Archives available at journals.mriindia.com

International Journal on Advanced Computer Engineering and Communication Technology

ISSN: 2278-5140
Volume 14 Issue 01, 2025

Human Scream Detection

Prof. Swati P Gade[1], Pranita More[2], Nikita Pawar[3], Vaishali Surwase[4], Ashutosh Kohinkar[5]
[1]Professor PDEA's College of Engineering, Pune.
[2,3,4,5]Student PDEA's College of Engineering, Pune.
swatiunique2006@gmail.com[1], pranitamore6301@gmail.com[2], nikitapawar9579@gmail.com[3], surwasevaishali595@gmail.com[4], kohinkarashu21@gmail.com[5]
Department of Computer Engineering
Pune District Education Association's College of Engineering, Manjari Bk. Hadapsar, Pune, Maharashtra, India. - 412307 Email: coem@pdeapune.org

Peer Review Information

Submission: 15 Jan 2025
Revision: 12 Feb 2025
Acceptance: 12 March 2025

Keywords

Human Scream Detection
Public Safety
Audio Analysis
Crime Rate
Support Vector Machine
Mel-Frequency Cepstral Coefficients

Abstract

Crime, including incidents like homicides, assaults, and robberies, is a common problem worldwide and a big concern for society. A common issue is that police often arrive at crime scenes too late, partly because they don't get timely, accurate information. To help address this, a disguised desktop application is proposed. This program uses modern technologies, including machine learning and deep learning techniques like Support Vector Machines (SVM) and Multilayer Perceptron (MLP), to quickly recognize and analyze human sounds while quietly running in the background. In an emergency, the program automatically sends SMS alerts to chosen contacts. This advanced technology improves accuracy in detecting threats and speeds up response times by identifying specific human sounds amid background noise. This project introduces a way to help control crime by detecting and analyzing human screams in real-time. Using machine learning and deep learning, the system can accurately pick out distress sounds even with background noise and quickly notify authorities. It works by processing audio, identifying key features, and using algorithms to tell apart different types and intensities of screams. When it detects a scream that signals danger, it sends alerts to nearby law enforcement, including the location and audio recording for evidence. The goal is to make communities safer and reduce the negative effects of crime by strengthening people's confidence in their ability to protect themselves and their neighborhoods.

Introduction

The project "Scream Detection" aims to develop a system that automatically detects and analyzes screams in audio recordings. This system uses machine learning, signal processing and sound classification methods to identify screams and distinguish them from other background sounds. The Human Scream Detection System is a groundbreaking innovation that uses real-time audio data to recognize and respond to human screams. Human scream detection is an emerging technology designed to help control crime rates by identifying distress signals in real time. By using advanced audio recognition algorithms, this technology can distinguish between regular background noise and a scream or cry for help. Once a scream is detected, an alert is sent to authorities or nearby security personnel, enabling a rapid response that can prevent further harm. This system is beneficial in high- risk areas or places with limited surveillance, as it adds a layer of protection by picking up sounds of distress even when visual cues may not be available. The implementation of scream detection technology has the potential to enhance public safety, act as a deterrent to potential criminals, and significantly improve emergency response times, ultimately contributing to a reduction in crime rates.

Literature Survey

Human scream detection has emerged as a critical research area in the field of audio signal processing, driven by its applications in safety, surveillance, and healthcare systems. The capability to identify screams accurately in real time enables prompt responses to emergencies and improves public safety. Over the years, researchers have explored various methods and technologies to enhance the efficiency and robustness of scream detection systems. This section reviews the existing literature on human scream detection, highlighting key methodologies, challenges, and advancements.

Early Studies on Scream Detection

Initial efforts in scream detection focused on basic acoustic signal processing techniques. These studies identified unique features of screams, such as high pitch, abruptness, and roughness, distinguishing them from other sounds like speech, laughter, or background noise. Traditional machine learning approaches, including Hidden Markov Models (HMMs), Gaussian Mixture Models (GMMs), and Support Vector Machines (SVMs), were employed for classification tasks. However, these methods faced challenges in noisy environments and lacked the flexibility to generalize across diverse datasets.

Deep Learning Approaches

The advent of deep learning revolutionized scream detection by enabling models to learn features directly from raw audio data. Convolutional Neural Networks (CNNs) became popular for analyzing spectrogram representations of audio signals, offering superior accuracy and robustness. Hybrid architectures combining CNNs and Long Short-Term Memory (LSTM) networks further enhanced performance by capturing temporal dependencies. Pretrained models, such as Wav2Vec and OpenL3, have also been fine-tuned for scream detection tasks, achieving state-of-the-art results in both accuracy and real-time applicability.

