Deep Neural Networks for Identifying Cough Sounds

Justice Amoh, *Student Member, IEEE*, and Kofi Odame, *Senior Member, IEEE*

***Abstract*—In this paper, we consider two different approaches of using deep neural networks for cough detection. The cough detection task is cast as a visual recognition problem and as a sequence-to-sequence labeling problem. A convolutional neural network and a recurrent neural network are implemented to address these problems, respectively. We evaluate the performance of the two networks and compare them to other conventional approaches for identifying cough sounds. In addition, we also explore the effect of the network size parameters and the impact of long-term signal dependencies in cough classifier performance. Experimental results show both network architectures outperform traditional methods. Between the two, our convolutional network yields a higher specificity 92.7% whereas the recurrent attains a higher sensitivity of 87.7%.**

***Index Terms*—Deep learning, machine learning, medical devices, wearables.**

I. Introduction

THE frequency and intensity of a patient's cough can be instrumental in the diagnosis and treatment of many respiratory conditions such as asthma, tuberculosis and pneumonia [1], [2]. Unfortunately, patient self-reports of cough history are often unreliable, as they tend to be fabricated or remembered inaccurately [3]. To circumvent the problems of self-reporting, automated cough detection systems have been proposed [4], [5], which provide an objective cough count by analyzing and classifying the patient's vocalizations.

The challenge with designing an automated cough detection system is that it must be highly accurate in order to be of any practical value. Since coughs are relatively rare events, even a cough detector with moderate specificity would produce an unacceptably large number of false alarms. Achieving high specificity is non-trivial, because coughs share similar acoustic properties to other sounds like throat-clearing, sneezing, laughter and even speech. In addition to high specificity, the detector must also exhibit good sensitivity in order not to under-report how often the patient is coughing.

The most common approach to addressing these challenges is to find features of the acoustic signal that offer good discriminability between coughs and non-cough sounds [7]. A wide array of features has been considered by previous researchers: classical speech recognition features such as the Mel-Frequency Cepstral Coefficients (MFCCs) and Linear Predictive Coding (LPC) analysis coefficients [4], [8]; adaptations of speech recognition features [9]; and custom-designed hand-crafted features [10].

Manuscript received February 8, 2016; revised May 12, 2016 and August 1, 2016; accepted August 1, 2016. This work was supported by the Neukom Institute for Computational Science at Dartmouth College. This paper was recommended by Associate Editor S. Ostadabbas.

The authors are with the Thayer School of Engineering, Dartmouth College, Hanover, NH 03755 USA (e-mail: justice.amoh.jr.th@dartmouth.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TBCAS.2016.2598794

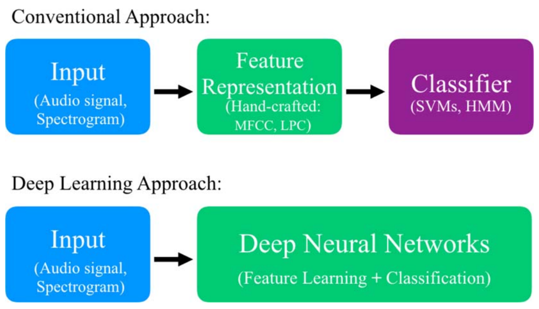


Fig. 1. An overview of the conventional approach for cough detection in comparison with the deep learning approach. The deep learning approach eliminates the need for hand-crafted feature representations by automatically learning task specific features directly from the audio data [6].

The attraction of custom-designed features is that they can take advantage of cough-specific characteristics that might not be evident from the more generic speech recognition features. Also, custom features are appropriate for our envisioned continuous monitoring application, where the sensor data is produced not by high-fidelity microphones, but instead by transducers that emphasize long-term wearability (size, physical comfort, battery life, privacy laws) over signal quality and linearity [10]–[12].

To avoid the cost of hand crafting each set of features for every new transducer, we previously proposed using a convolutional deep neural network to automatically identify an appropriate set of custom features [12].

Here we explore the deep neural network approach to identifying coughs more broadly by addressing questions of (1) what signal processing construct is appropriate for analyzing a potential cough event (2) the effect of the number of layers and number of neurons in a network and (3) the impact of long-term signal dependencies in cough classifier performance.

II. Problem Formulations

A. Visual Recognition

Cough detection can be cast as a visual recognition task by converting the one dimensional acoustic data into a two dimensional spectro-temporal'image', by computing, for example, a Short-Time Fourier Transform (STFT). Deep neural networks

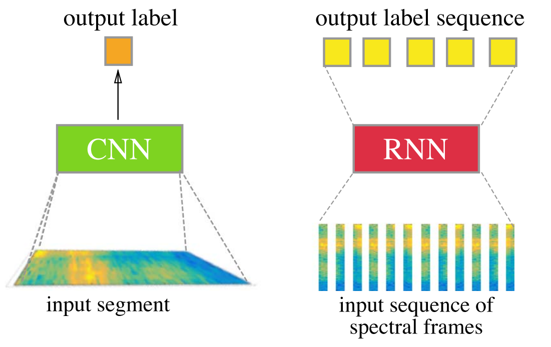


Fig. 2. An illustration of the convolutional and recurrent neural networks for the two formulations of cough detection. In the CNN, the input is a fixed-sized image, a segment of the spectrogram, and the output is a single label. The RNN takes in a sequence of spectral frames, and outputs a sequence of labels.

that work well in image recognition can then be employed to discriminate between cough and non-cough sounds based on'visual' patterns in the spectral contents. The advantage of this formulation is that the thoroughly studied neural network variant for images, Convolutional Neural Networks (CNN) [13], [14], can be adopted for cough detection (Fig. 2). Given their widespread usage recently, CNNs are very easy to train and there are a lot of software and hardware resources available for this.

There is, however, a major drawback to the visual formulation. It requires the input data dimensions to be fixed and known a-priori. While this works for traditional images since they originally have fixed 2D sizes, it proves challenging for time series data such as audio. With the same camera, all images would have the same size, but an audio recording of different coughs would have different lengths in time. Hence, a pre-segmentation step is needed to ensure spectro-temporal data from the acoustic signal are of a fixed dimension before feeding into the neural network. Irrespective of whatever pre-segmentation scheme is adopted, there will also arise edge cases where there is less content than the fixed segment width. Such segments would have to be either discarded or zero-padded. Discarding edge segments would reduce the amount of data available for training and this is not encouraged since deep learning works better with more data. On the other hand, zero-padding segments with little original content would introduce some ambiguity in training examples, which will in effect reduce classification accuracy. Besides the pre-segmentation requirement, the visual formulation would also require an additional post-processing stage to align predicted output labels with the audio signal.

