# A Deep Neural Network with Triplet Loss for Detecting Anomaly of Respiratory Sounds

Lam Pham[1], Dat Ngo[2], Alexander Schindler[1], Ross King[1]
[1] *Center for Digital Safety & Security, Austria Institute of Technology, Austria,*
*Email: {lam.pham, Alexander.Schindler, Ross.King}@ait.ac.at*
[2] *Electrical & Electronic Deparment, Ho Chi Minh City University of Technology, Viet Nam*
*Email: datt.ngo.hcmut@gmail.com*

## Abstract

In this paper, we propose a deep learning based system for detecting anomaly of respiratory sounds. The system is separated into two main steps: front-end feature extraction and back-end classification. In the first step, audio recordings of respiratory cycles collected from patients are transformed into Gammatone-filter based spectrograms, where both temporal and frequency features of respiratory sounds are presented. In the second step, a convolutional neural network (CNN) based architecture, referred to as the baseline, is proposed to detect whether the input sound contains anomaly. To further improve the baseline performance, we then propose a triplet-based network architecture in which parallel CNN baseline networks and triplet loss function are combined. To make our work comparable, we evaluate our systems on the 2017 Internal Conference on Biomedical Health Informatics (ICBHI), which is one of the largest public benchmark respiratory sound datasets. Our experimental results show that the proposed CNN baseline and triplet-network architecture outperform the ICBHI baseline, improving by 5.0% and 7.0%, respectively.

*Clinical relevance*— Wheeze, crackle, deep neural network, triplet loss, data augmentation.

## Introduction

According to the World Health Organization [1], the respiratory illness is one the most common mortality factors worldwide. The record [1] indicates that around 10 million people have tuberculosis (TB), 65 million people currently have chronic obstructive pulmonary disease (COPD), and 334 million people suffer from asthma. To deal with the morality from respiratory diseases, early detection, which is useful to prevent spread and further enhance treatment, is considered as the most effective way. The fact is that anomaly of lung sounds such as *Crackles* or *Wheezes* normally occur in respiratory cycles of patients who suffer from relevant-lung diseases. Therefore, detection of these abnormal respiratory sounds during a lung auscultation (i.e. an important aspect of a medical examination) is an effective way for early diagnosis of respiratory diseases. This motivates to develop automatically detected tools, which are able to detect abnormal sounds and aim to apply to a wider population. Recently, analysis of respiratory sounds has drawn increasing attention and many attempts have been made using machine learning, particularly deep learning. Machine learning systems proposed in [2] and [3] used Mel-frequency cepstral coefficients (MFCC) as extracting features. Next, MFCC features are fed into machine learning models such as Hidden Markov Model [2, 3], Support Vector Machine [4].Deep learning based systems made use of spectrogram where both spectral and temporal information are well presented. To explore spectrograms, diverse deep learning networks such as CNN [5] or RNN [6] have been proposed. Compared to systems relying on more conventional machine learning methods, deep learning based systems have been shown to be more advanced, evidenced by the comprehensive performance comparison in [6]. Inspired by the efficacy of deep learning based systems used for analysing respiratory sounds, a deep learning network combined with triplet loss is proposed in this paper. The proposed system is used for detecting respiratory cycles defined in 2017 Internal Conference on Biomedical Health Informatics (ICBHI) benchmark dataset [7].

## ICBHI dataset, task defined, data splitting, and evaluated metrics

The public ICBHI benchmark dataset consists of 920 audio recordings collected from a total of 128 patients over 5.5 hours. The recordings show a wide range of sampling frequencies ranging from 4 to 44.1 kHz and various lengths ranging from 10 to 90 s. In each audio recording, there are multiple of respiratory cycles marked with onset and offset time and labelled with one of four categories namely *Crackle*, *Wheeze*, *Both (Crackle & Wheeze)*, and *Normal*. It is fact that ICBHI dataset is unbalanced, especially between *Both* and *Normal* categories. Furthermore, *Both* cycles contains both *Crackle* and *Wheeze*, posing challenges for the back-end classifiers to distinguish Normal from Crackle and Wheeze.

