Deep Learning-Based Acoustic Mosquito Detection in Noisy Conditions Using Trainable Kernels and Augmentations

Sean Campos*
Devesh Khandelwal*
sean.campos@berkeley.edu
deveshkhandelwal@berkeley.edu
School of Information
University of California, Berkeley
Berkeley, California, USA

Shwetha C. Nagaraj
shwethacn@ischool.berkeley.edu
School of Information
University of California, Berkeley
Berkeley, California, USA

Fred Nugen
nooj@berkeley.edu
School of Information
Division of Computing, Data Science, and Society
University of California, Berkeley
Berkeley, California, USA

Alberto Todeschini
todeschini@berkeley.edu
School of Information
Division of Computing, Data Science, and Society
University of California, Berkeley
Berkeley, California, USA

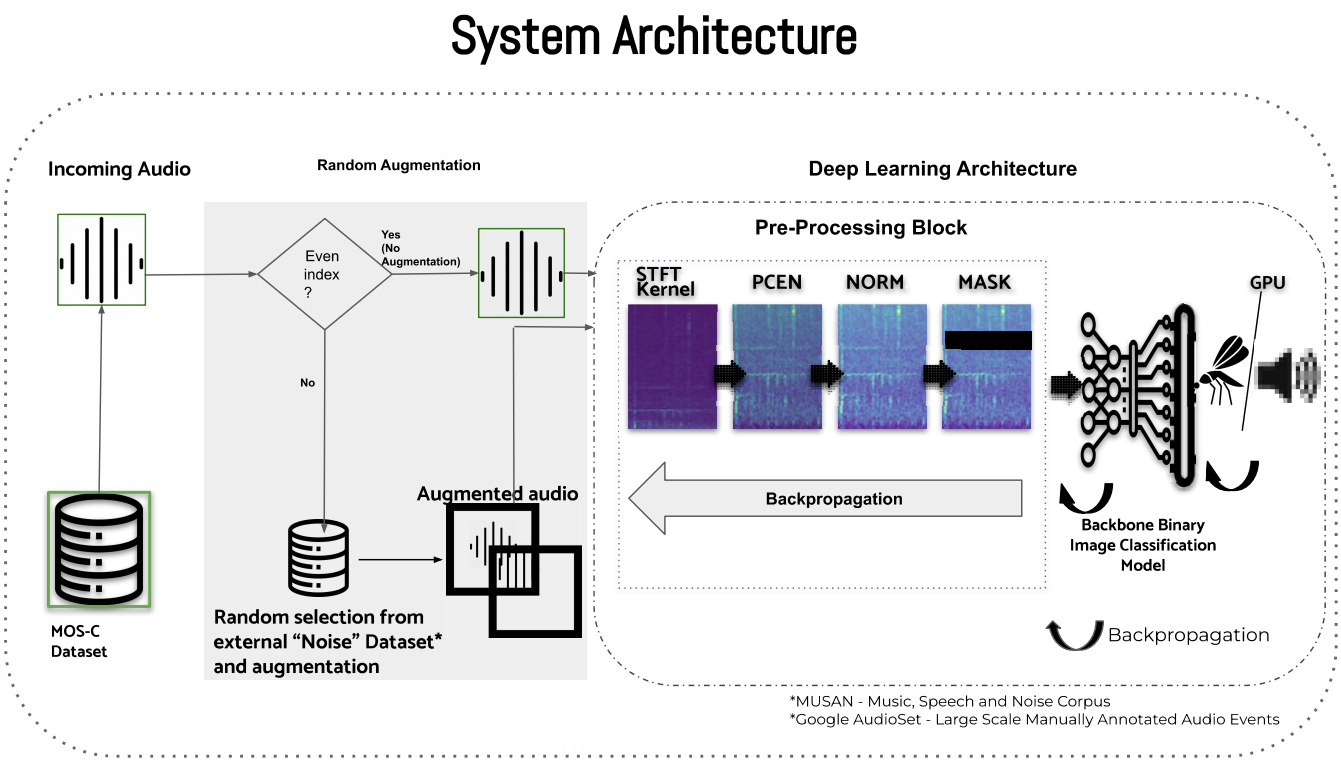

Figure 1: A High-level Overview of the VecNet Mosquito Detection System

ABSTRACT

In this paper, we demonstrate a unique recipe to enhance the effectiveness of audio machine learning approaches [3] by fusing pre-processing techniques into a deep learning model [12]. Our

*Both authors contributed equally to this research.

