TOWARDS A NEURAL COUGH CLASSIFIER FOR EDGE DEVICES

Hyfe Inc.

ABSTRACT

A persistent cough is a symptom that appears in several respiratory illnesses, but it can be difficult to monitor its frequency and severity accurately over a long period of time. The conventional approach involves subjective assessments by healthcare providers or patients reporting their coughing. As an alternative, Hyfe suggests an approach that embeds a microphone in wearable devices. Edge Impulse is selected as a development platform since it specializes in embedded devices for sensors, audio, and computer vision. The platform allows for the deployment of optimized ML on a wide range of hardware, from MCUs to CPUs and custom AI accelerators. Experiments are conducted to identify a classification method small in size and memory footprint while maintaining high performance. Hyfe's full dataset consisting of millions of cough/non-cough sounds is provided as a training/test set for experimentation. Based on this data, we propose novel cough detection methods using deep neural networks. Each sound is represented using Mel-Filterbank Energies (MFE). Experiments concluded that feature extraction and classification can be performed in less than 100 ms for an audio snippet of 0.5 s sampled at 16 kHz, with sensitivity of almost 91% and specificity of 99.7%. These results show that reliable and efficient cough detection in real-time on embedded devices is attainable.

Index Terms— Cough classification, acoustic signal, neural network, wearables, Edge Impulse

1. INTRODUCTION

Coughing is a prevalent and significant symptom reported by patients, and chronic coughing can have negative effects on both health and quality of life. Monitoring cough symptoms is crucial for detecting and treating respiratory conditions like COPD, asthma, pulmonary fibrosis, and tuberculosis. While subjective tests have been developed to assess the frequency and severity of coughs, these methods can be unreliable due to factors like patient mood and vigilance.

Cough counting using a computer or a mobile device allows subjective evaluation and observation to identify and quantify coughs in an automatic way. Digital Signal Processing (DSP) is applied in many works for this task [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14] since manual counting is a very laborious and error-prone task while it suffers from inter-expert subjectivity in classification.

Nevertheless, objective tools for studying coughs are currently lacking, but some systems, such as the Leicester Cough Monitor[1] and VitaloJAK[15], use wearable devices and algorithms to detect coughs from patient audio recordings. However, these systems can be semi-automated and require manual tuning or verification by personnel, and their validation is often limited to small datasets collected in artificial environments or with proprietary hardware. Recent interest in using deep learning techniques for automatic cough detection shows promise, but many of these methods have limited validation or use custom hardware for data collection.

Over the recent years, a group of technologies has emerged which enables running compact and well-tailored machine learning models on low-power microcontrollers. These microcontrollers can use machine learning to analyze sensor data right where it is generated, leading to more intelligent, faster, and less energy-consuming embedded applications. These applications can make decisions on their own instead of relying on cloud computing and waiting for a response. This idea is called TinyML, which is another term for embedded machine learning. Edge Impulse (EI) [16] leverages the TensorFlow ecosystem [17, 18] to train, optimize, and deploy deep learning models onto embedded devices. Despite being designed for non-expert engineers, EI follows a philosophy of being customizable and adaptable by machine learning specialists who can contribute their expertise through techniques such as hand-crafted model architectures and loss functions, and customized operator kernels.

In this work we describe Hyfe's attempts to identify and develop a classification method small in size and memory footprint while maintaining high performance. Among others, EI allows feature computation based on Mel-Filterbank Energies (MFEs), a set that is commonly used in audio processing. We developed neural models trained on MFE features aiming for a model that is lightweight, accurate, fast, and able to be deployed on many edge devices. For our purposes, the target device is an ARM Cortex M33 running at 128 MHz.

The rest of the paper is structured as follows: Section 2 describes the methodology, Section 3 presents the experimental setup and discusses the results, and Section 4 discusses integration with Hyfe SDK. Finally, Section 5 illustrates on-hardware challenges and real-time performance of the cough detection SDK pipeline and Section 6 concludes this work and suggests future directions.

