriented computer software. In its current form uml is comprised of two major components: a meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, uml.
- ✓ The unified modelling language is a standard language for specifying, visualization, constructing and documenting the artefacts of software system, as well as for business modelling and other non-software systems.
- ✓ The uml represents a collection of best engineering practices that have proven successful in the modelling of large and complex systems.

Use case diagram:

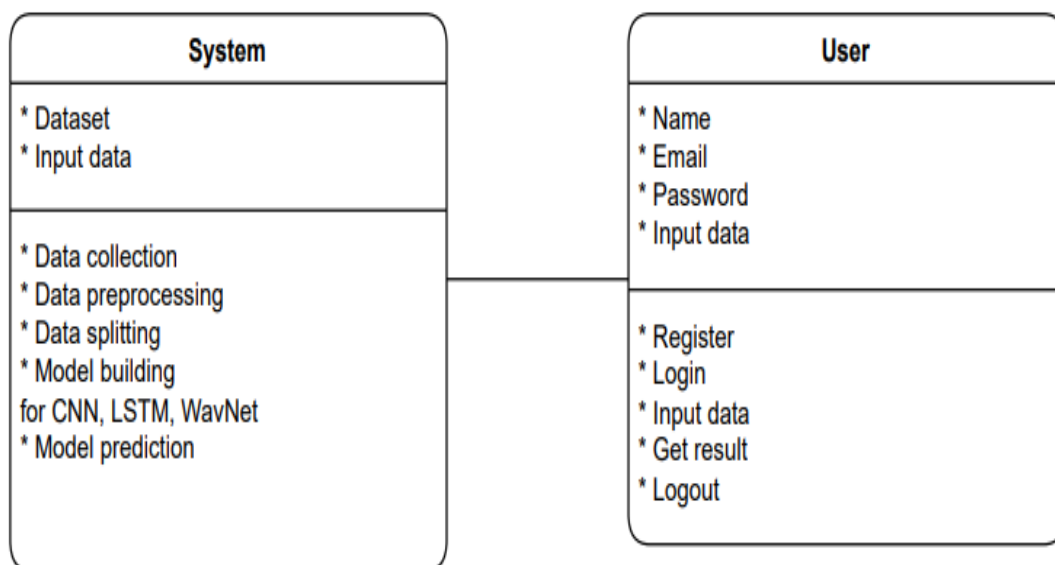
A use case diagram in the unified modeling language (uml) is a type of behavioral diagram defined by and created from a use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.



Activate Wi

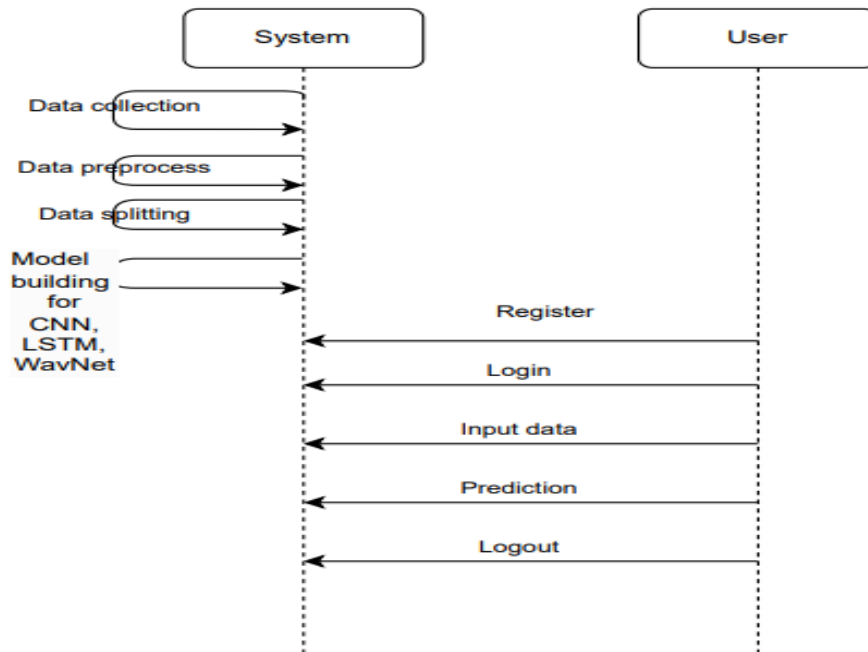
Class diagram:

In software engineering, a class diagram in the unified modeling language (uml) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.



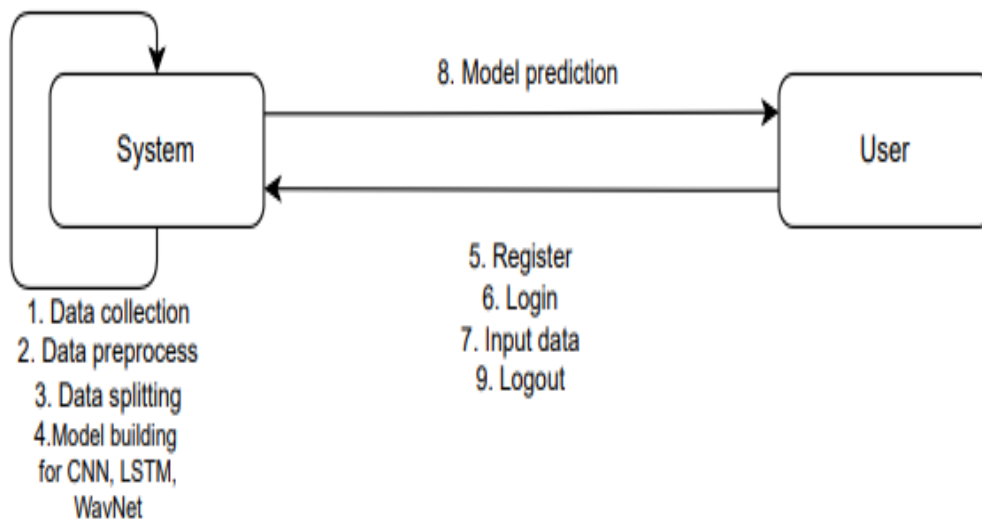
Sequence diagram:

A sequence diagram in unified modeling language (uml) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a message sequence chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.



Collaboration diagram:

In collaboration diagram the method call sequence is indicated by some numbering technique as shown below. The number indicates how the methods are called one after another. We have taken the same order management system to describe the collaboration diagram. The method calls are similar to that of a sequence diagram. But the difference is that the sequence diagram does not describe the object organization whereas the collaboration diagram shows the object organization.



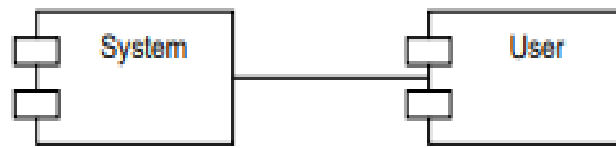
Deployment diagram:

Deployment diagram represents the deployment view of a system. It is related to the component diagram. Because the components are deployed using the deployment diagrams. A deployment diagram consists of nodes. Nodes are nothing but physical hard ware's used to deploy the application.



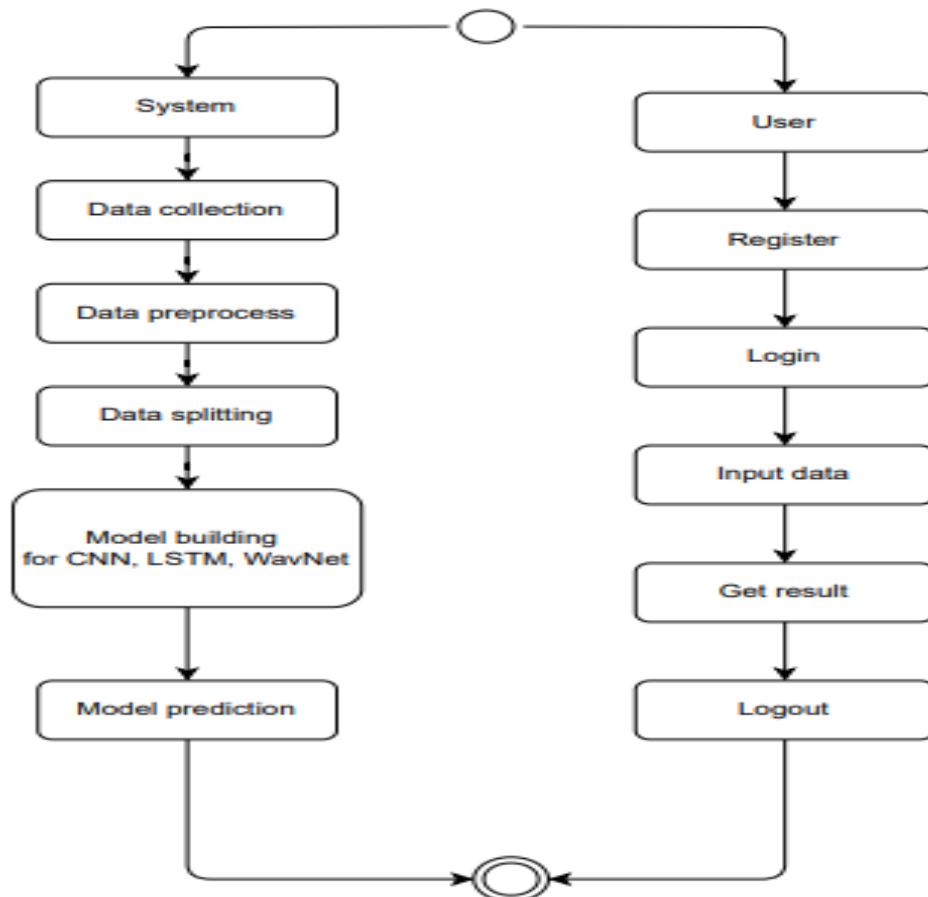
Component diagram:

Component diagrams are used to describe the physical artifacts of a system. This artifact includes files, executable, libraries etc. So the purpose of this diagram is different, component diagrams are used during the implementation phase of an application. But it is prepared well in advance to visualize the implementation details. Initially the system is designed using different uml diagrams and then when the artifacts are ready component diagrams are used to get an idea of the implementation.



