# Acoustic Shield: Proactive Incident Prevention via Generative Audio AI

### 1. Conceptual Framework: Solving the Pre-Incident Data Scarcity Problem

Conventional traffic safety systems are fundamentally reactive; they detect incidents *after* they have occurred.1 The frontier in road safety is proactive prevention—identifying high-risk situations in the critical moments *before* a collision. The primary barrier to developing such systems is a severe lack of training data. Near-miss events, which are characterized by distinct audio signatures like screeching tires or sudden, panicked horns, are far more common than actual crashes but are rarely recorded and labeled systematically.3 This data gap makes it nearly impossible to train a robust audio classification model capable of reliably detecting an impending incident.

Acoustic Shield directly confronts this data scarcity problem through a novel application of generative AI. The core innovation is the creation of a massive, high-fidelity, and contextually relevant synthetic dataset of "pre-incident" audio events. This approach circumvents the impractical task of collecting real-world near-miss audio by generating it on demand.4

The system's intelligence stems from its data synthesis workflow. It begins by analyzing historical, publicly available crash data to identify the specific locations and environmental conditions where incidents are most common. This real-world context then guides the programmatic generation of thousands of descriptive text prompts (e.g., "tires skidding on a wet road with background traffic"). These prompts are fed into generative AI models to produce a rich, diverse, and realistic audio dataset. This synthetic data is then used to fine-tune a state-of-the-art audio classification model, effectively solving the "cold start" problem and enabling the development of a truly proactive safety system.6 The final product is an AI model that can listen to a real-time audio stream and raise an alert during the crucial moments of a pre-incident event, providing drivers or autonomous systems with precious milliseconds to react and potentially avert a collision.

### 2. Data Fusion and Generation Strategy: From Public Records to Synthetic Reality

The project's success hinges on the creation of a high-quality, context-aware audio dataset. This will be achieved by fusing public crash data with open-source audio libraries and augmenting them with generative AI.

**Core Data Sources (Free & Open-Source):**

* **Historical Incident Data (The "Where" and "Why"):**
  + **San Jose Open Data Portal - Crashes Data:** This is the foundational dataset for establishing context. It provides detailed records of crash events, including precise latitude/longitude, timestamps, weather conditions, roadway conditions, and the primary collision factor (e.g., speeding, unsafe turn). Analyzing this data will reveal chronic accident hotspots and the typical scenarios that lead to incidents in those locations.
* **Environmental Context Data (The "When"):**
  + **NOAA Weather API:** The National Weather Service provides free API access to historical weather data. By correlating the timestamp and location of a crash from the San Jose dataset with historical weather records, we can add crucial environmental details (rain, fog, etc.) to our generative prompts, making the synthetic audio more realistic.
* **Real-World Audio Libraries (The "Foundation"):**
  + **NINA-Dataset:** This dataset is invaluable as it contains hundreds of pre-labeled, in-vehicle audio clips of Crash (751 clips) and Tire skidding (186 clips), which form the perfect baseline for our target classes.
  + **MIVIA Road Audio Events Dataset:** This dataset provides high-quality recordings of tire skidding and car crashes already superimposed on realistic road background noise, which is ideal for training the model to operate in noisy environments.
  + **UrbanSound8K:** This classic dataset will provide essential audio clips for classes like car\_horn and engine\_idling, allowing the model to distinguish between specific alert sounds and benign background traffic.
  + **ESC-50 (Environmental Sound Classification):** This library is a source for a wide variety of background sounds, such as rain and wind, which can be programmatically mixed with our primary audio events to dramatically increase the diversity and robustness of the training data.
* **Generative Audio Tools (The "Augmentation Engine"):**
  + **ElevenLabs Text to Sound Effects:** This service's API can generate highly realistic sound effects from text prompts. The free tier offers enough generations for a hackathon proof-of-concept and is perfect for creating specific, nuanced sounds that are missing from the static datasets. Note that the free tier requires attribution and is for non-commercial use.
  + **Hugging Face Models (e.g., Bark):** As a powerful open-source alternative, models available on the Hugging Face Hub can be run locally to generate an unlimited number of audio clips from text, offering maximum flexibility. Models like Bark can even generate non-speech sounds, adding another layer of realism.