Challenges in Real-World Implementation

Despite technological advancements, real-world deployment of scream detection systems poses several challenges. Noise robustness remains a major concern, as urban environments and crowded public spaces produce overlapping and unpredictable sounds. False positives, caused by sounds such as alarms, music, or children's voices, hinder the reliability of detection systems. Furthermore, the lack of large-scale, labeled datasets with diverse scream samples limits the ability to train and evaluate models effectively. Addressing latency issues in real-time applications, especially on resource-constrained devices, also remains a critical area of focus.

Methodology

Use SVM Classifier for Initial Scream Detection: Begin by using an SVM classifier to detect screams among various audio inputs, such as screams, shouts, and normal speech.

Noise Filtering: If the environment contains only background noise, shouts, or regular speech, the model exits immediately, as these sounds are not classified as emergencies.

Pass to Multilayer Perceptron (MLP) for Scream Verification: If the SVM classifier detects a scream, the audio input is passed to a Multilayer Perceptron (MLP) model for further analysis.

Scream Confirmation: The MLP model performs a detailed examination to confirm whether the sound is indeed a scream.

Alert Generation: If the MLP model verifies the sound as a scream, an alert SMS is sent to the nearest appropriate authority. If not, the model exits without action.

Workflow Overview

The workflow of a human scream detection system involves a structured series of processes that transform raw audio inputs into actionable outputs. This section provides a detailed overview of the workflow, outlining each stage from input acquisition to real-time detection and alert generation.

The process begins with audio acquisition, where sound data is captured from microphones or audio input devices. This input can be in the form of live streams from surveillance systems, wearable devices, or pre-recorded audio files. Ensuring high-quality audio capture is critical for accurate processing and detection.

Next, the preprocessing stage cleans and prepares the audio signals for analysis. Noise suppression techniques are applied to eliminate background interference and enhance the clarity of the scream signal. The audio is then segmented into fixed-duration frames using framing and windowing techniques, which divide the continuous signal into smaller, overlapping segments. These segments serve as the input units for feature extraction.

During the feature extraction stage, the system computes distinctive characteristics of the audio signal that are effective in differentiating screams from other sounds. Spectrograms are generated to represent the time-frequency characteristics of the audio, and features such as Mel-Frequency Cepstral Coefficients (MFCCs), zero-crossing rate, and spectral roughness are extracted. These features capture the high-pitched, abrupt, and chaotic nature of screams.

The extracted features are then passed into the classification stage, where a trained machine learning or deep learning model analyzes the input and determines whether the sound corresponds to a scream. Models like Convolutional Neural Networks (CNNs) or hybrid architectures (e.g., CNN + LSTM) are often employed for their ability to learn spatial and temporal patterns in the data. The model is trained on a labeled dataset of scream and non-scream audio samples to achieve high accuracy and generalization.

Once a scream is detected, the system triggers an alert mechanism. This mechanism generates real-time notifications or alarms, enabling rapid responses to potentially dangerous or emergency situations. The system can be configured to send alerts to designated personnel, activate security measures, or log the detected event for future analysis.

Throughout the workflow, real-time processing is prioritized to minimize latency, ensuring that detection and response occur without delay. Edge computing techniques are employed to optimize the system for deployment on resource-constrained devices, such as wearables or IoT platforms.

Finally, performance monitoring and feedback mechanisms allow for continuous improvement of the system. New audio samples can be incorporated into the training process to enhance model performance and adapt to evolving environments.

Yolov3 Algorithm:

Data Collection Preprocessing:

Audio Data: Collect a dataset of audio recordings containing both screams and non-screams (e.g., background noise, speech, music).

Preprocessing: The audio signals are pre-processed to remove noise and convert them into a suitable format for the

MLP:

Feature Extraction: Extract relevant features from the audio, such as MFCC (Mel-frequency cepstral coefficients), spectrograms, or zero-crossing rate, that represent the characteristics of the sounds.

Normalization/Scaling: The features are scaled or normalized to ensure the data is in a consistent range for training the MLP.

Data Split:

Data split refers to Split the data set into Training and Testing the set the MLP, the validation set to tune hyper parameters, and the test set to evaluate model performance.

Model Architecture:

The MLP consists of three main layers: Input Layer: The extracted features (e.g., MFCC values) are fed into the input layer. Hidden Layers: The MLP has one or more hidden layers, each with a set of neurons. These layers apply activation functions (like ReLU or sigmoid) to introduce non-linearity, allowing the model to learn complex patterns in the data. Output Layer: The output layer produces a classification result (e.g., 1 for scream and 0 for non- scream) based on the learned patterns. A softmax activation function is often used in the output layer for multi-class classification or a sigmoid function for binary classification (scream vs. non-scream).

Training:

Forward Propagation: The audio features are passed through the network, layer by layer, to produce an initial prediction (scream or non-scream).