Given ICBHI dataset, this paper aims to classify respiratory cycles mentioned, propose two main tasks of classifying four different types of cycles (*Crackle*, *Wheeze*, *Both* and *Normal*) and two types of cycles (group of abnormal cycles with *Crackle*, *Wheeze*, *Both* and *Normal* cycle), referred to as Task 1 and Task 2, respectively. To evaluate our proposed system, we use three evaluation metrics: Sensitivity (Sen.), Specitivity (Spec.) and ICBHI scores (Average of Spec. and Sen.), which are also the metrics used in the ICBHI challenge mentioned in [8, 9, 10]. To make our work comparable, we propose two ways of splitting data. Obeying the ICBHI challenge, the first uses a ratio of 60/40 for training/test subsets (note that this splitting way prevents a subject presenting in both subsets). The second splitting randomly splits the data with a ratio of 80/20 for training/test set regardless which subject a cycle belongs to.

## The baseline system proposed

We firstly present a baseline system as shown in Figure 1, which is composed of two main steps described below.

## The front-end feature extraction

The first step, referred to as front-end feature extraction, is illustrated in the upper part of Figure 1. Respiratory cycles are firstly separated from audio recordings. We re-sample all cycles to a common sampling frequency of 16000 Hz to deal with different sample frequencies. As respiratory cycles shows different lengths, cycles are duplicated to ensure the same length of 5 seconds. Next, these respiratory cycles are transformed into spectrograms by using Gammatone filterbank with the filter number set to 64, the window size of 1024 and the hop size of 400. Finally, the Gammatone spectrograms (Gamma) are split into non-overlapped patches of 64×128, which are then fed into the back-end classifier.

## The back-end classifier

Regarding the back-end classifier illustrated in the lower part of Figure 1, we propose a CNN-DNN based archiecture configured in Table 1. The proposed CNN-DNN network is composed of two main parts: CNN and DNN. The CNN part contains four sub-blocks which perform batch normalization (BN), convolution (Conv [kernel size]), rectified linear units (ReLU), max pooling (MP [kernel size]), global max pooling (GMP), dropout (Dr (percentage drop)) as shown in the upper part of Table 1. Due to use global max pooling (GMP) across channel dimension at the final block CNN-BL04, the output of CNN part is an embedding vector with size of 512 (i.e. the size of embedding vectors equals to the number of channels set to the convolutional layer (Conv)). The DNN part is composed of two sub-blocks of DNN-BL01 and DNN-BL02. While the fully-connected layer (FC) of DNN-BL01 is followed by rectified linear units (ReLU) and dropout (Dr (percentage drop)), Sigmoid is used after the fully-connected layer of DNN-BL02. We employ focal loss function for network training as it has been proven suitable for imbalanced data [11].

## Experimental setup for the baseline

To generate spectrogram, we used Gammatone-like spectrogram toolbox from [12]. Meanwhile, the CNN-DNN network was constructed on Tensorflow framework with the focal loss $L_f$ shown below

$$L_f = -\frac{1}{N}\sum_{i=1}^{N}\alpha(\mathbf{y}_i - \hat{\mathbf{y}}_i)^{\beta}\log(\hat{\mathbf{y}}_i) + (1-\alpha)\hat{\mathbf{y}}_i^{\beta}\log(1-\hat{\mathbf{y}}_i) \quad (1)$$

where $N$ is batch size, and constant $\alpha$ and $\beta$ are set to 0.25 and 2, respectively. $\mathbf{y}_i$ and $\hat{\mathbf{y}}_i$ denote expected and predicted results, respectively. The training was carried out for 100 epochs using Adam for optimization. As the proposed CNN-DNN network works on image patches of 64×128, the result of an entire respiratory cycle is obtained by an averaging result over its patches. Let us consider $\mathbf{p}^m = (p_1^m, p_2^m, \ldots, p_C^m)$ as the predicted probabilities obtained from the $m^{th}$ out of $M$ patches and $C$ as the number of categories classified. Then, the classification probabilite of an entire recording is denoted as $\bar{\mathbf{p}} = (\bar{p}_1, \bar{p}_2, \ldots, \bar{p}_C)$ where

$$\bar{p}_c = \frac{1}{M}\sum_{m=1}^{M} p_c^m \quad \text{for} \quad 1 \leq c \leq C. \quad (2)$$