MM '22, October 10–14, 2022, Lisboa, Portugal

ACM ISBN 978-1-4503-9203-7/22/10.
https://doi.org/10.1145/3503161.3551586

solution accelerates training and inference performance by optimizing hyper-parameters through training instead of costly random searches to build a reliable mosquito detector from audio signals. The experiments and the results presented here are part of the MOS-C submission of the ACM'22 challenge [20]. Our results outperform the published baseline by 212% on the unpublished test set. We believe that this is one of the best real-world examples of building a robust bio-acoustic system that provides reliable mosquito detection in noisy conditions.

CCS CONCEPTS

• **Computer systems organization**; • **Computing methodologies** → **Simulation evaluation**; **Computer vision representations**;

KEYWORDS

neural networks, bio acoustics, mosquito event detection

ACM Reference Format:
Sean Campos, Devesh Khandelwal, Shwetha C. Nagaraj, Fred Nugen, and Alberto Todeschini. 2022. Deep Learning-Based Acoustic Mosquito Detection in Noisy Conditions Using Trainable Kernels and Augmentations. In *Proceedings of the 30th ACM International Conference on Multimedia (MM '22), Oct. 10–14, 2022, Lisboa, Portugal.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3503161.3551586

1 INTRODUCTION

Mosquitoes kill one million people per year [16] and are the deadliest animal to humans on the planet. Global warming has increased the climatic suitability of mosquitoes in already endemic areas and about 1.4 billion additional people are at risk of malaria and dengue in urban areas in Africa and Southeast Asia alone. Due to Covid-19, critical 2020 milestones of The World Health Organization's global malaria strategy have been missed, and without immediate action, the 2030 targets will not be met [17]. There is an urgent need and an incredible opportunity amongst entomologists, researchers, and health organizations around the world, to use innovative approaches in the identification of mosquitoes at scale, which will, in turn, improve vector control and intervention strategies. Researchers have been exploring novel ideas to leverage budget smartphones as acoustic sensors [10]. Mobile services are expanding in sub-Saharan Africa [11] and these have the potential to become affordable, non-invasive, automated real-time mosquito monitoring tools that can be deployed in homes at no extra cost. The approach will help in producing much-needed surveillance data of mosquito species behavior and distribution over extensive temporal and spatial scales to assess ongoing vector-control measures. This AI-driven solution can replace time-consuming and expensive manual survey and classification tasks (e.g. Polymerase Chain Reaction (PCR) tests [22]) in remote and resource-constrained areas that bear the brunt of mosquito-borne diseases [13].

2 DATA

We use the publicly available dataset released as part of the ComParE challenge. The challenge dataset can be downloaded from Zenodo. Its attributes and the collection methods are described in detail in the Humbug system [10], while the splits and evaluation protocols are described as part of the challenge [20]. To validate model performance the organizers provide two development sets:

- Dev A: Recordings from Tanzania containing mosquito audio obtained from phones placed in bednets.
- Dev B: Recordings from the UK containing mosquito audio from lab-cultured mosquito larvae.

The baseline scores on the Dev sets are published as part of the challenge [20]. As evident from the scores, Dev B, which features a lower signal-to-noise ratio (SNR) [8], is a more challenging subset.

3 OUR APPROACH

We use a combination of a deep-learning model and data preprocessing techniques like Per-Channel Energy Normalization (PCEN), trainable kernels, and external data augmentation to create a model to detect mosquitoes. In the sections below we describe our unique data processing pipeline. A reproducible pipeline is available on our GitHub.[1]

3.1 Pre-Processing

To extract useful features from the audio samples, we convert them to their time-frequency representation: a spectrogram. This is done by applying the Short Time Fourier Transform (STFT) [5] operation to the audio signals. Converting an audio sample into its time-frequency representation is a common practice in extracting patterns from raw audio [1]. The spectrograms are then passed as input to a computer vision backbone [21] for further processing. Figure 2 below shows different representations of audio. The top portion represents the raw form (a time series), while the bottom part represents its time-frequency equivalent after applying STFT.

