Title: Acoustic Detection and Localisation of Drones using Audio Analysis and Real-time Audio Attribute Prediction

Abstract: Visual and Acoustic detection of drones has been a very challenging task since both of them produce a lot of erroneous predictions. Even more so, the acoustic prediction has a higher rate of getting prediction errors than image-based detection due to the uncertain nature of data it produces. But, even though visual detection of a drone is very prominent and frequently used in available technology, there may be cases where the view is obstructed, lighting conditions are poor, the field of view is narrow and so on. But, acoustic detection is tricky and localization is even harder for locating drones. Recent advances show that localization is possible using a microphone array to get audio signatures at different points and comparing them. This paper presents a method of predicting not only the presence of a drone but also its localization using a single microphone, by analyzing its spectral features and time-series nature of the signal, accounting for the model's response to false positives. Finally, the model was fed into a custom-made detector and tested in real time with promising results. Audio signatures were analyzed and various features were extracted after exhaustive filtering of the data to ensure the dataset we fed into the model is from a drone. Support Vector Machines were used because of their effectiveness and inexpensive computational cost. Finally, the model was fed into a custom-made detector and tested in real time.

Introduction:
[problem definition]Some detection and localization techniques have achieved up to 72.8% accuracy [urban sound classification and https://www.eit.lth.se/sprapport.php?uid=884], but these methods don't take into account signatures with similar frequency bands, such as stray noises and few harmonics.
[Related Work]
[Disposition]
[Scientific Background]

System Design:
Equipment Used:
A 3DR Iris+ provided by Duke Marine Lab was used to obtain the required audio dataset. The Iris+ weighs about 1.28kgs with average outdoor ambient sound as 39.4079dB and indoor as 33.0261dB [https://hal.pratt.duke.edu/sites/hal.pratt.duke.edu/files/u24/Small_UAV_Noise_Analysis_rqi.pdf]. The frequency band for a typical drone lies under 1500Hz with its harmonics within a short range of interval 130 - 180Hz, based on testing.

Detector Hardware Setup:
An Apex 220 microphone
[http://apexelectronics.com/microphones/special_applications/test_reference/product/apex220/] was

used for gathering audio signatures both in outdoor and indoor conditions because of its linear frequency response and omnidirectional nature. The microphone was connected to a Raspberry Pi 3+ that had been configured to get signals, which are parsed for detection and localization by Python modules developed by us. The modules sent the result to a server to be displayed on a screen or an app.
[insert picture]

Software Setup:
Python was used to write and compile scripts for recording, parsing, filtering, labeling, and training. Standard modules such as Numpy, Pandas, Libros, Scipy were used to get and analyze audio data, and Sklearn was used for training the generated model. Our focus was to use a simple Support Vector Machine (SVM) model with the polynomial kernel to train the dataset, as any other primitive algorithms would produce an insufficiently optimized model and anything higher would increase computational expense. The resulting SVM model was saved and a new script for real-time detection was made and run to detect and localize data using the available model.
[insert picture]

Algorithms Used:
We have tried to make our analysis as extensive as possible so as to get all aspects of the audio signal for efficient processing and target high accuracy. For this purpose, various audio features were extracted and the signatures were analyzed thoroughly. We found out that some features of audio were more distinct than others
[https://github.com/BenjaminDoran/Urban-Sound-Classification/blob/master/Urban_Sound_Classification_Paper.pdf]. These features were based on audio filter banks generated by auto-correlation methods (e.g. Mel scale, coefficients of Mel-frequency Cepstrum (spectrum of a spectrum), chromatogram, spectra-contrast and tonal centroid), as these have shown exemplary characteristics for classification [https://arxiv.org/pdf/1701.05779]. The initial training results were insufficient for generating a proper model of the detector due to random noise interference. To solve this, the data was passed through a Butterworth filter so that the accuracy increased to the desired level. Although the accuracy increased significantly with this filter, it was still inadequate to finalize the model as a few parameters would still detect stray audio and label them. Therefore, an improved filter was used with spectral subtraction [https://doi.org/10.1109/TASSP.1979.1163209], which is a transformation method analogous to Fourier transform that increases the signal to noise ratio. A power spectrum map was generated from the drone audio by analyzing a hanging window with the Welch method and its characteristics were noted to be similar for similar sounds. We used Total Power Spectra and its semilog to parametrize it, and we found out that the peaks were of recurring nature for certain frequencies (generally between 135-160 Hz), which formed a certain pattern. We concluded that this spectral estimator could be used to filter unwanted frequencies in the spectrum which basically comprises of noise by subtracting it from the main audio signal. To do it we used a separate noise spectrum whose magnitude was averaged from no flight audio. This noise suppressor was used as a filter to pass desired