The predicted label $\hat{y}$ is then determined as

$$\hat{y} = \underset{c\in\{1,2,\ldots,C\}}{\arg\max}\ \bar{p}_c. \quad (3)$$

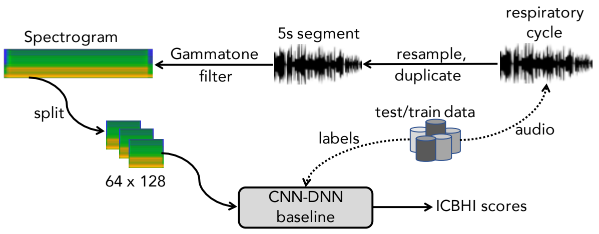

**Figure 1:** The architecture of the baseline system.

**Table 1:** The baseline network architecture proposed

| Blocks | Layers | Output |
|---|---|---|
| | Input layer (patch of 64×128×1) | |
| CNN-BL01 | BN - Conv [3×3] - ReLU - BN - MP [2×2] - Dr (10%) | 32×64×64 |
| CNN-BL02 | BN - Conv [3×3] - ReLU - BN - MP [2×2] - Dr (15%) | 16×32×128 |
| CNN-BL03 | BN - Conv [3×3] - ReLU - BN - MP [2×2] - Dr (20%) | 8×16×256 |
| CNN-BL04 | BN - Conv [3×3] - ReLU - BN - GMP - Dr (25%) | 512 |
| | Input layer (512-dimensional embeddings) | |
| DNN-BL01 | FC - Relu - Dr (30%) | 1024 |
| DNN-BL02 | FC - Sigmoid | 4 |

## Triplet-DNN network proposed for improving back-end classification

The proposed system improves upon the CNN-DNN baseline system. While the font-end feature extraction is kept unchanged, we seek to improve the back-end classifier. Specifically, we propose triplet-DNN network to replace the CNN-DNN architecture. As shown in Figure 2, the proposed triplet-DNN network follows a two-stage traning process. In the first stage shown in the left part of Figure 2, we employ three parallel CNN-DNN network architectures and make use of a combination of focal loss and triplet loss, referred to as the first-stage triplet network. Regarding the CNN-DNN architecture in the first-stage triplet network, we reused CNN-DNN baseline. By using three parallel CNN-DNN architectures, we obatain three probability outputs, which are considered as anchor, positive, and negative for triplet loss. While each probability output of individual CNN-DNN network is handled by a focal loss, triplet loss optimizes three probability outputs as below,

$$L_t = max(d(\mathbf{p}_a, \mathbf{p}_p) - d(\mathbf{p}_a, \mathbf{p}_n) + margin, 0) \quad (4)$$

where $d$ denotes Euclidean distance. $\mathbf{p}_a$, $\mathbf{p}_p$, $\mathbf{p}_n$ denote the probability outputs of the anchor, positive, and negative inputs respectively, the *margin* is set to 0.3 in this work. The final loss function *Loss* is a combination of the triplet loss function as Eq. (4) and three focal losses

$$Loss = \gamma(L_{f_a} + L_{f_p} + L_{f_n}) + (1-\gamma)L_t \quad (5)$$

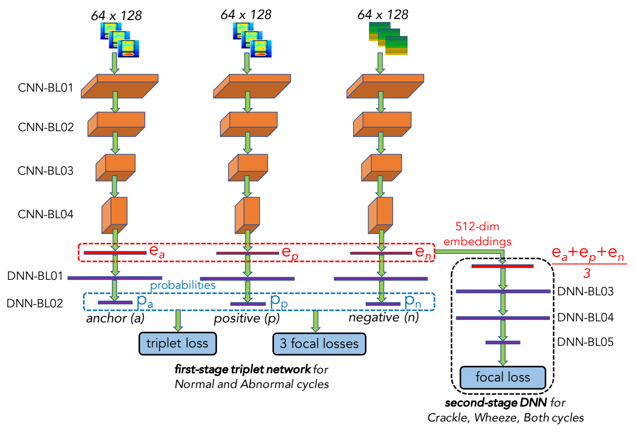

**Figure 2:** Triplet-DNN network architecture for further improving back-end classification

**Table 2:** The second-stage DNN network architecture

| Blocks | Layers | Output |
|---|---|---|
| | Input layer (512-dimensional embeddings) | |
| DNN-BL03 | FC - ReLU - Dr (30%) | 128 |
| DNN-BL04 | FC - ReLU - Dr (30%) | 128 |
| DNN-BL05 | FC - Softmax | 3 |