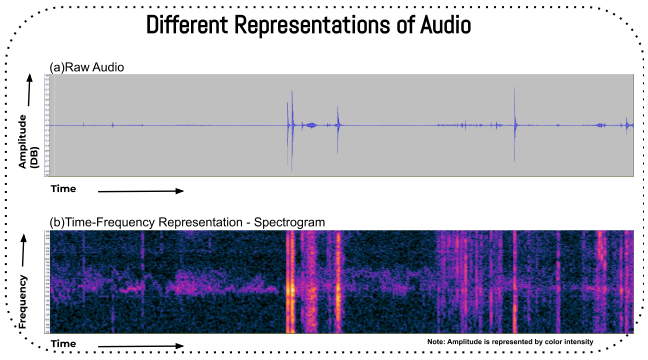

Figure 2: Waveform and Time-Frequency Representation of an Audio Signal

The audio samples in the dataset exhibit wide variations (owning to different recording conditions and the presence of background noise) and as a result, there is a wide variety of spectrograms that are sent as input to our backbone architecture. As an example consider Figure 3, which shows spectrogram representations of three randomly selected audio files in the dataset. The leftmost part shows a signal with uniform mosquito activity throughout its

[1] https://github.com/seancampos/ComParE2022_VecNet/

duration. The middle part shows a signal with sporadic instances of the presence of mosquito while the rightmost portion shows less noise and a clear pattern in fundamental mosquito frequency and its harmonics. The task to extract a discernible pattern gets even more complex due to the various hyperparameter settings of the STFT operation.

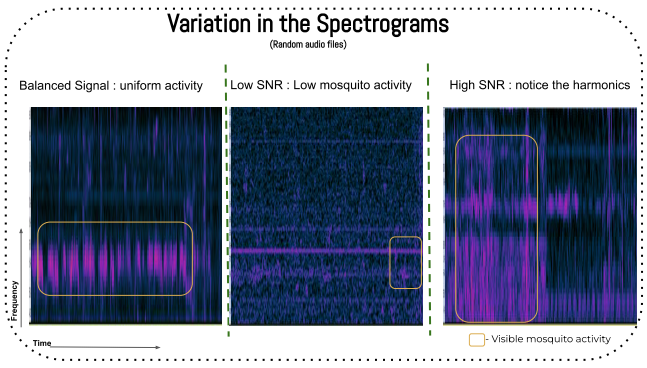

Figure 3: Spectrogram Representation of a Few Randomly Selected Files

3.2 Augmentation

We introduce random augmentations on the audio samples so that the model can learn to separate mosquito buzz from a variety of soundscapes. During training, for every other recording, we randomly sample files from an external database and add their corresponding tensor value to the tensor representation of the incoming audio (as indicated in Fig 1). The primary source of external sound files is the "noise" class in MUSAN [23], which is a general library of noise, speech, and music. We also supplement the raw files in the dataset with selected categories from Google Audioset [7] based on specific misclassifications in our model, including vehicles, other insects, and babies crying. Unlike the baseline solution, we do not discard any audio file owing to its duration. For a short audio file (less than 1.92 seconds), we either pad it with silence or augment it with audio from the two datasets, MUSAN and Audioset. Our training and inference sets consist of 1.92-second audio files. Please refer to our repository for examples. While random augmentations might impact reproducibility, but it adds robustness to the model and makes it more generic

3.3 Trainable Front-end

In deep learning based acoustic modeling the most widely used front-end is the log-mel front-end, consisting of melfilterbank energy extraction followed by log compression, where the log compression is used to reduce the dynamic range of filterbank energy. However, there are several issues with the log function. First, a log has a singularity [6] at 0. Common methods to deal with the singularity introduce uncertainity and may have different performance impacts on different signals. Second, the log function uses a lot of its dynamic range on low levels, such as silence, which is likely the least informative part of the signal. Third, the log function is loudness dependent. With different loudness, the log function can produce different feature values even when the underlying signalcontent (e.g. keywords) is the same, which introduces another factor of variation into training and inference. In their paper, Wang et al [24] introduce PCEN as an alternative to overcome the above issues with log filterbank. The key ingredient of PCEN is its use of automatic gain control [19]. The PCEN functions are also differentiable and hence they are included as a neural network layer so that it can learn from the attributes of the dataset [24]. PCEN is well-known to increase the performance of systems that work on keyword spotting tasks, hence we train a model that learns to spot mosquito buzz, using the same far-field detection method as Siri, Alexa, or Google [9].

