Grenze International Journal of Engineering and Technology, June Issue

Human Scream Detection and Analysis for Controlling Crime Rate

Sujata Joshi[1], Rajwardhann Surryavanshi[2], Rohit[3], Mahesh N[4] and Aniket Yadav[5]
[1-5]Computer Science and Engineering Nitte Meenakshi Institute of Technology Bengaluru, India
Email: sujata.joshi@nmit.ac.in, 1n21cs142.rajwardhann@nmit.ac.in, 1nt21cs149.rohit@nmit.ac.in, 1nt21cs099.mahesh@nmit.ac.in, 1nt21cs032.aniket@nmit.ac.in

***Abstract*— This paper affords an innovative framework for actual-time detection and evaluation of human screams to enhance public safety and enhance crime response. Leveraging both machine gaining knowledge of, our machine successfully discriminates scream sounds from ambient environmental noise, even in hard acoustic situations. A sturdy preprocessing pipeline is implemented, which involves loading audio files, padding, log-spectrogram extraction, and min–max normalization to extract reliable spectral features from audio records. these features are then utilized by our custom scream detection model to reap excessive detection accuracy even as minimizing false positives. Inte- grated with a geolocation-based totally alert mechanism, the device automatically sends SMS notifications to nearby regulation enforcement, ensuring timely intervention during capability crime incidents. Experimental reviews show that our approach significantly enhances response times and detection precision, presenting a promising tool for proactive crime prevention.**

I. INTRODUCTION

In these days's fast-evolving safety panorama, making sure public safety and stopping crime have come to be pinnacle priorities. advanced technologies are increasingly more deployed to deal with those demanding situations, and one rising solution is human scream detection. utilizing artificial intelligence (AI) and machine gaining knowledge of, this generation analyzes crucial acoustic capabilities—along with pitch, frequency, intensity, and period—to pick out misery sounds in actual time. by way of correctly distinguishing
screams from ambient noises like traffic, sirens, and informal conversations, the machine offers automated indicators that preventing crime to include emergency management in the event of accidents, natural disasters, or domestic abuse, supporting the effective distribution of vital resources.

In high-risk environments like parking lots, private alleyways, public transportation stations, and residential neighborhoods where criminal activity is more prevalent, scream detection is particularly helpful [14]. Law enforcement and security teams can enhance personal security and act more swiftly by integrating this technology with existing surveillance networks or incorporating it into mobile safety applications. Furthermore, its applications extend beyond crime prevention to include emergency management in the case of mishaps, natural disasters, or domestic abuse, facilitating the efficient allocation of essential resources.

A thorough framework for real-time scream detection and analysis is presented in this paper. Our method includes a geolocation-based alert system that quickly alerts local law enforcement and a strong preprocessing pipeline that transforms unprocessed audio signals into dependable spectral features.

Grenze ID: 01.GIJET.12.2.86_23

The suggested system offers a promising tool for proactive crime prevention and emergency management, demonstrating notable improvements in detection accuracy and response time through thorough evaluation.

II. LITERATURE SURVEY

Scream detection is especially useful in high-risk settings, such as parking lots, private alleyways, public transportation stations, and residential neighborhoods where criminal activity is more common [14]. By incorporating this technology into mobile safety applications or integrating it with current surveillance networks, law enforcement and security teams can intervene more quickly and improve personal security. Additionally, its uses go beyond in recent developments in acoustic event recognition, with the goal of supporting public safety systems and facilitating prompt emergency responses.

Researchers have concentrated on finding unique audio features that distinguish screams from typical ambient noises. In order to differentiate screams from normal speech or background noise, fundamental research highlighted the significance of factors like vocal pitch, energy intensity, and the distribution of spectral components. Mel-frequency cepstral coefficients (MFCCs) are a popular feature set among audio descriptors because they closely resemble human perception of sound.

In order to prepare audio inputs for analysis under difficult real-world acoustic conditions, early approaches relied on traditional audio processing techniques like noise suppression, zero-crossing rate measurement, and short-term energy evaluation. Machine learning has become popular in this field due to the increasing need for automation.