Activity diagram:

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the unified modeling language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.



D. Data Dictionary:

<https://github.com/AgaMiko/bird-recognition-review/blob/master/README.md#Open-Source-Projects>

<https://xeno-canto.org/>

https://github.com/AgaMiko/xeno-canto-download/blob/master/notebooks/download_data_example.ipynb

The data dictionary for the bird sound recognition system outlines the essential attributes used for species classification based on audio input. Each record in the dataset includes fields such as `species_id`, `species_name`, `audio_file_path`, and `recording_conditions`. Key extracted audio features include `zero_crossing_rate`, which reflects frequency changes; `rms_energy`, indicating the loudness; and `mfccs`, representing spectral characteristics of vocalizations. Additional fields include `augmented_type` (e.g., pitch shifted, noise added) for tracking data augmentation and label for supervised learning. These structured fields enable consistent data processing, model training, and performance evaluation across diverse bird species and recording conditions.

CHAPTER 6 - TECHNOLOGY DESCRIPTION

Technology Description

The Bird Sound Classification system leverages machine learning and deep learning techniques to accurately identify bird species based on their vocalizations. It integrates a robust audio preprocessing pipeline that extracts meaningful features such as Mel-frequency cepstral coefficients (MFCCs), zero-crossing rate (ZCR), and root mean square (RMS) energy from bird audio recordings. These features represent the frequency, intensity, and tonal characteristics essential for distinguishing bird calls. The system addresses data imbalance challenges using curated datasets and applies data augmentation techniques like noise addition and pitch shifting to improve model generalization. Deep learning architectures, including CNNs, LSTMs, and WavNet, are employed to learn spatial and temporal patterns from processed audio data. The entire workflow—from data ingestion to prediction—is designed for accuracy, scalability, and deployment in real-world biodiversity monitoring. This non-invasive approach provides researchers and conservationists with a powerful tool for passive acoustic monitoring and supports ecological studies and automated wildlife surveys.

- **Convolutional Neural Network (CNN):** CNNs are used to process 2D representations of audio (e.g., spectrograms, MFCCs). They detect spatial features such as pitch and frequency changes to classify bird species accurately.
- **Long Short-Term Memory (LSTM):** LSTMs are ideal for time-series data. They capture temporal dependencies in bird calls, learning how sounds change over time for better species recognition.
- **WavNet:** WavNet works on raw audio waveforms using dilated convolutions, allowing the model to understand fine-grained temporal features and vocal textures.
- **Data Augmentation:** Techniques like noise addition, pitch shifting, and time stretching are applied to expand the dataset and improve model robustness.
- **Feature Extraction:** Key features such as MFCCs, RMS energy, and zero-crossing rate are extracted to represent the tonal, rhythmic, and intensity characteristics of bird sounds.

CHAPTER 7 - TESTING & DEBUGGING TECHNIQUES

A. Unit Testing

- **Purpose:** To validate each core function (e.g., MFCC extraction, pitch shift augmentation) works as expected in isolation.
- **Tools:** PyTest, unittest, Jupyter cells with assertions.
- **Examples:** Test MFCC shape output for a given audio file.

Integration Testing

- **Purpose:** Ensures that all modules from audio upload to species prediction function as a complete pipeline.
- **Tools:** Postman for API tests, Selenium for UI flow, Jupyter integration.
- **Examples:** Upload audio via UI and confirm correct prediction in backend logs.

Model Evaluation Testing

- **Purpose:** To confirm the trained models achieve desired metrics (accuracy, precision, recall) and do not overfit or underfit.
- **Tools:** Scikit-learn metrics, TensorBoard, Confusion Matrix.
- **Examples:** Evaluate CNN model on validation data using classification report.

B. Debugging

- **Purpose:** To identify and fix functional errors in preprocessing, feature extraction, or model predictions.
- **Tools:** Visual Studio Code Debugger, print statements, TensorBoard logs.
- **Examples:** Trace incorrect MFCC values due to sample rate mismatch.

Boundary Testing

- **Purpose:** To test system stability with edge-case audio inputs like silence, overlapping sounds, or very short/long files.
- **Tools:** PyTest with custom test cases, Audacity to create test files.
- **Examples:** Upload 1-second clip and expect "unidentified" label or warning.

Real-Time Data Testing

- **Purpose:** To simulate real-world field scenarios and assess system speed, accuracy, and error handling with live recordings.
- **Tools:** Flask webhooks, live microphones, Postman.
- **Examples:** Real-time recording played into the model returns species prediction within 3 seconds.

User Interface (UI) Testing

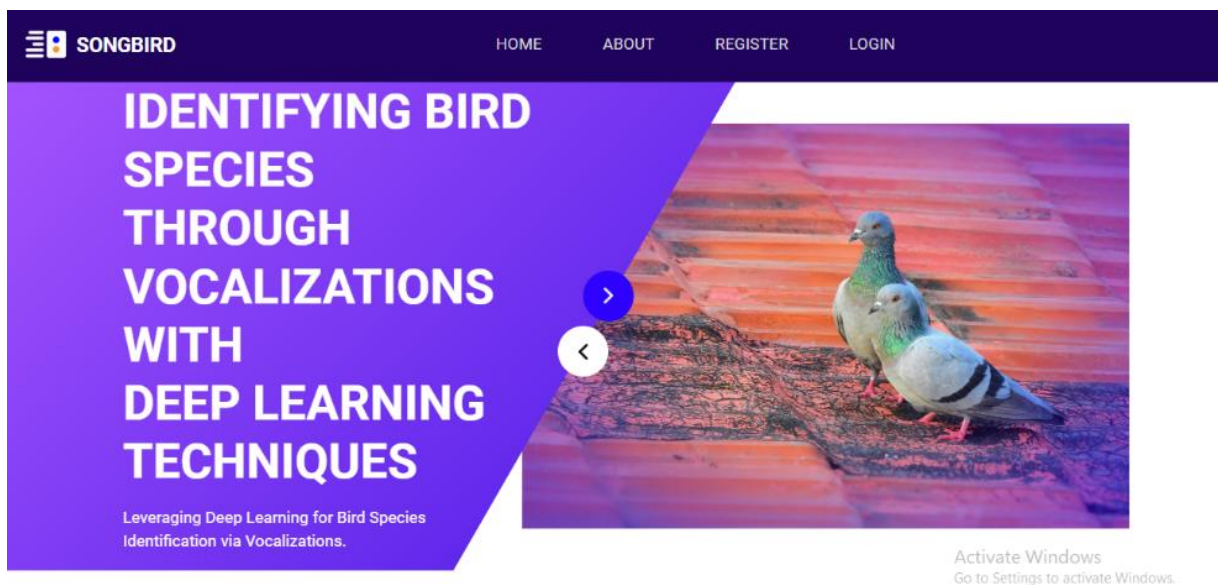
- **Purpose:** To ensure the frontend interface is responsive, accessible, and clearly communicates predictions and errors.
- **Tools:** Selenium, Cypress, manual walkthrough.
- **Examples:** Check if UI displays error for unsupported formats or blank input.

Performance Testing

- **Purpose:** To evaluate how well the system handles large audio files, multiple simultaneous uploads, and continuous requests.
- **Tools:** Locust, Apache JMeter, Python time module.
- **Examples:** Simulate 100 uploads in parallel and measure average response time.