2. METHODOLOGY

Hyfe's event detector [19] has been used to identify acoustic events in large sessions of audio. These events can be either cough or non-cough sounds. In order to compactly represent the audio input, MFEs were chosen. MFE features model the spectral energy distribution in a perceptually meaningful way (take into account human perception on the construction of the frequency scale), that is, MFEs are representations where the frequency bands are not linear but distributed according to the Mel-scale. This Mel-scale is defined as

$$Mel(f) = 2595 \log\left(1 + \frac{f}{700}\right) \qquad (1)$$

where f is the frequency in Hz, and $Mel(f)$ is the corresponding Mel-frequency. The Mel-scale is a perceptual scale of pitches judged by listeners to be equal in distance from one another. The Mel-filterbank is a set of triangular overlapping filters applied to the power spectrum (or magnitude spectrogram). These filters are spaced evenly on the Mel scale and are designed to mimic the non-linear sensitivity of the human auditory system to different frequencies. For each center frequency, we create a triangular filter, where the filter response is unitary at the center frequency and linearly decreases to 0 at the neighboring center frequencies. Each triangular filter in the Mel-filterbank is multiplied element-wise with the power spectrum (or magnitude spectrogram). The result is the so-called set of Mel-filterbank energies (MFEs), where each coefficient represents the energy in a specific frequency band on the Mel scale.

We trained three kinds of neural networks: Alison-based models (that is, models that rely on 2D-convolutional layers and have similar neural architecture to Alison, the current production model of Hyfe), 2D-CNNs (that is, models that also employ 2-dimensional convolutional layers but do not strictly follow Alison's architecture) and Fully Connected Deep Neural Networks (that is, models that consist of fully connected layers with dropout layers in-between). We also tested 1D-Convolutional Neural Networks but they were consistently performing poorly compared to the aforementioned models. Furthermore, we also used global pooling methods [20] in an attempt to reduce size and remove dense layers in the CNN-based models.

Besides performance, we are also very interested in low latency and small memory footprint. Thus we include inference time, peak RAM usage, and FLASH usage, along performance metrics, as provided by Edge Impulse platform. We remind that our current target chip is Nordic nRF5340 (Cortex-M33, 128MHz). The nRF5340 stands as the inaugural wireless System-on-Chip (SoC) to incorporate dual Arm Cortex-M33 processors. With its tandem of versatile processors, extensive advanced features, and the ability to function in temperatures reaching 105 C, it emerges as the prime selection for LE Audio, professional lighting, sophisticated wearables, and intricate Internet of Things (IoT) applications.

3. EXPERIMENTAL SETUP AND RESULTS

Cough sound duration is set to 0.5 s. The full dataset has been uploaded to EI platform and downsampled from 44.1 kHz to 16 kHz, encoded in 16-bit PCM files. An overview of this dataset is shown in Table 1.

	Dataset	
	Train	Test
Cough	85,705	560
Non-cough	3,388,168	7,191

Table 1. *Dataset details.*

A random 10% of the training dataset was selected as a validation set. Since we had no control on the split, some leak occurs in the validation set and validation performance is biased. However, test set contains no information from the training or validation set and thus model performance is objective.

For MFE extraction, we used an analysis window of 0.05 s with a hop size of 0.05 s, which is the maximum separation between frames without loss of information. The number of MFEs used were 40. An FFT of 256 bins was used in the spectral representation. An example of MFE features for two sounds is shown in Fig. 1.

A total of 40 MFE features were used as inputs to the models. The models were trained using Adam optimizer [21] using a categorical cross-entropy loss function. We employed a learning rate of 0.0001, and a variable batch size depending on the architecture but most of the models were trained with a batch size of 512. The network was set to be trained for 100 epochs. The model with the minimum validation loss was considered as the best one and moved to be evaluated on the test set. Python [22] and TensorFlow [17] were used in these experiments. EI platform provides confusion matrices, accuracy, precision and recall for each class as outputs. ROC-AUCs could not be computed from the platform, although they could be monitored for the validation set (and all of them were very close to 0.999).

Let us define TP as true positives (correctly classified coughs), TN as true negatives (correctly classified non-coughs), FP as false positives (incorrectly classified non-coughs as coughs), and FN as false negatives (incorrectly classified coughs as non-coughs). To measure classification performance, we present

- Sensitivity (or True Positive Rate - TPR): the probability of classifying a sound as cough, conditioned on the sound truly being a cough sound, i.e.