where $L_{f_a}, L_{f_p}$ and $L_{f_n}$ are focal loss functions used for anchor, positive and negative probability output, respectively. $\gamma$ is the hyper-parameter that trade-offs the focal losses and the triplet loss.

As there are three probability outputs from three CNN-DNN network streams, we fuse their probability outputs as

$$\mathbf{p}_{ave} = \frac{\mathbf{p}_a + \mathbf{p}_p + \mathbf{p}_n}{3} \qquad (6)$$

where $\mathbf{p}_{ave} = (\bar{p}_1, \bar{p}_2, \ldots, \bar{p}_C)$ is the average probability output. Finally, predicted result over an entire cycles is obtained by applying Eq. (2) and (3). Since the data is considerably imbalanced between *Normal* and *Both*, and a *Both* cycles contains both *Crackle* and *Wheeze*, only three categories of *Crackle, Wheeze* and *Normal* are used for training the first-stage triplet network. For instance, when positive and anchor input are *Crackle* or *Wheeze*, the negative input is always set to *Normal*, and vice versa.

To further encourage the first-stage triplet network to enlarge Fisher's criterion (i.e. the ratio of the between- class variance to the within-class variance in the feature space) between *Normal* and others, we apply mixup data augmentation [13] over input patches. By using mixup augmentation, when a *Crackle* and *Wheeze* cycle are mixed with a certain ratio, new image patches of *Both* cycles are created. Therefore, although *Both* cycles are not used for training the first-stage triplet network, when they are fed into the network for inference, classification results could be *Crackle* or *Wheeze*. To detect *Both* cycles, the embedding features of 512-dimension vectors, are extracted from output of CNN part of CNN-DNN network streams as shown in right part of Figure 2. As using three CNN-DNN architectures, we obtain anchor, positive and negative embeddings denoted as $\mathbf{e}_a$, $\mathbf{e}_p$, and $\mathbf{e}_n$, respectively. We then compute the average of three embeddings as

$$\bar{\mathbf{e}} = \frac{\mathbf{e}_a + \mathbf{e}_p + \mathbf{e}_n}{3}. \qquad (7)$$

**Table 3:** Variants of CNN architecture evaluated

| Blocks | Layers | Output |
|---|---|---|
| **Vgg-8** | Input layer (patch of 64×128×1) | |
| CNN-BL01 | BN - Conv [3×3] - ReLU - BN - MP [2×2] - Dr (10%) | 32×64×64 |
| CNN-BL02 | BN - Conv [3×3] - ReLU - BN - MP [2×2] - Dr (15%) | 16×32×128 |
| CNN-BL03 | BN - 2×(Conv [3×3] - ReLU) - BN - MP [2×2] - Dr (20%) | 8×16×256 |
| CNN-BL04 | BN - 2×(Conv [3×3] - ReLU) - BN - GMP - Dr (25%) | 512 |
| **Vgg-10** | Input layer (patch of 64×128×1) | |
| CNN-BL01 | BN - 2×(Conv [3×3] - ReLU) - BN - MP [2×2] - Dr (10%) | 32×64×64 |
| CNN-BL02 | BN - 2×(Conv [3×3] - ReLU) - BN - MP [2×2] - Dr (15%) | 16×32×128 |
| CNN-BL03 | BN - 2×(Conv [3×3] - ReLU) - BN - MP [2×2] - Dr (20%) | 8×16×256 |
| CNN-BL04 | BN - 2×(Conv [3×3] - ReLU) - BN - GMP - Dr (25%) | 512 |
| **Vgg-12** | Input layer (patch of 64×128×1) | |
| CNN-BL01 | BN - 2×(Conv [3×3] - ReLU) - BN - MP [2×2] - Dr (10%) | 32×64×64 |
| CNN-BL02 | BN - 2×(Conv [3×3] - ReLU) - BN - MP [2×2] - Dr (15%) | 16×32×128 |
| CNN-BL03 | BN - 3×(Conv [3×3] - ReLU) - BN - MP [2×2] - Dr (20%) | 8×16×256 |
| CNN-BL04 | BN - 3×(Conv [3×3] - ReLU) - BN - GMP - Dr (25%) | 512 |

