

# Gunshot Detection Prototype Report

---

Author: Soham Sahare  
ASU Smart City Cloud Innovation Center

# Table of Contents

<b>Introduction</b> .....	2
<b>ASU Smart City Cloud Innovation Center Powered by AWS</b> .....	2
<b>Problem Description</b> .....	2
<b>Architecture Diagram</b> .....	3
Bills of Services .....	4
<b>Hardware</b> .....	6
Major Components .....	6
End device .....	7
Cost of components .....	7
<b>Algorithm</b> .....	7
Machine Learning Model .....	8
<b>Implementation Outcomes</b> .....	10
<b>Lessons Learned</b> .....	11
<b>Appendices</b> .....	13
Appendix A: Project Visualization .....	13
Appendix B: Credits .....	13
Appendix C: License .....	14
Appendix D: Bill of Materials .....	14
Appendix E: Machine Learning Model Filter .....	16

## Introduction

This purpose of this document is to provide an overview of Arizona State University's Smart City Cloud Innovation Center (ASU CIC) engagement with City of Phoenix Police Department to develop a low-cost Gunshot Detection Prototype. It illustrates how the ASU CIC architected, implemented and tested the solution. The outcomes of the engagement are available open source on the [ASU Smart City Cloud Innovation Center Website](#) for anyone to use and scale.

## ASU Smart City Cloud Innovation Center Powered by AWS

Also known as the ASU CIC, the ASU Smart City Cloud Innovation Center employs Amazon's innovation processes, cloud expertise, and global solution platforms to solve pressing community and regional challenges.

The center is designed as part of a long-term collaboration between ASU and AWS to improve digital experiences for smart city designers, expand technology alternatives while minimizing costs, spur economic and workforce development and facilitate sharing public sector solutions within the region.

The mission of the Smart City Cloud Innovation Center is to drive Innovation Challenges that materially benefit the greater Phoenix metro area and beyond, by solving pressing community and regional challenges, using shareable and repeatable technology solutions from ideation through prototype, as a service for the greater human good.

## Problem Description

On average, over 2,000 gun incidents occur in the City of Phoenix each year and this rate of gun violence is increasing according to [EveryStat for Gun Violence](#). Gun violence costs Arizona taxpayers \$253.2 million each year. According to the [CDC](#), in 2019, 1,136 people died of firearm injury in Arizona. Getting the right information, to the right officers, at the right time is critical to identifying, catching, and prosecuting offenders.

Incidents of gun violence are either reported by a 911 emergency call or by the officer's present in the area. The time delay between the guns firing and the citizens reporting them to the police can be costly and usually lead to the perpetrators running away from the scene or even worse, causing more harm. Existing Gunshot detection systems eliminate this lag time, automate notifications, and enable the police to arrive at the site more quickly than before. The cost of gunshot detection systems, however, prevents the wide scale deployment, which limits effectiveness.

The ASU CIC and the Phoenix Police Department worked together to reimagine how to develop a solution that would be inexpensive, easy to deploy extensively, and turn gunshot detection alerts into identified offenders.

There 3 goals for the ASU CIC's Gunshot Detection Prototype implementation:

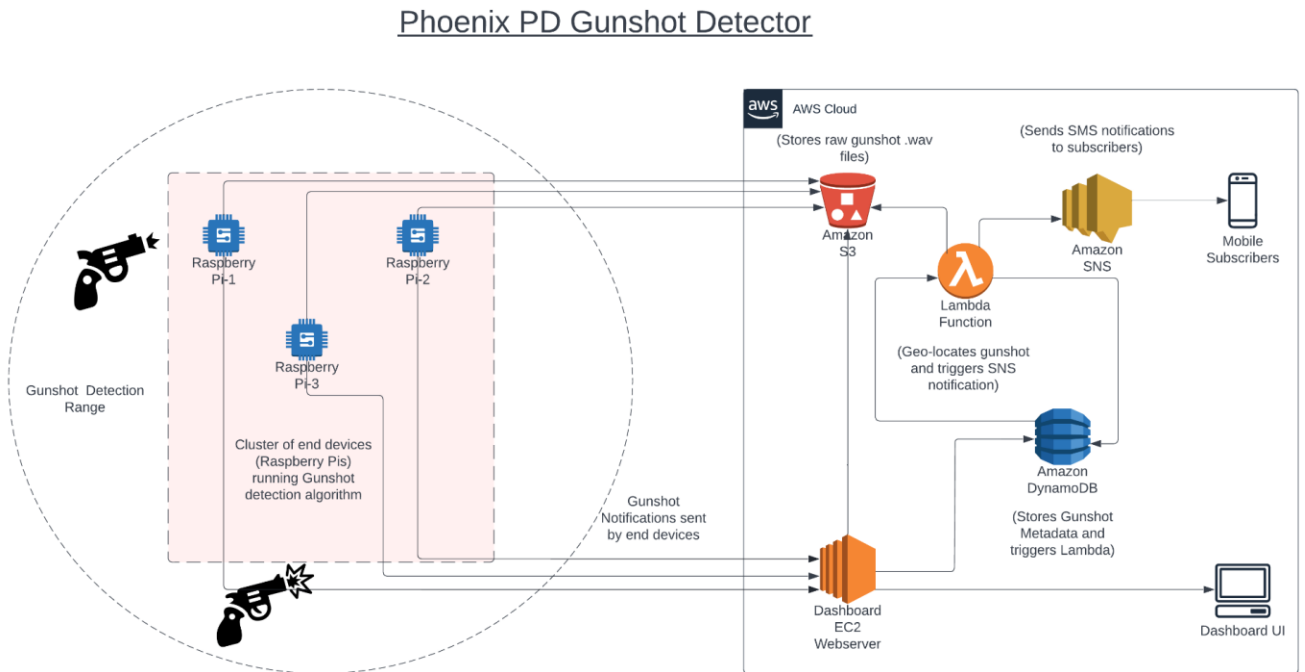
1. Primary Goal
  - a. Develop a low-cost Audio Detection System that differentiates between Gunshot and other detected sounds.
2. Secondary Goals
  - a. Evaluate range and triangulation of devices.
  - b. Environmental impact on hardware.

## Architecture Diagram

The ASU CIC Team and the City of Phoenix Police Department deployed ten prototype devices in central Phoenix to listen for peculiar sounds and detect gunshots using deep machine learning models. The ASU CIC Gunshot Detection project utilizes a combination of [Gunshot detection algorithm for Raspberry Pi](#) and AWS services to capture audio events, identify gunshots and trigger notifications.

Once a gunshot is detected, the gunshot clip is stored in AWS S3 (Simple Storage Service), the timestamp and device-id is stored in AWS DynamoDB and a notification is sent to the server, which waits for a second for other devices to listen and report the same sound clip. The information reported by multiple devices is collated to triangulate the approximate location for the source of the gunshot. The officers that are subscribed to the system are then notified with a link to the Maps location of the gunshot source which can be used to respond immediately.

Figure 1: Gunshot Detection Prototype Architecture Diagram



## Bills of Services

The ASU CIC Team uses several AWS Cloud Services throughout the implementation of the Gunshot Detection Prototype. The details of how these services are used, their cost and estimated average monthly costs are outlined below in Table 1. Note, the costs of AWS Cloud Services vary with time and regions. These prices might not reflect the actual price in the future, in order to get the latest estimate of cost of resources, refer to [AWS Cost Calculator](#).

Table 1: AWS Cloud Service Cost of Ownership

AWS Cloud Service	Utilization	Cost/Month
<a href="#">Amazon SNS</a>	SNS is used to notify officers with the location of the gunshot by sending them a text message using a low-volume Dedicated 10-digit long codes (10DLC) <ul style="list-style-type: none"> <li>Monthly \$2 for each approved low-volume 10DLC number (we need 1)</li> <li>Each text notification at \$0.00581 Base Price + \$0.00266 Carrier fee for a total of \$0.00847</li> <li>For the most active month, July 2022, we sent 4000 SMS at a cost of \$33</li> <li>For worst case of 10,000 SMS per month, monthly cost is \$84.7</li> </ul>	\$84.7
<a href="#">Amazon DynamoDB</a>	DynamoDB is a database used for storing intermediate and final results. Read and Write	\$1.51

