CLASSIFICATION OF AVIATION NOISE AND FALSE TRIGGERS FOR AIRPORT NOISE MONITORING

PROBST Lukas, TSCHAVOLL Jakob, YIJUN Zhao

l.probst@campus.tu-berlin.de, j.tschavoll@campus.tu-berlin.de, yijun.zhao@tu-berlin.de

ABSTRACT

Manual classification of recordings of aircraft noise at Berlin Brandenburg Airport (BER) is costly and therefore in need of automation. Such a manual verification process is time-consuming and necessary to ensure compliance with sound pressure level limits. Machine learning models, particularly Convolutional Neural Networks (CNNs), are suitable for (partial) automation of this classification task. A dataset was provided, which was obtained from 10 noise monitoring stations located in and around the airport and almost exclusively contained flight noise without interference (*clean*). Due to the imbalance of *clean* data and files with false alarms due to urban or rural noises (*contaminated*), artificial *contaminated* files were generated via augmentation. The authors trained different CNN models on this dataset by extracting Mel-spectrograms and evaluated them using test data. An accuracy of 91% suggests that CNN models can be of use for automated noise classification in the field of aviation.

Index Terms— aviation noise, convolutional neural networks, noise classification

1. INTRODUCTION

The aviation industry is an important part of our modern society, but it causes a lot of emissions. Not only pollutants - this is usually the first thing that comes to mind - but also aircraft noise pose a challenge to a desired neutrality in terms of environmental impact. In this respect, airports, especially Berlin Brandenburg Airport BER, have a duty to check emissions among residents - including aircraft noise. To this end, several protection zones have been set up around BER with limits on sound pressure levels [1]. The verification of compliance with these limits is carried out by means of permanent monitoring of aircraft noise with the help of more than 30 measuring points distributed in and around the area of the airport. At each measuring point, the monitoring system automatically creates audio recordings of every aircraft movement in the vicinity of the measuring point. If a limit is exceeded, BER is obliged to compensate local residents monetarily for the excess. This requires a manual check of the flight movement, or more precisely of the audio recording, in order to verify whether the exceeding of the limit value was triggered by the flight event. In some cases, disturbing noises (e.g. animals, construction noise, traffic) in the vicinity of the measurement points are triggers for an exceedance, in which case the flight movement cannot be sanctioned.

The manual review or classification of audio recordings (*clean* or *contaminated*) takes a lot of time, due to the number and duration of files per flight movement. The supportive use of machine learning for the classification of audio recordings helps the airport to get an estimation of how high the aircraft noise or *contaminated* noise content of a recording is already without having listened into an audio file. Probable classifications can possibly already be excluded from the manual process on the basis of the classification, thus saving time.

1.1. State of the art

In similar works, Ju-won and Min-koo [2] were able to obtain an aircraft identification accuracy with less than 1% false-postives and false-negatives using a CNN model with 21 layers and Mel-, MFCC- and derivative-features of aviation noise sourced from Jeju international airport, Korea. Asensio et al. [3] developed a likeliness detector for aviation noise with MFCC features and advanced Gaussian statistics in real time which lead to an accuracy of 93%. Training data is either gathered by the conducting parties or often sourced from pre-built data sets such as *ESC-50*[4], *UrbanSound8K*[5], or *FSD50K*[6]. A well-known and successful model for universal audio classification is *YAMNet*[7] from Google which is often used as comparison to newly trained models.

2. METHODS

2.1. Data set

In cooperation with Germany-based airport BER a total of 107k mp3-files sourced from 10 noise monitoring stations around the airport's perimeter were available as training data. The files consist of start and landing events by aviation vehicles of 13 different types (*clean* data) and unwanted rural or urban noise (*contaminated* data) lasting anywhere from 40 to

80 seconds. Despite the data set's large size, the amount of data containing contamination only makes out about 1% of the total data set. To combat this disparity, the total amount of *clean* data gets reduced to 5% of the whole set and the *contaminated* data gets bootstrapped up to the same amount with different methods of augmentations, described in section 2.2.

2.2. Pre-processing

Besides mandatory train and test splits the data was separated into audio slices of 5 seconds each while omitting the last slice smaller than this time frame. This time was chosen because the authors believe that such a time frame is representative of the duration of an unwanted noise event in addition to being the same length as similar data sets. The amount of *contaminated* data was increased by mixing *clean* slices with clips from the *UrbanSound8k*[8] and *ESC-50*[4] data sets, from which only suitable noise events were chosen. These include rural and urban noises labeled as *dog, rooster, frog, cat, insects, crow, rain, crickets, chirping birds, wind, thunderstorm, chainsaw, siren, car horn, engine, train, church bells, children playing, street music, drilling* and reflect the natural occurrences of noise in the towns and villages around the airport where the noise monitors are located. After both *clean* and *contaminated* data sets were brought up to the same size and re-sampled to 16 kHz various augmentation methods including *pitch shifting, time stretching, time offset* were applied to both classes until a total of 100000 audio slices were present in the data set. This paper does not further discuss methods or advantages of data augmentation.