3.4 Random Masking

We also randomly warp blocks of frequency channel and time steps. This approach has shown promising results in end-to-end ASR tasks [18]. This step is part of our pre-processing pipeline and we use it as a regularization technique to avoid overfitting.

3.5 Trainable Kernels

We move spectrogram generation from CPU to GPU and generate them on the fly in the training loop [4]. We found an approximately 250% speedup in generation time moving from an AMD 3.9 GHz 3990X Threadripper CPU to a NVIDIA V-100 GPU. Additionally, since spectrogram generation has a large number of hyperparameters that become fixed after generation, it becomes difficult to search for optimal parameters during model training. Moving the spectrogram generation as part of the model training process onto the GPU provides several benefits. Hyper-parameters can be searched without running a separate pre-processing pipeline for each iteration. Additionally, the STFT is implemented as a 1-D convolution, meaning that the kernel is trainable. Thus, the model can learn a custom Fourier transform suited for our bio-acoustic domain. Finally, at inference time, there is significant performance improvement and a simpler pipeline that does not require a pre-processing stage.

Figure 4 below shows how an audio signal is processed through our pipeline before going to a deep-learning backbone for prediction.

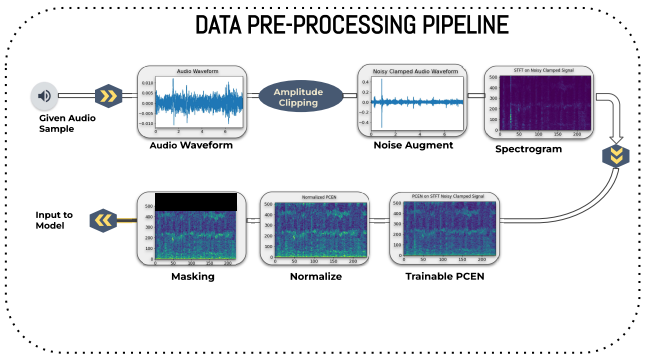

Figure 4: Data Pre-Processing on Raw Audio

4 ARCHITECTURE

We use Pytorch-based backbones to train our models. Figure 1 above represents an end-to-end pipeline from training to inference. We primarily focus on the Hierarchical Vision Transformer using Shifted Windows (Swin) architecture [14] and on CNNs (ConvNeXt) [15]. Our findings indicate that ConvNeXt outperforms Swin on this task. To perform an STFT, we use the default STFT parameters as defined in the baseline configuration[2].

5 RESULTS

Table 1 summarizes the Polyphonic Sound Detection Scores (PSDS) [2] obtained on the development sets by our models. The training was done on an NVIDIA V-100, 16GB GPU. We were able to achieve significant improvements on Dev B, which is a more challenging subset that has a lower SNR [20], without causing any reduction to the baseline score on Dev A. Our best-performing model showed an improvement of approximately 2100% over the published baseline. The ROC curves for PSDS for ConvNext is depicted in Figure 5.

Table 1: Summary of Dev Results (PSDS)

Backbone	Dev-Set	Baseline	Best-Score
ConvNext	**A**	**.614**	**.613**
ConvNext	**B**	**.034**	**.758**
Swin	A	.614	.610
Swin	B	.034	.040

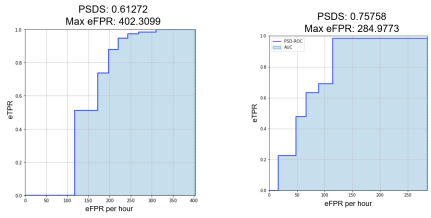

Figure 5: ConvNext (Left): PSDS Score on Dev A (Right): PSDS Score on Dev B

5.1 Test Set Results

The access to the test set was hidden and submissions were done by leveraging a docker template designed to automatically score the model results over the test data. Out of the allowed five submissions, our best performing model outperformed the published baseline by 212%. The table below summarizes the results (PSDS) obtained on the test data.