B. Sequence-To-Sequence Labelling

A sequence-to-sequence labeling formulation sidesteps the pre-segmentation and post-processing challenges of visual recognition. The goal in sequence-to-sequence labeling is to map one unsegmented sequence (input data) to another (output labels). Speech recognition, handwriting recognition and machine translation are typical examples of sequence-to-sequence learning problems. Specific to our application, sequence-to-sequence labeling can potentially model the long-term dependencies in a cough sound. In particular, it can capture the temporal and spectral dependencies between the three characteristic phases (initial burst, middle phase and the final burst) of a cough [7], [15].

To cast our cough identification task in this light, the spectro-temporal data from the acoustic signal can still be used except considered as a time series of spectral features. Unlike in visual recognition, we can have varying input sequence lengths in this formulation; this removes the need to discard or pad training samples. Furthermore, since output labels are also sequential, there is no need for post-processing or alignment of predicted labels. A classical model for solving the sequence labeling problem is the Hidden Markov Model (HMM) [16]. However, HMMs are limited by their ability to capture long- term dependencies [17]. During HMM training, remote past events have much less influence on the state variable than more recent events. The resulting model is therefore fairly independent of remote past inputs and outputs [17]. Recurrent Neural Networks (RNNs) are variants of neural networks that enable a deep learning approach for handling sequence-to- sequence labeling problems (Fig. 2) [18]. While RNN models have traditionally also suffered the long-term dependency limitation, two recent developments have helped circumvent that issue. First, it has been shown that a hierarchy of RNNs (deeper networks) can better model long-term dependencies since they afford both fine and coarse temporal resolutions of sequences [19]. Secondly, there are now new types of neurons (Long Short-Term Memory units) that specifically enable RNNs to have more control over their internal memory [20]. As a result, unlike HMMs, current deep RNNs with the specialized neurons are able to capture and model long-term contexts in sequences. Besides that, RNNs are also known to be robust to noise in sequential data [21].

The downside to RNN based learning of sequence labeling is that the training process can be difficult and requires a lot of examples. As a result, past RNN applications failed to perform as expected for such sequential tasks like speech recognition [22]. However, recent studies in deep learning have introduced additional techniques and modifications such as gradient clipping, other types of neurons and attention mechanisms, which have made RNN training tractable and more efficient [22]–[24]. The exponential growth in computational resources over the years has also contributed significantly to RNN training and to all deep learning methods. Currently, RNNs have achieved state of the art performances in most sequence to sequence learning problems including speech recognition and machine translation [24], [25].

III. Visual Recognition Implementation

A. Preprocessing

During preprocessing, the stream of acoustic data is segmented into frames which are each 4 ms long. To eliminate irrelevant data such as silence and background noise, the preprocessor implements the frame admission process suggested by *Lu et al.* [26]. For each 16-frame (64 ms) window, the RMS energy is calculated and compared with a predetermined threshold. Windows with low energy are assumed to be silence or ambient sounds and are discarded whereas high energy ones are

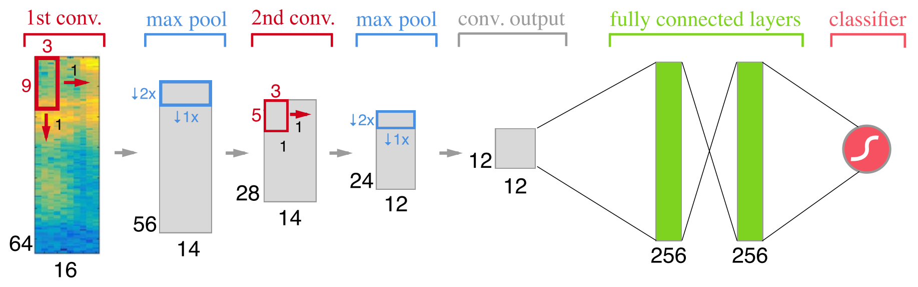


Fig. 3. A depiction of the architecture of our CNN. The input to the network is a 64 ms STFT spectrogram. Network consists of 2 convolutional layers, 2 dense layers and a softmax classification layer. Each conv. layer has 16 filters, with rectified linear unit (ReLU) activations. The filters have sizes 9×3 and 5×3 respectively and convolution is performed with a stride of 1. Filter and pooling sizes are chosen to encourage learning of patterns across both temporal and spectral domains.

"admitted." Since some sounds like speech can be intermittent or discontinuous, there is a possibility of low energy windows occurring within one sound event. To avoid discarding such windows, once a window is admitted, it is assumed to depict the start of an acoustic event and the next 4 windows are admitted as well irrespective of their energies. Hence, a minimum of 320 ms event is admitted at a time, which is also the average length of cough sounds. Admitted acoustic events are normalized by a running RMS average, and then undergo the spectro-temporal conversion.

For each admitted event, a 128-bin Short Time Fourier Transform (STFT) is performed yielding a spectrogram with 64 frequency points and varying number of time frames depending on the length of the event. The spectrograms are segmented into 16-frame segments with 4 frame overlaps corresponding to 64 ms of the original sound data. For edge cases where there a fewer than 16 frames of content, the segment is zero-padded accordingly. The 64×16 spectral segments are each assigned a single label pertaining to the class (cough or non-cough) from which they were obtained. These spectral segments and their labels are the inputs which are fed to the CNN for classification.

B. *Convolutional Neural Network Architecture*

Our Convolutional Neural Network architecture was inspired by the popular LeNet-5 architecture [27] which yielded state of the art performance on the MNIST handwritten digits dataset. Compared to other well known architectures such as the AlexNet [14], LeNet-5 is a much smaller network and more suitable for smaller datasets. However, since our dataset is even smaller than the MNIST dataset, we reduced the number of neurons in each layer in accordance with common heuristics such as ensuring number of hidden units are only a fraction of the input.