The average embedding feature $\bar{\mathbf{e}}$ is then fed into a DNN based network for classifying *Crackle, Wheeze* and *Both*, referred to as the second-stage DNN. The second-stage DNN is configured by DNN-BL03, DNN-BL04, and DNN-BL05 as shown in Table 2. Notably, if an image patch is classified as Crackle or Wheeze by the first-stage triplet network, it will be further presented to the second-stage DNN network to classify to Crackle, Wheeze, or Both.

As the CNN part in an CNN-DNN architecture is considered as a high-level feature extractor which condenses an image patch input into a discriminative embedding, we evaluate three different CNN variants to study their effects on the performance. These variants are constructed by replicating the two layers Conv-ReLU in the CNN sub-blocks (i.e., CNN-BL01/02/03/04) while keeping the DNN fixed, resulting in CNN-DNN architectures similar to VGG-8, VGG-10, and VGG-12 as shown in the upper, middle and lower part of Table 3, respectively. The triplet-DNN networks which use VGG-8, VGG-10 and VGG-12 architectures are called as triplet(VGG-8)-DNN, triplet(VGG-10)-DNN and triplet(VGG-12)-DNN, respectively. Regarding experimental settings, both the first-stage triplet and second-stage DNN networks are constructed with Tensorflow framework. Other experimental settings are similar to those in CNN-DNN baseline system.

## Experimental results and Discussion

We firstly compare the CNN-DNN baseline and the triplet-DNN network with different $\gamma$ used in Eq. (5). According to the results shown in Table 4, the proposed triplet-DNN networks outperform the CNN-DNN baseline with the ICBHI challenge's data split regardless the value of $\gamma$, achieving the best ICBHI scores of 0.50 and 0.57 at $\lambda = 0.8$ for Task 1 and Task 2, respectively. However, the triplet-DNN is ineffective with random splitting method. The gap of scores between two splitting methods also reveals that the performance highly depends on the subject.

Fixing the optimal $\gamma = 0.8$, we evaluated the effect of using different CNN variants. The obtained results in Table 5 show that the triplet(VGG-10)-DNN network with random split achieves the best scores, gaining an ICBHI score of 0.03 and 0.06 over the CNN-DNN baseline on Task 1 and Task 2, respectively. With the ICBHI chal-