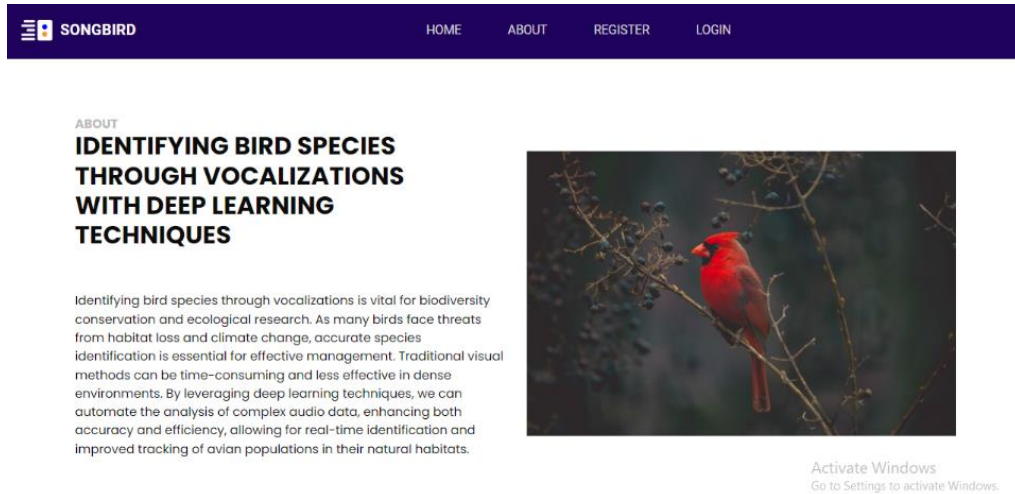
CHAPTER 8 – OUTPUT SCREENS

INDEX PAGE: This is the main landing page of the Songbird system, which introduces the core functionality of identifying bird species through vocalizations using deep learning techniques. The purpose of this page is to provide an overview of the platform, allowing users to navigate to different sections such as registration, login, and uploading audio files for bird species identification.



ABOUT PAGE: The About page provides detailed information on the importance of identifying bird species through vocalizations for conservation and ecological research. It outlines the challenges of traditional methods and highlights how deep learning techniques streamline and improve bird sound identification for tracking avian populations. This page serves to inform users about the platform's relevance and impact.

DETERMINING SPECIES OF BIRD USING THEIR VOICE




REGISTRATION PAGE: This is the registration form output screen. It allows new users to input their details like full name, email, mobile number, age, gender, and password to create an account in the SongBird system.

The screenshot shows the 'REGISTER' page of the SongBird website. The header is dark blue with the 'SONGBIRD' logo and navigation links: HOME, ABOUT, REGISTER, and LOGIN. The main content area contains a registration form with the following fields: Username, Email, Mobile Number, Age, Gender (radio buttons for Male and Female), Password (min 8 characters), and Confirm Password. Below the form is a blue 'Register' button. At the bottom of the form, there is a link that says 'Already registered? [Already have an account](#)'. At the bottom right of the page, there is a Windows watermark that says 'Activate Windows Go to Settings to activate Wi'.

LOGIN PAGE: The login screen allows registered users to enter their email and password to access the SongBird system. It's a simple, secure interface for account verification and accessing personalized features like Bird voice identification.

DETERMINING SPECIES OF BIRD USING THEIR VOICE

 SONGBIRD

HOMEABOUTREGISTERLOGIN

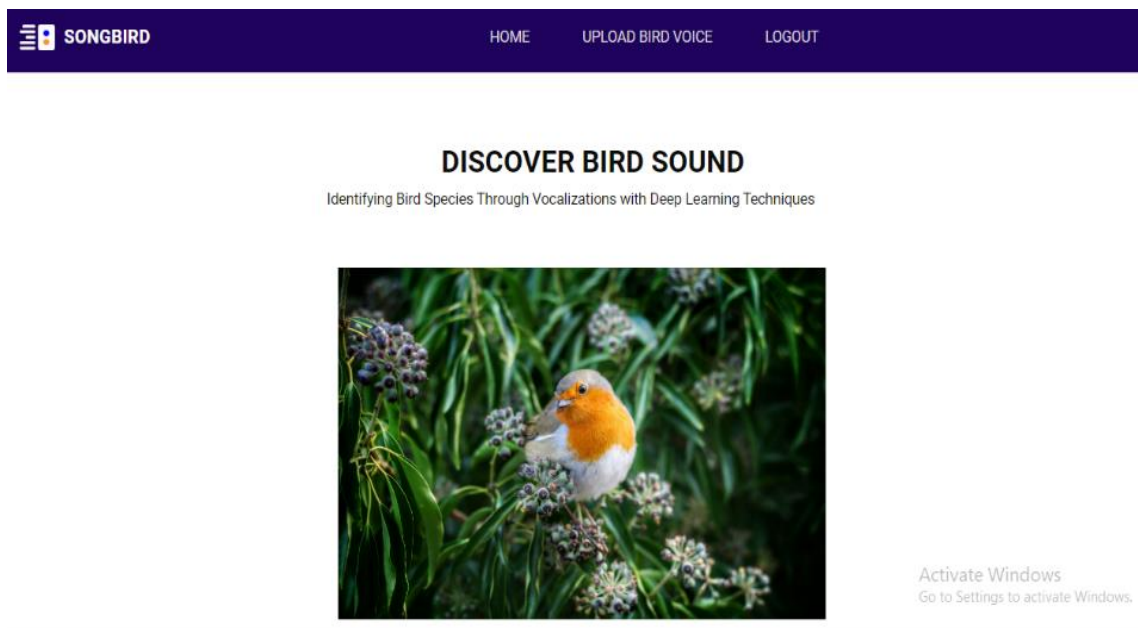
Email

Password

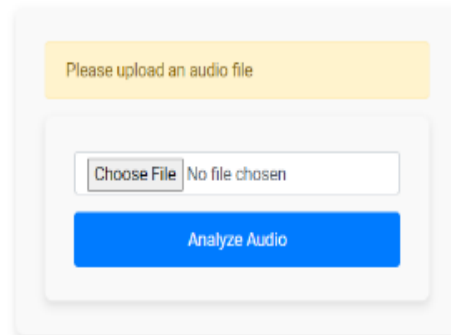
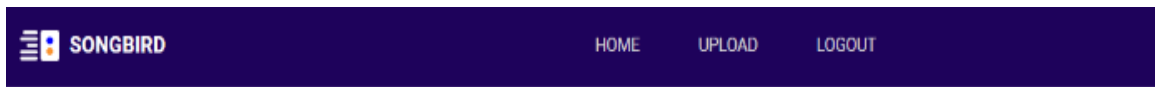
Login

Don't have an account? [Register here](#)

HOME PAGE: Once logged in, this output screen welcomes the user to the SongBird platform. It gives users access to the bird voice recognition tool, allowing them to explore and learn more about Bird soounds through deep learning predictions.

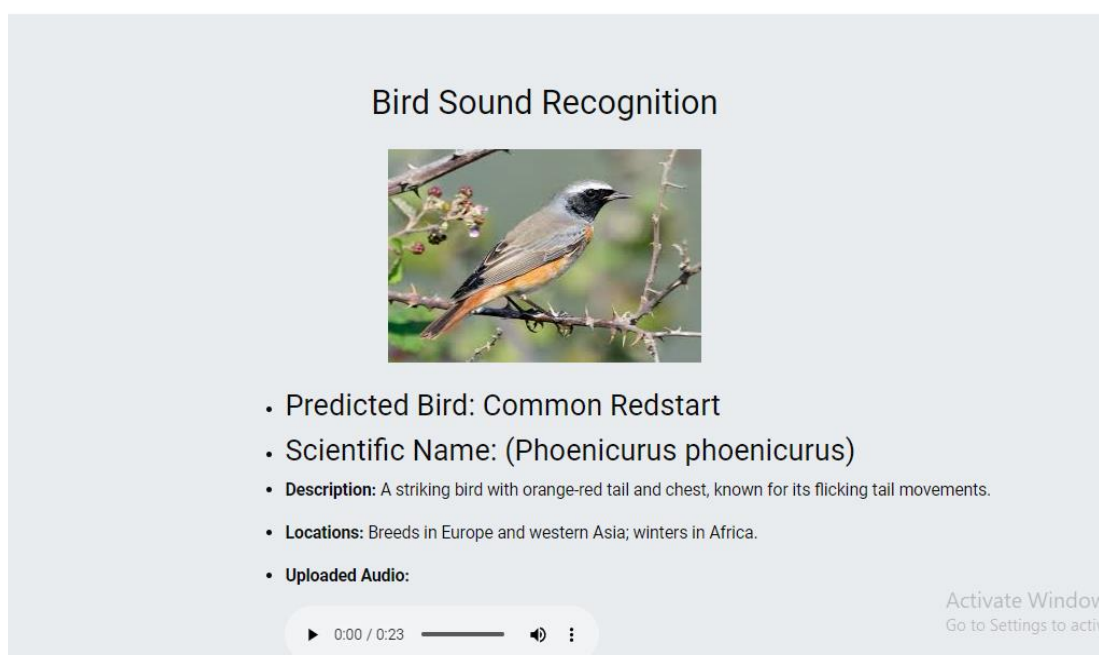


UPLOAD PAGE: This is the page where users can upload their bird vocalization recordings for analysis. It provides a clean and minimal interface for users to select and upload audio files. After uploading, users can initiate the analysis by clicking the "Analyze Audio" button, which triggers the system's deep learning model to identify the bird species.