$$\text{Sensitivity} = \frac{TP}{TP + FN} \qquad (2)$$

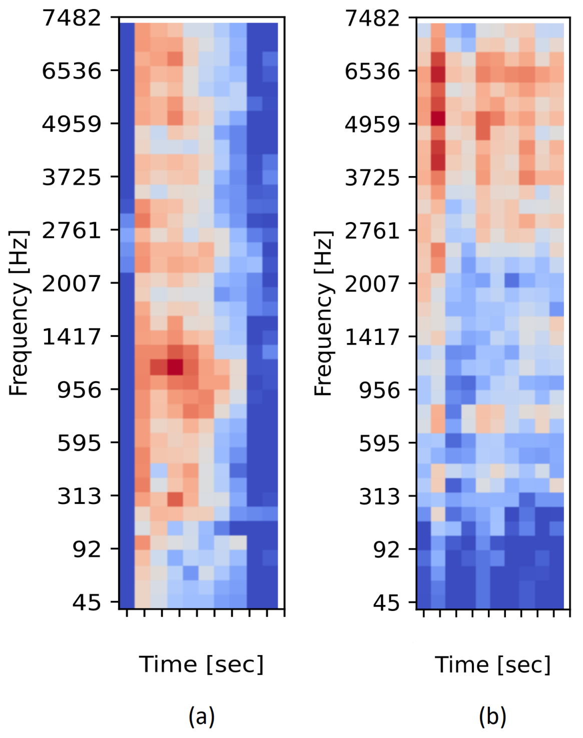

Fig. 1. *MFE features for (a) a cough sound, (b) another impulsive event [obtained from Edge Impulse platform].*

- Specificity (or True Negative Rate - TNR): the probability of classifying a sound as non-cough, conditioned on the sound truly being a non-cough, i.e.

$$\text{Specificity} = \frac{TN}{TN + FP} \tag{3}$$

- Precision (or Positive Predictive Value - PPV): the fraction of sounds truly classified as coughs among all sounds classified as coughs, i.e.

$$\text{Precision} = \frac{TP}{TP + FP} \tag{4}$$

and

- F1-score (for both classes): the harmonic mean of precision and sensitivity, i.e.

$$\text{F1-score} = \frac{2TP}{2TP + FP + FN}$$

$$= 2\frac{\text{Precision} \times \text{Sensitivity}}{\text{Precision} + \text{Sensitivity}} \tag{5}$$

Table 2 presents the performance of a small but representative subset of the actual models developed.

The chosen model in terms of performance and latency is a 2D-convolutional neural network followed by a series of fully connected layers. It achieved a sensitivity of 90.9%, a specificity of 99.7%, at a total (feature extraction plus classification) latency time of less than 100 ms.

4. SDK INTEGRATION

Hyfe has developed a Software Development Kit (SDK) that allows third parties to integrate Cough Detection into their own devices. It is programmed in C++ and has no external dependencies. It can therefore be built on all platforms: Windows, Linux, Mac OS, Android, iOS and embedded targets.

Table 2 presents model performance on audio chunks post event detection. In a real world scenario, audio is feed into the SDK in real-time, which then runs event detection on the audio stream and then feeds the events found as 0.5 audio chunks to the classification model (classifier). Since coughs missed by the event detector will never reach the classifier, the sensitivity of the event detector is a multiplicative factor to the sensitivity of the classifier when calculating the overall performance of the SDK. The SDK validation process involves feeding pre-recorded, hand-labeled, audio files (of lengths between 30 s and 5 minutes) and comparing the output of the SDK with the hand-labeled annotations.

However, there is always a trade-off between true positive and false positive rate in classification tasks. Adjusting the trade-off requires to tune the decision threshold (DT). DT is a value that determines how the model assigns input data to one of the two possible classes or categories (cough and non-cough). However, the decision threshold is not always fixed at 0.5, as one may assume. It can be adjusted based on the specific requirements of the problem and the trade-offs between different types of errors (Type I error: false positive, and Type II error: false negative). This is especially relevant in situations where the cost of false positives and false negatives is not equal. Thus, we present *overall sensitivity* versus *false positives per hour (FPph)* at specific thresholds in Table 3.

5. REAL TIME COUGH DETECTION ON AN EMBEDDED TARGET

Running real-time cough detection on embedded devices presents many challenges. Among them, one that stands out is the trade-off between accuracy and real-time audio processing feasibility. Can a good enough model which can run in real-time on a severely under-resourced, in terms of computation, environment be devised? Table 4 shows the amount of time each component takes in the execution pipeline.

The maximum number of cough events that can be detected in a 500 ms input audio chunk equals 3. Of course, the actual number of events found in such a chunk varies depending on the acoustic environment. Table 5 illustrates the differences in CPU usage for three different acoustic scenes.