	<p>requests cost \$0.25 and \$1.25 per million requests respectively.</p> <ul style="list-style-type: none"> <li>For storage, the first 25 GB is free per month and we never cross the 25 GB limit (we used a maximum of 3 MB) and hence we are not charged for storage.</li> <li>For backup storage, we don't quite use it yet but in case you wish to backup the results it would cost \$0.1 per GB per month. For backing up the results of the entire setup for a maximum of 1 GB would cost a maximum of \$0.1 per month.</li> <li>Total cost of using DynamoDB ranges from \$0 to \$1.51 per month</li> </ul>	
<a href="#">AWS Lambda</a>	<p>Lambda functions are short lived functions meant to process tasks on demand and die down after processing. They run for a maximum of 15 minutes and they are charged on memory used per second, for a price of \$0.0000166667 per GB of memory used per second.</p> <ul style="list-style-type: none"> <li>We have 4 lambda functions that run at 128 MB of memory for a maximum duration of 3 seconds.</li> <li>Max monthly forecast for Lambda function would be \$0.0000166667 per month</li> </ul>	\$0.0000166667
<a href="#">Amazon EC2</a>	<p>EC2 instance is an Infrastructure as a Service provided from AWS which we use to host the website. EC2 charges on an hourly rate. Until we move the website to S3 for a static hosting cost, the cost for running EC2 are as follows.</p> <ul style="list-style-type: none"> <li>We use the <i>t2.micro</i> instance, which qualifies for 750 hours of free tier usage for the first 12 months, in case you have already used the free tier previously, it costs \$0.0104 per hour, we run the instance 24/7, so monthly cost of the instance will come as <math>\\$0.0104 * 24 * 30 = \\$7.488</math> per month</li> </ul>	\$7.488
<a href="#">Amazon EC2-EBS Storage</a>	<p>Every EC2 instance uses EBS storage for persisting data on an non-ephemeral device, these block storage are charged at \$0.08 per GB used per month.</p> <ul style="list-style-type: none"> <li>For a standard t2.micro instance, we need 24GB storage per month, which would make the monthly cost as <math>\\$0.08 * 24 = \\$1.92</math> per month</li> </ul>	\$1.92
<a href="#">Amazon Cloudwatch</a>	<p>In build alarm system which triggers whenever the DynamoDB needs scaling out and increases the number of instances because it cannot handle the incoming requests.</p> <ul style="list-style-type: none"> <li>Average cost of these CloudWatch Alarms come to around \$4.20 per month</li> </ul>	\$4.20
<a href="#">Amazon Pinpoint</a>	<p>AWS Pinpoint is a service which assigns us a US Toll-free number that sends SMS to officers regarding the position of the estimated gunshot detected by our system.</p>	\$2

<a href="#">Amazon S3</a>	<ul style="list-style-type: none"> <li>The flat cost comes as \$2 per month</li> </ul> <p>S3 is a storage service that is used to save the gunshot audio files in the bucket.</p> <ul style="list-style-type: none"> <li>S3 charges \$0.023 for the first 50 TB each month. To put this number in perspective, each audio file we save is around 100 KB, to exceed 50 TB we would need to store around 500 Million audio files.</li> <li>S3 charges \$0.0005 for every 1000 new files saved in to S3, looking at our use case of approximately 10,000 requests a month, we would be charged \$0.005 per month</li> <li>S3 charges \$0.0004 for every 1000 files fetched from S3, looking at our use case of approximately 10,000 requests per month, we would be charged \$0.004 per month</li> <li>Total cost of S3 → Storage + PUT + GET = \$0.032 per month</li> </ul>	<p>\$0.032</p>
<p>Total</p>		<p>\$101.85</p>

## Hardware

ASU CIC's prototype implementation of the Gunshot Detection Project consists of both Software and Hardware components. The specific details of Hardware elements and their costs are listed in the section below.

### Major Components

There are three required components to develop a Gunshot Detection Prototype:

1. Development Board
  - a. The requirements of the project necessitated two parallel threads running consistently – one to continuously listen to audio and the other running the Machine Learning Algorithm to identify gunshots. While both Raspberry Pi-4 and Arduino devices were tested during the development of the project, only Raspberry Pi-4 supported multi-thread processing and was therefore selected.
2. Microphone
  - a. The goal of the project was to find inexpensive microphones with omnidirectional hearing capabilities with at least 100 ft range. After testing multiple microphone devices from online sellers, SiZHENG microphones performed the best.
3. Connectivity Device
  - a. For hotspot, the project would require high speed internet connectivity through SIM access and can be easily connected to the Raspberry Pi. The hotspot should have support for never sleeping while connected and be small enough to fit in the casing while being inexpensive. The ZTE Max WiFi hotspot checked all boxes.

## End device

In order to develop a functional Gunshot Detection Prototype, the ASU CIC utilized the following components to build the edge device:

- Microphone
- Wi-Fi Hotspot using a SIM
- Power supply cord for Light Poles
- Raspberry Pi for processing
- Casing to house the components
- Waterproofing gel to encase the microphone
- Plastic encasing to waterproof the microphone

## Cost of components

Refer to [Appendix D: Bill of Materials](#) for a detailed cost breakdown of every component used to create the end devices. The development team and the City of Phoenix Police Department installed 10 end devices in City of Phoenix. The total hardware cost of these devices is \$3982.