Table 2: Test Set Results (PSDS scores) on a ConvNext Backbone

Data-Set	Baseline	Best-Score
Test	.142	**.443**

[2]https://github.com/EIHW/ComParE2022/blob/MOS-C/src/config.py

6 EXPERIMENTS AND RESULTS: ABLATION STUDY

To verify the importance of the pre-processing techniques in a quantitative way, we performed an ablation study. We removed each component from our pipeline one at a time and retrained our model with the component missing. Figures 7 below show how the model performed with each component removed. The shorter bars are more important because that means that the model performs significantly worse when that component is missing from the pipeline. We notice that "random-masking" [18] on spectrograms and "PCEN" are the most significant components of the pipeline; without these components in place, the performance of the model suffers the most. This should come as no surprise, as indicated in the section on Trainable Front End (PCEN); the inclusion of this technique makes our model less vulnerable to the variations in the incoming audio. Random Masking, on the other hand, prevents overfitting.

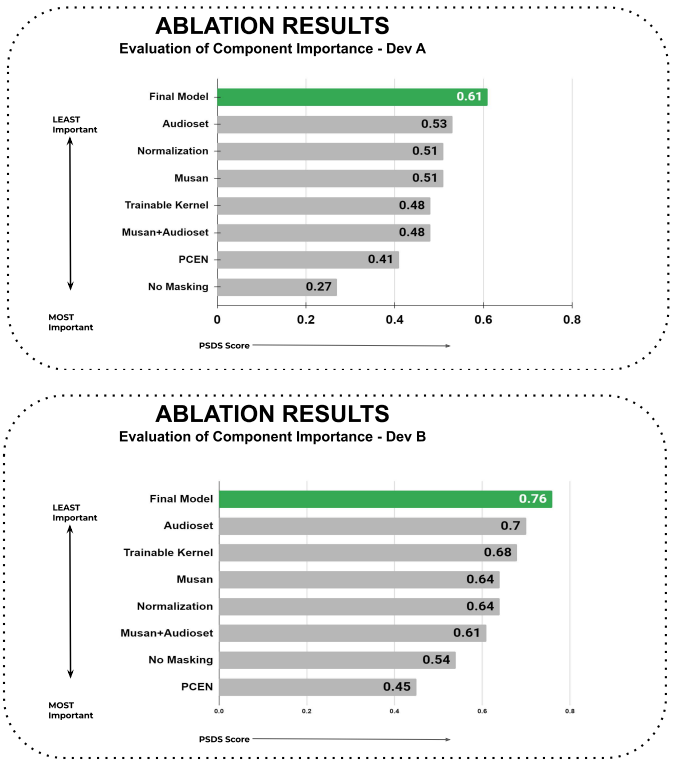

Figure 7: Ablation Study Results

7 CONCLUSION

Our models outperformed the published baseline on the test set by 212% and 2100% on Dev B, which is known to have a very low signal-to-noise ratio (SNR). The improved sensitivity to low SNR events comes without any decrease in performance on Dev A. While detecting the presence of mosquitoes in a real-world setting presents considerable challenges, our results show that with a well-designed pipeline and a judicious choice of neural net architecture, the baseline results can be significantly improved.

REFERENCES

[1] Richard A Altes. 1980. Detection, estimation, and classification with spectrograms. *The Journal of the Acoustical Society of America* 67, 4 (1980), 1232–1246.

[2] Çağdaş Bilen, Giacomo Ferroni, Francesco Tuveri, Juan Azcarreta, and Sacha Krstulović. 2020. A framework for the robust evaluation of sound event detection. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 61–65.

[3] Francesco Camastra and Alessandro Vinciarelli. 2015. *Machine learning for audio, image and video analysis: theory and applications*. Springer.

[4] Kin Wai Cheuk, Hans Anderson, Kat Agres, and Dorien Herremans. 2020. nnaudio: An on-the-fly gpu audio to spectrogram conversion toolbox using 1d convolutional neural networks. *IEEE Access* 8 (2020), 161981–162003.

[5] Lutfiye Durak and Orhan Arikan. 2003. Short-time Fourier transform: two fundamental properties and an optimal implementation. *IEEE Transactions on Signal Processing* 51, 5 (2003), 1231–1242.

[6] Philippe Flajolet and Andrew Odlyzko. 1990. Singularity analysis of generating functions. *SIAM Journal on discrete mathematics* 3, 2 (1990), 216–240.