	Test set				Performance: ARM Cortex-M33, 128 MHz					
Model /Metrics	F1-score (C/NC)	Sensitivity	Specificity	Precision	Inferencing Time	Peak RAM usage	FLASH usage	MFE latency	MFE RAM	Total latency
DNN	0.89 / 0.99	82.3%	99.7%	0.96	25 ms	2.0K	187.9K	17 ms	7.3K	42 ms
Alison	0.92 / 0.99	88.8%	99.6%	0.95	149 ms	23.0K	208.4K	17 ms	7.3K	166 ms
Reduced Alison$_1$ with GAP	0.92 / 0.99	88.6%	99.7%	0.96	69 ms	20.9K	58.1K	17 ms	7.3K	86 ms
Reduced Alison$_2$ with GMP	0.92 / 0.99	89.1%	99.6%	0.95	67 ms	20.8K	58.1K	17 ms	7.3K	84 ms
Reduced Alison$_3$ with GAP	0.91 / 0.99	89.3%	99.4%	0.93	79 ms	21.2K	67.8K	17 ms	7.3K	96 ms
2D-CNN plus Dense$_1$	0.94 / 1.00	91.3%	99.7%	0.96	97 ms	19.3K	309.1K	17 ms	7.3K	114 ms
2D-CNN plus Dense$_2$	0.93 / 1.00	90.9%	99.7%	0.96	78 ms	19.3K	172.6K	17 ms	7.3K	95 ms

Table 2. *Model performance using MFEs. Selected model in red.*

Hyfe SDK Performance of 2D-CNN plus Dense		
Sensitivity	FPph	DT
85.29%	3.58	0.5
82.72%	2.26	0.6
81.99%	1.88	0.7
80.51%	1.51	0.75

Table 3. *Sensitivity vs False Positives per Hour (FPph) at a specific Decision Threshold (DT) on an ARM Cortex-M33.*

6. CONCLUSIONS AND FUTURE WORK

Edge Impulse platform is used to develop and train models for cough classification on an ARM Cortex-M33. A variety of neural architectures were tested, ranging from simple fully connected to complex two-dimensional convolutional neural networks. Besides model performance, inference time and small memory footprint are crucial factors for edge devices. Our best model (2D-CNN model followed by a fully connected network) achieved a sensitivity of 90.9% with specificity of 99.7% when trained on Mel-Filterbank Energy features. Feature extraction and classification is performed in less than 100 ms for a 0.5 s sound signal sampled at 16000 Hz. Using Hyfe's SDK, cough detection requires 108.8 ms, on average, with a sensitivity of 81.99% and 1.88 false positives per hour. This work demonstrates that reliable and efficient cough detection in real-time on embedded devices is attainable. Further work can be focused on increasing sensitivity while approximately maintaining the same performance on hardware.

7. REFERENCES

[1] SS Birring, T Fleming, S Matos, AA Raj, DH Evans, and ID Pavord, "The leicester cough monitor: preliminary validation of an automated cough detection system in chronic cough," *European Respiratory Journal*, vol. 31, no. 5, pp. 1013–1018, 2008.

[2] Filipe Barata, Kevin Kipfer, Maurice Weber, Peter Tinschert, Elgar Fleisch, and Tobias Kowatsch, "Towards device-agnostic mobile cough detection with convolutional neural networks," in *2019 IEEE International Conference on Healthcare Informatics (ICHI)*. IEEE, 2019, pp. 1–11.

[3] Thomas Drugman, Jerome Urbain, and Thierry Dutoit, "Assessment of audio features for automatic cough detection," in *2011 19th European Signal Processing Conference*. IEEE, 2011, pp. 1289–1293.

[4] Renard Xaviero Adhi Pramono, Syed Anas Imtiaz, and Esther Rodriguez-Villegas, "Automatic cough detection in acoustic signal using spectral features," in *2019 41st Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*. IEEE, 2019, pp. 7153–7156.

[5] Jesús Monge-Álvarez, Carlos Hoyos-Barceló, Paul Lesso, and Pablo Casaseca-De-La-Higuera, "Robust detection of audio-cough events using local hu moments," *IEEE journal of biomedical and health informatics*, vol. 23, no. 1, pp. 184–196, 2018.

[6] Yan Shi, He Liu, Yixuan Wang, Maolin Cai, and Weiqing Xu, "Theory and application of audio-based assessment of cough," *Journal of Sensors*, vol. 2018, 2018.

[7] Brian H Tracey, Germán Comina, Sandra Larson, Marjory Bravard, José W López, and Robert H Gilman, "Cough detection algorithm for monitoring patient recovery from pulmonary tuberculosis," in *2011 Annual international conference of the IEEE engineering in medicine and biology society*. IEEE, 2011, pp. 6017–6020.

