



INTELLIGENT CLASS ROOM - NOISE PREDICTION AND ALERT SYSTEM USING PYAUDIO

Dr.S.Gandhimathi ph.D., Project supervisor, Assistant Professor Department of Computer Science,

K.Madhupriya B.Sc CS(2021-2024),

M.Sahanas Fathima B.Sc CS(2021-2024),

R.Saranya B.Sc CS(2021-2024),

Valluvar college of Science and Management, Karur-3

Abstract: A lot of time is wasted queuing to enter the classroom and collect materials. It is very difficult for teachers to deal with many students without technology as they are always there to answer questions. The American student spends 1025 hours a year following the instructions given to him. In daily teaching activities, teachers and professors try to find out if the students (or the audience) were satisfied with the lecture, which parts of the lecture were interesting to them, which presentation techniques and methods were more effective and motivating. something more Previous research has shown that students' attention spans begin to wane after about 10 minutes. As a result, at the end of the lecture, students remember about 65% of the information given in the first 10 minutes, and only 25% in the last 10 minutes. The main contribution of this paper is its innovative approach to the smart classroom environment and the multidisciplinary research project. Our work focuses on using observational and cognitive techniques to explore audiences and their behavior in cognitive environments. The information collected can provide insight into classroom performance levels by correlating sound and movement, presence and intensity. These smart environments can keep students engaged. It provides responses to lessons, useful for instructors to use Python skills to improve the quality of lessons and deliver reminders. A comprehensive review of the Smart Classroom project will highlight new ideas and research.

Index Terms - Python, Classroom, pyaudio, Pyttex3, Voice Recognition

I. INTRODUCTION

The Shrewdly Classroom over Sound with Caution Framework speaks to a cutting-edge activity to revolutionize the learning environment by consistently joining progressed innovations. This extend utilizes deliberately set sound sensors and modern machine learning calculations to screen and analyse surrounding sounds inside the classroom. The framework is planned to recognize between typical classroom exercises and potential disturbances, activating a responsive caution framework when required. A central preparing unit encourages real-time information investigation, whereas a customizable communication framework alarms instructors and chairmen to address troublesome circumstances expeditiously.



Figure: 1

The joining of a user-friendly interface, open through a portable app or web stage, engages teachers with farther observing and control capabilities. With highlights such as versatile affectability, verifiable information investigation, and crisis cautions, this shrewdly classroom framework points to upgrade in general classroom administration, cultivating an ideal learning environment for understudies whereas prioritizing security and security contemplations.

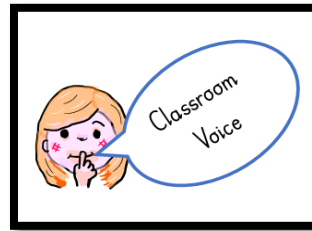


Figure: 2

The Shrewdly Classroom Over Sound with Caution Framework addresses basic challenges in advanced instructive settings, giving a multifaceted arrangement to upgrade the in general learning involvement. In today's energetic classrooms, overseeing troublesome behaviours and guaranteeing a conducive environment for viable educating and learning can be a noteworthy challenge. This extend looks for to relieve these challenges by leveraging progressed sound sensors and machine learning calculations to scholarly people screen classroom exercises. By recognizing and reacting to troublesome sounds in real-time, the framework engages teachers to preserve a cantered and beneficial learning climate. The customizable nature of the framework permits for adjustment to different instructing styles and exercises, making it a flexible device for teachers. In addition, the integration of crisis alarms and authentic information examination not as it were improves classroom administration but too contributes to the security and security of understudies. In an period where innovation plays a significant part in instruction, the Cleverly Classroom Over Sound with Caution Framework develops as a significant advancement, pointing to cultivate an environment conducive to successful instructing, learning, and understudy well-being.

VOICE RECOGNITION

The visibility and use of voice recognition has increased with the rise of artificial intelligence and smart assistants such as Amazon and Alexa, Apple and Siri, and Microsoft and Cortana. Voice recognition systems allow consumers to interact with technology simply by speaking to it, enabling hands-free requests, reminders and other simple actions.