Including the recurring AWS Cloud Services costs, the total operating cost of implementing 10 Gunshot Detection Prototype end devices for a year is shown in Table 2.

Table 2: Total Cost of Gunshot Detection Prototype

Component	Cost/Year
10 Gunshot Prototype End Devices	\$3982
AWS Cloud Services	\$1222.20
<b>Total</b>	<b>\$5204.20</b>

## Algorithm

Existing commercial Gunshot Detection Systems require human verification of algorithm results prior to notifying authorities. This process adds additional lag time to the verification workflow. One of the goals of the ASU CIC’s Gunshot Detection Prototype is to eliminate the extra approval steps and reduce the time for notifications to be sent.

Four factors were considered to improve workflow efficiency while designing the ASU CIC’s Gunshot Algorithm:

1. Edge device (machine learning & sound capture)
  - a. Installed devices listen continuously and detect gunshots using a machine learning model. The edge devices are capable of listening to gunshots in a radius of ~100 feet, this can be improved greatly with the use of better microphones which come at a higher cost or by building a custom microphone array.



2. Cloud (machine learning model)
  - a. While the Gunshot Detection Model is trained to pick up gunshots, there are cases where it picks up on similar sounds such as fireworks, car backfire, flat tires, rainfall, etc. To mitigate this limitation, the ASU CIC added a filter machine learning model that is trained to pick up gunshots among other similar sounds.
3. Location determination (triangulation)
  - a. If a gunshot is still detected after these two models, then gunshot location is determined by using multiple end devices to triangulate a Latitude and Longitude of the event.
4. Notification of gun shot
  - a. The triangulated gunshot location is sent to the server, which then notifies all officers/ point of contacts that are subscribed to the gunshot detection system.

## Machine Learning Model

The following techniques are employed for preprocessing and extracting features from the audio files.

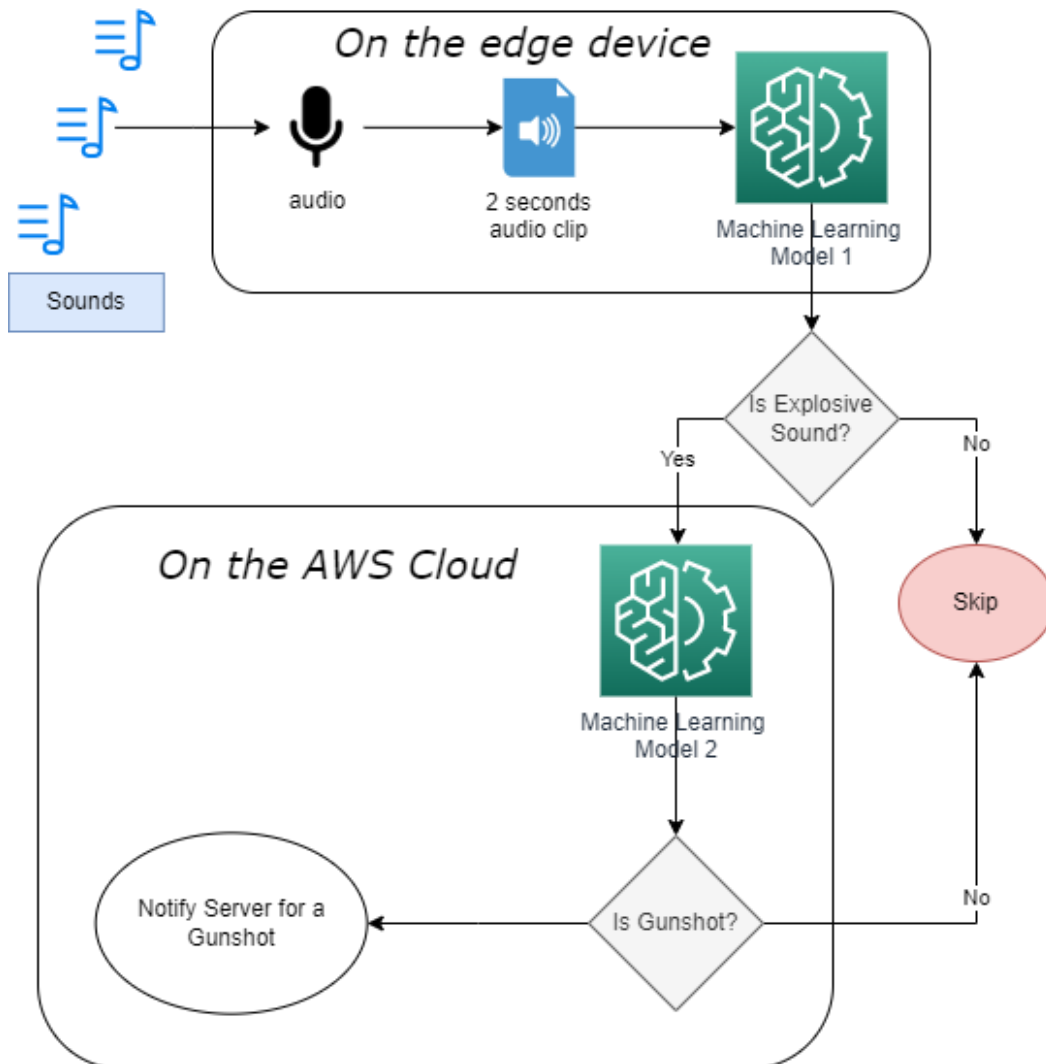
1. Elimination of noise, to make the audio clear using Short-time Fourier Transform and Convolved Fast Fourier Transformation
2. Padding with silence to make it of equal length
3. Making the audio files Stereo channel by duplicating the wave file (if mono)
4. Spectral Roll off feature
5. Spectral Centroid feature
6. Mel Frequency Cepstral Coefficient features
7. Mel Spectrogram Analysis features

