Using Machine Learning to Classify Gunshots and Gunshot-like Sounds

Barbara Teterycz

Table of Contents

Abstract

Sounds differ from each other. Having a strong ML model that can distinguish between the sounds and correctly classify them, may help with a variety of social problems, such as human and animal well being, (e.g. some sounds cause harm to autistic people and/or animals), safety drills (people with hearing disabilities may not be aware of the alarms), mass shootings, etc. Although reducing a false positive rate (e.g. a rate of other sounds falsely classified as gunshots) would help decrease the number of false alarms, it is more important for the safety and security reason, to actually reduce the false negative rate (e.g. a rate of actual gunshots not being recognized as such).

Our research started with searching for and collecting data. The first database found in one of the public github repositories is a collection of plastic bag pop sounds. The second database is a collection of various gunshot recordings. We used an identical number of data points from each database as an input to our machine learning models. We developed two types of ML models: classification linear regression model and convolutional neural network (CNN). Furthermore, in order to prevent our models from overfitting by learning from only these two different types of sound, we implemented data augmentation and then added a random noise to the half of this enlarged database.

Our first classification model used with the original database resulted in 0.561 accuracy. When used with the augmented dataset with half of it being slightly distorted, its accuracy level was ranging from 0.557 to 0.610. Our CNN model, on the other hand, produced much better results with both the original and augmented data. The accuracy level of the convolutional neural network used with the original data reached 0.972 for 14 epochs and 0.996 for 30 epochs, whereas the accuracy level of the CNN model used with the augmented data was 0.983 for 14 epochs and 0.972 for 30 of them.

We concluded that while our CNN reported back very high accuracy, it is still not good enough for recognizing actual gunshots among a variety of sounds, which we did not use in our project, and for this reason it is not meant to be the sole tool for recognizing an active shooter [16].

Background

There have been about 2,000 school shooting events in the U.S. between 1970 and 2022 [3]. Based on the research, it's often difficult to recognize the sound and the direction of the gunshot inside the building [7]. Furthermore, since some sounds have a very similar waveform to the one of the gunshot, it's often difficult for human beings to differentiate life-threatening gunshot events from non-life-threatening gunshot-like events (e.g. plastic bag explosion). However, recognizing the key features that distinguish between various sounds, can be very helpful in building an effective gunshot detection system that can save innocent lives[9]. By taking our dataset, which uses audio samples both original and augmented ones with half of them being slightly distorted, and plugging it into two different supervised machine learning models (a

classification model and a convolutional neural network), we aimed to see the accuracy in which a machine could detect the difference between gunshots and gunshot-like sound events.

Dataset

The dataset used in our project included 1226 audio files, 612 of which was a collection of plastic bag pop sounds captured in different environments at different distances from different microphones and found in one of the public github repositories[15]. The second database was a collection of 615 gunshot recordings influenced by firearm and ammunition type, the scene geometry, and the recording device used [14]. We labeled each audio in the plastic bag pop sounds dataset using 0, and each audio in the gunshots dataset using 1.

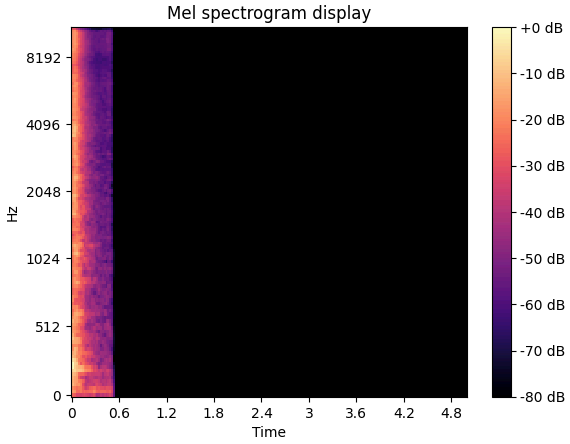
	File Name:	Sample Rate	Duration (s)	Label
0	IP_162A_S01.wav	44100	1.875283	1
1	IP_219B_S06.wav	44100	1.927891	1
2	ZM_089A_S02.wav	96000	2.027781	1
3	IP_110B_S04.wav	44100	1.968571	1
4	SA_153B_S02.wav	48000	2.057167	1
...	...	...	...	...
608	50001010800.wav	44100	0.512358	0
609	100020620400.wav	192000	0.500000	0
610	50004060500.wav	192000	0.500000	0
611	30002020800.wav	192000	0.450000	0