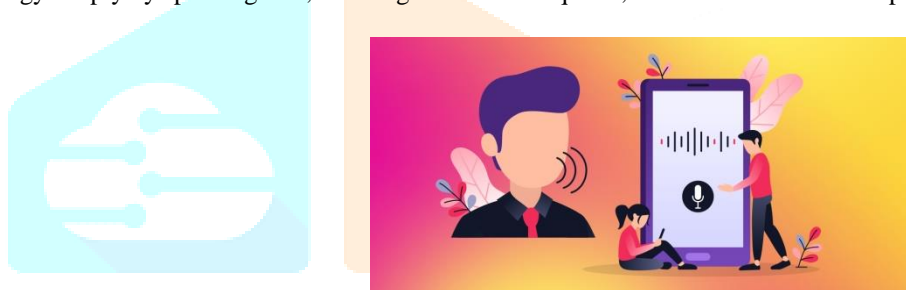


Figure: 3

Two types of speech recognition:

- Text-dependent - depends on what the person says about a certain set of words - requires advanced authentication and identity verification, and the user must say the required phrase to activate.
- Text-independent - does not depend on a specific text, but is based on conversational speech. Authentication does not require the user to say a set of required phrases.

Voice Recognition Job

Voice Recognition uses technology to evaluate the biometric data of your voice. This includes the frequency and flow of your voice and accent. Every word you speak is divided into polyphonic parts. It is then digitized and translated to create its own unique sound model. Artificial intelligence, deep learning and machine learning are behind speech recognition. Artificial intelligence is used to understand the colloquialisms, abbreviations and acronyms we use. Machine learning then assembles the patterns and evolves from this data using neural networks.

PyAudio

Pyaudio may be a flexible Python library that encourages sound preparing assignments, making it an fundamental instrument for engineers working on applications including sound recording, playback, and control. With Pyaudio, designers can effectively interface with sound gadgets, capture and handle sound information in real-time, and produce a wide extend of sound impacts. Its effortlessness and adaptability make it available for both tenderfoots and experienced designers looking for to consolidate sound usefulness into their Python applications.



Pyaudio underpins different stages, making it a cross-platform arrangement for errands such as voice acknowledgment, music applications, and real-time sound handling. Its open-source nature empowers a collaborative community, driving to continuous changes and overhauls. Whether utilized for instructive purposes, imaginative sound ventures, or proficient applications, Pyaudio gives a capable and user-friendly interface for dealing with sound errands inside the Python programming dialect.

PYTTSX3

Pyttxs3 may be a Python library that gives a helpful interface for text-to-speech (TTS) amalgamation. This effective and easy-to-use library permits designers to consolidate discourse union capabilities into their Python applications easily. The title "Pyttxs3" stands for "Python Text-to-Speech adaptation 3 and it builds upon its forerunner, Pyttxs. One of the key focal points of Pyttxs3 is its cross-platform compatibility, supporting Windows, macOS, and Linux working frameworks. Engineers can utilize this library to change over literary data into talked words, empowering applications to communicate with clients through discourse.

Pyttxs3 comes with different setups, permitting engineers to control discourse rate, volume, and voice choice, giving a customizable and immersive involvement. Whether it's including voice prompts to applications, creating availability highlights, or making intelligently chatbots, Pyttxs3 demonstrates to be a important device for upgrading the client encounter by coordination natural-sounding discourse blend consistently into Python ventures.

APPLICATIONS OF VOICE RECOGNITIONS

- Audio Recording and Playback
- Voice Assistants and Recognition
- Telecommunications and VoIP Applications
- Audio Processing and Analysis
- Music and Sound Synthesis
- Educational Tools
- Automated Testing and Quality Assurance
- Game Development
- Bioacoustics and Research
- Accessibility Solutions

2. SYSTEM ANALYSIS**2.1 EXISTING SYSTEM**

Controlling students in classroom manually is one among the most challenging task for the teachers to manage in a blended learning environment; with the target of accelerating classroom strength, teachers aim to force students to silence by making voice manually. This is a very bad one and teachers suffer a lot through it. Parallel to this, the current students also refuse to listen to the teachers' advice. It is very regrettable that most of the teachers in many schools, colleges and institutes are facing great difficulty in keeping the students calm.