Feature extraction follows a standard combination of FFT and Mel-windowing to create Mel-spectrograms up to 8 kHz. Parameters like window size, hop size, normalization and cutoff cause the features to take on a specific shape and size and are summed up in a configuration described as F_x in table 1. For the sake of comparison, more than one settings configuration was used to create the training data.

	window size	hop size	normalization	cutoff
F_1	2048	256	no	None
F_2	2048	256	yes	$-40\,\mathrm{dB_{FS}}$

Table 1. Configurations for feature extractions.

2.3. Network architecture

Similar to the the feature configurations different model architectures are compared to each other. Every model features convolutional and multiple dense layers, closer described as M_x in table 2. Each layer is described in the *KERAS* API nomenclature and follows established methods like convolution layers and deep layers.

M_1
Conv2D: 32 filters, (3, 3) kernel size, 'relu' activation, (32, 321, 1) input shape *MaxPooling2D*: (2, 2) pool size *Flatten* *Dense*: 128 neurons, 'relu' activation *Dropout*: 0.5 rate *Dense*: 2 classes, 'softmax' activation
M_2
Conv2D: 32 filters, (3, 3) kernel size, 'relu' activation, (32, 321, 1) input shape *MaxPooling2D*: (2, 2) pool size *Dropout*: 0.2 rate *Conv2D*: 64 filters, (3, 3) kernel size, 'relu' activation *MaxPooling2D*: (2, 2) pool size *Dropout*: 0.2 rate *Conv2D*: 64 filters, (3, 3) kernel size, 'relu' activation *Flatten* *Dense*: 64 neurons, 'relu' activation *Dropout*: 0.2 rate *Dense*: 32 neurons, 'relu' activation *Dense*: 2 classes, 'softmax' activation

Table 2. Model architectures.

M_1 is shallow and has $\approx 10^7$ trainable parameters with a size of $120\,\mathrm{MB}$. It features a single convolution and hidden layer in order to reduce complexity and training time. It is compared with a more complex but less flexible architecture in M_2 which has $\approx 10^6$ trainable parameters and a size of $15\,\mathrm{MB}$. By comparing the accuracy, size and training time these models can be fit to their applications which may have strict requirements regarding size or training speed.

2.4. Post-processing

Obtained output consists of percentages per 5 second slice $t_{s,5}$ describing the binary distribution between the classes *clean* and *contaminated*. Since flight events are usually longer than a single slice, each 5 second part of a file gets analyzed individually. To avoid missing transient noises right at the edges of such slices, the classification includes a hop distance of $\frac{t_{s,5}}{2}$, which causes twice as many outputs to be generated. These show classifications over time and are able to capture transient changes in noise sources, similar to the recommended post processing stage of YAMNet [7]. To break down the results into a single user friendly value, different methods for interpreting the multiple classifications per noise event are introduced, see table 3. These and above mentioned iterations are combined to obtain the best pre-, intra- and post-processing combination.

P_1	A single contamination classification is greater than 50%
P_2	The mean of contamination classification is greater than 50%

Table 3. Evaluation methods for contamination flagging.

3. RESULTS

Accuracy scores are obtained via evaluations of the data set's test split. This metric is valid for single 5 second samples and is listed in table 4. Since their deviations from each other are less than 5%, the metrics *precision* and *recall* behave similar to *accuracy* and are therefore not listed.

	F_1M_1	F_1M_2	F_2M_1	F_2M_2
Accuracy	0.88	0.91	0.84	0.85
Loss	0.79	0.37	0.59	0.55
F1-score (macro)	0.87	0.91	0.85	0.85

Table 4. Validation results.

Extraction method F_1 in combination with model M_2 scores highest on accuracy and exits training with the lowest validation loss. Normalization in extraction method F_2 lead to lower performance but less score difference between model M_1 and M_2. Despite the accuracy's steady growth the loss rises as well, visible in figure 1. This somewhat surprising phenomenon is further discussed in section 4.

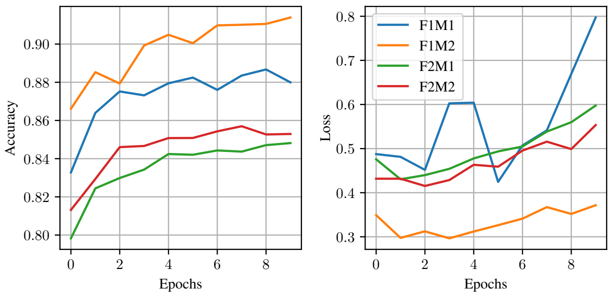

Fig. 1. Accuracy and loss comparison of the validation set for all models.

All models were also tested against both post-processing methods from table 3. For this, a total of 120 full length noise event recordings per class were used for inferencing the models. Figure 2 shows the effect of post-processing methods P_1 and P_2. Method P_2 has a large bias towards *clean* classifications and therefore performs bad on *contaminated* noise events. In tandem with its per slice accuracy combination F_1M_2 performs best in both cases in regards to contamination detection.