[7] Jort F Gemmeke, Daniel PW Ellis, Dylan Freedman, Aren Jansen, Wade Lawrence, R Channing Moore, Manoj Plakal, and Marvin Ritter. 2017. Audio set: An ontology and human-labeled dataset for audio events. In *2017 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE, 776–780.

[8] Don H Johnson. 2006. Signal-to-noise ratio. *Scholarpedia* 1, 12 (2006), 2088.

[9] Byeonggeun Kim, Mingu Lee, Jinkyu Lee, Yeonseok Kim, and Kyuwoong Hwang. 2019. Query-by-example on-device keyword spotting. In *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*. IEEE, 532–538.

[10] Ivan Kiskin, Marianne Sinka, Adam D Cobb, Waqas Rafique, Lawrence Wang, Davide Zilli, Benjamin Gutteridge, Rinita Dam, Theodoros Marinos, Yunpeng Li, et al. 2021. HumBugDB: a large-scale acoustic mosquito dataset. *arXiv preprint arXiv:2110.07607* (2021).

[11] Nir Kshetri. 2022. Economics of the Internet of Things in Sub-Saharan Africa. *IT Professional* 24, 1 (2022), 81–85.

[12] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. 2015. Deep learning. *nature* 521, 7553 (2015), 436–444.

[13] Timothy M Lenton, Hermann Held, Elmar Kriegler, Jim W Hall, Wolfgang Lucht, Stefan Rahmstorf, and Hans Joachim Schellnhuber. 2008. Tipping elements in the Earth's climate system. *Proceedings of the national Academy of Sciences* 105, 6 (2008), 1786–1793.

[14] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. 2021. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 10012–10022.

[15] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. 2022. A convnet for the 2020s. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 11976–11986.

[16] Sylvie Manguin. 2013. *Anopheles mosquitoes: new insights into malaria vectors*. BoD–Books on Demand.

[17] April Monroe, Nana Aba Williams, Sheila Ogoma, Corine Karema, and Fredros Okumu. 2022. Reflections on the 2021 World Malaria Report and the future of malaria control.

[18] Daniel S Park, William Chan, Yu Zhang, Chung-Cheng Chiu, Barret Zoph, Ekin D Cubuk, and Quoc V Le. 2019. Specaugment: A simple data augmentation method for automatic speech recognition. *arXiv preprint arXiv:1904.08779* (2019).

[19] Juan Pablo Alegre Pérez, Santiago Celma Pueyo, and Belén Calvo López. 2011. *Automatic gain control*. Springer.

[20] Björn W. Schuller, Anton Batliner, Shahin Amiriparian, Christian Bergler, Maurice Gerczuk, Natalie Holz, Pauline Larrouy-Maestri, Sebastian P. Bayerl, Korbinian Riedhammer, Adria Mallol-Ragolta, Maria Pateraki, Harry Coppock, Ivan Kiskin, Marianne Sinka, and Stephen Roberts. 2022. The ACM Multimedia 2022 Computational Paralinguistics Challenge: Vocalisations, Stuttering, Activity, & Mosquitos. In *Proceedings ACM Multimedia 2022*. ISCA, Lisbon, Portugal. to appear.

[21] Mitt Shah, Nandit Pujara, Kaushil Mangaroliya, Lata Gohil, Tarjni Vyas, and Sheshang Degadwala. 2022. Music Genre Classification using Deep Learning. In *2022 6th International Conference on Computing Methodologies and Communication (ICCMC)*. IEEE, 974–978.

[22] Georges Snounou, Suganya Viriyakosol, William Jarra, Sodsri Thaithong, and K Neil Brown. 1993. Identification of the four human malaria parasite species in field samples by the polymerase chain reaction and detection of a high prevalence of mixed infections. *Molecular and biochemical parasitology* 58, 2 (1993), 283–292.

[23] David Snyder, Guoguo Chen, and Daniel Povey. 2015. Musan: A music, speech, and noise corpus. *arXiv preprint arXiv:1510.08484* (2015).

[24] Yuxuan Wang, Pascal Getreuer, Thad Hughes, Richard F. Lyon, and Rif A. Saurous. 2017. Trainable Frontend For Robust and Far-Field Keyword Spotting. In *Proc. IEEE ICASSP 2017*. New Orleans, LA.