DISADVANTAGES

- Not Communicating Expectations Clearly
- Being Inconsistent
- Waiting Too Long to Intervene

2.2 PROPOSED SYSTEM

This project aims to provide an easy and efficient way to interact with classroom students by giving voice commands in human language. The proposed system has a great flexibility by using voice assistant and PyAudio technology to recognize noises and alert students to keep silent by making a sound. Also flexibility is increased by giving users an option to specify the commands according to their comfort and also in the human language.

ADVANTAGES

- Minimal Effort
- Eyes Free
- Hands-free
- Fast response

3. IMPLEMENTATION**IMPORTING THE REQUIRED LIBRARY**

Detecting classroom yelling involves processing audio data, so you'll need libraries that can handle audio input and perform signal processing. One commonly used library for this purpose is pyaudio for audio analysis, and numpy for numerical operations. Additionally, you may use voice processing libraries like pyttxs3 and portaudio for building a classroom noise detector.

CREATE A VOICE ASSISTANT

Creating a voice detector for classroom yelling involves a combination of audio signal processing and machine learning techniques. First, you'll need to collect and pre-process audio data containing examples of both normal classroom sounds and instances of yelling. The PYAUDIO and NUMPY library can be used to extract relevant features from the audio, such as spectral characteristics or Mel-frequency cepstral coefficients (MFCCs). Once you recognize the classroom noise our model will be worked to predict a classroom noise and alert the students to keep silent.

TRAIN ASSISTANT WITH COMMANDS

Training a voice assistant for the classroom yelling detection project involves developing a system that can not only recognize instances of yelling but also respond appropriately. First, you need to integrate a speech recognition library, such as PYTTSX3 to convert audio input into text. Once the voice input is transcribed, you can use the voice detector model, previously trained for yelling detection, to analyse whether the spoken words include yelling. If the voice detector identifies yelling, the voice assistant can trigger a predefined response to alert the students.

MAKE THE ASSISTANT WORK IN REAL TIME

To make the assistant work in real-time for classroom yelling detection, you need to implement a continuous audio input processing pipeline. Utilize a library like PYAUDIO to capture and stream real-time audio input. The assistant should constantly feed the incoming audio data to the pre-trained voice detector model, allowing it to make instant predictions on whether yelling is detected.

4. EXPERIMENT RESULT

PYTHON

Python is a translated, object-oriented, high-level programming dialect with energetic semantics. It's advanced built-in data structures with dynamic typing and dynamic binding; makes it very attractive both for rapid application development and for use as scripting or glue languages to connect existing components. Python's simple, easy-to-learn syntax emphasizes readability and thus reduces programming costs. Python supports modules and packages, which promote program modularity and code reuse. The Python interpreter and extensive standard library are freely available in source or binary form for all major platforms and are freely redistributable. Often, developers fall in love with Python because of the increased productivity it offers. Because there is no compile step, the edit-test-debug cycle is incredibly fast. Debugging Python programs is easy: a bug or bad input never causes segmentation faults. Instead, the interpreter throws an exception when an error is detected. If the program does not catch an exception, the interpreter prints a stack trace. An entry-level debugger allows you to inspect local and global variables, evaluate arbitrary expressions, set breakpoints, step through code line by line, etc. The debugger is written in Python itself, which is a testament to Python's introspection. On the other hand, often the fastest way to debug a program is to add some printable statements to the source: the fast edit-test-debug cycle makes this simple approach very powerful.

Applications

The Python Bundle File (PyPI) has thousands of third-party modules for Python. Both Python's standard library and the community-contributed modules permit for perpetual conceivable outcomes.

- Web and Web Improvement
- Database Get to
- Desktop GUIs
- Scientific and Numeric
- Education
- Network Programming
- Software and Amusement Advancement

Open-source

Python is created beneath an OSI-approved open source permit, making it unreservedly usable and distributable, indeed for commercial utilize. Python's permit is managed by the Python Computer program Establishment.

- Learn more almost the permit
- Python permit on OSI
- Learn more around the Establishment.