Since these data points had different starting point (onset) of the sound, duration, and sample rate, we first had to perform data preprocessing by 1) applying the same sample rate to each audio, 2) finding the starting point of each sound, 3) clipping each audio starting from the onset sample, 4) adjusting the length of each audio. We performed the above steps using functions from the librosa library. After that, we combined all the sounds in one np.array called data, and all the labels in another np.array called labels.

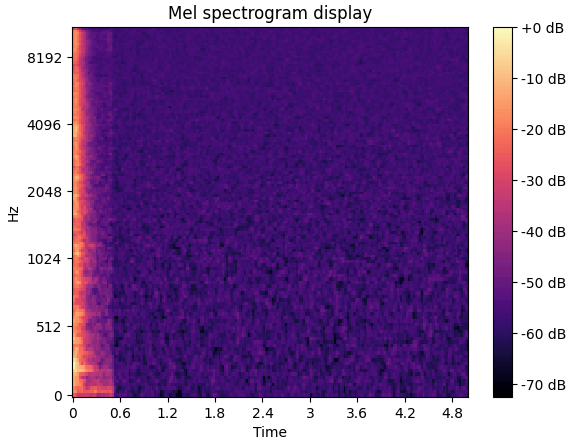
The next step was splitting our dataset into 80% training and 20% testing data. Also, in order to prevent our ML models from overfitting by learning from only these two types of sound, we performed data augmentation by doubling our training data and slightly distorting a half of it by adding a random noise (using random normal/Gaussian distribution) on top of the original waveforms. This way, the total number of our training data points increased to 1960. Without this random distortion of the original audio, our model might not make accurate predictions from any data other than the training data [19].

In order to observe the difference between the original and distorted audio, we added a code for presenting a given data point in the form of a spectrogram.

Original audio



Distorted audio



Methodology/Models

Two supervised machine learning models were used to classify the data: logistic regression and convolutional neural network. Both of them are known as classifiers. The other type of the supervised ML models is regression. The most significant difference between regression and classification is that regression helps predict a continuous quantity, whereas classification predicts discrete (separate) class labels [8].

Logistic Regression

Logistic regression is used for binary classification problems (those, where the data has only two classes). In other words, logistic regression estimates the probability of the outcome being 0 or 1 [12]. And as such, it uses sigmoid function that maps any real-valued number to a value between 0 and 1:

$$P(y = 1|x) = \frac{1}{1 + e^{-(\beta_0+\beta_1 x_1+\beta_2 x_2+...+\beta_n x_n)}}$$

Where:

- P(y = 1|x) represents the probability that the dependent variable y is 1 given the input features x.
- e is the base of the natural logarithm.
- $\beta_0, \beta_1, \beta_2, \ldots \beta_n$, are the coefficients (weights) of the logistic regression model.
- $x_1, x_2, \ldots x_n$ are the input features.

In logistic regression, the goal is to find the best-fitting set of coefficients β that maximizes the likelihood of the observed data [11].

Convolutional Neural Network (CNN)