[8] Matthijs D Kruizinga, Ahnjili Zhuparris, Eva Dessing, Fas J Krol, Arwen J Sprij, Robert-Jan Doll, Frederik E

Components	Execution time	Notes
Event detection	85 ms	Runs once for every 500 ms of audio
MFE features	17 ms	Computed for each event found
Inference	78 ms	Same as above

Table 4. *Execution time for each component in the cough detection pipeline on an ARM Cortex-M33.*

Environment	CPU usage in ms (%)	Notes
Silence	85 (17%)	Best case scenario
Loud and noisy	85 + (17+78) $\times$ 3 = 370 (74%)	Worst case scenario
Real-life conditions	85 + (17+78) $\times$ 0.25 = 108.8 (21.8%)*	On average

Table 5. *CPU usage of the cough detection pipeline in three different acoustic environments on an ARM Cortex-M33.* **On average, the event detector finds* ≈ 1800 *events/hour, which translates to* $0.25/500$ *ms.*

Stuurman, Vasileios Exadaktylos, Gertjan JA Driessen, and Adam F Cohen, "Development and technical validation of a smartphone-based pediatric cough detection algorithm," *Pediatric Pulmonology*, vol. 57, no. 3, pp. 761–767, 2022.

[9] Jaclyn A Smith, Kimberley Holt, Rachel Dockry, Shilpi Sen, Kitty Sheppard, Philip Turner, Paul Czyzyk, and Kevin McGuinness, "Performance of a digital signal processing algorithm for the accurate quantification of cough frequency," *European Respiratory Journal*, vol. 58, no. 2, 2021.

[10] Mingyu You, Huihui Wang, Zeqin Liu, Chong Chen, Jiaming Liu, Xiang-Huai Xu, and Zhong-Min Qiu, "Novel feature extraction method for cough detection using nmf," *IET Signal Processing*, vol. 11, no. 5, pp. 515–520, 2017.

[11] YH Hiew, JA Smith, JE Earis, Barry MG Cheetham, and AA Woodcock, "Dsp algorithm for cough identification and counting," in *2002 IEEE International Conference on Acoustics, Speech, and Signal Processing*. IEEE, 2002, vol. 4, pp. IV–3888.

[12] Mingyu You, Zeqin Liu, Chong Chen, Jiaming Liu, Xiang-Huai Xu, and Zhong-Min Qiu, "Cough detection by ensembling multiple frequency subband features," *Biomedical Signal Processing and Control*, vol. 33, pp. 132–140, 2017.

[13] Thomas Drugman, "Using mutual information in supervised temporal event detection: Application to cough detection," *Biomedical Signal Processing and Control*, vol. 10, pp. 50–57, 2014.

[14] Samantha J Barry, Adrie D Dane, Alyn H Morice, and Anthony D Walmsley, "The automatic recognition and counting of cough," *Cough*, vol. 2, no. 1, pp. 1–9, 2006.

[15] K McGuinness, K Holt, R Dockry, and J Smith, "P159 validation of the vitalojak™ 24 hour ambulatory cough monitor," *Thorax*, vol. 67, no. Suppl 2, pp. A131–A131, 2012.

[16] Shawn Hymel, Colby Banbury, Daniel Situnayake, Alex Elium, Carl Ward, Mat Kelcey, Mathijs Baaijens, Mateusz Majchrzycki, Jenny Plunkett, David Tischler, et al., "Edge impulse: An mlops platform for tiny machine learning," *arXiv preprint arXiv:2212.03332*, 2022.

[17] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al., "Tensorflow: a system for large-scale machine learning.," in *Osdi*. Savannah, GA, USA, 2016, vol. 16, pp. 265–283.

[18] Robert David, Jared Duke, Advait Jain, Vijay Janapa Reddi, Nat Jeffries, Jian Li, Nick Kreeger, Ian Nappier, Meghna Natraj, Tiezhen Wang, et al., "Tensorflow lite micro: Embedded machine learning for tinyml systems," *Proceedings of Machine Learning and Systems*, vol. 3, pp. 800–811, 2021.

[19] Hyfe Inc., "On the onset detection of cough audio signals," Tech. Rep., Hyfe Inc., 01 2023.

[20] Min Lin, Qiang Chen, and Shuicheng Yan, "Network in network," *arXiv preprint arXiv:1312.4400*, 2013.

[21] Diederik P Kingma and Jimmy Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[22] Guido Van Rossum, Fred L Drake, et al., *Python reference manual*, Centrum voor Wiskunde en Informatica Amsterdam, 1995.