History:

Python could be a broadly utilized general-purpose, high-level programming dialect. It was at first outlined by Guido van Rossum in 1991 and created by Python Computer program Establishment. It was basically created for accentuation on code readability, and its sentence structure permits software engineers to specific concepts in less lines of code. Within the late 1980s, history was approximately to be composed. It was that time when working on Python begun. Before long after that, Guido Van Rossum started doing its application-based work in December of 1989 at Centrum Wiskunde and Informatics (CWI) which is situated within the Netherlands. It was begun firstly as a pastime extend since he was trying to find an curiously venture to keep him possessed amid Christmas. The programming dialect in which Python is said to have succeeded is ABC Programming Dialect, which had meddle with the Single adaptable cell Working Framework and had the highlight of special case dealing with. He had as of now made a difference to form ABC prior in his career and he had seen a few issues with ABC but enjoyed most of the highlights. After that what he did was truly exceptionally intelligent. He had taken the language structure of ABC, and a few of its great highlights. It came with a parcel of complaints as well, so he fixed those issues totally and had made a great scripting dialect that had expelled all the imperfections. The motivation for the title came from BBC's TV Appear – 'Monty Python's Flying Circus', as he was a enormous fan of the TV appear additionally he needed a brief, special and somewhat secretive title for his innovation and thus he named it Python! He was the "Benevolent tyrant for life" (BDFL) until he ventured down from the position as the pioneer on 12th July 2018. For very some time he utilized to work for Google, but as of now, he is working at Dropbox. The dialect was at last discharged in 1991. When it was discharged, it utilized a lot fewer codes to specific the concepts, when we compare it with Java, C++ and C. Its plan reasoning was quite great as well. Its main objective is to supply code coherence and progressed developer productivity. When it was discharged, it had more than sufficient capability to supply classes with legacy, a few center information sorts special case dealing with and capacities..

PYAUDIO

PyAudio gives Python ties for PortAudio, the cross-platform sound I/O library. With PyAudio, you'll effectively utilize Python to play and record audio on a assortment of platforms.



Pyaudio could be a Python library that gives bindings for the PortAudio library, permitting Python applications to work with sound. PortAudio may be a cross-platform sound I/O library that gives a uniform API over different working frameworks. PyAudio acts as a wrapper around PortAudio, making it available and simple to utilize for Python engineers. The advancement of PyAudio begun

around 2001-2002. The library has experienced upgrades and advancements over the a long time, reflecting changes in both Python and PortAudio.

APPLICATIONS OF PYAUDIO

- Audio Recording and Playback
- Real-Time Audio Processing
- Telecommunications and VoIP
- Music and Sound Processing
- Educational Projects
- Speech Recognition
- Voice Assistants and Automated Systems
- Audio Testing and Measurement

PYTTSX3

pyttsx3 is a text-to-text library in Python. Unlike other libraries, it works offline and supports Python 2 and 3. The application obtains a reference to pyttsx3 by calling the pyttsx3.init() factory function. A mechanical model. It is a very simple tool to convert written text to speech. The pyttsx3 module supports two languages. The first is female and the second is male by "sapi5" for Windows. It supports three TTS engines.

- sapi5 – SAPI5 on Windows
- nsss – NSSpeechSynthesizer on Mac OS Uses different OS-based speech engines, such as SAPI5 on Windows and NSSpeechSynthesizer on macOS. The main application of pyttsx3 is to convert text to speech, which is useful in a variety of situations.