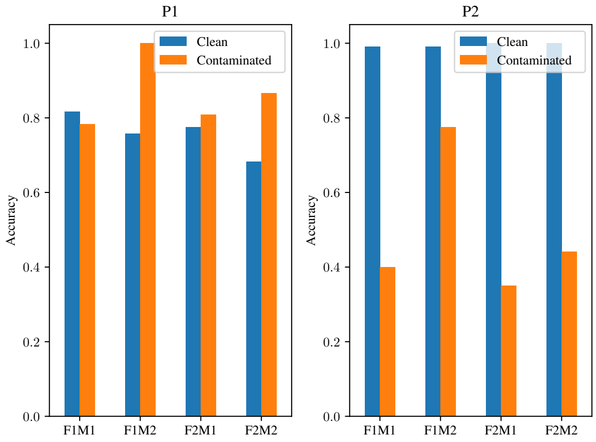

Fig. 2. Model accuracy with post-processing.

To compare F_1M_2's performance against related works, accuracy of a transfer learning model of YAMNet [7] is measured as well. A model with two fully connected layers on top of YAMNet's embeddings were trained with only 200 slices of audio, as adding more training files to the data set leads to overfitting due to YAMNet's already accurate embeddings.

	F_1M_2	**YAMNet TL**
Accuracy	0.91	0.90
Loss	0.37	0.35

Table 5. Performance comparison of F_1M_2 and a YAMNet transfer learning model

YAMNet does not significantly improve results when compared to F_1M_2, visible in table 5. However, it achieves the same *accuracy* with only 0.2% of the data set due to its pretraining. A transfer learning approach should be taken in case of training data scarcity.

4. DISCUSSION

In this study, we explored the problem of distinguishing airplane noise from other environmental sounds using deep learning methods. We first trained a model using a standard approach and achieved an accuracy of 91% by by adjusting different combinations of model architectures as well as pre- and post-processing methods. Then, we applied transfer learning to fine-tune a pre-trained YAMNet model and obtained a similar accuracy of 90%.

A comparison of the results of the two models in Table 4 shows that the complex model has a higher accuracy with the same pre-processing treatment, which is understandable, as in this similar article [2], the authors used a much more complex model in comparison and got a higher accuracy than ours. With the same model, the data set without normalization and cutoff was found to cause a higher accuracy, most likely due to the flight events having similar $\mathrm{dB_{FS}}$ already while normalization raises the noise floor and cutoff limits the dynamic range. This might cause the model to miss out on essential details. Fine tuning the cutoff might alleviate some of these problems. The surprising phenomenon of simultaneously rising accuracy and loss can be explained by outliers in the loss distribution. While a single very wrong classification has a great impact on the overall loss function due to its relatively large numeric size, this very same single classification is easily overshadowed by many true classifications, causing a rising accuracy and loss over epochs.

5. LIMITATIONS

First, the data set we used was relatively small and may not fully capture the variability in airplane noise across different environments and conditions. Future studies could benefit from larger and more diverse data sets. Additionally, we only considered a binary classification task of airplane noise vs. non-airplane noise, but in practice, there may be other types of noise that could be present in the audio recordings. Another problem is the potential for location bias in the data set, which could lead to location-specific predictions by the model. For example, if the data set is predominantly composed of recordings from a particular airport or geographic region, the model may not generalize well for other locations or situations. To mitigate this risk, future studies could explore strategies for collecting more representative and diverse data sets. Additionally, more feature vector dimensions like MFCCs, their derivatives and SNR combined with a more complex network architecture could result in higher accuracy.

6. REFERENCES

[1] Topsonic Systemhaus GmbH, "Travis - flughafen berlin brandenburg," 12 2021.

[2] Ju-won Pak and Min-koo Kim, "Convolutional neural network approach for aircraft noise detection," in *2019 International Conference on Artificial Intelligence in Information and Communication (ICAIIC)*, 2019, pp. 430–434.

[3] C. Asensio, M. Ruiz, and M. Recuero, "Real-time aircraft noise likeness detector," *Applied Acoustics*, vol. 71, no. 6, pp. 539–545, June 2010.

[4] Karol J. Piczak, "ESC: Dataset for Environmental Sound Classification," in *Proceedings of the 23rd Annual ACM Conference on Multimedia*. 2015, pp. 1015–1018, ACM Press.

[5] J. Salamon, C. Jacoby, and J. P. Bello, "A dataset and taxonomy for urban sound research," in *22nd ACM International Conference on Multimedia (ACM-MM'14)*, Orlando, FL, USA, Nov. 2014, pp. 1041–1044.

[6] Eduardo Fonseca, Xavier Favory, Jordi Pons, Frederic Font, and Xavier Serra, "FSD50K: an open dataset of human-labeled sound events," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 30, pp. 829–852, 2022.

[7] Manoj Plakal and Dan Ellis, "Yamnet," 2020.

[8] Justin Salamon and Juan Pablo Bello, "Deep convolutional neural networks and data augmentation for environmental sound classification," *IEEE Signal Processing Letters*, vol. 24, no. 3, pp. 279–283, 2017.