### 3. AI-Powered Audio Engine: Generation and Classification

The AI architecture is a two-stage pipeline: first, generating a robust dataset, and second, training a high-performance classifier.

**3.1. Stage 1: Context-Aware Synthetic Data Generation**

This stage transforms insights from public data into a machine-learning-ready audio dataset.

1. **Hotspot Analysis:** A Python script will analyze the San Jose crash data to identify geographic hotspots and common crash profiles. For example, it might find that a specific intersection has a high incidence of broadside collisions during rainy evenings.
2. **Prompt Engineering:** Based on these profiles, the script will programmatically generate thousands of descriptive text prompts. The prompts will be designed to create varied and realistic audio signatures of pre-incident events.
   * *Example Prompt (based on data):* `"Sound of a sedan's tires skidding for 1.2 seconds on a rain-slicked highway, followed by a loud, panicked car horn, with the ambient sound of moderate traffic in the background."\*
3. **Audio Generation & Mixing:** These prompts are fed into a generative audio API (like ElevenLabs) or a local Hugging Face model to create the core synthetic audio files. To further enhance realism, these generated sounds will be programmatically mixed with background noises (e.g., rain, wind, engine idling) sourced from the ESC-50 and UrbanSound8K datasets.

**3.2. Stage 2: Pre-Incident Audio Classification**

This stage uses the newly created dataset to train a model capable of real-time detection.

* **Model Selection:** A pre-trained **Wav2Vec2** model is the recommended architecture. This model, available from the Hugging Face Hub, is a state-of-the-art foundation for audio and speech recognition tasks and is well-suited for fine-tuning on a custom dataset.8
* **Training Workflow:** The model will be fine-tuned to perform a classification task on short audio segments (e.g., 1-2 seconds). The key classes will be:
  + Normal\_Traffic: Benign sounds like engine noise, wind, and normal driving.
  + Pre\_Incident\_Alert: High-urgency sounds like tire skidding, hard braking, and panicked horns.
  + Collision\_Impact: The sound of an actual crash.
* **Performance:** The goal is to achieve high precision and recall for the Pre\_Incident\_Alert class, ensuring the system can reliably detect warning signs without generating excessive false positives.

### 4. Implementation Architecture on AWS

This architecture is designed for a hackathon, prioritizing rapid development and leveraging the specified AWS services for a powerful, end-to-end ML pipeline.

* **Data Lake and Staging:**
  + **Amazon S3:** S3 will serve as the central data repository. It will store the raw open-source audio datasets (NINA, MIVIA, etc.), the contextual data from the San Jose portal, and, most importantly, the final, augmented training dataset of labeled .wav files generated by the AI.
* **Data Generation and Orchestration:**
  + **Amazon EC2:** A general-purpose EC2 instance will act as the primary "worker" node. It will host the Python scripts responsible for:
    1. Downloading and preprocessing the open-source audio and crash data.
    2. Programmatically generating the text prompts for the generative AI.
    3. Calling the generative audio APIs (ElevenLabs or a model hosted via Hugging Face) to create the synthetic sound files and uploading them to S3.
* **Model Training and Deployment:**
  + **Amazon SageMaker:** This is the core of the machine learning pipeline.
    1. **Fine-Tuning:** A SageMaker Training Job will be launched to fine-tune the Wav2Vec2 model. The training job will be configured to use the augmented dataset stored in S3 as its input.
    2. **Deployment:** The best-performing model will be deployed as a **SageMaker real-time inference endpoint**. This provides a scalable, low-latency API that can accept a raw audio chunk and return classification probabilities in milliseconds.11
* **Generative AI for Incident Reporting (Advanced Feature):**
  + **Amazon Bedrock:** To showcase a more advanced, agentic workflow, Bedrock can be used to create an "AI Incident Analyst."
    1. When the SageMaker model detects a Collision\_Impact event with high confidence, the application can trigger an **Agent for Amazon Bedrock**.
    2. The agent will receive the event data (timestamp, classification, confidence score) and a prompt instructing it to generate a structured incident report. Using a model like Anthropic's Claude 3.7 Sonnet, it can reason over the inputs and produce a JSON object suitable for logging or a natural language summary for a human operator, demonstrating a complete, automated incident detection-to-reporting pipeline.12