Like the LeNet-5, our network consists of five layers: 2 convolutional layers, 2 fully connected layers and a softmax classification layer. Each convolutional layer has 16 rectified linear units (ReLU). The first convolutional layer takes in the 64×16 spectral segments as inputs, and has filters of size 9×3. This is followed by a 2×1 max-pooling layer. The second convolutional layer has filters of size 5×3 and is also followed by a 2×1 max-pooling layer. Convolutions are performed with a stride of 1. The convolutional layers are followed by 2 fully connected layers with 256 rectified linear units for each layer. The fully connected layers also employ dropout regularization ($p = 0.5$) to reduce overfitting. Finally, the last layer takes the outputs of the second fully connected layer and classifies the input as either a cough or speech event using the softmax function. The network architecture is illustrated in Fig. 3.

We chose ReLU activations over the traditional tanh or sigmoid functions because ReLU does not have the vanishing gradient problem and often leads to a faster convergence [14]. The convolutional filter sizes are chosen to enable 2D convolutions: across both frequency and temporal domains. Previous applications of convolutional networks in audio sometimes convolved along either time or frequency axis [28]. For our application however, since we know both short-term temporal and spectral patterns can be discriminative for cough and speech events, we convolve along both dimensions. In addition, since our input segments cover a relatively short time window (16 frames, 64 ms), we fix the size of the filters along the time axis (at 3 frames). Pooling layers down-sample outputs of convolutions to make computations manageable in subsequent layers. Similar to our filter sizing, we perform no pooling along the time axis to avoid further reducing the limited temporal resolution of segments.

IV. Sequence-to-Sequence Labeling Implementation

A. *Preprocessing*

The signal is vectorized into 4 ms frames, and a frame admission protocol is employed to reject silence and low energy windows. In this case, however, admitted windows can have varying lengths (as opposed to the fixed 64 ms). The admitted windows are converted to sequences of 64 frequency components per frame, which is a sequential perspective of the STFT spectral segments from the visual recognition set up. These sequences and their corresponding labels are employed to train the recurrent neural network.

B. *Recurrent Neural Network Architecture*

We implemented our RNN with a 6 layer encoder-decoder architecture, which allows the network to process and classify input sequences of arbitrary length [24]. The encoder is made

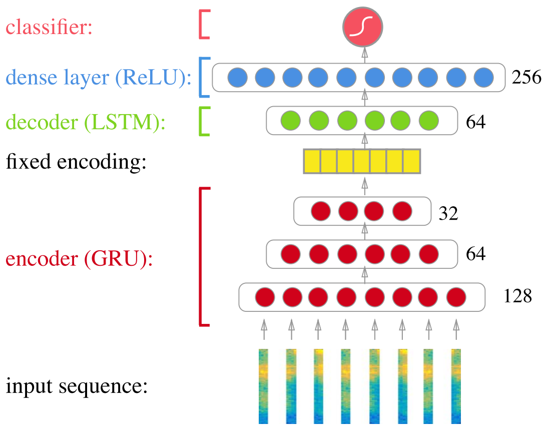
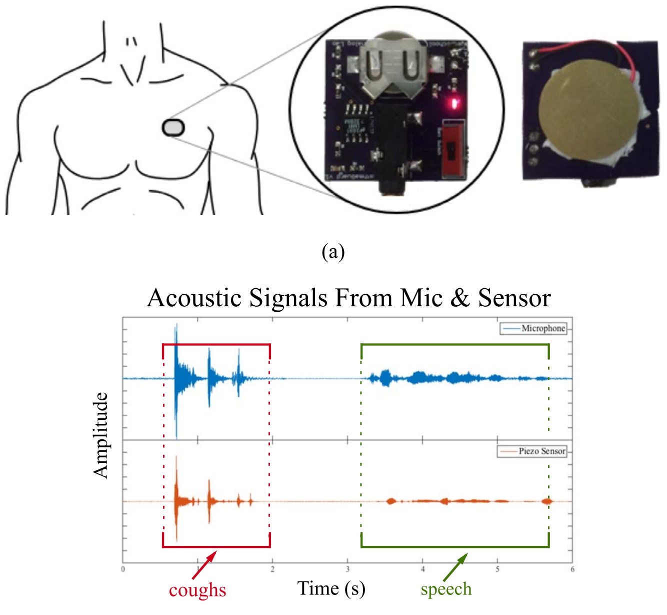


Fig. 4. An overview of our encoder-decoder RNN architecture for cough detection. The encoder consists of three layers; the first two have bidirectional units and the third is unidirectional. All neurons in encoder are Gated Recurrent Units (GRU). The decoder is a single LSTM layer with a built in attention mechanism. Next is a dense layer with ReLU units, and finally a softmax classification layer.



(b)

Fig. 5. (a) An illustration of how the proposed sensor is worn. Front and back images of sensor show the front-end electronics and the piezoelectric transducer. (b) Plots of cough and speech sounds from a standard microphone (top) and from the piezo sensor (bottom). Speech sounds are evidently more attenuated in the sensor output.

up of 3 layers: 2 bidirectional recurrent layers with 128 and 64 units respectively, and a unidirectional layer with 32 recurrent units. Our encoder is set up to process sequences of up to a certain maximum length we set depending on the experiment (see experiment section below). All recurrent neurons in the encoder are Gated Recurrent Units (GRU), which are able to identify long-term dependencies in a sequence of input data [24]. The last layer of the encoder outputs a fixed representation (the 32 activations) which is then used to initialize the decoder. The decoder is a single recurrent layer of 64 Long Short Term Memory (LSTM) units [20], combined with an attention mechanism. The attention mechanism enables the network to focus on salient parts of the input features, and ultimately results in improved classification performance [25]. Currently, our decoder is set up to output a single label for each input sequence. Following the decoder, we have a fully connected layer with 256 ReLU neurons. Finally the classification layer outputs a class label using the softmax function. The encoder-decoder model is also illustrated in Fig. 4.

V. Experimental Methods

A. Cough Sensor

For measuring the cough sounds, we devised a wearable sensor that acquires a patient's respiratory and vocal sounds in real-time. The sensor, shown in Fig. 1(a), is attached to the chest via a medical-grade foam adhesive. Once worn, it streams lung and abdominal auscultation sounds to a computer or smartphone for further processing and classification of events. In contrast to the conventional condenser microphones used in most prior work [10], [29], ours is a more application-specific sensor consisting of a piezoelectric transducer and additional signal conditioning electronics that enhance acoustic events of interest. In particular, the sensor amplifies respiratory sounds, attenuates voiced speech and eliminates environmental sounds altogether. Also, the contact piezo-transducer captures the additional vibrational energy that coughs induce in the body, and thus, further amplifies cough signals as illustrated in Fig. 5 (b). From our earlier work, we observed that our piezo cough sensor works just as well as a conventional microphone in a controlled environment, and would potentially have an advantage in a noisy external environment; therefore we will focus solely on the piezo sensor data for the experiments in this paper.