For the Hyper-parameter and ML model setup, refer to [Appendix E: Machine Learning Model Filter](#).

As the device is constantly listening to the sounds in the environment, it makes a 2 second audio clip, sends it to the first Machine learning model to determine if the sound is similar to an explosion sound. Many different sounds share this feature, namely, car backfiring, fireworks, raindrops falling on the device, balloons popping, etc. All sound clips which do not exhibit this feature are first eliminated. Once a sound clip that is similar to an explosion is identified, the audio clip is sent to the cloud and run against the next machine learning model which is trained to identify Gunshots among other types of loud sounds. If the model categorizes a sound clip as gunshot, the information of the device sent previously is used to trigger the latter half of the system. The general flow of the machine learning model is described below in Figure 2.

Figure 2: Gunshot Detection Model

## Gunshot Detection Model



## Implementation Outcomes

The ASU CIC with the support of City of Phoenix Police Department installed 10 devices in central Phoenix over a 3-month period. At the end of the testing cycle, only 3 of the 10 installed devices were still functional.

After uninstalling the hardware, further analysis brought additional insights to light:

1. Even in harsh conditions such as high temperatures and heavy rain in the middle, the devices survived the conditions quite well
  - a. 9/10 devices survived the heavy rains and there was not even a single drop that leaked inside.
  - b. 9/10 devices had functioning microphones, even though there were significant burns on the top of the microphones, they did function well.
2. A common trend in the device condition was that most, if not all microphones, could not bear the constant heat exposure and had their upper protective lining burnt off and melted inside the microphone, compromising the mic quality.

Out of the 7 devices that lost connectivity with the AWS cloud, there were three reasons for the malfunction:

1. Raspberry Pis for 2/10 devices burnt out and don't turn on after uninstall. This could be due to the heat or other unknown factors.
2. One device was flooded completely because of a tear on the water proof sealing. Even though the device did not function after that, the fuse protected the components and once dried up, most components worked well.
3. A few hotspot devices lost internet connection and never connected back to the AWS cloud rendering the end devices dead.

On full analysis of the Gunshot Detection Prototype implementation, the Primary and Secondary goals of the project were achieved as outlined in Table 3. A low-cost audio detection framework was developed and the infrastructure to improve the algorithm was designed. While the limited microphone range impacted the distance between devices, this can easily be mitigated by customizing a microphone array with a longer range. Furthermore, the devices functioned reasonably well against the Phoenix summer heat and monsoon showers with only one device stop functioning due to flooding. While a longer-term installation would need to account for winterization as well as heat mitigation, the initial implementation allowed the ASU CIC Team to test the goals of the prototype without any issues.