**Table 4:** Performance comparison between the CNN-DNN baseline and the proposed Triplet-DNN

| **Task & Systems** | **Spec. (ICBHI/random)** | **Sen. (ICBHI/random)** | **ICBHI score (ICBHI/random)** |
|---|---|---|---|
| Task 1, CNN-DNN baseline | 0.63/0.85 | 0.33/**0.61** | 0.48/**0.73** |
| Task 1, triplet-DNN ($\gamma = 0.9$) | 0.69/0.90 | 0.30/0.48 | 0.49/0.69 |
| Task 1, triplet-DNN ($\gamma = 0.8$) | **0.70**/0.86 | 0.30/0.55 | **0.50**/0.71 |
| Task 1, triplet-DNN ($\gamma = 0.7$) | 0.67/0.90 | 0.31/0.49 | 0.49/0.70 |
| Task 1, triplet-DNN ($\gamma = 0.6$) | 0.67/**0.91** | 0.31/0.49 | 0.49/0.70 |
| Task 1, triplet-DNN ($\gamma = 0.5$) | 0.65/0.90 | **0.32**/0.50 | 0.49/0.70 |
| Task 1, triplet-DNN ($\gamma = 0.4$) | 0.67/0.90 | 0.28/0.50 | 0.48/0.70 |
| Task 1, triplet-DNN ($\gamma = 0.3$) | 0.69/0.90 | 0.26/0.48 | 0.48/0.69 |
| Task 1, triplet-DNN ($\gamma = 0.2$) | **0.70**/0.89 | 0.27/0.50 | 0.49/0.69 |
| Task 1, triplet-DNN ($\gamma = 0.1$) | 0.68/0.90 | 0.29/0.46 | 0.49/0.68 |
| Task 2, CNN-DNN baseline | 0.63/0.85 | 0.47/**0.71** | 0.55/**0.78** |
| Task 2, triplet-DNN ($\gamma = 0.9$) | 0.69/0.90 | 0.45/0.63 | **0.57**/0.77 |
| Task 2, triplet-DNN ($\gamma = 0.8$) | **0.70**/0.86 | 0.44/0.70 | **0.57**/**0.78** |
| Task 2, triplet-DNN ($\gamma = 0.7$) | 0.63/0.90 | 0.45/0.64 | 0.56/0.77 |
| Task 2, triplet-DNN ($\gamma = 0.6$) | 0.67/**0.91** | 0.46/0.62 | **0.57**/0.77 |
| Task 2, triplet-DNN ($\gamma = 0.5$) | 0.65/0.90 | **0.48**/0.64 | **0.57**/0.77 |
| Task 2, triplet-DNN ($\gamma = 0.4$) | 0.67/0.90 | 0.44/0.63 | 0.55/0.76 |
| Task 2, triplet-DNN ($\gamma = 0.3$) | 0.69/0.90 | 0.40/0.62 | 0.55/0.76 |
| Task 2, triplet-DNN ($\gamma = 0.2$) | **0.70**/0.89 | 0.40/0.63 | 0.56/0.76 |
| Task 2, triplet-DNN ($\gamma = 0.1$) | 0.68/0.90 | 0.43/0.60 | 0.56/0.75 |

**Table 5:** Performance with variants of CNN ($\gamma = 0.8$)

| **Task & Systems** ($\lambda = 0.8$) | **Spec. (ICBHI/random)** | **Sen. (ICBHI/random)** | **ICBHI score (ICBHI/random)** |
|---|---|---|---|
| Task 1, CNN-DNN baseline | 0.63/0.85 | **0.33**/0.61 | 0.48/0.73 |
| Task 1, triplet-DNN | **0.70**/0.86 | 0.30/0.55 | **0.50**/0.71 |
| Task 1, triplet(vgg8)-DNN | 0.67/**0.91** | 0.21/0.51 | 0.44/0.71 |
| Task 1, triplet(vgg10)-DNN | 0.64/0.88 | 0.27/**0.63** | 0.45/**0.76** |
| Task 1, triplet(vgg12)-DNN | 0.67/0.85 | 0.19/0.61 | 0.43/0.73 |
| Task 2, CNN-DNN baseline | 0.63/0.85 | 0.47/0.71 | 0.55/0.78 |
| Task 2, triplet-DNN | **0.70**/0.86 | 0.44/0.70 | 0.57/0.78 |
| Task 2, triplet(vgg8)-DNN | 0.67/**0.91** | 0.46/0.72 | 0.56/0.81 |
| Task 2, triplet(vgg10)-DNN | 0.64/0.88 | **0.51**/0.80 | **0.58**/**0.84** |
| Task 2, triplet(vgg12)-DNN | 0.67/0.85 | 0.49/**0.82** | **0.58**/**0.84** |

lenge's data split, the three CNN variants result in the modest improvement in Task 2 and underperform in Task 1. Comparing to prior works, the performance obtained by our proposed system is very competitive to the current state-of-the-art as illustrated in Table 6 and 7 for the two data splitting methods.

## Conclusion

In this paper, we have explored a deep learning based method and triplet loss for detecting respiratory anomaly from lung sound recordings. Experiments over the ICBHI benchmark dataset showed that our proposed system obtained competitive results compared with the current state-of-the-art, achieving our best ICBHI scores (Task 1/Task 2) of 0.50/0.57 and 0.76/0.84 with ICBHI and random splitting, respectively.

## References

[1] World Health Organization: The global impact of respiratory diseases (second edition), 2017.