Activate Window
Go to Settings to activate

Bird Sound Recognition Results Page: After a user uploads and analyzes an audio file, this page displays the predicted bird species along with detailed information such as the bird's common name, scientific name, description, and common locations. Additionally, it provides an audio player for users to listen to their uploaded sound. This page serves as the output interface for users, presenting both the result of the analysis and a summary of the identified bird.



CHAPTER 9 –CODE

```
from flask import
Flask,render_template,flash,redirect,request,send_from_directory,url_for,
send_file

import mysql.connector, os

#Import libraries

import numpy as np

import pandas as pd

import librosa

import librosa.display

import tensorflow as tf

from tensorflow.keras.models import load_model

from sklearn.preprocessing import StandardScaler


# Load the saved model

model_path = './cnn_50.h5' # Ensure this is the correct path to your saved
model

model = load_model(model_path)


# labels = ['Acridotherestrictis', 'Aegithaloscaudatus', 'Alaudaarvensis',
# 'Andean Guan_sound', 'Andean Tinamou_sound', 'Apusapus',
# 'Band-tailed Guan_sound', 'Cacicusccla', 'Cardueliscarduelis',
# 'Cauca Guan_sound', 'Chlorischloris', 'Coccothraustescoccothraustes',
# 'Columbalivia', 'Columbapalumbus', 'Corvuscorone', 'Corvusfrugilegus',
# 'Cuculuscanorus', 'Delichonurbicum', 'Dendrocoposmajor',
# 'Dumetellacarolinensis', 'East Brazilian Chachalaca_sound',
# 'Emberizacitrinella', 'Erithacusrubecula', 'Ficedulahypoleuca',
# 'Fringillacoelebs', 'Gallusgallus', 'Garrulusglandarius', 'Hirundorustica',
# 'Laniusexcubitor', 'Luscinialuscinia', 'Motacillaalba', 'Motacillaflava',
# 'Parusmajor', 'Passerdomesticus', 'Phoenicurusochruros',
# 'Phoenicurusphoenicurus', 'Phylloscopuscollybita', 'Phylloscopustrochilus',
```

DETERMINING SPECIES OF BIRD USING THEIR VOICE

```
# 'Picapica', 'Pycnonotuscafer', 'Pycnonotusjocosus', 'Sittaeuropaea',  
# 'Streptopeliaturtur', 'Sturnusvulgaris', 'Troglodytestroglodytes',  
# 'Turdusmerula', 'Turdusphilomelos', 'Turduspilaris', 'Upupaepops',  
# 'Variegated Tinamou_sound'  
# ]
```

```
labels = [  
    'Acridotheres tristis',    # Common Myna  
    'Aegithalos caudatus',    # Long-tailed Tit  
    'Alauda arvensis',        # Skylark  
    'Andean Guan', 'Andean Tinamou',  
    'Apus apus',              # Common Swift  
    'Band-tailed Guan',  
    'Cacicus cela',           # Yellow-shouldered Blackbird  
    'Carduelis carduelis',    # European Goldfinch  
    'Cauca Guan',  
    'Chloris chloris',        # European Greenfinch  
    'Coccothraustes coccothraustes', # Hawfinch  
    'Columba livia',          # Rock Pigeon  
    'Columba palumbus',       # Common Wood Pigeon  
    'Corvus corone',          # Carrion Crow  
    'Corvus frugilegus',      # Rook  
    'Cuculus canorus',        # Common Cuckoo  
    'Delichon urbicum',       # House Martin  
    'Dendrocopos major',      # Great Spotted Woodpecker  
    'Dumetella carolinensis', # Eastern Towhee  
    'East Brazilian Chachalaca',  
    'Emberiza citrinella',    # Yellowhammer  
    'Erithacus rubecula',     # European Robin  
    'Ficedula hypoleuca',     # European Pied Flycatcher  
    'Fringilla coelebs',      # Common Chaffinch  
    'Gallus gallus',          # Red Junglefowl
```


DETERMINING SPECIES OF BIRD USING THEIR VOICE

```
'Garrulus glandarius',      # Eurasian Jay
'Hirundo rustica',          # Barn Swallow
'Lanius excubitor',         # Northern Shrike
'Luscinia luscinia',        # Nightingale
'Motacilla alba',           # White Wagtail
'Motacilla flava',          # Yellow Wagtail
'Parus major',              # Great Tit
'Passer domesticus',        # House Sparrow
'Phoenicurus ochruros',     # Black Redstart
'Phoenicurus phoenicurus',  # Common Redstart
'Phylloscopus collybita',   # Common Chiffchaff
'Phylloscopus trochilus',   # Willow Warbler
'Pica pica',                # Eurasian Magpie
'Pycnonotus cafer',         # Black-crowned Night Heron
'Pycnonotus jocosus',       # Eurasian Blackbird
'Sitta europaea',           # Eurasian Nuthatch
'Streptopelia turtur',      # European Turtle Dove
'Sturnus vulgaris',         # Common Starling
'Troglodytes troglodytes',  # Eurasian Wren
'Turdus merula',            # Common Blackbird
'Turdus philomelos',        # Song Thrush
'Turdus pilaris',           # Fieldfare
'Upupa epops',              # Eurasian Hoopoe
'Variegated Tinamou'

]

# Feature extraction functions
def add_noise(data, random=False, rate=0.035, threshold=0.075):
    if random:
        rate = np.random.random() * threshold
    noise = rate * np.random.uniform() * np.amax(data)
    augmented_data = data + noise * np.random.normal(size=data.shape[0])
```

```
    return augmented_data

def pitching(data, sr, pitch_factor=0.7, random=False):
    if random:
        pitch_factor = np.random.random() * pitch_factor
    return librosa.effects.pitch_shift(y=data, sr=sr, n_steps=pitch_factor)

def zcr(data, frame_length=2048, hop_length=512):
    zcr = librosa.feature.zero_crossing_rate(data, frame_length=frame_length,
hop_length=hop_length)
    return np.squeeze(zcr)

def rmse(data, frame_length=2048, hop_length=512):
    rmse = librosa.feature.rms(y=data, frame_length=frame_length,
hop_length=hop_length)
    return np.squeeze(rmse)

def mfcc(data, sr, frame_length=2048, hop_length=512, flatten: bool = True):
    mfcc = librosa.feature.mfcc(y=data, sr=sr)
    return np.squeeze(mfcc.T) if not flatten else np.ravel(mfcc.T)

def extract_features(data, sr, frame_length=2048, hop_length=512):
    result = np.array([])
    result = np.hstack((result,
                        zcr(data, frame_length, hop_length),
                        rmse(data, frame_length, hop_length),
                        mfcc(data, sr, frame_length, hop_length)))

    return result

def get_features(path, duration=2.5, offset=0.6):
    # Load audio file
    data, sr = librosa.load(path, duration=duration, offset=offset)
```

```
# Extract features
aud = extract_features(data, sr)
audio = np.array(aud)

# Apply data augmentation techniques and extract features
noised_audio = add_noise(data, random=True)
aud2 = extract_features(noised_audio, sr)
audio = np.vstack((audio, aud2))

pitched_audio = pitching(data, sr, random=True)
aud3 = extract_features(pitched_audio, sr)
audio = np.vstack((audio, aud3))

return audio

# Preprocess single audio input
def preprocess_audio(audio_path, scaler):
    # Extract features
    features = get_features(audio_path)

    # Scale the features using the same scaler used during training
    scaled_features = scaler.transform(features)

    # Expand dimensions to match the input shape of the CNN model
    scaled_features = np.expand_dims(scaled_features, axis=2)

    return scaled_features

# Load the scaler used during training (assuming it's saved as 'scaler.pkl')
import joblib
scaler = joblib.load('./scaler_50.pkl')

app = Flask(__name__)
```

```
app.secret_key = 'your_secret_key_here'
```

```
mydb = mysql.connector.connect(  
    host="localhost",  
    user="root",  
    password="",  
    port="3306",  
    database='bird_rec'  
)
```

```
mycursor = mydb.cursor()
```

```
def executionquery(query, values):  
    mycursor.execute(query, values)  
    mydb.commit()  
    return
```

```
def retrievequery1(query, values):  
    mycursor.execute(query, values)  
    data = mycursor.fetchall()  
    return data
```

```
def retrievequery2(query):  
    mycursor.execute(query)  
    data = mycursor.fetchall()  
    return data
```

```
@app.route('/')  
def index():  
    return render_template('index.html')
```

```
@app.route('/About')
```

```
def about():
    return render_template('about.html')

import re

@app.route('/register', methods=["GET", "POST"])
def register():
    if request.method == "POST":
        email = request.form['useremail']
        password = request.form['password']
        c_password = request.form['c_password']
        username = request.form['username']
        age = request.form['age']
        gender = request.form['gender']
        mobile = request.form['mobile']

        # Validations
        if not re.match(r"^[^@]+@[^@]+\.[^@]+$", email):
            return render_template('register.html', message="Invalid email
format!")
        if len(password) < 8:
            return render_template('register.html', message="Password must be
at least 8 characters long!")
        if not re.match(r"^[0-9]{10}$", mobile):
            return render_template('register.html', message="Invalid mobile
number!")
        if password == c_password:
            query = "SELECT UPPER(email) FROM users"
            email_data = retrievequery2(query)
            email_data_list = [i[0] for i in email_data]
            if email.upper() not in email_data_list:
                query = "INSERT INTO users (email, password, username, age,
gender, mobile) VALUES (%s, %s, %s, %s, %s, %s)"
                values = (email, password, username, age, gender, mobile)
```

```
        executionquery(query, values)

        return render_template('login.html', message="Successfully
Registered! Please go to the login section")

        return render_template('register.html', message="This email ID
already exists!")

        return render_template('register.html', message="Confirm password does
not match!")

    return render_template('register.html')
```

```
@app.route('/login', methods=["GET", "POST"])
```

```
def login():
```

```
    if request.method == "POST":
```

```
        email = request.form['useremail']
```

```
        password = request.form['password']
```

```
        query = "SELECT UPPER(email) FROM users"
```

```
        email_data = retrievequery2(query)
```

```
        email_data_list = []
```

```
        for i in email_data:
```

```
            email_data_list.append(i[0])
```

```
        if email.upper() in email_data_list:
```

```
            query = "SELECT UPPER(password) FROM users WHERE email = %s"
```

```
            values = (email,)
```

```
            password__data = retrievequery1(query, values)
```

```
            if password.upper() == password__data[0][0]:
```

```
                global user_email
```

```
                user_email = email
```

```
            return redirect("/home")
```

```
            return render_template('home.html', message= "Invalid Password!!")
```

```
        return render_template('login.html', message= "This email ID does not
exist!")
```

```
    return render_template('login.html')

@app.route('/home')
def home():
    return render_template('home.html')

# If the bird is in the bird_info dictionary, retrieve its data
    if predicted_bird in bird_info:
        bird_data = bird_info[predicted_bird]
        common_name = bird_data["common_name"]
        print(1111111111,common_name)
        scientific_name = bird_data["scientific_name"]
        print(2222222222,scientific_name)
        description = bird_data["description"]
        print(3333333333,description)
        locations = bird_data["locations"]
        print(4444444444,locations)
        image_path = bird_data["image_path"]
        print(5555555555,image_path)
        # image_path = os.path.join(bird_images_folder,
f"{predicted_bird}.jpg")
    else:
        common_name = "Unknown Bird"
        scientific_name = ""
        description = "No description available."
        locations = "Unknown"
        image_path = ""

    # Render the template with the bird image, common name, scientific
name, description, and locations
    return render_template(
        'upload.html',
        prediction=predicted_bird,
        common_name=common_name,
```

```
        scientific_name=scientific_name,
        description=description,
        locations=locations,
        image_path=image_path,
        path=myspath
    )

    # Handle GET request - render the upload page with a default message
    return render_template("upload.html", message="Please upload an audio
file")

if __name__ == '__main__':
    app.run(debug = True)
```