Table 3: Goal Analysis

	Solution Goal	Outcome	Challenges
Primary Goal	Develop a low-cost Audio Detection System that differentiates between	Successfully developed a framework for low cost audio detection system that	Training of edge and cloud models can be improved with the use

	Gunshot and other detected sounds.	could be iterated and improved further	of different real-world audio samples
Secondary Goal	Evaluate range and triangulation of devices	Range is around 30 ft and triangulation implementation is rudimentary	Improvement needed in triangulation (cloud-assisted GNSS, Wi-Fi scan, cellular triangulation)
	Environmental impact on hardware	<ul style="list-style-type: none"> <li>• Edge devices were constructed and packaged by students with easy to access and cost-efficient hardware devices</li> <li>• 10 devices were successfully deployed on various lamp posts</li> </ul>	<p>Edge devices can be made smaller, more efficient and weather-proof</p> <ul style="list-style-type: none"> <li>• One device was flooded</li> <li>• One other device's microphone went out of service</li> <li>• Five devices had trouble with internet communication</li> </ul>

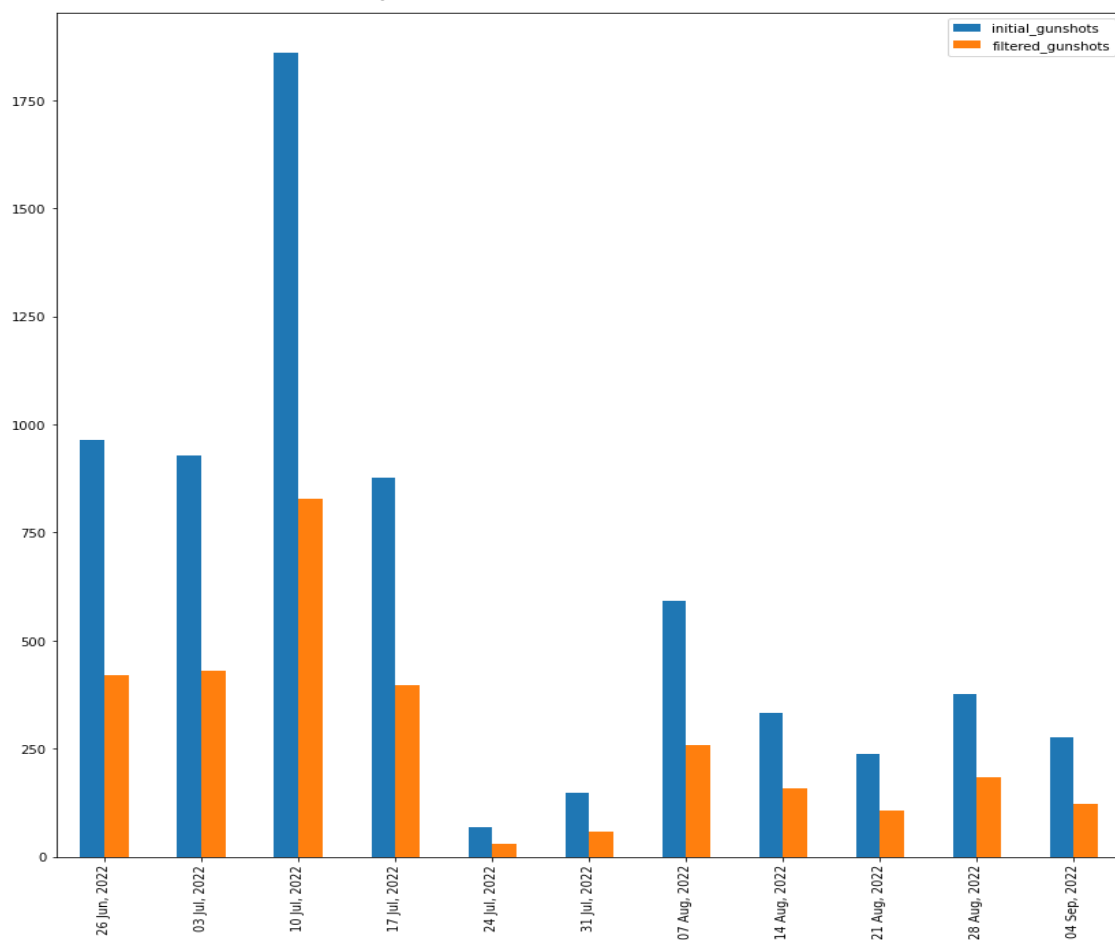
## Lessons Learned

- **A good quality microphone** is the crux of the prototype. It should be small enough to fit inside the casing while also having high quality sound recording capabilities, omnidirectional, high range, and inexpensive. Such a microphone is hard to find in the market, a better solution is to build your own microphone array using breadboards and electret microphones. These arrays are highly customizable and can carry as many microphones to enhance hearing capabilities.
- **Making a small, robust, weather proof, and inexpensive casing** is very important as end devices should fit around poles or high reaching structures easily and not spotted easily. They should be capable of handling high heat, rainfall, snowfall, birds, etc so the devices does not lose power/ fail randomly.