B. Data Collection

To build and evaluate the proposed system, we created a database of lung sound recordings from 14 healthy volunteers: 7 males and 7 females. Subject ages range from 21 to 30 years, with ethnicities consisting of African, Asian and Caucasian. All subjects provided informed consent, and the experimental protocol was approved by the Dartmouth College Institutional Review Board. The piezo sensor was used to collect acoustic data as subjects were asked through a series of procedures including producing forced bouts of coughs, reading provided prompts and breathing normally. Each subject produced an average of 40 cough sounds, yielding a total 627 cough examples in our database. For the speech data, subjects read 20 phonetically-balanced prompts. The prompts used are two lists from the Harvard Sentences that are used for speech quality measurement and telephone systems testings [30]. Besides the cough and speech sounds, we also identified a third category of sounds which we will refer to as“other sounds” from here on. Other sounds included all non-cough and non-speech sounds that the sensor picked up such as breathing, heartbeats, crackings, and even the beep sounds used to signal subjects during procedures. For all recordings, the piezo sensor was sampled at a 44.1 kHz rate and later down-sampled to 16 kHz. All events in recordings (cough, speech, or others) were manually labelled on a PC using the Audacity sound editor.

The duration of coughs in all the collected data ranges from 250 ms to a maximum of about 800 ms. On the other hand, speech and other sounds tend to be much longer in the time range. To ensure our training examples are of the same length, we split the speech and other sound recordings into smaller segments with random durations generated from a Gaussian distribution of the durations of the cough examples. This results in an average duration of around 320 ms for all examples in our database, irrespective of their class.

C. Network Training

Before training, a small subset of the database is run several times in different configurations to find optimal training hyper-parameters (e.g., learning rate, momentum, etc.). Once these hyper-parameters are set, we proceed to build the actual networks with the complete data. For training both neural networks, a 10-fold cross-validation scheme is employed. The database is divided into 10 parts: 9 parts for building the model, and the remaining part for testing the model. This is done iteratively over all parts in 10 different runs and average evaluation metrics are computed. All data is shuffled before partitioning and hence there is a likelihood of identity overlap between training and testing data. In other words, it is possible that audio samples from the same subject are in both training and testing data. However, in the experiments section, we investigate how well our networks generalize to sounds from completely unseen subjects.

Deep networks benefit significantly from a large set of training data. So meaningful data augmentation schemes that increase the number of training examples can be useful. In our application, input data is augmented to introduce some translational invariance in the learning. This is done by re-buffering the spectral segments from the same events to have a 25% overlap. Our training database resolves into 11,126 segments with which we train our network. We also standardize the entire training data across all components as is often done in training deep neural networks. Below, we highlight the additional training details of the two different networks.

1) CNN Training: The convolutional network is trained using stochastic gradient descent (SGD), with a learning rate of 0.001, batch size of 20 and a Nesterov's momentum of 0.9. The network has 660,690 learnable parameters and training converges after roughly 50 epochs, with an average run time of about 5 hours for all 10-folds.

2) RNN Training: The recurrent network has 323,983 parameters and is trained using the'adadelta' optimizer: a gradient descent method with an adaptive learning rate, that is less sensitive than other optimizers to the initial learning rate hyper-parameter [31]. Although adadelta was not the fastest optimizer for the application, it was found to converge more smoothly during training, and to yield better test accuracies than other optimizers tried such as vanilla SGD, rmsprop [32], and adagrad [33]. An initial learning of 0.005 and a mini-batch size of 40 are used. The network is trained for 35 epochs, which takes about 7 hours for all 10-folds.

Other techniques employed to make the recurrent net training effective were gradient clipping and batch normalization. Gradient clipping imposes a threshold on gradients as a means to restrict the effect any one neuron's activation can have on the overall loss during back-propagation [34]. Gradient clipping is found to make training of recurrent nets tractable as reported in other studies [35]. Batch normalization is a recently introduced technique for addressing the'internal covariate shift problem': the change in distribution of network activations in the course of training [36]. It involves normalizing the inputs to each layer using the mini-batch statistics. Batch normalization improves training speed, acts as a regularizer to reduce over fitting, and generally leads to higher validation accuracies even in recurrent nets [37].

Both networks are implemented using Lasagne [38], a Theano-based python library for training neural networks. Training is performed on a single PC with a 3.3 GHz Core i7 CPU, and 16 GB of memory.

D. Experiments

To compare and evaluate the performance of the two deep learning based formulations of cough detection, five experiments are undertaken First, we investigate the claim that both neural networks extract features that are effective for identifying cough segments. Next, we compare the CNN and RNN with each other on a more rigorous classification task to further explore their cough discriminatory abilities. In the third experiment, we investigate how well both the CNN and RNN capture long-term dependencies by testing both models on longer sequences. In the fourth experiment, we verify how well our models perform on data from subjects that are outside of our database. Finally, in the last experiment, we examine how the performance of both networks is affected by their sizes. All experiments are performed in a 10-fold cross validation scheme and performance metrics are averaged over all folds.

Our metrics for evaluating the models are sensitivity, specificity and accuracy. Sensitivity or true positive rate, is computed as the ratio correctly identified coughs to the total number of coughs in a test set. Specificity, the false negative rate, refers to the ratio of correctly identified non-coughs to the total number of non-coughs. Accuracy, a composite of both sensitivity and specificity, is the ratio of correctly identified samples (whether coughs or non-coughs) to the total number of samples in the test set.

1) Experiment 1: To verify how effective the learned RNN and CNN features are for cough classification, we compare them against the commonly used MFCC features. In this experiment, we only focus on two classes: cough and speech sounds. We extract 13 MFCC coefficients from the sounds in our database using an analysis window width of 32 ms and a hop length of 16 ms (50% overlap). These analysis parameters are similar to those typically used in cough studies and speech recognition [8], [39]. The MFCC analysis yields 3 frames for every 64 ms of sound. The MFCC features are therefore fragmented into 13×3 segments to make for a comparable set up with the spectral segments used to train the CNN and RNN. Also, although the RNN can handle sequences of arbitrary length, we set a maximum sequence length of 16 frames (64 ms) to enable a direct comparison with the CNN and MFCC features at that time resolution. With this approach, each feature extraction method (CNN, RNN, and MFCC) yields meaningful features for classifying any 64 ms audio segment as a cough event or not.