features to the above-mentioned model and the results were astonishingly good. The summary of the process is given below:

[write algorithm/code]

Dataset Generation and Processing:
Audio recordings from Iris+ flights were collected at different locations at regular intervals for the generation of the dataset. The recordings were properly labeled with heights and distances in a vertical 2D plane, with the Apex 220 designated as the origin. The drone was flown for about 1 min at each location, covering about 150 coordinate points and gathering as much as 14932 filtered 'feature rich' recordings of 1 second to be analyzed by the SVM model. Each of them was labeled according to radial distances with 5 classes: 'very near': less than 15m, 'near': 15m to 35m, 'midrange': 35m to 60m, ' far': 60m to 120m and 'very far': beyond 120m for 5-fold cross-validation. This cross-validation avoids the data to generate over-optimistic results from training. This dataset was taken and stored as .wav files, then parsed using Soundfile and Libros library in order to get frames of arrays with numbers for further analysis. We also took some recordings of wind and blizzard, so that we can generate an additional model to distinguish between a drone and white noise to improve detection confidence.
[insert comparison pics]
[insert frequency pics]
[insert spectrum pics]

Experimental Methods:
Experiment Walkthrough and Underlying Techniques:
The labeled audio data was processed for feature extraction using a Butterworth filter to cancel out noises. As these noises cannot be fully eradicated, we analyzed hanging window using Welch method to get power spectral peaks and mapped them for different frequencies and found that a set of these formed a pattern of frequency ranges, which was recurrent in a similar audio dataset. A separate audio with no flight sound was used to compute the average noise power spectrum in the same environment which was then subtracted from the original audio to filter out the noise.

[add formulae and snippets][add pics of power spectra]

We used the same method as a parameter to distinguish between files that genuinely comprise drone audio signature and those which don't. We then set the following criteria to get a set of 6 peak frequencies for comparison to generate a model to distinguish misclassified data with properly classified data.

[add the snippet of formulae]

The audio dataset that passed the above criteria were 1-second samples with a sample rate of 44100. These samples were again parsed to create a Mel spectrogram, Chromagram, a Mel-Frequency CC, a Spectral Contrast and a Tonal Feature by analyzing short, partially overlapping frames in the sample. The above features are common characteristics used for speech recognition. After slicing the signal into frames, we applied the Hamming Window Function [https://haythamfayek.com/2016/04/21/speech-processing-for-machine-learning.html] to each of the frames for a length of approximately 20 ms and apply filter banks to these power spectra and sum them up. The resulting data is highly correlated and can interfere with our model, so we used the Discrete Cosine Transform to decorrelate them and stored the coefficients generated (MFCCs).
[add mfcc generation formulae]
Similarly, the other autocorrelation features, such as Mel spectrogram, chromic features, spectral contrast, and tonal centroid, are separated from the data with the help of Libros library and are concatenated to get a concentrated feature-rich array that is passed into our model to train it.
[insert formulae for the above features][insert pic if possible]

Model Generation and Training:
Since all the data we have are from drone audio taken at different distances from the microphone, its degree of similarity in the recording is very high. For this purpose, a nu supported SVM classifier was generated using a 3rd-degree polynomial basis function as a one-vs-rest classifier to categorize the labels according to their corresponding features. It is interesting to note that, for unfiltered data, a nu value of 0.8 gave as much as 60% accuracy, whereas a Butterworth filtered data boosted the accuracy over 80%, even though the data is correlated to a certain degree. At this nu value, the persistently filtered data using our algorithm did not converge the model, suggesting that this limit for training error is inadequate. Pushing the nu to about 0.02 gave us exceptional results, with accuracy over 95% and confirmed that the filter used was indeed an exceptional one. Thus, we generated an apt detector which would not only detect data but also classify them based on its distance, solving our localization problem.
[insert confusion maps]