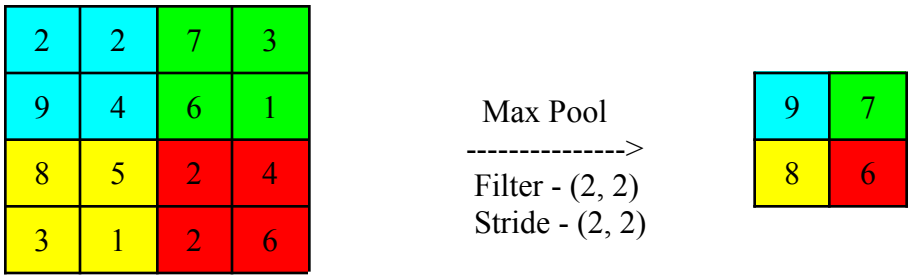
Convolutional Neural Network is a type of deep learning algorithm used for image classification and object recognition tasks. Like artificial neural networks, they also use node layers, such as input layer, one or more hidden layers, and an output layer. Each node connects to another and has an associated weight and threshold. If the output of any individual node is above the specified threshold value, that node is activated, sending data to the next layer of the network. Otherwise, no data is passed along to the next layer of the network [17]. However, CNNs also have convolutional layers that use feature detectors (also known as kernels or filters) that move across the image checking if the feature is present. This process is known as convolution. The filter is a two-dimensional (2-D) array of weights with a typical size of a 3x3 matrix. It is applied to an area of the image and a dot product is calculated between the input pixels and the filter. This dot product is then fed into an output array. With each layer, the CNN increases in its complexity identifying greater portions of the image. Earlier layers focus on simple features, such as colors and edges, and the next layers recognize additional elements or shapes of the image until the intended object is finally identified.
Our model used seven convolutional layers, each of them activated by Rectified Linear Units (ReLU) function that outputs the input if it's positive, and zero otherwise [1]. The first five convolutional layers used 32 kernels, and the last two of them used 64 kernels. All the kernels are 3x3 matrices. The fifth and seventh convolutional layers were followed by the pooling layers that reduced the spatial dimensions (width and height) of the input data, while retaining the most significant information [18]. In other words, the purpose of the pooling layer is to summarize the features covered by the two-dimensional filter called pooling window. The filter slides over each channel of feature map and reduces its dimension by following this formula [10]:

$$\text{ceil}[(n_h - f + 1) / s] * \text{ceil}[(n_w - f + 1)/s] * n_c$$

where:
n_h - height of feature map
n_w - width of feature map
n_c - number of channels in the feature map
f - size of filter
s - stride length (it's the number of jumps a pooling window must make per pool operation. The larger the stride value the more pixels to jump over).

We used max pooling, as one of the types of the pooling layers, to select the maximum element from the region of the feature map covered by the filter [4]. For example, for a feature map having dimensions 4 x 4 x 4, the dimensions of output, obtained after a pooling layer that uses 2x2 window and a stride size defaulting to pool size, is:

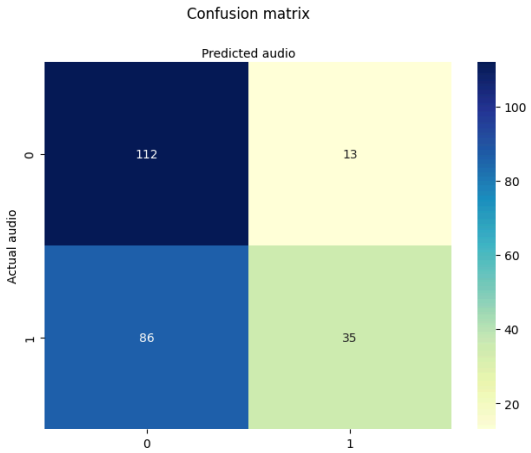
$$\text{ceil}[(4 - 2 + 1) / 2] * \text{ceil}[(4 - 2 + 1)/2] * 4 = \text{ceil}[3/2] * \text{ceil}[3/2] * 4 = 2 * 2 * 4$$