Since the classification layer of both the RNN and CNN are softmax functions, a softmax function (SM) is also trained on the MFCC features. With the same type of classifier and approximately the same number of features, a direct comparison of the classification accuracies will inform on the representational abilities of the MFCC, CNN, and RNN features with respect to our cough detection task. We also train a radial basis function Support Vector Machine (SVM) on the MFCC features to observe how a more complex classifier compares with the deep neural networks. In addition, we train an SVM on the raw STFT data to serve as a reference bar for comparison with the RNN and CNN features.

2) Experiment 2: In the second experiment, we use the same setup from the first experiment, except we only focus on the CNN and RNN. The discrimination task is made more realistic by including sounds other than cough and speech that the audio sensor would measure in a practical setting. These sounds include: heartbeat sounds, breathing sounds, laughter, throat-clearing, cracklings and sounds resulting from physical disturbances to the sensor (e.g., subjects touching the sensor). The goal is to evaluate the networks' performance in a multi-class classification problem.

3) Experiment 3: Here, we investigate the extent to which both network architectures can capture long-term dependencies and whether this improves cough detection. This is done by running both models on longer sequences. Since this experimental setup allows both models to be run on whole cough events, we also compare with the conventional MFCC-Hidden Markov Model cough detection approach [8]. As in the first experiment, we focus only on two classes: cough and speech sounds.

While the Hidden Markov Model (HMM) and the RNN can both handle input sequences of variable length, the CNN model also requires a fixed input. Hence, we set our maximum sequence length as the average duration of coughs in our database: 320 ms. This is 5 times the previously used window length (64 ms, 16 frames) and yields 64×80 spectral segments. Database entries with longer durations are split into two with a 25% overlap, and zero-padded as necessary. To adjust the CNN model accordingly, we scale the width of its temporal convolutions by 5. The RNN's input sequence length is also updated for a maximum of 80. Both networks are then retrained on the revised data.

Using the same data and testing framework, an MFCC-HMM model is implemented for comparison. An HMM with 10 states is trained for each class. The first and last states are non-emitting, but all middle states have an emission probability distribution modeled by a 7-dimensional mixture of Gaussians. For each training example, 13 MFCC coefficients are computed in the same way as in experiment 3, except this results in a longer 15 frame sequence for the elongated training examples. The 13×15 MFCC features are then used to train the HMMs. At test time, a similar feature vector sequence extracted from the test example is fitted to both HMMs. The resulting log-likelihood values of both fits determines whether the sound pertains to a cough or speech event. This HMM configuration is fairly common in cough studies and speech recognition [8].

TABLE I
EXPERIMENT 1: COMPARISON OF CNN, RNN, AND MFCC FOR COUGH CLASSIFICATION

Model	Sensitivity (%)	Specificity (%)	Accuracy (%)
MFCC+SM	94.3 ± 3.1	68.5 ± 9.4	81.4 ± 3.6
MFCC+SVM	74.9 ± 7.6	91.1 ± 1.5	87.6 ± 4.8
STFT+SVM	76.9 ± 3.4	74.4 ± 4.8	77.2 ± 3.3
STFT+CNN	86.8 ± 1.5	92.7 ± 2.4	89.7 ± 1.5
STFT+RNN	87.7 ± 7.9	82.0 ± 11.6	84.9 ± 3.6

4) Experiment 4: In the fourth experiment, we investigate the network performance when there is no identity overlap between the training and testing data. Both models are tested on samples from two subjects outside of the database; one male and one female. The test data in this set up consists of 128 samples each, of cough and speech sounds. The sounds are collected using the piezo-sensor in the same way as the original database sounds were collected. The test is to verify that our models generalize well to coughs from unseen persons.

5) Experiment 5: The neural network size is characterized by two parameters: the number of hidden units in a layer, and the total number of layers in the network. In the final experiment, we examine how modifying either of these parameters affects model performances. To investigate the effects of the number of layers, we train networks with half as many layers as there are in the original models. This results in a smaller 3-layer network for both the RNN and CNN, compared to the original 5-layer CNN and 6-layer RNN. The three layers are: the first convolutional or recurrent layer from the original models, a 256-unit fully connected layer and the final sigmoid classification layer. We also train a 3-layer regular dense neural network for comparison. For the number of units, we generate multiple network models by reducing the number of units in each layer of the original by a factor of 2, 4, and 8. For instance the'half' RNN model, corresponding to a reduction of 2, has 64, 32, 16, 32, 128, 1 number of units in the 6 respective layers (from the original 128, 64, 32, 64, 256, 1 configuration).

VI. RESULTS

The results for Experiment 1 are reported in Table I. First, we observe that both neural network models perform better than merely training an SVM on the raw data, which is the baseline test. In addition, the two networks seem to perform better than both MFCC based models. While the MFCC with softmax (MFCC + SM) appears to have a high sensitivity, it admits a lot of false positives and results in a poor accuracy. Comparing the CNN and RNN directly, the CNN yields an overall higher accuracy of 89.7%. While the RNN appears to yield a slightly better mean sensitivity across the 10-folds, it has a much higher variance than the CNN. On the other hand, CNN attains a significantly larger specificity than the

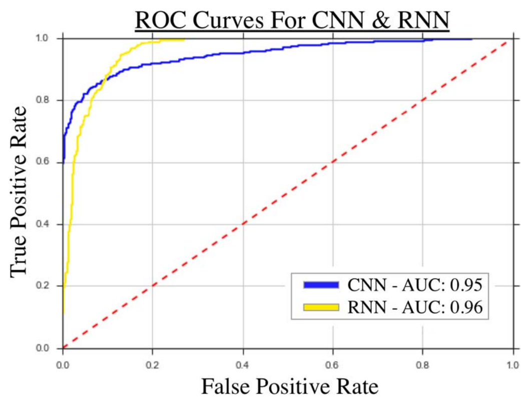


Fig. 6. ROC Curve generated by varying threshold on the output of the last node in networks. RNN appears to have a slightly higher AUC of 0.96 than the CNN.

