**AUDIO CLASSIFICATION USING**

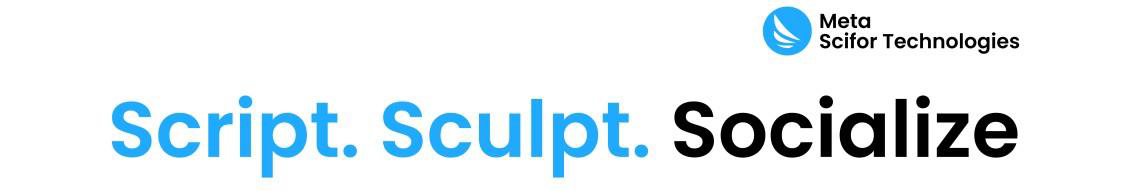
**DEEP LEARNING**

**by**

**D.SAKTHI ESWARAN**

**SUBMITTED TO SCIFOR TECHNOLOGIES**





**UNDER THE GUINDANCE OF**

**UROOJ KHAN**

**ABSTRACT**

This thesis explores the development and training of a neural network model for Audio classification Cat and Dog audio dataset. The process involves importing necessary libraries, loading and preprocessing the data, constructing a neural network architecture, training the model, and evaluating its performance. This cat and dog dataset, consisting of of total 167 cat audios and 112 dog barking audios.Through the utilization of TensorFlow and Keras frameworks, a Sequential model architecture is built, comprising layers such as convolution and pooling with appropriate activation functions. The model is trained using the Adam optimizer and sparse categorical cross-entropy loss function. Throughout the training process, performance metrics including accuracy and loss are monitored and visualized using matplotlib. The trained model achieves a high accuracy on the test dataset, demonstrating its effectiveness in accurately classifying the audios. This research contributes to the understanding and application of deep learning techniques in the field of audio classification, with implications for various real-world applications such as speak recognition ,emotion recognition and medical diagnosis.

**INTRODUCTION**

In an increasingly digitized world, audio data has become abundant, with applications ranging from speech recognition and music analysis to environmental sound monitoring and medical diagnosis. Audio classification, the process of automatically categorizing audio samples into predefined classes, plays a crucial role in extracting meaningful insights and information from this data.

In this project, we aim to develop an audio classification system using Convolutional Neural Networks (CNNs), a powerful deep learning architecture known for its effectiveness in image classification tasks. While CNNs are traditionally used for processing visual data, they can also be adapted to analyze sequential data such as audio waveforms or spectrograms.

Our project will focus on building a CNN-based model capable of classifying audio samples into multiple predefined classes, such as music genres, spoken languages, or environmental sounds. We will explore various aspects of the audio classification pipeline, including data preprocessing, model architecture design, training, evaluation, and deployment.

By leveraging the power of CNNs and deep learning techniques, our project aims to advance the state-of-the-art in audio classification and contribute to the development of intelligent systems capable of understanding and analyzing audio data in diverse real-world scenarios.

**TECHNOLOGY USED**

1. Programming Language: Python
2. Deep Learning Framework: TensorFlow
3. Libraries and functions used:
   * pandas: Data manipulation and analysis library.
   * numpy: Library for numerical computations.
   * tensorflow: Deep learning library for building neural networks.
   * sklearn (scikit-learn): Machine learning library for various algorithms and metrics.
   * matplotlib: Data visualization library for plotting graphs and images.
   * tensorflow\_io (tfio): TensorFlow I/O library for loading and preprocessing audio files.
   * streamlit: Python library for building interactive web applications.
   * Pickle
   * Os
   * tempfile

**DATASET INFORMATION**

The Audio cats and dogs dataset consists in many "wav" files for both the cat and dog classes :

* cat has 164 WAV files to which corresponds 1323 sec of audio.
* dog has 113 WAV files to which corresponds 598 sec of audio.
* Link:[click here](https://www.kaggle.com/datasets/mmoreaux/audio-cats-and-dogs)

**METHODOLOGY**

The methodology employed in this project involves several steps:

**1. Importing Libraries:**

- The code begins by importing necessary libraries like TensorFlow, Keras, Matplotlib, and Streamlit for building and deploying the model, loading and visualizing data, and creating the web application.

**2. Loading and Preprocessing Data:**

- The Audio cats and dogs dataset, a collection of wav audio files of cats and dogs , is loaded using Keras.

- Data preprocessing involves the conversion of audio files into speactrograms , which are fitted into the model to help the model.