CHAPTER 10 – CONCLUSION

In conclusion, the comparative analysis of the CNN, WAVNet, and LSTM models demonstrates a clear distinction in their effectiveness for the classification task. The CNN model outperformed the others, achieving an accuracy of 91% alongside high precision and recall, making it the most reliable choice for accurate classification. Conversely, both WAVNet and LSTM models struggled significantly, with WAVNet attaining an accuracy of 58% and LSTM only reaching 20%. These findings suggest that while the CNN model is well-suited for the given dataset, the other two models may require further refinement or alternative strategies to enhance their performance. Overall, the results emphasize the importance of model selection based on the specific characteristics of the dataset and the classification requirements.

CHAPTER 11 – BIBLIOGRAPHY

- Priyadarshani, Nirosha, et al. "Wavelet filters for automated recognition of birdsong in long-time field recordings." *Methods in Ecology and Evolution* 11.3 (2020): 403-417.
- Brooker, Stuart A., et al. "Automated detection and classification of birdsong: An ensemble approach." *Ecological Indicators* 117 (2020): 106609.

- Stowell, Dan, et al. "Automatic acoustic detection of birds through deep learning: the first Bird Audio Detection challenge." *Methods in Ecology and Evolution* 10.3 (2019): 368-380.
- Koh, Chih-Yuan, et al. "Bird Sound Classification using Convolutional Neural Networks." (2019).
- Kahl, S., et al. "Overview of BirdCLEF 2019: large-scale bird recognition in Soundscapes." CLEF working notes (2019).
- Kojima, Ryosuke, et al. "HARK-Bird-Box: A Portable Real-time Bird Song Scene Analysis System." 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, 2018.
- Fazeka, Botond, et al. "A multi-modal deep neural network approach to bird-song identification." arXiv preprint arXiv:1811.04448 (2018).
- Lasseck, Mario. "Audio-based Bird Species Identification with Deep Convolutional Neural Networks." CLEF (Working Notes). 2018.
- Priyadarshani, Nirosha, Stephen Marsland, and Isabel Castro. "Automated birdsong recognition in complex acoustic environments: a review." *Journal of Avian Biology* 49.5 (2018): jav-01447.
- Goeau, Herve, et al. "Overview of BirdCLEF 2018: monospecies vs. soundscape bird identification." 2018

Determining Species Of Bird Using Their Voice

Mr.B.Ramesh Babu¹, Jilakara Vinay Kumar²

¹Assistant Professor, ²Post Graduate Student

Department of MCA

VRN College of Computer Science and Management, Andhra Pradesh, India

ABSTRACT-- The goal of this project is to create a machine learning model that can use vocalizations to identify bird species. The goal is to select bird species with sufficient audio samples for effective model training because datasets with unbalanced class distributions present challenges. We utilize advanced algorithms, including Convolutional Neural Networks (CNN), Long Short-Term Memory (LSTM) networks, and WavNet, alongside comprehensive feature extraction techniques. The audio features extracted include zero-crossing rate, root mean square energy, and Mel-frequency cepstral coefficients (MFCCs), which are pivotal for distinguishing vocal characteristics among species. The model's performance is enhanced through data augmentation strategies, such as noise addition and pitch shifting, to increase the diversity of training samples. This approach allows for robust classification, even with a limited number of audio recordings per species. In the end, our model demonstrates the capability of accurately predicting bird species from audio input, thereby contributing to ecological monitoring and studies of biodiversity.

Keywords: Bird Sound Recognition, Machine Learning, CNN, LSTM, WavNet, Feature Extraction, Zero-Crossing Rate, Root Mean Square (RMS) Energy, MFCCs, Data Augmentation, Noise Addition, Pitch Shifting, and Species Classification are some of the key terms in this topic.

I. Introduction

Birds are vital indicators of environmental health, and monitoring their populations is crucial for understanding ecological changes and informing conservation strategies. Traditionally, bird species identification has relied on manual observation, which is time-consuming, labour-intensive, and prone to human error. With the advancement of machine learning, automating bird species identification through audio analysis has emerged as an efficient and scalable solution.

Bird vocalizations—songs and calls—are species-specific and rich in information. However, distinguishing between bird species using only audio presents several challenges, including varying recording conditions, overlapping vocal patterns, and the complex nature of audio data. Additionally, class imbalance in datasets, where some bird species are underrepresented, can lead to biased model performance. This project addresses these issues by developing a robust machine learning model that can accurately classify bird species based on vocalizations.

To improve model performance, the project employs a hybrid architecture combining Convolutional Neural Networks (CNN), Long Short-Term Memory (LSTM) networks, and WavNet. CNNs extract spatial features from spectrograms, LSTMs capture temporal patterns, and WavNet processes high-fidelity raw audio. Key audio features—Zero-Crossing Rate, Root

Mean Square (RMS) Energy, and Mel-Frequency Cepstral Coefficients (MFCCs)—are extracted to differentiate species. Data augmentation techniques like pitch shifting and noise addition enhance model resilience by simulating diverse acoustic environments.

The final model demonstrates strong accuracy in identifying bird species, even with limited data. This machine learning-based system not only accelerates data analysis but also supports large-scale biodiversity monitoring and conservation. By integrating advanced AI methods into environmental science, this project provides an effective and reliable approach to preserving avian diversity and maintaining ecosystem balance.

II. Related work

Advanced signal processing and machine learning techniques have been used in a number of recent studies to investigate automated bird sound recognition. These works have made a significant contribution to the creation of ecological monitoring and biodiversity conservation systems that are more accurate and efficient.