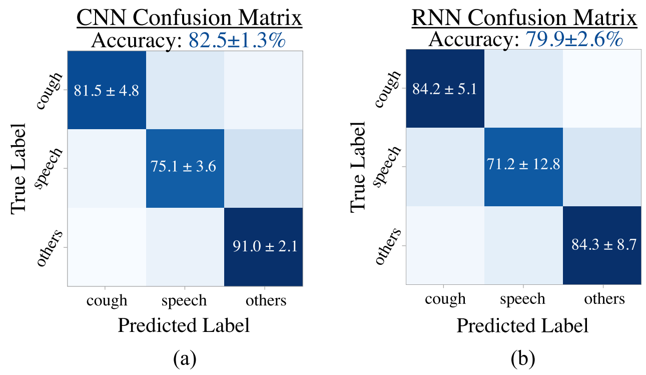


Fig. 7. Experiment 2: Confusion matrices for (a) CNN and (b) RNN in multi-class classification problem. The task involves discriminating between the three categories: cough, speech and other sounds. CNN attains a higher overall accuracy of 82.5%, although RNN compares well across all classes.

RNN, with a relatively minimal standard deviation. We also generate a Receiver Operating Characteristic (ROC) curve for both networks by varying the threshold on the output of the last sigmoid unit (Fig. 6). This informs on how far apart the networks separate the two classes. From the curves, we observe both networks do fairly well in this regard, with ROC Area Under the Curve (AUC) values of 0.96 (RNN) and 0.95 (CNN).

Fig. 7 shows the confusion matrices for both the CNN and RNN in the harder multi-class classification problem of Experiment 2. As expected, classification accuracy drops in general for the two networks. However, we still observe a higher accuracy for the CNN (82.5%) than for the RNN (79.9%). Across the three classes, we observe the same trend as in the first experiment, where cough sensitivity is slightly higher in the RNN case whereas non-cough (speech and other) accuracies remain significantly higher in the CNN.

In Table II, we report the performance of the CNN, RNN, and MFCC-HMM models in experiment 3. Note that the RNN, with a classification accuracy of 85.5% appears to perform better than both the CNN and the MFCC-HMM model. The CNN seems to perform almost as well as the MFCC-HMM model, although with a much larger variance. Compared to the RNN, the long-term adjusted CNN performance is significantly lower

TABLE II
EXPERIMENT 3: PERFORMANCE ON LONGER SEQUENCES

Model	Sensitivity (%)	Specificity (%)	Accuracy (%)
MFCC-HMM	79.1 ± 11.7	80.8 ± 5.9	79.9 ± 4.0
CNN	76.2 ± 24.6	82.2 ± 6.4	79.2 ± 15.0
RNN	81.7 ± 16.9	89.29 ± 18.4	85.5 ± 8.6

TABLE III
EXPERIMENT 4: PERFORMANCE ON UNSEEN SUBJECTS

Model	Sensitivity (%)	Specificity (%)	Accuracy (%)
CNN	82.0	93.2	87.6
RNN	84.2	75.2	79.7

and this seems to suggest that the CNN lacks in its ability to capture long-term dependencies.

Also, Table III shows the performance of the networks on samples from unseen subjects. The sensitivity, specificity and accuracy values for both RNN and CNN are within the ranges that were observed for the 10-fold tests of Experiment 1.

Finally, Fig. 8 shows the box plots comparing the performance of both CNN and RNN with different size configurations across the 10-folds. The 3-layer networks attain a better performance than the original models with accuracies of 90.9% for CNN, and 88.2% for RNN [Fig. 8(a)]. In comparison, a regular fully connected 3-layer network exhibited an accuracy of 82.8% ± 2.5. On the other hand, when the number of units is halved, the CNN accuracy falls whereas the RNN performance improves [Fig. 8(b)]. Further reduction in number of units beyond the half yields a poorer performance in both models. This trend is more visible for the CNN than the RNN since the RNN accuracy appears to remain fixed. However, observing the sensitivity and specificity values for the RNN models reveals that beyond the reduction by two, specificity only increases at the expense of sensitivity [Fig. 8(c)]. As such, the optimal number of units for the RNN appears to be half those in the original model. The reason this is optimal is because the specificity is maximized, with the accuracy remaining roughly the same; we want very high specificity for a rare event such as a cough. The half-unit CNN and RNN models yield accuracies of 85.3% and 87.6% respectively. In general, it is noted that RNN seems to outperform CNN in models with fewer units, whereas the vice-versa is true for models with fewer layers.

VII. DISCUSSION

From the first experiment and the results in Table I, we can confirm that our neural network models are indeed learning good features. This is made evident by the fact that they outperform the baseline SVM classifier on the raw STFT. Furthermore, both models perform better than the MFCC based ones, attesting to the notion that deeply learned features are more effective than hand-crafted once for cough detection. An interesting observation, however, is that the MFCC + SVM model attains very high specificity. A possible reason for this could be that since MFCCs are designed specifically for speech, they extract good features for speech. Since specificity in experiment 1 refers to correctly identifying speech sounds, the

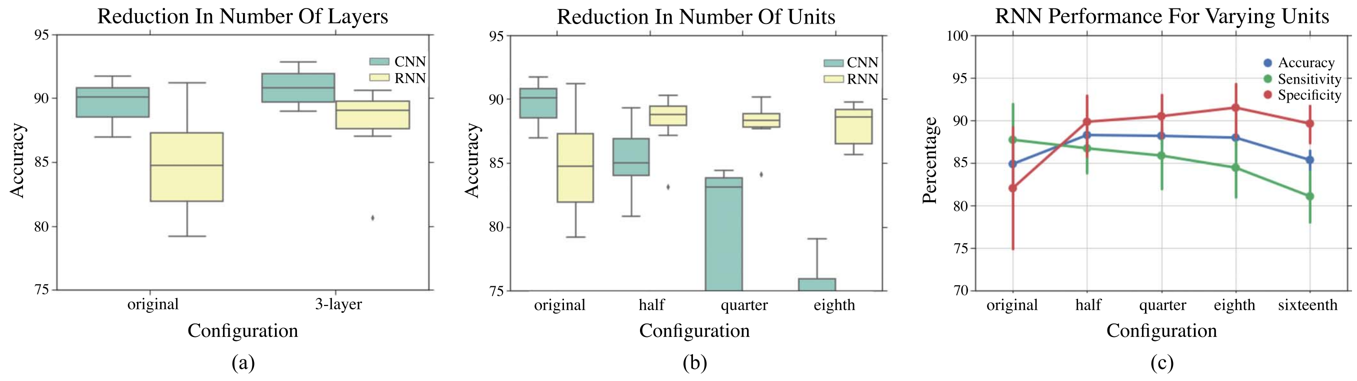


Fig. 8. Experiment 5: Plots comparing accuracies of networks with different size configurations. (a) Box-plots for accuracies of original models and 3-layer models. The 3-layer network attains better performance with accuracies of 90.9% (CNN) and 88.2% (RNN). (b) Box-plots for models with number units reduced by a factor of 2 (half), 4 (quarter), and 8 (eighth) as compared to original model. (c) Comparison of RNN accuracy, sensitivity and specificity across models with reduced number of units.

extra benefit of MFCCs is observed when the fairly robust SVM classifier is used.