**3. Model Architecture:**

- The neural network model is defined using Keras Sequential API.

- The input layer consists of input of shape [772, 257, 1].

- Dense layers with ReLU activation functions are added to introduce non-linearity and learn complex patterns.

- The output layer consists of 1 node, which gives 0 and1 for respective class and utilizes sigmoid activation for classification.

**4. Model Compilation:**

- The model is compiled with appropriate loss function ('BinaryCrossentropy'), optimizer ('Adam'), and evaluation metric ('accuracy') using the `compile()` method.

**5. Model Training:**

- The compiled model is trained on the training data using the `fit()` method.

- The training process involves iterating through epochs (10 in this case), with a validation split of 20% to monitor the model's performance on unseen data.

**6. Model Evaluation:**

- After training, the model's performance is evaluated on the test set using accuracy as the metric.

- The accuracy score is computed using `accuracy`

**7. Visualization:**

Matplotlib is used to visualize training and validation loss, as well as training and validation accuracy, over epochs.

**8. Model Saving:**

- The trained model is saved in the Hierarchical Data Format (HDF5) using Keras's `save()` method.

**9.Streamlit Setup:**

import streamlit as st: Imports the Streamlit library, which is used to create interactive web applications.

import numpy as np: Imports NumPy, which is used for numerical computations.

from tensorflow.keras.models import

load\_model: Imports the function load\_model from Keras to load the pre-trained model.

Load Pre-trained Model:

The pre-trained neural network model for handwritten digit recognition, saved as 'class\_model.h5', is loaded using the load\_model() function from Keras.

Audiofile Preprocessing Function:

This function converts the audio file into spectrogram.

Streamlit App Configuration:

Title and Description:

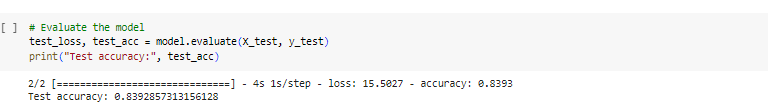
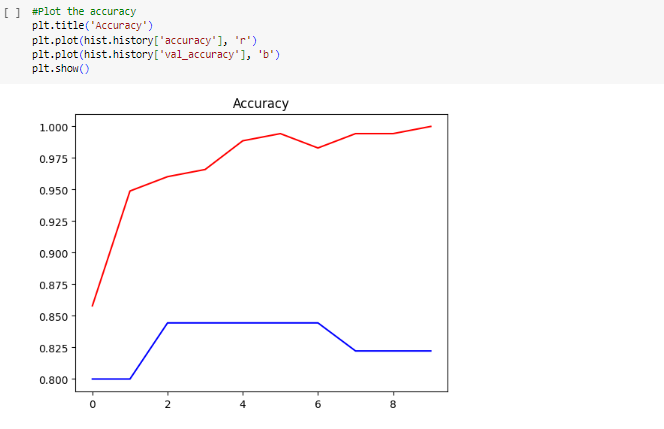
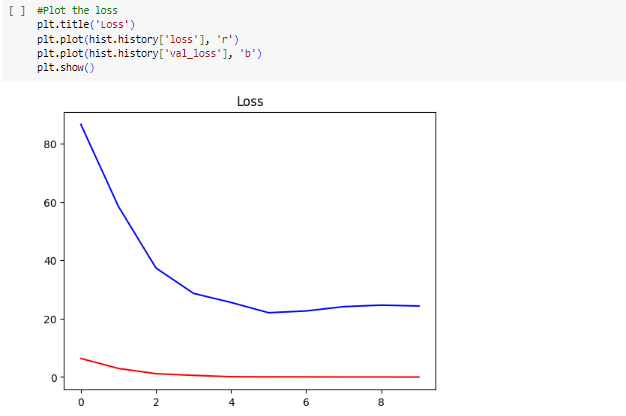
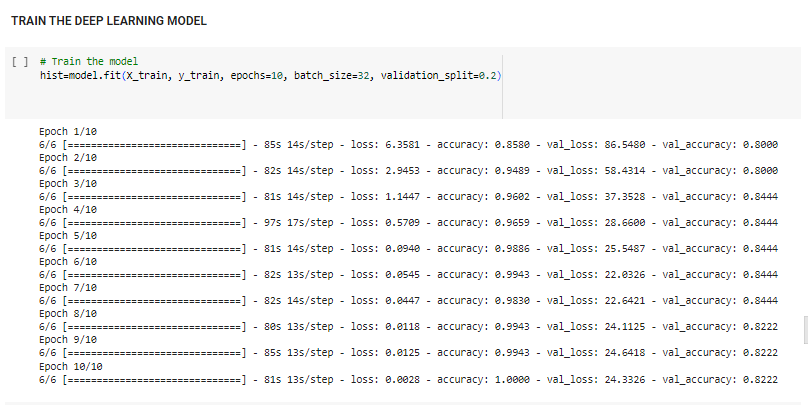
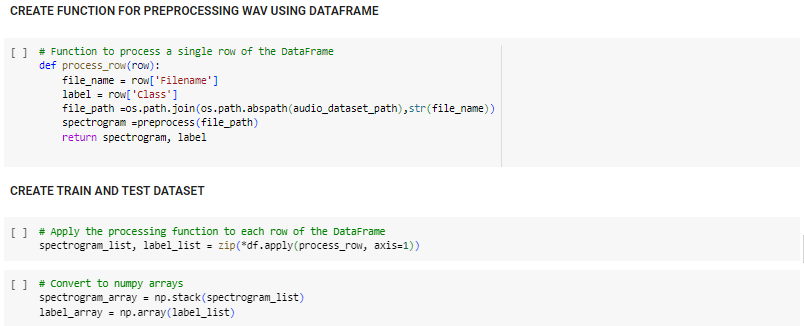
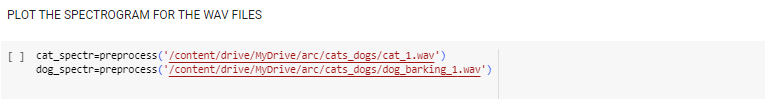
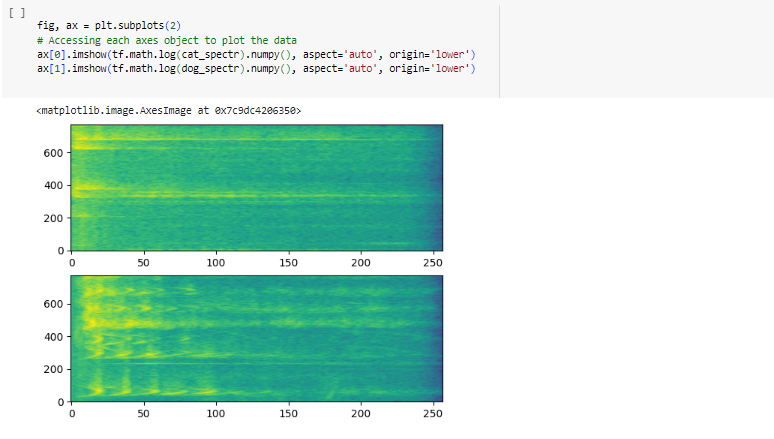
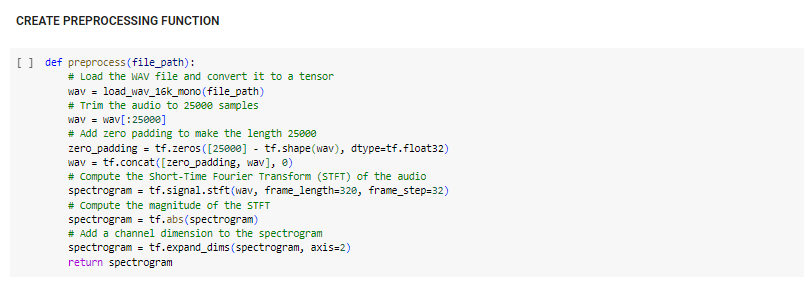
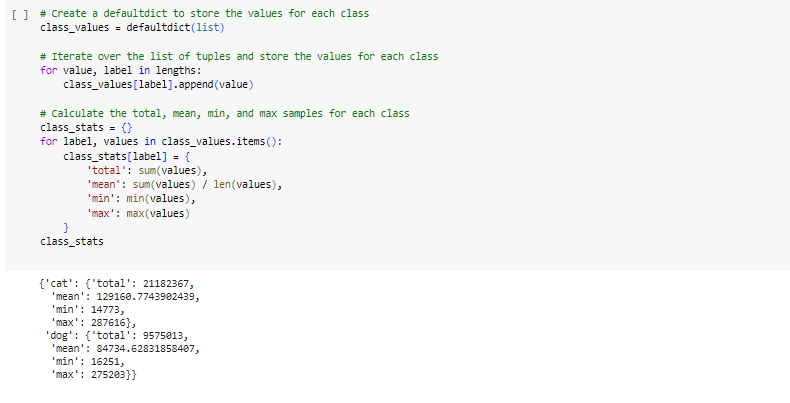
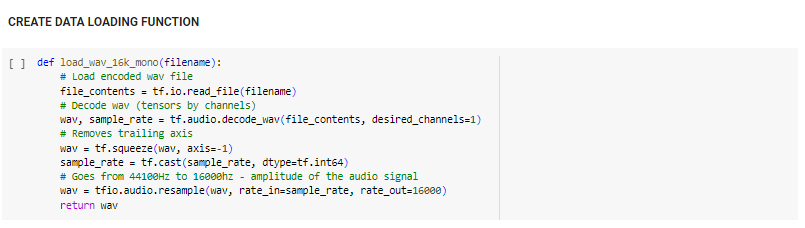
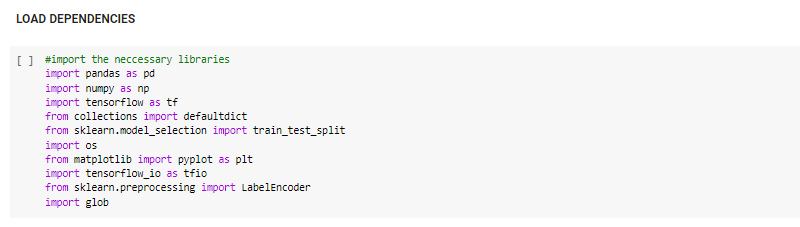
st.title(): Displays a title for the web application.