Our model also used four dropout layers that dropped out some neurons by randomly setting input units to 0 with a frequency of 0.5 rate at each step during training time to prevent overfitting [6]. We then flattened the data in the Flatten layer that converted multi-dimensional input data into a one-dimensional array, which allowed us to pass our images to the dense layer with 512 neurons fully connected to every neuron in this previous flatten layer. The dense layer ran the data through the equation: output = activation(dot(input, kernel) + bias) to classify data [5]. At the end, we added another dense layer with one neuron only, which was squished by the activation sigmoid function to anything between 0 and 1 [13].
While training the data, our model first used forteen, and then thirty epochs before being tested. In each epoch, every sample in a batch (the number of samples from the training data, which was 10 in our case) was used to update the internal model parameters [2]. At the end of the batch, the predictions were compared to the expected output and an error was calculated and then used to improve the model. Each epoch reported such an individual loss (a mathematical function that quantifies the difference between predicted and actual values, also known as the cost function) and accuracy (a metric used to measure the proportion of correct predictions made by the model out of the total predictions made). When training, we aim to minimize the loss between the predicted and target outputs [20]. Thus, the goal in training a machine learning model is to minimize the loss function and maximize the accuracy score.

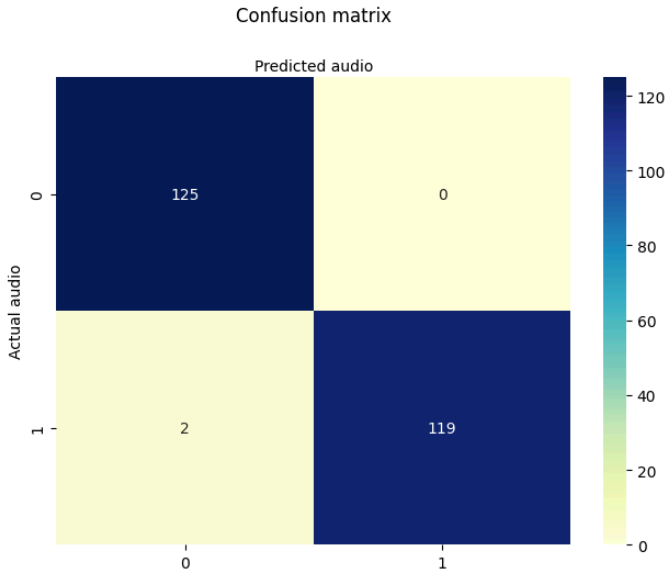
Results and Discussion

The accuracy score of the logistic regression model applied to the original dataset was 0.597. As shown below, there are more true positives (plastic bag pop sounds /0/ and gunshots /1/ predicted as such) than false positives (plastic bag pop sounds classified as gunshots) and false negatives (gunshots classified as plastic bag pop sounds).



These results however, are not good enough and not promising at all. 86 plastic bag pop sounds falsely classified as gunshots would cause too many false alarms, whereas 13 actual gunshots falsely classified as plastic bag pop sounds might result in deaths.
When used with the augmented data, the logistic regression model fluctuated between 0.557 and 0.610.

On the other hand, the accuracy score of the convolutional neural network used to train the model with the original dataset and 14 epochs was 0.972, and 0.996 with 30 epochs.



When training the model with the augmented data, the accuracy score after running the model for 14 epochs was 0.983, and 0.972 after 30 epochs. When running this model several times, its accuracy score reached 1.0 at some point.

```
[ ]  cnn_aug.score(X_test, y_test)

     8/8 [==============================] - 18s 2s/step
     1.0
```

Conclusion

Based on the results, we can conclude that the convolutional neural network significantly outperformed the linear regression model. When trained with more variety of sounds (while using augmented data with half of them having some added random noise), its accuracy score dropped a bit after the first few runs, but then increased. This only proves that training a model with only two types of sound, causes the model to overfit and makes it less reliable. If such a model were used with a larger range of different sounds, which it was never trained on, it would perform much worse.

Acknowledgements

We would like to thank the Inspirit AI mentorship program and its mentor Ivan Villa-Renteria for their dedication and wonderful support as well as this great learning opportunity.