- **Making the device heat resistant** is crucial for places like Arizona, as devices overheat very easily. Painting the device white, using heat resistant semiconductor casings can be a few ways to employ this. Similarly, the devices would need to be made rain and snow resistant according to the areas it is housed in.
- Because the devices are put on high-rise structures, it is extremely difficult to remove them and make alterations. There should be **methods for remotely pushing updates to the device and enabling remote troubleshooting** to prevent uninstalling devices every time changes need to be made. AWS IoT Greengrass can be utilized in the future.
- Internet connectivity is a requirement since the devices need to interact with the cloud. The project employed hotspot devices with SIM cards for continual internet connection; these hotspots must be properly installed and turned on before the

device is put at its designated location. Instead, **less expensive communication technologies, such as LoRaWAN** can be leveraged, which is also scalable.

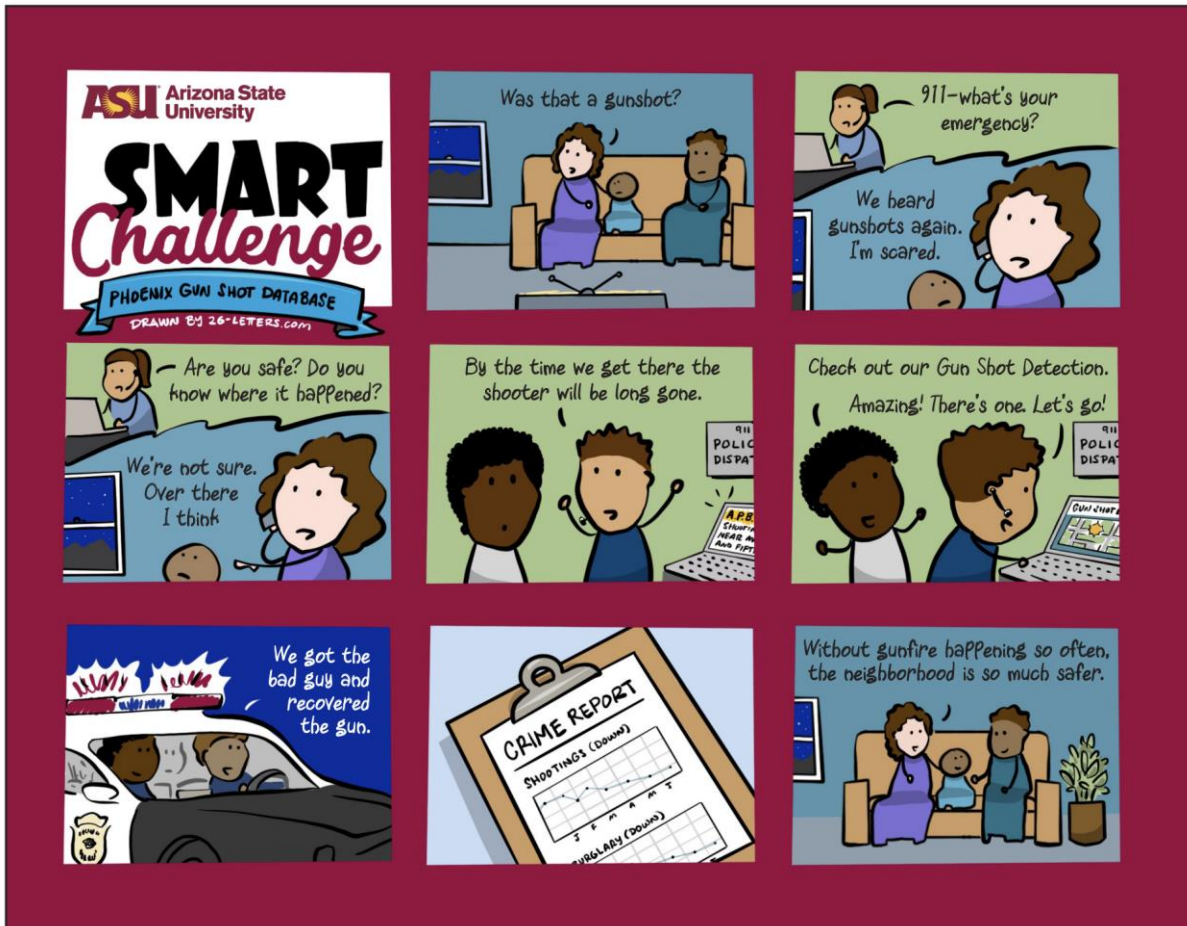
- A well-trained collection of machine learning models that were trained using the audio captured by the devices to better recognize gunshots and distinguish non-gunshots as previously mentioned. Due to variances in the recording technology used in the instances, using external datasets of gunshots and non-gunshots does not yield satisfactory performance on recorded audio files; therefore, gather **more real-world sounds and keep refining the model**. In Figure 3 below, the initial gunshots detected (which had significant false positives) were eliminated after the second ML model filter was added.