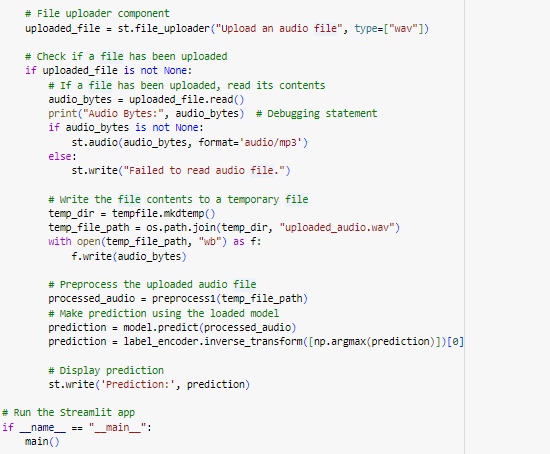
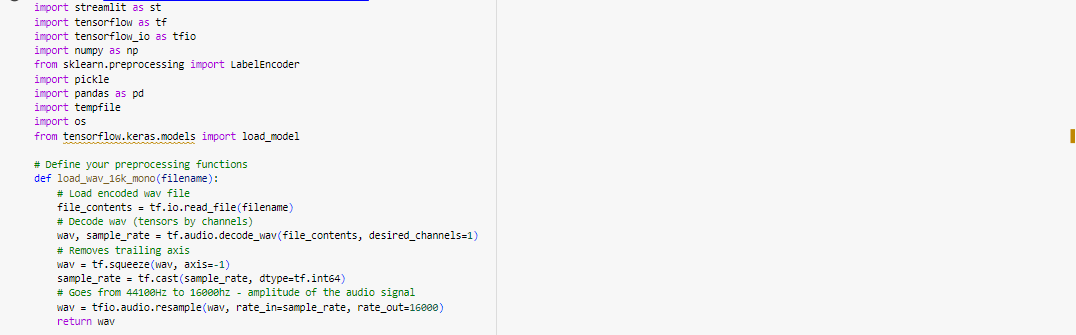
st.write(): Writes a description for the user, instructing them to upload an image of a handwritten digit for prediction.

File Uploader:

st.file\_uploader(): Displays a file uploader widget where users can upload an audio file (wav) containing the audio.

**CODE SNIPPET**

****

STREAMLIT SNIPPET

**RESULT AND DISCUSSION**

The results obtained from training the CNN model show promising performance in terms of accuracy and loss. The model achieves a high accuracy on both the training and validation datasets (got 83.92% accuracy), indicating good generalization capability. Additionally, the deployed Streamlit application provides a seamless user experience for predicting handwritten digits.



**CONCLUSION**

In conclusion, this project successfully demonstrates the implementation of Audio classification using CNN and its deployment on Streamlit.