SYSTEM ARCHITECTURE

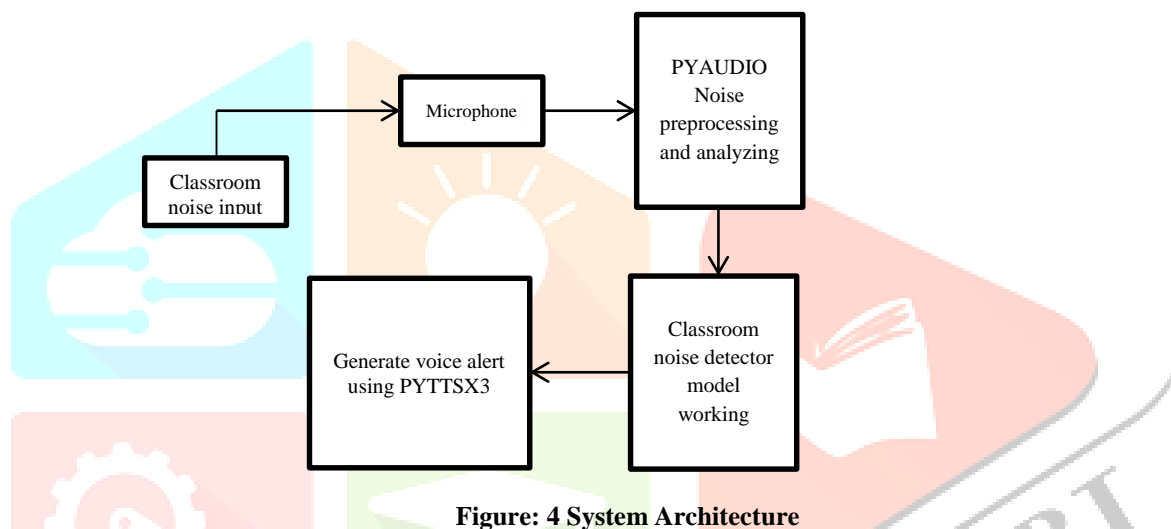


Figure: 4 System Architecture

```

import pyaudio
import numpy as np

def detect_yelling():
    CHUNK = 1024 # Number of frames per buffer
    FORMAT = pyaudio.paInt16 # Audio format (16-bit PCM)
    CHANNELS = 1 # Number of audio channels (1 for mono, 2 for stereo)
    RATE = 44100 # Sample rate (samples per second)

    p = pyaudio.PyAudio()

    stream = p.open(format=FORMAT,
                    channels=CHANNELS,
                    rate=RATE,
                    input=True,
                    frames_per_buffer=CHUNK)

    print("Listening for yelling...")

    try:
        while True:
            data = np.frombuffer(stream.read(CHUNK), dtype=np.int16)

            # Calculate the root mean square (RMS) to measure the amplitude
            rms = np.sqrt(np.mean(np.square(data)))

            # Adjust this threshold according to your environment and microphone sensitivity
            threshold = 1000

            if rms > threshold:
  
```

Figure: 4 Voice input

```

            rate=RATE,
            input=True,
            frames_per_buffer=CHUNK)

    print("Listening for yelling...")

    try:
        while True:
            data = np.frombuffer(stream.read(CHUNK), dtype=np.int16)

            # Calculate the root mean square (RMS) to measure the amplitude
            rms = np.sqrt(np.mean(np.square(data)))

            # Adjust this threshold according to your environment and microphone sensitivity
            threshold = 1000

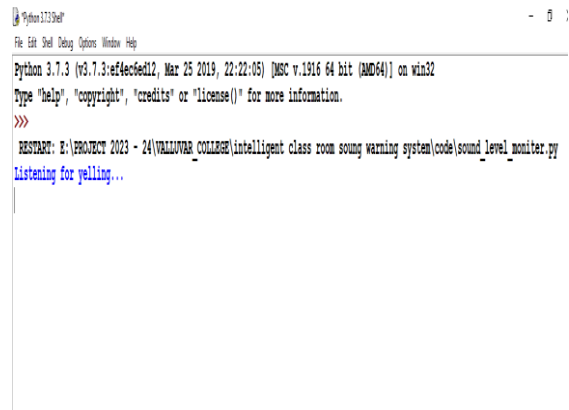
            if rms > threshold:
                print("Yelling detected!")

    except KeyboardInterrupt:
        print("Stopped listening.")

    finally:
        stream.stop_stream()
        stream.close()
        p.terminate()

if __name__ == "__main__":
    detect_yelling()
  
```

Figure: 4 Voice Output



```
Python 3.7.3 (tags/v3.7.3:ef4ec0ed12, Mar 25 2019, 22:22:05) [MSC v.1916 64 bit (AMD64)] on win32
Type "help()", "copyright()", "credits()" or "license()" for more information.
>>>
RESTART: E:\PROJECT 2023 - 24\VALIUVAR COLLEGE\intelligent class room sound warning system\code\sound_level_monitor.py
Listening for yelling...
```

Figure: 4 Voice recognition

5. CONCLUSION

In conclusion, the implementation of INTELLIGENT CLASSROOM VOICE DETECTOR project using pyttsx3 in Python has provided a versatile solution for converting written text into spoken words. This project has demonstrated its utility across various applications, including accessibility tools for visually impaired individuals, educational software to aid language learning, notification systems for alerts and reminders, and voice interfaces for interactive applications. Additionally, the versatility of pyttsx3 makes it suitable for integration into automation scripts, interactive games, and even multimedia production projects. While this library serves well for basic TTS needs, developers may explore more advanced solutions for applications requiring nuanced voice synthesis. Overall, the project showcases the simplicity and effectiveness of pyttsx3 in enabling text-to-speech functionality in diverse scenarios, contributing to improved accessibility and user experience in different domains.