Figure 3: Filtered Gunshot Detection Algorithm



## Appendices

### Appendix A: Project Visualization



### Appendix B: Credits

Phoenix-pd-gunshot-detection (<https://github.com/ASUCICREPO/phoenix-pd-gunshot-detection>) is an open source software. The following people have contributed to this project.

#### Student Developers

- Krishna Teja Kalaparty
- Risabh Raj
- Soham Sahare
- Sameet Kumar
- Yug Gulati
- Harshita Jain

#### ASU Cloud Innovation Center Team

- Sr. Program Manager, AWS: Jubleen Vilku

- Digital Innovation Lead, AWS: Jason Whittet
- Solutions Architect, AWS: Arun Arunachalam
- General Manager, ASU: Ryan Hendrix

This project is designed and developed with guidance and support from the ASU Cloud Innovation Center and the City of Phoenix, Arizona teams.

## Appendix C: License

This project is distributed under the Apache License 2.0

(<https://github.com/ASUCICREPO/phoenix-pd-gunshot-detection/blob/main/LICENSE>)






## Appendix D: Bill of Materials

The specific cost details of the Gunshot Detection Prototype hardware components is split into Tables 4, 5 & 6 below. They outline the cost of casing, electronic components and ancillary costs of developing the prototype.

Table 4: Gunshot Detection Prototype Casing

Hardware Component	Cost
<a href="#">Waterproof Cable Strain - 20 pieces</a> 	\$15.80
<a href="#">Hose Clamps - 39 Ft cable + 13 clamps</a> 	\$24.99
<a href="#">Waterproof &amp; Dustproof Case - 10 piece</a> 	\$209.90

Table 5: Gunshot Detection Prototype Electronic Components

Hardware Component	Cost
<p data-bbox="204 331 815 365"><a href="#">Streetlight Mounted Auxiliary Power Adapter - 10 piece</a></p>  <p data-bbox="272 611 411 633">24/7-SF Shown</p>	\$984.50
<p data-bbox="204 674 783 707"><a href="#">12V 3A 2.5A AC/DC Adapter Power supply - 10 piece</a></p> 	\$179.90
<p data-bbox="204 987 584 1021"><a href="#">Audio AUX to USB cable - 10 piece</a></p> 	\$109.90
<p data-bbox="204 1290 699 1323"><a href="#">3.5mm to 2-Male RCA cable 4 feet - 10 piece</a></p> 	\$63.80
<p data-bbox="204 1592 451 1626"><a href="#">Microphone - 10 piece</a></p> 	\$339.50
<p data-bbox="204 1883 683 1917"><a href="#">Unlocked 4G LTE Mobile hotspot - 10 piece</a></p>	\$499.90





	
<a href="#">Raspberry Pi 4 (over 4GB should work fine) - 10 piece</a>	\$1499.90
	

Table 6: Ancillary Gunshot Prototype Components

Hardware Component	Cost
<a href="#">Plastic gloves for covering the microphone - 100 pack</a>	\$8.99
<a href="#">Gorilla electrical all-weather duct tape - 1 roll</a>	\$11.98
<a href="#">Gorilla double sided tape - 2 pack</a>	\$23.38
<a href="#">Waterproof sealant - 1 pack</a>	\$9.84

## Appendix E: Machine Learning Model Filter

Details for the second ML filter model, which is responsible for filtering gunshots from similar sounding audio files →

```

model = Sequential(
    [
        layers.Dense(1000,activation='relu',input_shape=(128,)),
        layers.Dense(750,activation='relu'),
        layers.Dense(500,activation='relu'),
        layers.Dense(250,activation='relu'),
        layers.Dense(100,activation='relu'),
        layers.Dense(50,activation='relu'),
        layers.Dense(10,activation='relu'),
        layers.Dense(2,activation='softmax')
    ]
)

```