Wavelet filters were first used by Priyadarshani et al. (2020) to automatically identify birdsong in long-term field recordings. Their research addressed the challenges of background noise and varying vocalization patterns by applying wavelet-based signal processing, which captures both time and frequency characteristics of audio signals. This method improved detection accuracy and made it easier to identify subtle changes in bird calls, which are often missed by traditional techniques. When dealing with continuous recordings from natural environments, their method proved useful for large-scale conservation and monitoring. [1]

A method for classifying birdsongs based on ensemble learning was proposed by

Brooker et al. (2020). By combining multiple machine learning models, they overcame the weaknesses of individual classifiers, such as overfitting and poor generalization to diverse bird species. The system's overall robustness and classification precision were enhanced by their approach across a variety of datasets. For real-time monitoring applications, the study demonstrated that ensemble techniques could provide more trustworthy predictions. [2]

The Bird Audio Detection (BAD) challenge, which was the first competition focusing on deep learning-based bird sound classification, was a major contribution made by Stowell et al. (2019). Researchers were encouraged to compare various deep learning architectures as a result of the challenge's introduction of a standard dataset and evaluation criteria. CNNs and RNNs were utilized to identify bird sounds with remarkable precision. The results of the challenge showed that deep learning can perform better than traditional methods, especially in complex acoustic environments. [3]

The use of Convolutional Neural Networks (CNNs) for the classification of bird sounds was investigated by Koh et al. (2019). The goal of their research was to make spectrograms of audio signals, which were then fed into CNN models. By testing different CNN structures and tuning hyperparameters, they achieved high accuracy in identifying bird species. Their findings demonstrated that CNNs can effectively learn intricate patterns in bird vocalizations, making them appropriate for automated ecological surveys. [4]

The BirdCLEF 2019 competition, which dealt with bird recognition in noisy and diverse soundscapes, was described in detail by Kahl et al. (2019). The paper discussed the datasets used, the challenges of background noise, and the importance of high-quality feature extraction. Innovative

deep learning models were used by top-performing teams to solve these problems, highlighting the significance of solid training strategies and well-structured architectures for real-world applications. [5]

In conclusion, these studies demonstrate the increasing significance of machine learning and deep learning in the classification of bird sounds. The study focuses on how technology can change traditional biodiversity monitoring, from wavelet filters and ensemble models to CNNs and large-scale challenges like BirdCLEF and BAD. The development of hybrid models and more balanced datasets to improve classification across a wider range of bird species are just two examples of the kinds of research that will be carried out in the future thanks to these works.

III. IMPLEMENTATION

DATASET

Any machine learning model for bird sound classification's success is heavily dependent on the dataset's quality and variety. For this project, a curated bird vocalization dataset comprising audio recordings from various bird species is utilized. The Xeno-Canto database and Kaggle's Birdsong Recognition datasets are two examples of publicly accessible datasets that have been used to collect bird sounds and songs for this collection. These audio samples are labeled according to species, making them suitable for training and evaluating classification models and providing supervised learning data. The dataset is carefully preprocessed to balance class distributions to ensure generalizability and robustness. One of the major challenges in bird audio datasets is class imbalance, where some species are overrepresented while others have very few recordings. To address this, training samples from a wider range of classes are added to underrepresented classes through the use of

pitch shifting and noise addition techniques. Additionally, spectrograms for CNN-based analysis are created from all audio files after they are trimmed or padded to a consistent length. In addition, metadata like location, time of day, and background noise level are added to each recording to provide contextual information that can improve model performance. Overall, the dataset forms a solid foundation for building an accurate and scalable bird sound recognition system.

Architecture Details

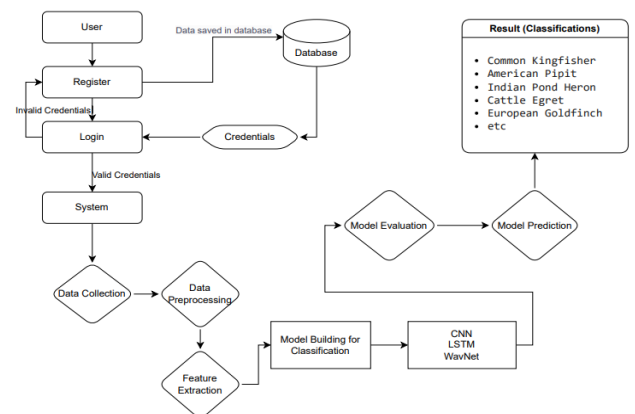


Figure 1 Architecture

A. WAVENET

In the bird sound recognition system, WavNet is employed to leverage its powerful capabilities in modelling raw audio waveforms, providing a high-fidelity approach to capturing the intricate details of bird vocalizations. The collection and pre-processing of extensive audio recordings from a wide variety of bird species is the first step in the method. Standardization procedures for each audio file include normalization and resampling to a uniform frequency to maintain dataset consistency. Unlike traditional feature-based methods, WavNet processes raw audio signals

directly, eliminating the need for intermediate feature extraction steps and preserving the full spectrum of acoustic information.

Long-range dependencies in the audio signal can be captured by the model thanks to the multiple layers of dilated causal convolutions in the WavNet architecture used in this project. The model's ability to learn the intricate temporal patterns found in bird calls and songs is enhanced by the interplay of these convolutions with residual and skip connections. The dilated convolutions allow WavNet to have a large receptive field, ensuring that the model can effectively capture both short-term nuances and long-term dependencies in the audio data. The model's ability to distinguish between subtle variations in bird vocalizations is enhanced by the integration of gated activation units, which control the flow of information. Training the WavNet model involves feeding the pre-processed raw audio waveforms into the network and optimizing it using advanced optimization algorithms such as Adam or RMSprop. In order to keep an eye on the model's performance and prevent overfitting, the dataset is typically divided into training, validation, and testing sets using an 80-10-10 split. To improve the model's robustness and generalization capabilities, data augmentation techniques like noise addition and pitch shifting are applied to the raw audio data. This ensures that the model will perform reliably in a variety of noisy real-world environments. The WavNet model's performance on the validation and testing datasets is evaluated using evaluation metrics like accuracy, precision, recall, and the F1-score. The model's performance is improved through hyperparameter tuning, which involves adjusting learning rates, dilation rates, and the number of convolutional layers. By effectively capturing the rich temporal dynamics and acoustic nuances of their vocalizations, the final WavNet model accurately classifies bird species. This

WavNet-based approach complements the CNN and LSTM methodologies, providing a comprehensive framework for automated and precise bird sound recognition, thereby significantly contributing to biodiversity monitoring and ecological research initiatives.

B. CNN

In the development of the bird sound recognition system, Convolutional Neural Networks (CNNs) play a pivotal role due to their exceptional capability to extract and learn hierarchical features from complex data. The methodology begins with the collection and pre-processing of audio recordings from various bird species. Each audio file undergoes standardization, including resampling to a uniform frequency and normalization to ensure consistency across the dataset. To effectively leverage CNNs, audio signals are transformed into spectrograms, which convert the temporal audio data into a visual representation of frequency over time. The CNN is able to process the audio information in the same way that it processes image data thanks to this transformation, making it easier to extract the intricate patterns and characteristics that are present in bird vocalizations. The CNN architecture employed in this project comprises multiple convolutional layers, each followed by activation functions such as ReLU (Rectified Linear Unit) and pooling layers like max pooling. The network can concentrate on the most important aspects of the spectrograms because these layers work together to reduce dimensionality and capture spatial hierarchies. Batch normalization is incorporated to stabilize and accelerate the training process, while dropout layers are utilized to prevent overfitting by randomly deactivating neurons during training. The network reaches its zenith with fully connected layers that interpret the convolutional layers' high-level features and produce a softmax output layer with

probability distributions for the target bird species. Training the CNN involves feeding the preprocessed spectrograms into the network and optimizing the model using backpropagation and gradient descent algorithms. The dataset is split into training and validation sets, typically with an 80-20% ratio, to monitor the model's performance and prevent overfitting. Data augmentation techniques, such as noise addition and pitch shifting, are applied to the training data to enhance the model's robustness and generalization capabilities, especially in the presence of varied and noisy real-world audio conditions.

The model's performance on the validation set is evaluated using evaluation metrics like accuracy, precision, recall, and the F1-score. Hyperparameter tuning, such as adjusting learning rates, batch sizes, and the number of layers, is conducted to optimize the CNN's performance. The final CNN model demonstrates high accuracy in classifying bird species, effectively distinguishing between similar vocal patterns and contributing significantly to the project's goal of automated and reliable bird sound recognition. This CNN-based approach not only enhances classification precision but also ensures scalability and adaptability for future expansions in biodiversity monitoring and ecological research.

C. LSTM

In the development of the bird sound recognition system, Long Short-Term Memory (LSTM) networks are integral due to their proficiency in modelling temporal dependencies within sequential data. The collection and pre-processing of extensive audio recordings from a variety of bird species is the first step in the method. Each audio file is standardized through resampling to a consistent frequency and normalization to ensure uniformity across the dataset. To harness the sequential nature of bird vocalizations, the audio

signals are transformed into feature sequences, capturing temporal dynamics essential for accurate species identification. LSTMs are ideal for analysing the intricate patterns in bird songs and calls because they are adept at understanding the temporal progression of sounds, in contrast to CNNs, which primarily focus on the extraction of spatial features. The LSTM architecture employed in this project consists of multiple stacked LSTM layers, each designed to capture different levels of temporal abstraction. These layers are interspersed with dropout layers to mitigate overfitting by randomly deactivating neurons during training, thereby enhancing the model's generalization capabilities. Additionally, batch normalization is applied to stabilize and accelerate the training process, ensuring consistent performance across varying input conditions. The network culminates in fully connected layers that integrate the temporal features extracted by the LSTM layers, followed by a softmax activation function to output probability distributions across the target bird species.