Across most experiments (1, 2, and 5) we observed the intriguing pattern where CNN yields a much higher specificity whereas the RNN yields a better cough sensitivity. One thought that could explain this is that the CNN does much better at detecting speech because the speech spectra has characteristic harmonics and patterns that are more defined than the cough spectrum. The CNN, which is really good at capturing visual patterns, is able to better model such cues in the spectrum than the RNN. On the other hand, it can be reasoned that the RNN yields a better sensitivity because the sequence labeling formulation is more true to the actual cough detection task.

Another observation is that RNN outperforms both the CNN and the MFCC-HMM on the longer sequences. The GRU and LSTM units of the RNN enable it to better model the long-term dependencies in cough sounds. The CNN yields accuracies similar to the HMM, especially when one considers the high variance of its accuracies across the 10-folds. The CNN's under-performance is interesting since we showed in our previous work [12] that CNNs can yield good performance on long sequences if their output labels for short windows are averaged over the entire, longer sequence. Considering both factors, we suppose that it is better to use CNN on short sequences than on longer ones. In general, performance decreases for the longer sequences and this makes sense as the number of training examples decrease when the sequences are elongated.

Concerning the network parameters, we noted that the 3-layer models performed better than our original models. This could mean that our initial models were slightly overfitting our data. More layers affords a neural network more nonlinearity and hence increases its complexity. A more complex model can easily overfit to any given training data but would perform poorly on test data. The 3-layer models are less complex than the original models and thus, their better performance hints that the original model is likely overfitting the training data. That said, reducing the number of units is seen to hurt performance in general although the optimal number of units for the RNN is half of those in the original model. Finally, we show that our networks generalize well to unseen subjects, yielding performances close to that of entries in training sets.

VIII. Conclusion

In summary, our work first casts cough detection in the lights of visual recognition and of sequence labeling problems. We implemented a convolutional and a recurrent neural network for tackling the two formulations respectively. From our model evaluations, we show that both networks are able to learn good features for the cough discrimination task. We identified that for our dataset and set up, the CNN yields a better specificity whereas the RNN produces the better sensitivity. We also show how changing factors such as input sequence length, classification task and network parameters affects model performance.

Even though we chose our models and hyper-parameter values manually, the resulting networks still performed better than traditional classifiers. An interesting future study would be to evaluate the performance of an optimal network, after using the techniques of Grid/Random Search [40] or Bayesian Optimization [41] to systematically explore the network model and hyper-parameter space.

Acknowledgment

The authors would like to thank Prof. L. Toressani (Computer Science Department, Dartmouth College) for discussions and suggestions on deep learning methods and resources.

References

[1] S. S. Birring, S. Matos, R. B. Patel, B. Prudon, D. H. Evans, and I. D. Pavord, "Cough frequency, cough sensitivity and health status in patients with chronic cough," *Respiratory Med.*, vol. 100, pp. 1105–1109, Jun. 2006.

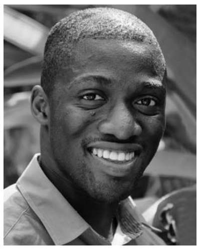
[2] L. J. Toop, C. W. Thorpe, and R. Fright, "Cough sound analysis: a new tool for the diagnosis of asthma," *Family Practice*, vol. 6, pp. 83–85, Jun. 1989.

[3] P. Verschelden, A. Cartier, J. L'archeveque, C. Trudeau, and J. L. Malo, "Compliance with and accuracy of daily self-assessment of peak expiratory flows (PEF) in asthmatic subjects over a three month period," *Eur. Respiratory J.*, vol. 9, no. 5, pp. 880–885, 1996.

[4] S. J. Barry, A. D. Dane, A. H. Morice, and A. D. Walmsley, "The automatic recognition and counting of cough.," *Cough (London, U.K.)*, vol. 2, p. 8, Jan. 2006.

[5] S. Matos, S. S. Birring, I. D. Pavord, and D. H. Evans, "An automated system for 24-h monitoring of cough frequency: the leicester cough monitor.," *IEEE Trans. Biomed. Eng.*, vol. 54, pp. 1472–1479, Aug. 2007.