Strong classification performance and robustness in low signal-to-noise scenarios have been shown by algorithms like Support Vector Machines (SVM), which is essential for use in crowded urban areas [7]. Kim and Chung's [7] comparison of SVM and Gaussian Mixture Models (GMM) revealed the advantages of each model in terms of accuracy and flexibilityThe resilience to noisy environments has additionally been improved by developments in feature extraction. To attain high recognition rates in various acoustic environments, Sharma et al. [12] proposed

the integration of MFCCs with chroma features and spectral descriptors. Ensemble learning methods have additionally been observed to perform well for complicated classification problems. In their research paper, Lopez et al. [9] showed that ensemble classifiers such as Random Forest and AdaBoost are more effective for emergency sound detection. To support this methodology, Singh et al. [13] proposed a real-time random forest-based classifier for achieving high accuracy and response rates suitable for security applications. Multimodal approaches are also becoming increasingly popular. Log spectrogram features, which are efficient for representing

both time and frequency domain information, were studied in the research paper by Patel and Kumar [11].

Liu et al. [8] emphasized the growing need for strong privacy-preserving mechanisms as surveillance systems continue to spread throughout public areas. In order to ensure that sensitive audio data can be processed efficiently without jeopardizing individual privacy, their extensive research looked at a number of cutting-edge methods, including federated learning and secure computation.

By putting these strategies into practice, the authors demonstrated that, in a world where surveillance is becoming more widespread, safer environments can be created while preserving the confidentiality and integrity of personal data.

Liu et al. [8] emphasized the growing need for strong privacy-preserving mechanisms as surveillance systems continue to spread throughout public areas. In order to ensure that sensitive audio data can be processed efficiently without jeopardizing individual privacy, their extensive research looked at a number of cutting-edge methods, including federated learning and secure computation.

By putting these strategies into practice, the authors demonstrated that, in a world where surveillance is becoming more widespread, safer environments can be created while preserving the confidentiality and integrity of personal data.

Zhang et al. [15] created a real-time classification model that employs several base learners, building on ensemble approaches. Their method strengthened its suitability for deployment in safety-critical infrastructure by improving the identification of emergency audio cues, such as screams

In order to identify distress signals, Miller et al. [10] evaluated the synergy between MFCCs and other spectral features.

According to their findings, a fused feature set greatly increases the precision of detecting emergency sounds produced by people in noisy settings.

III. METHODOLOGY

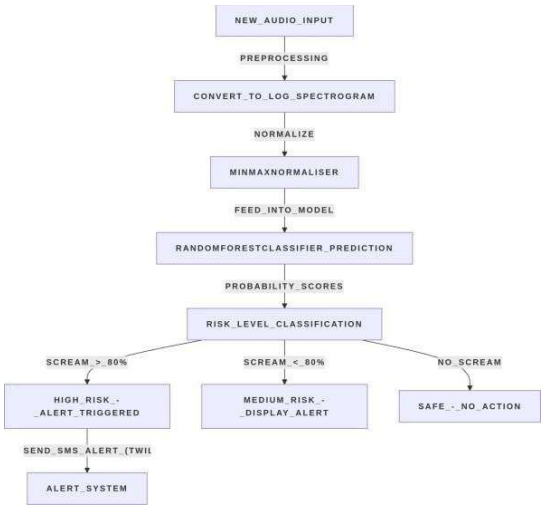

Fig. 1. ML-based audio processing and risk classification pipeline

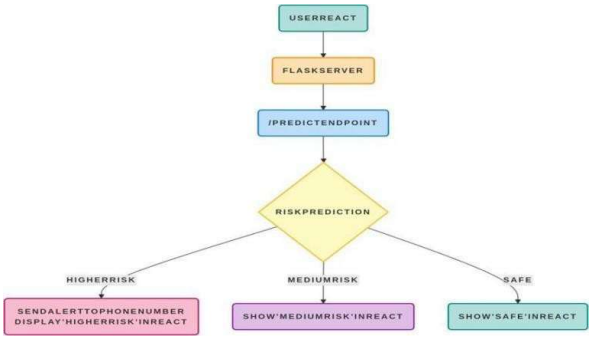

Fig. 2. alert dispatch flow

The proposed system is composed of two main modules: the machine learning-based audio processing pipeline and the frontend-backend integration for risk alerting, as shown in Figures 1 and 2 respectively.