Validation:
To validate this, we created a separate script, which would periodically gather a chunk of the audio signal of length 1 sec and sample rate of 44100, and parse them after filtering using our algorithm so that it generated a feature list similar to what we fed the SVM for training. We would then load our SVM model and classify the recordings for us. This process is repeated until the user wishes to close the program, thus fulfilling real-time detection and localization problem.
[insert field test pics]

Post Classification Analysis

Our original intention was to predict the location of drone using the model generated above. With preliminary tests, we found out that although the results of training were really good for classifying the gathered data, we wanted to see if this result can be extrapolated to additional data collected during actual flights in real time. So, we created python scripts which would get 1 sec chunk of audio data, filter, process and analyze it to classify them into the given labels. The results were seen to vary within a range of about 50-70% accuracy, given that we changed the surroundings and used a different drone after every test. We took new recordings after each testing which had about 2 mins of audio data at each label with varying distances and trained them again to generate a new model for that particular set of surrounding and drone. After conducting tests with this new model in the same {surrounding, drone} set, we found out similar accuracy range with about 70% being in the same set and 50% in a new set. Due to this, we used a simple algorithm which made sure that an identical model can be used for a different set of {surrounding, drone}, by determining its confidence. In this algorithm, we took the most recent 10 predictions generated by the old model and, using the number of occurrence of the similarly classified labels, we got a relative confidence measure. This was found to be very effective in predicting the labels efficiently in different {surrounding, drone} set with accuracy over 80% in the number of field tests done.
[add a table for test logs]

Results and Discussion:
Audio Analysis Results:
We saw that the final data generated after filtering were highly correlated for the model to classify, so we used special features to decorrelate the dataset. This gave us a database of size 8732 and all of its characteristic rich features were logged into a CSV file for easy import and analysis. The final extracted features were highly distinguishable from one another and thus were used to train the model.

[We have to talk about the confidence level]

[insert pic]

Training Results:
Training the above set with SVM gave exceptional results, suggesting that Neural Network based models are not always required. A simple model such as SVM, which is computationally inexpensive, can be used with high accuracy of 96.886%.
[insert filter vs nu vs accuracy table]
We also saw that for an increase in the size of the database, a more precise model is generated that can be used for training.

Conclusion:
A multilayered filter using power spectral mapping was introduced to the model, which spiked up training accuracy even for highly correlated audio data. With this filter, we were able to present a very accurate multilabel classification with the same drone recording based on distance. Then we evaluated this model with confusion maps and analysed our features. We ruled out most of the misclassified data, since the filter in itself would not allow the 'false' data to pass into the classifier. This method enhanced our detection and localisation distance using only one microphone. The device can be used as a primitive drone detector that can also classify the drone based on its distance, thus partially solving the problem of localisation. With only one microphone, we can use the model to determine different distance classes for different ranges.

Appendix A for Mobile App:

Overview of PRIS

The Prison Reconnaissance Information System (PRIS), a cost-efficient solution developed by HAL, detects drones and human intruders and alerts and provides relevant detection event information to prison officials. It comprises three components (HAL)—server, detection equipment, and client interface. The detection equipment consists of thermal cameras for taking thermal images, microphones gathering acoustic data, and Raspberry Pi for transmitting these data to the server via wireless network. The server receives, processes and analyzes the information from Raspberry Pi (thermal images and sound data) based on algorithms developed by HAL. If the analysis determines there is an intruder, the server will securely send the relevant information to the client interface and alert the user through the mobile app.

Client Interface

After receiving information from the server, the client interface (mobile app) will display where equipment detected intruders on Google Maps, relevant thermal images (only for thermal detection),

the time of event occurrence, how confident the algorithm is about the result, how far the intruder is from the detection equipment, and the general area of where the intruder may be in (only applies to drones). Below is the user interface and a more detailed explanation of the mobile application features.