### 5. Hackathon MVP & Quantifiable Impact

The Minimum Viable Product (MVP) will be a real-time desktop application that demonstrates the core functionality of the Acoustic Shield system.

**MVP Scope:**

The MVP will be a Python application with a simple GUI that:

1. Captures audio from the system's microphone or simulates an audio stream from a pre-recorded file containing a mix of sounds.14
2. Continuously sends short audio chunks to the deployed SageMaker inference endpoint.
3. Displays the live classification probabilities for each class (Normal\_Traffic, Pre\_Incident\_Alert, Collision\_Impact).
4. Triggers a prominent visual and audible alert on the screen whenever the Pre\_Incident\_Alert class is detected with a confidence score above a set threshold (e.g., 90%).
5. (Stretch Goal) When a Collision\_Impact is detected, it will make a call to the Bedrock agent and display the generated incident summary.

**Quantifiable Impact:**

* **Saving Lives (Safety):** The primary impact is the system's ability to provide a pre-incident warning. The demonstration can emphasize that even a few hundred milliseconds of advance warning can be the difference between a near-miss and a fatal collision. Research shows that acoustic detection can be nearly instantaneous (<1.4 seconds), far faster than traditional methods.16
* **Innovation (Efficiency):** The project's core technical achievement is solving a critical data scarcity problem. The presentation should highlight the innovative use of generative AI to create a bespoke, high-quality training dataset, a process that would otherwise be impossible or prohibitively expensive.4 This demonstrates a powerful, scalable method for building specialized AI models in data-constrained environments.

### Table 1: Open-Source Data Acquisition Plan

| **Data Category** | **Dataset/Source** | **Key Information / Audio Classes** | **Purpose in Project** |
| --- | --- | --- | --- |
| **Incident Context** | San Jose Open Data Portal - Crashes | Lat/Long, Timestamp, Weather, Collision Type | Identify real-world crash hotspots and scenarios to guide synthetic data generation. |
| **Environmental Context** | NOAA Weather API | Historical weather conditions (rain, fog, etc.) | Add realistic environmental context to generative AI prompts. |
| **Core Audio Events** | NINA-Dataset 17 | Crash, Tire skidding, Harsh acceleration | Provide a high-quality baseline of real-world pre-incident and incident audio clips. |
| **Core Audio Events** | MIVIA Road Audio Events | tire skidding, car crashes (in noisy backgrounds) | Train the model to be robust against background noise. |
| **Supporting Audio** | UrbanSound8K | car\_horn, siren, engine\_idling | Provide distinct alert sounds and common, benign traffic noises for classification. |
| **Background Ambience** | ESC-50 Dataset 18 | rain, wind, thunderstorm | Source of environmental sounds to be mixed with other clips for data augmentation. |

### Table 2: AWS Service Architecture for Acoustic Shield

| **Component** | **AWS Service** | **Role in Architecture** | **Justification for Hackathon** |
| --- | --- | --- | --- |
| **Data Storage** | Amazon S3 | Central data lake for all raw and generated audio datasets. | Provides a simple, scalable, and easily integrated storage solution for the ML pipeline. |
| **Data Generation** | Amazon EC2 | Hosts Python scripts for data analysis, prompt generation, and calling generative AI APIs. | A flexible compute environment for running the data preparation and synthesis workflows. |
| **Model Training** | Amazon SageMaker | Manages the end-to-end process of fine-tuning the Wav2Vec2 model on the custom dataset. | Simplifies the training process with managed infrastructure and pre-built deep learning containers. |
| **Real-Time Inference** | SageMaker Endpoint | Deploys the trained model as a low-latency, real-time API for audio classification. | Provides a production-ready, scalable endpoint for the live demonstration application. |
| **AI Reporting** | Amazon Bedrock | Powers an "AI Incident Analyst" agent to generate structured reports from model detections. | Demonstrates an advanced, agentic workflow and leverages a key generative AI service. |