In the audio processing and ML pipeline (Fig. 1), the system begins by receiving a new audio input, which undergoes several preprocessing steps such as trimming, resampling, and silence removal. The preprocessed audio is then trans-formed into a log-mel spectrogram to extract meaningful time-frequency domain features. These spectrogram features are normalized using a MinMax scaler to ensure uniform input scaling for the machine learning model. The normalized output is then passed to a pre-trained Random Forest classifier, which generates probability scores corresponding to different risk levels. Based on the probability score of a scream being detected, the system classifies the result into three categories: if the scream probability is greater than 80%, it is labeled as high risk, triggering an immediate alert; if the probability is less than 80%, it is considered medium risk, prompting a visual alert without escalation; and if no scream is detected, the situation is marked as safe, resulting in no further action. Frontend-backend integration facilitates user interaction and end-to-end system response (Fig. 2). A React-based frontend interface that records audio and transmits it to the backend server allows the user to interact with the application. This Flask-built server exposes a /predict API endpoint that handles incoming requests and launches the machine learning pipeline for risk classification and scream detection. Depending on the result from the backend, the system takes appropriate action: in a high risk scenario, an SMS alert is sent to a registered contact using Twilio, and the frontend displays a high-risk warning; for medium risk, a cautionary message is shown on the interface; and for safe conditions, a reassurance message is displayed, indicating that no danger has been detected.

A. Preprocessing

The PreprocessingPipeline processes audio files in a direc-tory by applying a sequence of operations to each file. Initially, it loads an audio file, then applies padding if required to meet a fixed length. Following this, the pipeline extracts a log spectrogram from the signal and performs normalization on the spectrogram. Finally, the normalized spectrogram is saved. In addition to these steps, the pipeline also stores the minimum and maximum values computed across all log spectrograms for later use.

- Loader: The Loader class is responsible for loading audio files with a specified sample rate, target duration, and mono channel conversion. It reads an audio file using the model librosa.load() function and returns the audio signal as a NumPy array.
- Padder: The Padder class provides methods to apply padding to one-dimensional arrays. It offers two functions: one for left padding and another for right padding. Both functions use numpy.pad() internally to add the required number of items to the array either at the beginning or the end.
- LogSpectrogramExtractor: The LogSpectrogramEx tractor class computes the log-scaled spectrogram from a given time-series audio signal. It performs a Short-Time Fourier Transform (STFT) on the signal, converts the magnitude spectrum into decibel units, and returns the resulting log spectrogram.

- MinMaxNormaliser: The MinMaxNormaliser class ap-plies min-max normalization to numerical arrays. The nor-malize method rescales the input array to a predefined range, typically [min, max]. The class also provides a denormalize method, which restores the normalized array to its original scale using the original minimum and maximum values.
- Saver: The Saver class is responsible for persisting the extracted features and the normalization metadata. It provides methods to save a spectrogram feature as a NumPy file and to store the min-max normalization values using the pickle format for future reference.

B. Model Training

The extracted features and the corresponding target labels are used to train the classification model. Loading the feature dataset (features.csv) is the first step in the training process. as well as the corresponding target labels (targets.csv). To properly validate the model's performance, the data is split into training and testing sets using a 70-30 split after it has been loaded.
The training subset of the data is then used to train a Random Forest Classifier.

C. Frontend Deployment

To make interacting with the system easier, a frontend interface that is both responsive and easy to use has been created. Users can upload audio files straight from their devices using this web-based interface. After the file is submitted, the frontend processes it and uses HTTP requests to send it to the backend server for additional analysis. Because of the design's emphasis on usability and simplicity, both technical and non-technical users can utilize it. The user is also given visual feedback that shows the risk prediction results returned from the backend and indicates the status of their submission. The front-end guarantees a seamless and easy experience to show the model's real-time applicability.

IV. Results and Discussion

A scream detection system that processes input audio files was used to evaluate the trained models. Every audio file is first normalized and converted into a log spectrogram. A trained classifier is then applied to this processed spectrogram in order to produce predictions. The system categorizes the risk level connected to the detected sound based on the model's output probability scores.
Table I presents the performance of different classification models. Random Forest outperformed other conventional and ensemble-based techniques, achieving the highest accuracy of 95.345% among all tested models. Neural networks and logistic regression both achieved an accuracy of about 90.7%. With an accuracy of 88.372%, Gradient Boosting and Support Vector Machines (SVM) both performed comparably.