[6] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 8, pp. 1798–1828, 2012.
[7] J. Amoh and K. Odame, "Technologies for developing ambulatory cough monitoring devices," *Crit. Rev. Biomed. Eng.*, vol. 41, no. 6, 2013.
[8] S. Matos, S. Member, S. S. Birring, I. D. Pavord, D. H. Evans, and S. Member, "Detection of cough signals in continuous audio recordings using hidden Markov models," *IEEE Trans. Biomed. Eng.*, vol. 53, no. 6, pp. 1078–1083, 2006.
[9] T. Drugman, J. Urbain, and T. Dutoit, "Assessment of audio features for automatic cough detection," in *Proc. 19th Eur. Signal Processing Conf.*, no. 32, 2011.
[10] E. C. Larson, T. Lee, S. Liu, M. Rosenfeld, and S. N. Patel, "Accurate and privacy preserving cough sensing using a low-cost microphone," in *Proc. 13th Int. Conf. Ubiquitous Computing*, p. 375, 2011.
[11] S. S. Kraman, G. R. Wodicka, G. A. Pressler, and H. Pasterkamp, "Comparison of lung sound transducers using a bioacoustic transducer testing system," *J. Appl. Phys.*, vol. 101, no. 2, pp. 469–476, 2006.
[12] J. Amoh and K. Odame, "DeepCough: A Deep Convolutional Neural Network in A Wearable Cough Detection System," in *Proc. IEEE Biomedical Circuits and Systems Conf.*, pp. 1–4, 2015.
[13] B. B. Le Cun, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Handwritten digit recognition with a back-propagation network," *Adv. Neural Inf. Process. Syst., Citeseer*, 1990.
[14] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Adv. Neural Inf. Process. Syst.*, pp. 1–9, 2012.
[15] P. M. Olia, P. Sestini, and M. Vagliasindi, "Acoustic parameters of voluntary cough in healthy non-smoking subjects," *Respirology (Carlton, Vic.)*, vol. 5, pp. 271–275, Sep. 2000.
[16] L. R. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proc. IEEE*, pp. 257–286, 1989.
[17] Y. Bengio and P. Frasconi, "Diffusion of context and credit information in Markovian models," *J. Artif. Intell. Res.*, vol. 3, pp. 249–270, 1995.
[18] P. J. Werbos, "Backpropagation through time: What it does and how to do it," *Proc. IEEE*, vol. 78, no. 10, pp. 1550–1560, 1990.
[19] S. E. Hihi and Y. Bengio, "Hierarchical recurrent neural networks for long-term dependencies," *Nips*, pp. 493–499, 1995.
[20] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1–32, 1997.
[21] K.- C. Jim, C. L. Giles, S. Member, and B. G. Horne, "An analysis of noise in recurrent neural networks: Convergence and generalization," *IEEE Trans. Neural Netw.*, vol. 7, no. 6, pp. 1424–1438, 1996.
[22] R. Pascanu, T. Mikolov, and Y. Bengio, "On the difficulty of training recurrent neural networks," *arXiv preprint arXiv:1211.5063*, 2012.
[23] A. Graves, "Generating sequences with recurrent neural networks," *arXiv preprint arXiv:1308.0850*, pp. 1–43, 2013.
[24] K. Cho, B. van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," *Proc. Conf. Empirical Methods in Natural Language Processing*, pp. 1724–1734, 2014.
[25] J. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio, "*Attention-based models for speech recognition*," pp. 1–19, 2015.
[26] H. Lu, W. Pan, N. Lane, T. Choudhury, and A. Campbell, "SoundSense: scalable sound sensing for people-centric applications on mobile phones," in *Proc. 7th Int. Conf. Mobile Systems, Applications, and Services*, pp. 165–178, 2009.
[27] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.
[28] H. Lee, P. Pham, Y. Largman, and A. Y. Ng, "Unsupervised feature learning for audio classification using convolutional deep belief networks," in *Adv. Neural Inf. Process. Syst.*, pp. 1096–1104, 2009.
[29] S. S. Birring, T. Fleming, S. Matos, A. A. Raj, D. H. Evans, and I. D. Pavord, "The leicester cough monitor: Preliminary validation of an automated cough detection system in chronic cough," *Eur. Respiratory J.*, vol. 31, no. 5, pp. 1013–1018, 2008.
[30] E. H. Rothauser, W. D. Chapman, N. Guttman, H. R. Silbiger, and J. L. Sullivan, "IEEE recommended practice for speech quality measurements," *IEEE Trans. Audio and Electroacoust.*, vol. AU-17, no. 297, pp. 225–246, 1969.
[31] M. D. Zeiler, "ADADELTA: An adaptive learning rate method," *arXiv*, p. 6, Dec. 2012.
[32] Y. N. Dauphin, H. de Vries, J. Chung, and Y. Bengio, "RMSProp and equilibrated adaptive learning rates for non-convex optimization," *arXiv preprint arXiv:1502.04390*, 2015.
[33] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," *J. Mach. Learn. Res.*, vol. 12, pp. 2121–2159, 2011.
[34] R. Pascanu, T. Mikolov, and Y. Bengio, "Understanding the exploding gradient problem," *Comput. Res. Repository (CoRR) abs/1211.5063*, 2012.
[35] Y. Bengio, N. Boulanger-Lewandowski, and R. Pascanu, "Advances in optimizing recurrent networks," in *Proc. IEEE Int. Conf. Acoustics, Speech and Signal Processing*, pp. 8624–8628, 2013.
[36] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv*, 2015.
[37] C. Laurent, G. Pereyra, P. Brakel, Y. Zhang, and Y. Bengio, "Batch normalized recurrent neural networks," *arXiv preprint arXiv:1510.01378*, 2015.
[38] S. Dieleman, J. Schlüter, C. Raffel, E. Olson, S. K. Sønderby, D. Nouri, D. Maturana, M. Thoma, E. Battenberg, J. Kelly, J. D. Fauw, M. Heilman, Diogo149, B. McFee, H. Weideman, Takacsg84, Peterderivaz, Jon, Instagibbs, D. K. Rasul, CongLiu, Britefury, and J. Degrave, "*Lasagne: First Release.*," Aug. 2015.
[39] C. Ittichaichareon, S. Suksri, and T. Yingthawornsuk, "Speech recognition using MFCC," in *Proc. Int. Conf. Computer Graphics, Simulation and Modeling*, pp. 135–138, 2012.
[40] J. Bergstra, R. Bardenet, Y. Bengio, and B. Kegl, "Algorithms for hyperparameter optimization," *Adv. Neural Inf. Process. Syst.*, pp. 2546–2554, 2011.
[41] J. Snoek, O. Rippel, K. Swersky, R. Kiros, N. Satish, N. Sundaram, M. Mostofa Ali Patwary MOSTOFAALIPATWARY, I. Prabhat PRABHAT, and L. P. Ryan Adams, "Scalable Bayesian optimization using deep neural networks," in *Proc. 32nd Int. Conf. Machine Learning*, 2015.



Justice Amoh (S'13) received the A.B. and B.E. degrees in electronic and computer engineering from Dartmouth College, Hanover, NH, USA.

Currently, he is working toward the Ph.D degree at Thayer School of Engineering at Dartmouth College. His research interests are embedded systems, machine learning, and wearable devices.

Kofi Odame (S'06–M'08–SM'15) is an Associate Professor of Electrical Engineering at the Thayer School of Engineering at Dartmouth College, Hanover, NH, USA. His primary interest is in analog integrated circuits for nonlinear signal processing. This work has applications in low-power electronics for implantable and wearable biomedical devices, as well as in autonomous sensor systems.