Loss Calculation: The difference between the predicted output and the actual label (ground truth) is calculated using a loss function (e.g., cross-entropy loss for classification tasks). Backpropagation: The model adjusts the weights of the network through backpropagation, using the loss value to update weights and biases by applying gradient descent. The model iteratively adjusts weights over several epochs (iterations) to minimize the loss and improve accuracy.

Testing Evaluation:

After training, the model is tested on the unseen test data to evaluate its performance. Metrics like accuracy, precision, recall, and F1-score are calculated to assess how well the MLP classifies screams and non-screams.

Prediction:

For real-time scream detection, the trained MLP model can be deployed to classify incoming audio streams. When a new audio clip is received, the model extracts the same features (e.g., MFCCs), passes them through the MLP, and outputs a prediction (scream or non-scream). If a scream is detected, an alert can be triggered (e.g., notifying authorities, caregivers, or security personnel)

Implementation and Features

The implementation of a human scream detection system involves the design and deployment of a robust audio processing pipeline that can identify scream sounds accurately, even in noisy or dynamic environments. This section describes the key steps and features involved in implementing such a system.

Implementation Steps

Audio Data Collection and Preparation

Dataset Collection: Gather audio recordings containing scream sounds, non-scream sounds (e.g., speech, laughter, ambient noise), and diverse environmental noises. Publicly available datasets like Audio Set or ESC-50 can serve as a foundation, augmented by custom-recorded screams.

Annotation: Label the dataset to differentiate between scream and non-scream audio segments. Ensure diversity in scream types (e.g., loud, muffled, high-pitched) for better generalization.

Preprocessing

Noise Reduction: Apply filtering techniques (e.g., spectral subtraction or bandpass filters) to suppress background noise and enhance scream detection clarity.

Framing and Windowing: Divide audio signals into overlapping frames (e.g., 25ms frames with 10ms overlap) to capture temporal features.

Normalization: Standardize audio intensity to ensure consistency across recordings.

Feature Extraction

Extract acoustic features that characterize screams effectively:

Mel-Frequency Cepstral Coefficients (MFCCs): Capture the spectral properties of screams. Spectrograms: Represent audio signals in the time-frequency domain for visualization and analysis.

Zero-Crossing Rate (ZCR): Measure the rate of signal sign changes, often higher in screams. Spectral Roughness and Energy Entropy: Identify the chaotic and high-energy nature of screams.

Model Training

Model Selection: Use deep learning models like Convolutional Neural Networks (CNNs) for analysing spectrograms or hybrid models (e.g., CNN + LSTM) for capturing temporal patterns.

Training Process: Train the model on the prepared dataset using a supervised learning approach. Utilize techniques like data augmentation (e.g., pitch shifting, time stretching) to improve model robustness.

Loss Function and Optimizer: Employ binary cross-entropy as the loss function for binary classification and optimize using Adam or RMSProp optimizers.

Real-Time Detection and Deployment

Inference Pipeline: Implement a streaming audio pipeline that processes live audio in overlapping windows.

Edge Optimization: Optimize the model for low-latency inference on resource-constrained devices using techniques like quantization or pruning.

Alert System: Integrate with an alert mechanism (e.g., notifications, alarms) triggered upon detecting a scream.

Summary And Conclusions

screaming alters speech parameters significantly, which affects how well speaker recognition systems work. According to this investigation, speaker ID utilizing screaming is neither a reliable nor efficient use of current technologies. algorithm-based scream detection system is set to make a substantial contribution to security infrastructure, public safety, and medical applications. This system is a useful tool for quickly and accurately responding to distress signals since it combines machine learning, adaptive learning processes, and advanced audio processing techniques, all of which offer potential for future improvements.

References

M. Akbacak and J.H.L. Hansen, “Environmental sniffing: noise knowledge estimation for robust speech systems,” in Acoustics, Speech, and Signal Processing. Proceedings. (ICASSP’03). IEEE International Conference on. IEEE, vol. 2, pp. II–113Leonardo Software, Santa Monica, CA 90401, Conference on, Oct 2009, pp. 2695–2700.

L.S. Kennedy and D.P.W. Ellis, “Laughter detection in meetings,” in NIST ICASSP 2004 Meeting Recognition Workshop, Montreal. National Institute of Standards and Technology, pp. 118–121.

F. Afghah, A. Razi, J. Chakareski, and J. Ashdown, Wildfire Monitoring in Remote Areas using Autonomous Unmanned Aerial Vehicles. 2019.

C.KaoandS.Chang, “An Intelligent Real-Time Fire-Detection Method Based on Video Processing,” IEEE 37th Annu. 2003 Int. Carnahan Conf. OnSecurity Technol. 2003 Proc., 2003.

N.I. Binti Zaidi, N. A. A. Binti Lokman, M. R. Bin Daud, H. Achmad, and K.A. Chia, “Fire recognition using RGB and YCbCr color space,” ARPN J. Eng. Appl. Sci., vol. 10, no. 21, pp. 9786–9790, 2015.