References

1. Brownlee, J. (2020, August 20). *A Gentle Introduction to the Rectified Linear Unit (ReLU) - MachineLearningMastery.com*. Machine Learning Mastery. Retrieved April 28, 2024, from https://machinelearningmastery.com/rectified-linear-activation-function-for-deep-learning-neural-networks/

2. Brownlee, J. (2022, August 15). *Difference Between a Batch and an Epoch in a Neural Network - MachineLearningMastery.com*. Machine Learning Mastery. Retrieved April 28, 2024, from https://machinelearningmastery.com/difference-between-a-batch-and-an-epoch/

3. Center for Homeland Defense and Security. (2022, July 30). *Shooting Incidents at K-12 Schools (Jan 1970-Jun 2022) - CHDS School Shooting Safety Compendium*. Center for Homeland Defense and Security. Retrieved April 7, 2024, from https://www.chds.us/sssc/data-map/

4. *Convolution and Max Pooling*. (n.d.). Colby Computer Science. Retrieved April 28, 2024, from https://cs.colby.edu/courses/F19/cs343/lectures/lecture11/Lecture11Slides.pdf

5. *Dense layer*. (n.d.). Keras. Retrieved April 22, 2024, from https://keras.io/api/layers/core_layers/dense/

6. *Dropout layer*. (n.d.). Keras. Retrieved April 28, 2024, from https://keras.io/api/layers/regularization_layers/dropout/

7. Ellifritz, G. (2022, June 27). *Recognizing the Sound of Gunfire*. Active Response Training. Retrieved April 7, 2024, from https://www.activeresponsetraining.net/recognizing-the-sound-of-gunfire

8. Gupta, S. (2021, October 6). *Regression vs. Classification in Machine Learning: What's the Difference?* Springboard. Retrieved April 24, 2024, from https://www.springboard.com/blog/data-science/regression-vs-classification/

9. IEEE Xplore. (2020, July 16). *Gunshot Detection Using Convolutional Neural Networks*. Abstract. Retrieved April 7, 2024, from https://ieeexplore.ieee.org/abstract/document/9141621

10. Jain, S. (2023, April 21). *CNN | Introduction to Pooling Layer*. GeeksforGeeks. Retrieved April 28, 2024, from https://www.geeksforgeeks.org/cnn-introduction-to-pooling-layer/

11. *Logistic Regression | STAT 462*. (2018). STAT ONLINE. Retrieved April 22, 2024, from https://online.stat.psu.edu/stat462/node/207/

12. MathWorks. (n.d.). *Choosing the Best Machine Learning Classification Model and Avoiding Overfitting*. MathWorks. Retrieved April 24, 2024, from https://www.mathworks.com/campaigns/offers/next/choosing-the-best-machine-learning-classification-model-and-avoiding-overfitting.html

13. Ming, Z. (2022, August 6). *Using Activation Functions in Neural Networks - MachineLearningMastery.com*. Machine Learning Mastery. Retrieved April 22, 2024, from https://machinelearningmastery.com/using-activation-functions-in-neural-networks/

14. National Institute of Justice. (2016, September 18). *Development of Computational Methods for the Audio Analysis of Gunshots*. National Institute of Justice. Retrieved April 7, 2024, from https://nij.ojp.gov/funding/awards/2016-dn-bx-0183

15. *Data Collection, Modeling, and Classification for Gunshot and Gunshot-like Audio Events: A Case Study*. MDPI. Retrieved April 7, 2024, from https://www.mdpi.com/1424-8220/21/21/7320

16. Sonoma State University. (n.d, n.d n.d). *Active Shooter Response*. Emergency Services at Sonoma State University. Retrieved April 7, 2024, from https://emergency.sonoma.edu/procedures/active-shooter-response

17. *What are Convolutional Neural Networks?* (n.d.). IBM. Retrieved April 22, 2024, from https://www.ibm.com/topics/convolutional-neural-networks

18. *What is a max pooling layer in CNN?* (n.d.). Educative.io. Retrieved April 28, 2024, from https://www.educative.io/answers/what-is-a-max-pooling-layer-in-cnn

19. *What is Overfitting?* (n.d.). IBM. Retrieved April 22, 2024, from https://www.ibm.com/topics/overfitting

20. Yathish, V. (2022, August 4). *Loss Functions and Their Use In Neural Networks*. Towards Data Science. Retrieved April 22, 2024, from https://towardsdatascience.com/loss-functions-and-their-use-in-neural-networks-a470e703f1e9