TABLE I: PERFORMANCE OF INDIVIDUAL CLASSIFICATION MODELS

Model	Accuracy (%)
Support Vector Machines	88.372
Logistic Regression	90.698
Gradient Boosting	88.372
Random Forest	95.345
Neural Networks (1 layer, 1024 neurons)	90.700

The Neural Network did not perform better after one layer. Therefore, deeper architectures may not provide any additional advantages for smaller datasets in scream detection. The ensemble models' performance is summed up in Table II. The combined accuracy of Support Vector Machines and Logistic Regression was 90.698%. The accuracy remained almost unchanged at 90.6976% despite the inclusion of Random Forest. Furthermore, there was no discernible improvement when Random Forest and Neural Networks were combined. This occurs as a result of neural networks' inability to efficiently reduce variance when trained on smaller datasets.

TABLE II: PERFORMANCE OF ENSEMBLE CLASSIFICATION MODELS

Ensemble Model	Accuracy (%)
SVM + Logistic Regression	90.698
SVM + Logistic Regression + Random Forest	90.6976

The system uses a Random Forest model to detect screams from audio in real time. After converting the audio into a spectrogram, it classifies the risk as:
- High Risk: > 80% probability

- Medium Risk: ≤ 80% probability
- Safe: No scream detected

A. Testcases

The following test cases were created to ensure the scream detection system works well and is reliable:

- Handling Input Audio: The system should be able to handle WAV and MP3 audio files. Also, it needs to give the user useful feedback and be able to handle audio files that are incomplete or damaged correctly.
- Preprocessing and Spectrogram Generation: It must be checked that the system can always make log spectrograms from audio files. To do this, we need to look at the normalization process of the spectrogram data to make sure that it works the same way with different inputs.
- Model Performance: It's important to see how well the model predicts different types of screams, like high-pitched, muffled, and distressed ones. It is important to test the system with non-scream audio, like background or ambient sounds, to make sure it doesn't give too many false positives.
- System Integration: The backend model, the React frontend, and the Flask API endpoint at /predict must all work together perfectly. Testing must cover everything from audio input to risk prediction, including real-time interactions and UI updates.
- Alert and Notification System: To guarantee prompt and effective message delivery, the Twilio-based SMS alert system should be tested for high-risk forecasts. To find the closest police station and plan a response, location tracking needs to function properly. Lastly, once an alert has been resolved, it should be updated in the system and transferred to the "Solved" collection in Firestore.

B. Integration and Deployment

The integration and deployment process for the scream detection system is structured into several stages:

- Backend Integration: The Flask-based backend is in-tegrated with the Random Forest model. The '/predict' endpoint processes audio input, converts it to a log spectrogram, and feeds it to the classifier for scream detection.
- Frontend Integration: A React-based user interface allows users to upload audio files. The frontend communicates with the backend API to display the prediction result and the risk level.(Fig. 3)

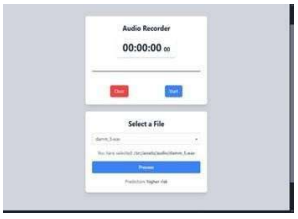

Fig. 3. Frontend Application

- Twilio API Integration: For high-risk scream detection, the backend triggers the Twilio API to send alert mes-sages, including the user's location, to the nearest police station (Fig. 4).

Fig. 4. Alert Message

- Firebase Integration: Alerts are stored in the Firestore database. The police panel can access these alerts and update their status as ”Solved.”.(Fig. 5)

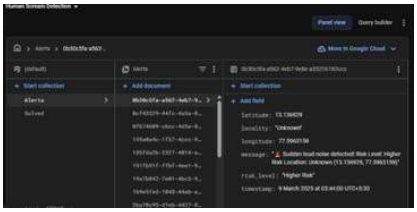

Fig. 5. Database

- Real-time Monitoring and Logging: Mechanisms for tracking performance and identify mistakes. Firebase Analytics monitors alert trends and user interactions.(Figure 6)

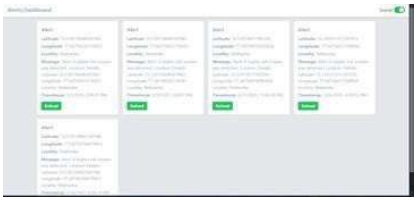

Fig. 6. police panel for logs

- Security Measures: Secure data transmission is ensured by end-to-end encryption. Access to the admin panel is limited by role-based access control and user authentication.