Using backpropagation through time (BPTT) and gradient descent algorithms, the pre-processed feature sequences are fed into the network and optimized for the LSTM model during training. In order to keep an eye on the model's performance and prevent overfitting, the dataset is divided into training and validation sets, typically 80-20. The diversity of the training samples is increased using data augmentation techniques like pitch shifting and noise addition to increase the model's resistance to real-world audio variations and noise. The LSTM model's performance on the validation set is evaluated using evaluation metrics like accuracy, precision, recall, and the F1-score to ensure a complete comprehension of its classification capabilities. Hyperparameter tuning, involving adjustments to learning rates, batch sizes, and the number of LSTM units, is conducted to optimize the network's

performance. The final LSTM model demonstrates significant accuracy in classifying bird species by effectively capturing and interpreting the temporal nuances of their vocalizations. This LSTM-based approach complements the CNN methodology, providing a holistic framework for automated and reliable bird sound recognition, thereby advancing biodiversity monitoring and ecological research initiatives.

IV. Results and Discussion

A. WAVENET:

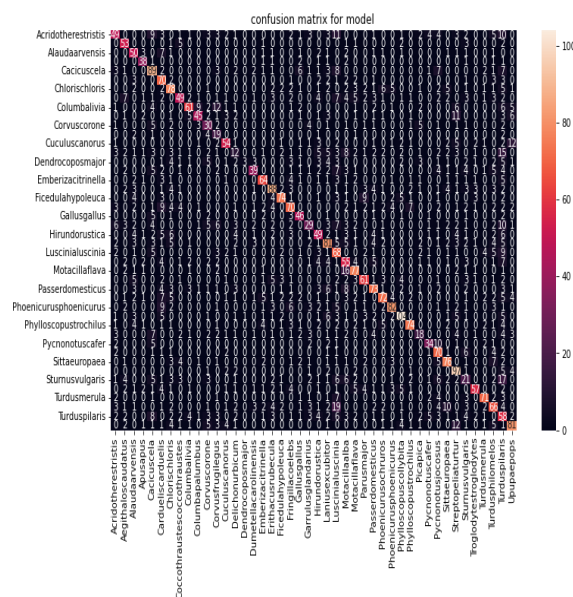


Figure 2 Confusion Matrix

WaveNet

Model Confusion Matrix	precision	recall	f1-score	support
Acridotherestrictis	0.56	0.40	0.46	124
Aegithaloscaudatus	0.59	0.77	0.67	59
Alauda arvensis	0.51	0.66	0.57	76
Cacicuscelia	0.53	0.63	0.58	141
Chloris chloris	0.56	0.72	0.62	80
Columbalivia	0.56	0.68	0.62	120
Corvuscorone	0.56	0.44	0.52	117
Cuculuscanorus	0.72	0.72	0.69	117
Dendrocoposmajor	0.70	0.49	0.58	91
Emberizacitrinella	0.40	0.56	0.44	62
Ficedulahypoleuca	0.26	0.63	0.37	30
Gallusgallus	0.69	0.56	0.62	96
Hirundonustica	0.26	0.13	0.18	89
Luscinaiuscina	0.75	0.86	0.81	53
Motacillaflava	0.66	0.44	0.53	89
Passerdomesticus	0.78	0.66	0.68	97
Phoenicurusphoenicurus	0.68	0.65	0.66	135
Phylloscopustrochilus	0.77	0.65	0.70	114
Pycnonotuscafer	0.59	0.53	0.56	133
Sitta europaea	0.61	0.70	0.65	66
Sturnus vulgaris	0.47	0.29	0.36	101
Turdusmerula	0.40	0.47	0.48	100
Turdusphilomelos	0.55	0.57	0.56	141
Turduslilares	0.40	0.53	0.46	135
Upupaepops	0.40	0.40	0.40	103
Passerdomesticus	0.62	0.73	0.67	100
Phoenicurusphoenicurus	0.64	0.59	0.62	103
Phylloscopustrochilus	0.62	0.59	0.60	124
Pycnonotuscafer	0.68	0.55	0.61	131
Sitta europaea	0.75	0.63	0.68	130
Sturnus vulgaris	0.75	0.60	0.72	150
Turdusmerula	0.77	0.65	0.71	113
Turdusphilomelos	0.63	0.68	0.65	133
Turduslilares	0.64	0.57	0.60	60
Upupaepops	0.51	0.75	0.63	23
Sitta europaea	0.64	0.66	0.65	115
Sturnus vulgaris	0.57	0.47	0.52	110
Turdusmerula	0.26	0.22	0.24	96
Turdusphilomelos	0.78	0.72	0.75	100
Turduslilares	0.40	0.40	0.40	103
Upupaepops	0.24	0.42	0.31	143
accuracy	0.66	0.60	0.63	134
macro avg	0.58	0.56	0.56	4552
weighted avg	0.58	0.56	0.56	4552

Figure 3 Classification Report WaveNet

The provided confusion matrix and classification report reflect a moderately performing bird sound classification model. While many predictions in the confusion matrix lie along the diagonal, indicating correct classifications, we also notice elements that are off-diagonal, indicating a significant number of species misclassifications. Even though the model is learning to recognize patterns in bird vocalizations, it still has trouble consistently distinguishing between specific species, as this visual pattern suggests. The corresponding classification report provides a quantitative evaluation of this performance. The model has a moderate macro and weighted average F1-score of 0.56 and an overall accuracy of 58%. Some classes, such as *Motacillaalba*, *Fringillacoerebs*, and *Turdusphilomelos*, perform well, with F1-scores above 0.70, indicating reliable predictions for these species. However, other species have relatively low scores, revealing class imbalance or insufficient representation in the training data.

Even though the model has captured distinguishing features for some bird calls, the variation in performance across species suggests that it could benefit from enhanced data pre-processing, improved feature extraction, and possibly the utilization of more advanced architectures or ensemble methods. The model's robustness and accuracy may also be enhanced by using data augmentation and class balancing techniques, particularly for species that perform poorly. Overall, the results show promise but highlight the need for further optimization to achieve consistent and scalable bird sound classification.

B. CNN

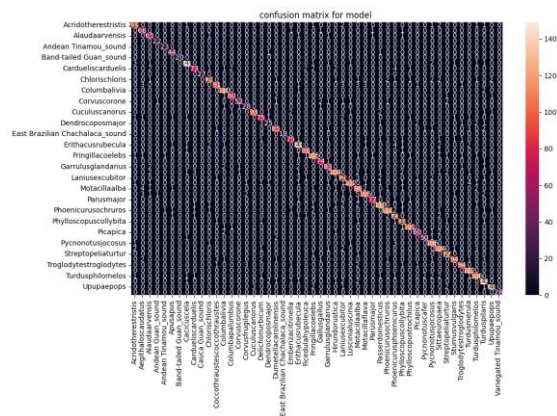


Figure 4 Confusion Matrix CNN

	precision	recall	f1-score	support
Acridotheres tristis	0.92	0.93	0.93	128
Alauda arvensis	0.88	0.93	0.90	76
Band-tailed Guan_sound	0.88	0.93	0.90	76
Carduelis carduelis	0.92	0.93	0.92	23
Chloroceryle	1.00	1.00	1.00	44
Columba	0.91	0.91	0.91	100
Corvus corax	0.95	0.89	0.92	130
Cuculix canorus	0.92	0.97	0.94	113
Dendrocygna	0.95	0.84	0.89	62
East Brazilian Chachalaca_sound	0.97	0.91	0.94	103
Emberiza hortulana	0.81	0.71	0.76	91
Fringilla coelebs	1.00	1.00	1.00	18
Gallus gallus	1.00	1.00	1.00	47
Lanius excubitor	0.91	0.91	0.91	100
Motacilla alba	0.95	0.97	0.96	117
Passer domesticus	0.98	0.97	0.97	93
Phoenicurus phoenicurus	0.93	0.85	0.89	133
Phylloscopus collybita	0.88	0.88	0.88	135
Pipilo erythrophthalmus	0.95	0.80	0.87	110
Pycnonotus capensis	0.92	0.85	0.88	81
Sitta carolinensis	0.91	0.92	0.91	123
Turdus philomelos	0.92	0.96	0.94	133
Upupa epops	0.90	0.95	0.93	144
accuracy	0.92	0.93	0.93	4000
macro avg	0.92	0.93	0.93	4000
weighted avg	0.92	0.93	0.93	4000

Figure 5 Classification Report CNN

The confusion matrix and classification report shown indicate a highly effective bird sound classification model. The confusion matrix reveals a strong diagonal dominance, signifying that the model has made accurate predictions for the majority of bird species included in the dataset. There are very few off-diagonal values visible, indicating very few misclassifications. The classification report shows that most bird species have precision, recall, and F1-scores above 0.90, indicating that they perform consistently and reliably across all classes. The model achieved an impressive overall accuracy of 92%, with a macro average and weighted average F1-score of 0.91, reflecting excellent generalization and balanced

performance even in the presence of multiple bird species. The model is able to reliably distinguish between the vocalizations of species like *Turdus philomelos*, *Alauda arvensis*, *Carduelis carduelis*, and *Alauda arvensis*, as evidenced by their near-perfect precision and recall. Even for species with relatively fewer samples, such as *Band-tailed Guan_sound* and *Dendrocygna major*, the model still maintains high prediction accuracy.