FUTURE WORK

For future implementations of this text-to-speech project using pyttsx3, several enhancements can be considered to elevate its capabilities and user experience. First and foremost, integrating more advanced natural language processing (NLP) techniques and machine learning models could improve the overall quality and naturalness of the generated speech. This might involve exploring cloud-based TTS services that leverage deep learning algorithms for better voice synthesis. Additionally, incorporating support for multiple languages and dialects would enhance the project's global applicability, making it more inclusive for users from different linguistic backgrounds. Furthermore, introducing user customization options, such as voice selection and speech rate adjustments, could provide a more personalized experience. To expand accessibility, integration with voice recognition systems could enable bidirectional communication, allowing users to interact verbally with the application. Lastly, exploring opportunities for cloud-based deployment would facilitate scalability and ensure seamless performance in resource-intensive scenarios. These future enhancements aim to refine the project, making it more sophisticated, adaptable, and user-friendly for a broader range of applications and user preferences.

REFERENCE

1. Asha G. Hagargund, Sharsha Vanria Thota, Mitadru Bera, Eram Fatima Shaik (2017) "Image to Speech Conversion for Visually Impaired", International Journal of Latest Research in Engineering and Technology, ISSN: 2454-5031, Issue 06, Vol. 03, No. 0, pp. 09-15.
2. Deepa V. Jose, Alfateh Mustafa, Sharan R (2014) "A Novel Model for Speech To Text Conversion", International Refereed Journal of Engineering and Science, ISSN(online): 2319-183X, (print) 2319-1821, Issue 1, Vol. 3, No. 0, pp. 39-41.
3. Kiran Rakshana R, Chitra C (2019) "A Smart Navguide System for visually Impaired", International Journal of Innovative Technology and Exploring Engineering, ISSN: 2278-3075, Issue 6S3, Vol. 8, No. 0, pp. 0.
4. Vaibhav V. Govekar, Meenakshi A (2018) "A Smart Reader for Blind People", International Journal of Science Technology & Engineering, ISSN: 2349-784X, Issue 1, Vol. 5, pp. 0.
5. L. Latha, V. Geethani, M. Divyadharshini, P. Thangam (2019) "A Smart Reader for Blind People", International Journal of Engineering and Advanced Technology, ISSN: 2249-8958, Issue 6S3, Vol. 8, No. 0, pp. 0.
6. Shraddha Hingankar, Prachi Tardekar, Santoshi Pote (2020) "A Smart Reader for Visually Impaired Individuals", International Research Journal of Engineering and Technology, P-ISSN: 2395-0072, E-ISSN: 2395-0056, Issue 07, Vol. 07, No. 0, pp. 0.
7. Ram Nivas Duraisamy, Sathya Manoharan (2018) "A Smart Reader for Visually Impaired People", International Journal of Latest Engineering and Management Research, ISSN: 2455-4847, Issue 10, Vol. 03, No. 0, pp. 39-46.
8. Aravind S, Roshna E (2013) "A Text Reding System for the Visually Disabled", International Journal of Research in Computer Application & Management, ISSN: 2231-1009, Issue 12, Vol. 3, No. 0, pp. 0.
9. Akhilesh Panchal, Shrugal Varde, M.S. Panshe (2016) "Automatic Scene Text Detection and Recognition system for visually Impaired People", International Journal for Research in Emerging Science and Technology, E-ISSN: 2349-761, Issue 6, Vol. 3, No. 0, pp. 0.
10. N. S. Lokhande, P.B. Pawar, S.J. Shelke, A.R. Wagh (2019) "Book Reader for Visually Impaired Using Raspberry Pi", International Research Journal of Engineering and Technology, P-ISSN: 2395-0072, E-ISSN: 2395-0056, Issue 02, Vol. 06, No. 0, pp. 0.