V. CONCLUSION

Using audio data, this study investigated several machine learning models for scream detection. The Random Forest classifier outperformed all other models, including ensemble approaches, with the highest accuracy of 95.345%, according to the results.

The experiments yielded several important conclusions. Neural networks' classification accuracy did not increase with the number of layers. In a similar vein, using ensemble models to combine several classifiers did not yield any appreciable performance improvements. On the other hand, all individual and combined methods were surpassed by the Random Forest model alone. Furthermore, the preprocessing pipeline was simplified because it was discovered that the Random Forest classifier did not require normalization.

A real-time scream detection system was developed using the trained model. After converting an input audio file into a log spectrogram and extracting relevant features, the system divides it into three risk levels based on the expected likelihood of a scream event: High Risk, Medium Risk, or Safe.

Because it offers a reliable and efficient real-time audio classification solution, the Random Forest-based scream detection system is a strong candidate for deployment in safety and security applications.

REFERENCES

[1] K. Ahmed, L. Smith, and R. Brown. Multi-modal fusion of audio and video for emergency situation detection in smart cities. IEEE Transactions on Smart Cities, 4(2):345–358, 2025.

[2] S. Birru, K. Reddy, and J. Thompson. A layered detection framework for audio-based security systems in smart cities. Smart Cities, 8(2):214– 229, 2025.

[3] Saikumar Birru, Ayeshafathima, Saideep Reddy G, Vaishnavi Dharma-puri, and Prasanna Kumari Kathi. Human scream detection and analysis for controlling crime rate. International Journal of Scientific Research in Engineering and Management (IJSREM), 2025. Published March 3, 2025. Available at:.

[4] Nardjes Bouchemal, Aissa Serrar, Yehya Bouzeraa, and Naila Bouch-memal. Scream to survive(s2s): Intelligent system to life-saving in disasters relief. page 414–430, Berlin, Heidelberg, 2019. Springer-Verlag.

[5] Gonc¸alo Cosme, Vaˆnia Tavares, Guilherme Nobre, Ce´sar Lima, Rui Sa´, Pedro Rosa, and Diana Prata. Cultural differences in vocal emotion recognition: a behavioural and skin conductance study in portugal and guinea-bissau. Psychological Research, 86(2):597–616, 2022.

[6] R. Garcia, P. Martinez, and L. Wong. Geolocation-based alert systems for acoustic emergency detection. In IEEE International Conference on Systems, Man, and Cybernetics (SMC), pages 1845–1850. IEEE, 2025.

[7] Sangwoo Kim and Yongdae Chung. Comparative analysis of svm and gaussian mixture models for human scream classification in surveillance applications. IEEE Transactions on Information Forensics and Security, 17:2612–2624, 2022.
[8] J. Liu, H. Wang, and S. Zhang. Privacy-preserving audio surveillance: Challenges and solutions for public safety applications. IEEE Transac-tions on Information Forensics and Security, 20(1):187–199, 2025.
[9] M. Lopez, D. Garcia, and C. Rodriguez. Ensemble methods for audio-based emergency detection: A comprehensive evaluation. Expert Systems with Applications, 234:121013, 2024.
[10] T. Miller, S. Johnson, and K. Davis. Mfcc and spectral features for human distress sound recognition. Computer Speech & Language, 80:101404, 2024.
[11] R. Patel and A. Kumar. Log spectrogram features for acoustic event classification in surveillance systems. Digital Signal Processing, 134:103571, 2024.
[12] A. Sharma, P. Gupta, and V. Kumar. Feature extraction techniques for acoustic event detection in urban environments. Applied Acoustics, 185:108424, 2023.
[13] R. Singh, A. Joshi, and K. Mehta. Random forest algorithms for real-time acoustic event classification in security applications. Pattern Recognition Letters, 178:71–79, 2024.
[14] L. Wang, S. Chen, and J. Taylor. Preprocessing techniques for environ-mental audio classification: A comparative study. Journal of the Audio Engineering Society, 71(3):145–158, 2023.
[15] X. Zhang, Y. Li, and P. Chen. Real-time audio classification for public safety applications using ensemble learning. IEEE Sensors Journal, 24(5):6723–6735, 2024.