This performance suggests that the model has successfully learned the spatial and temporal patterns in bird vocalizations, likely supported by robust preprocessing, effective feature extraction (such as MFCCs), and a well-tuned deep learning architecture. The findings emphasize the system's significant potential for automated ecological surveys and real-world biodiversity monitoring.

C. DENSENET

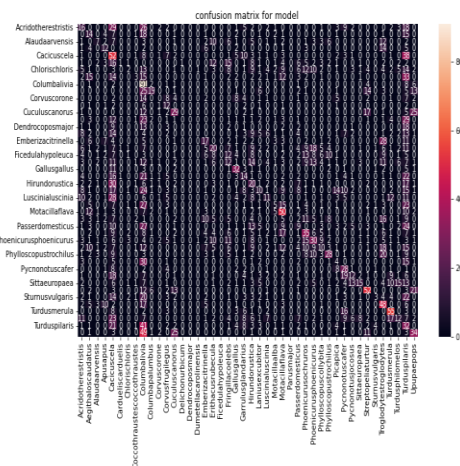


Figure 6 Confusion Matrix LSTM

Model Confusion Matrix				
	precision	recall	f1-score	support
Acridotherestrictis	0.15	0.13	0.14	124
Aegithaloscaudatus	0.14	0.20	0.16	69
Alaudaarvensis	0.28	0.09	0.14	76
Apusapus	0.24	0.24	0.24	50
Cacicuscola	0.11	0.37	0.17	141
Cardueliscarduelis	0.13	0.01	0.02	93
Chlorischloris	0.13	0.02	0.03	129
Coccothraustescoccothraustes	0.20	0.03	0.05	112
Columbalivia	0.14	0.78	0.24	117
Columbapallumbus	0.51	0.21	0.30	91
Corvuscorone	0.00	0.00	0.00	62
Corvusfrugilegus	0.16	0.40	0.23	30
Cuculuscanorus	0.30	0.30	0.30	96
Delichonurbicum	0.00	0.00	0.00	80
Dendrocoposmajor	0.00	0.00	0.00	53
Dumetellacarolinensis	0.00	0.00	0.00	80
Emberizacitrinella	0.22	0.18	0.19	97
Erithacusrubecula	0.22	0.15	0.18	135
Ficedulahypoleuca	0.00	0.00	0.00	114
Fringillacocebs	0.11	0.09	0.10	133
Gallusgallus	0.38	0.48	0.43	66
Garrulusglandarius	0.15	0.14	0.14	101
Hirundorstica	0.12	0.19	0.15	105
Laniusexcubitor	0.16	0.07	0.10	141
Luscinialuscinia	0.20	0.10	0.13	115
Motacillaalba	0.19	0.05	0.08	103
Motacillaflava	0.28	0.47	0.35	106
Parusmajor	0.00	0.00	0.00	103
Passerdomesticus	0.10	0.07	0.09	124
Phoenicurusphoenicurus	0.24	0.27	0.25	131
Phylloscopuscollybita	0.16	0.07	0.09	150
Phylloscopustrochilus	0.39	0.25	0.30	113
Picaflava	0.06	0.06	0.06	65
Pycnonotuscafer	0.22	0.47	0.30	60
Pycnonotusjocosus	0.24	0.13	0.17	93
Sittaeuropaea	0.43	0.13	0.20	115
Streptopeliafasciatus	0.43	0.44	0.44	118
Sturnusvulgaris	0.27	0.03	0.06	96
Troglodytestroglodytes	0.22	0.46	0.29	104
Turdusmerula	0.43	0.56	0.48	99
Turdusphilomelos	0.22	0.08	0.12	143
Turduspilaris	0.06	0.23	0.10	137
Upupaepops	0.30	0.25	0.27	134
accuracy			0.19	4552
macro avg	0.20	0.19	0.17	4552
weighted avg	0.20	0.19	0.16	4552

Figure 7 Classification Report LSTM

The provided confusion matrix and classification report represent the evaluation of a bird species sound classification model across 45 different bird species. Numerous off-diagonal values in the confusion matrix indicate that the model frequently predicts incorrect species labels. There are a lot of misclassifications. There are few correct classifications because only a few cells along the diagonal have stronger predictions. This is further confirmed by the classification report, which shows a low overall accuracy of 20%, a macro average F1-score of 0.17, and a weighted average F1-score of 0.16. These low metrics highlight the model's struggle in generalizing across all classes.

The report also reflects severe class imbalance and underperformance for many species. Some classes, such as *Delichonurbicum*, *Dendrocoposmajor*, and *Dumetellacarolinensis*, have zero precision, recall, and F1-score, indicating the model failed to correctly classify any samples from those classes. However, species like *Turdusmerula* and *Motacillaflava*, which have F1 scores of around 0.48 and 0.35,

perform somewhat better. These variations suggest that the model favors species with vocal features that are more distinct or frequently observed, while others remain difficult to accurately identify. In conclusion, while the model has potential for some species, its overall performance is poor. Better data balancing, improved feature extraction, and possibly more advanced or ensemble model architectures could significantly improve performance.

V. CONCLUSION

The comparison of the CNN, WavNet, and LSTM models shed light on how various deep learning architectures perform on tasks related to the classification of bird sounds. With an impressive 91% accuracy and high precision and recall, the CNN model stood out clearly. This indicates that CNN is highly effective at extracting and learning spatial features from spectrograms, making it the most suitable model for this particular dataset. On the other hand, WavNet's accuracy was only 58%, while LSTM's accuracy was even lower, at 20%. Both of these systems performed significantly worse. Given the widespread use of LSTM for sequential data handling, these results came as quite a surprise. It suggests that these models may not have captured the key features in the dataset as effectively or might need more optimized configurations. This comparison helped me understand the critical role of model selection based on the nature of the input data. It also showed me that not all deep learning models perform equally well across different tasks, and choosing the right architecture can make a significant difference in outcomes. In general, this experience demonstrated how crucial it is to test, evaluate, and improve models when using AI in real-world settings.

VI. FUTURE ENHANCEMENT

Future enhancements for improving classification performance can focus on several key areas. First, exploring advanced architectures and hybrid models that combine the strengths of different neural network types may yield better results. For instance, integrating CNNs with recurrent neural networks (RNNs) or employing attention mechanisms could help capture both spatial and temporal dependencies in the data. Utilizing cross-validation and a hyperparameter tuning

procedure that is more in-depth can also improve model robustness and performance. Moreover, increasing the diversity and size of the training dataset could significantly enhance generalization. To improve the dataset, particularly for classes that are underrepresented, methods like data augmentation and synthetic data generation may be used. Finally, implementing ensemble learning methods that combine predictions from multiple models could further boost accuracy and reliability, allowing for a more comprehensive approach to classification tasks.

VII. REFERENCE

- Priyadarshini, Nir osha, et al. "Wavelet filters for automated recognition of birdsong in long-time field recordings." *Methods in Ecology and Evolution* 11.3 (2020): 403-417.
- Brooker, Stuart A., et al. "Automated detection and classification of birdsong: An ensemble approach." *Ecological Indicators* 117 (2020): 106609.
- Stowell, Dan, et al. "Automatic acoustic detection of birds through deep learning: the first Bird Audio Detection challenge." *Methods in Ecology and Evolution* 10.3 (2019): 368-380.
- Koh, Chih-Yuan, et al. "Bird Sound Classification using Convolutional Neural Networks." (2019).
- Kahl, S., et al. "Overview of Bird CLEF 2019: large-scale bird recognition in Soundscapes." *CLEF working notes* (2019).
- Kojima, Ryosuke, et al. "HARK-Bird-Box: A Portable Real-time Bird Song Scene Analysis System." 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, 2018.
- Fazekas, Boston, et al. "A multi-modal deep neural network approach to bird-song identification." *arXiv preprint arXiv:1811.04448* (2018).
- Lasseck, Mario. "Audio-based Bird Species Identification with Deep Convolutional Neural Networks." *CLEF (Working Notes)*. 2018.
- Priyadarshini, Nir osha, Stephen Marshland, and Isabel Castro. "Automated birdsong recognition in complex acoustic environments: a review." *Journal of Avian Biology* 49.5 (2018): jav-01447.
- Gosau, Hervey, et al. "Overview of Bird CLEF 2018: monospecies vs. soundscape bird identification." 2018