#### Works cited

1. AutoAlert: Automated Acoustic Detection of Incidents - ROSA P, accessed October 24, 2025, <https://rosap.ntl.bts.gov/view/dot/39341>
2. Slow down, you're going too fast - Smart Cities World, accessed October 24, 2025, <https://www.smartcitiesworld.net/news/slow-down-youre-going-too-fast-1641>
3. Getting Started | INRIX documentation, accessed October 24, 2025, <https://docs.inrix.com/>
4. Role of Generative AI in Data Augmentation: Models, Use Cases, and Benefits - upGrad, accessed October 24, 2025, <https://www.upgrad.com/blog/the-role-of-generativeai-in-data-augmentation/>
5. Why Data Augmentation Is Key for Vision, Language, and Audio AI | DigitalOcean, accessed October 24, 2025, <https://www.digitalocean.com/community/tutorials/data-augmentation-vision-language-audio-research>
6. Generative AI for Data Augmentation in Small Biomedical Datasets - Frontiers, accessed October 24, 2025, <https://www.frontiersin.org/research-topics/72904/generative-ai-for-data-augmentation-in-small-biomedical-datasets>
7. [2501.13372] Generative Data Augmentation Challenge: Zero-Shot Speech Synthesis for Personalized Speech Enhancement - arXiv, accessed October 24, 2025, <https://arxiv.org/abs/2501.13372>
8. Fine-tuning for Audio Classification with Transformers - Colab, accessed October 24, 2025, <https://colab.research.google.com/github/huggingface/notebooks/blob/master/examples/audio_classification.ipynb>
9. Audio classification - Hugging Face, accessed October 24, 2025, <https://huggingface.co/docs/transformers/main/tasks/audio_classification>
10. mkunes/w2v2\_audioFrameClassification: wav2vec2 audio classification for prosodic boundary detection and other tasks - GitHub, accessed October 24, 2025, <https://github.com/mkunes/w2v2_audioFrameClassification>
11. Amazon SageMaker Model Dashboard, accessed October 24, 2025, <https://docs.aws.amazon.com/sagemaker/latest/dg/model-dashboard.html>
12. Exploring Claude 3.7 Sonnet's Hybrid Reasoning on Amazon Bedrock | AWS Builder Center, accessed October 24, 2025, <https://builder.aws.com/content/2tZDHRTW9tn83oxVR7p2PTDECU2/exploring-claude-37-sonnets-hybrid-reasoning-on-amazon-bedrock>
13. Tool use - Amazon Bedrock - AWS Documentation, accessed October 24, 2025, <https://docs.aws.amazon.com/bedrock/latest/userguide/model-parameters-anthropic-claude-messages-tool-use.html>
14. aiXander/Realtime\_PyAudio\_FFT: Realtime audio analysis in Python, using PyAudio and Numpy to extract and visualize FFT features from streaming audio. - GitHub, accessed October 24, 2025, <https://github.com/aiXander/Realtime_PyAudio_FFT>
15. How to do real-time audio signal processing using python, accessed October 24, 2025, <https://python-forum.io/thread-21674.html>
16. Instantaneous Incident Detection System Based on Analysis of Acoustic Signal from Crash and Skid in Tunnel - The Open Transportation Journal, accessed October 24, 2025, <https://opentransportationjournal.com/VOLUME/12/PAGE/344/FULLTEXT/>
17. axa-rev-research/NINA-Dataset: Script to create the crash sound dataset - GitHub, accessed October 24, 2025, <https://github.com/axa-rev-research/NINA-Dataset>
18. ESC-50: Dataset for Environmental Sound Classification - GitHub, accessed October 24, 2025, <https://github.com/karolpiczak/ESC-50>