[2] Hitoshi Y, Shoichi M, Masaru Y, Katsuya Y, and Sueharu M: Classification between normal and abnormal respiratory sounds based on stochastic approach, in Proc. 20th International Congress on Acoustics, 2010.

[3] Takanori O, Naoki N, Masaru Y, and Shoichi M: Classification of healthy subjects and patients with pulmonary emphysema using continuous respiratory sounds, in Proc. EMBC, 2014, pp. 70–73.

[4] Morten G, Juan S, Einar H, Hasse M, and Lars B: Feature extraction for machine learning based crackle detection in lung sounds from a health survey, arXiv preprint arXiv:1706.00005, 2017.

[5] Diego Perna: Convolutional neural networks learning from respiratory data, in Proc. BIBM, 2018, pp. 2109-–2113.

**Table 6:** Task 1 comparison against the state of the art with the ICHBI challenge's split (our system in **bold**).

| Features | Classifiers | Spec. | Sen. | ICBHI score |
|---|---|---|---|---|
| MFCC | DT [14](ICBHI baseline) | 0.75 | 0.12 | 0.43 |
| MFCC | HMM [15] | 0.38 | 0.41 | 0.39 |
| Wavelet | SVM [16] | 0.78 | 0.20 | 0.47 |
| Wavelet | CNN-RNN [9] | 0.62 | 0.37 | 0.50 |
| STFT | CNN-RNN [9] | 0.69 | 0.36 | 0.50 |
| STFT+Wavelet | CNN-RNN [9] | 0.81 | 0.28 | 0.54 |
| Wavelet+STFT | bi-ResNet [17] | 0.69 | 0.31 | 0.50 |
| **Gamma** | **triplet-DNN** | **0.70** | **0.30** | **0.50** |

**Table 7:** Task 1 & 2 comparison against the state of the art with the random split (our system in **bold**).

| Features | Classifiers | train/test | Spec. | Sen. | ICBHI score |
|---|---|---|---|---|---|
| **Task 1** | | | | | |
| MFCC | CNN [5] | 80/20 | 0.77 | 0.45 | 0.61 |
| Wavelet+STFT | bi-ResNet [17] | 10 folds | 0.80 | 0.58 | 0.69 |
| MFCC | LSTM [6] | 80/20 | 0.85 | 0.62 | 0.74 |
| Gamma | CNN-MoE [10] | 5 folds | 0.90 | 0.68 | 0.79 |
| **Gamma** | **triplet(VGG-10)-DNN** | 80/20 | **0.88** | **0.63** | **0.76** |
| **Task 2** | | | | | |
| MFCC | LSTM [6] | 80/20 | - | - | 0.81 |
| MFCC | CNN [5] | 75/25 | - | - | 0.82 |
| Gamma | CNN-MoE [10] | 5 folds | 0.90 | 0.78 | 0.84 |
| **Gamma** | **triplet(VGG-10)-DNN** | 80/20 | **0.88** | **0.80** | **0.84** |

[6] Diego P and Andrea T: Deep auscultation: Predicting respiratory anomalies and diseases via recurrent neural networks, in Proc. CBMS, 2019, pp. 50–55.

[7] Rocha BM, Filos D, Mendes L, et al: A respiratory sound database for the development of automated classification, Precision Medicine Powered by pHealth and Connected Health, 2018, pp. 33–37.

[8] Lam P, Ian M, Huy P, Minh T, Truc N, and Ramaswamy P: Robust deep learning framework for predicting respiratory anomalies and diseases, arXiv preprint arXiv:2002.03894, 2020.

[9] Minami K at al: Automatic classification of large-scale respiratory sound dataset based on convolutional neural network, in Proc. ICCAS, 2019, pp. 804–807.

[10] Lam P, Huy P, Ramaswamy P, Alfred M, and Ian M: Cnn-moe based framework for classification of respiratory anomalies and lung disease detection, arXiv preprint arXiv:2004.04072, 2020.

[11] Tsung L et al: Focal loss for dense object detection, in Proc. IEEE international conference on computer vision, 2017, pp. 2980-–2988.

[12] Ellis D Gammatone-like spectrograms. web resource: http://www. ee. columbia. edu/dpwe/resources/matlab/gammatonegram.

[13] Xu K et al: Mixup-based acoustic scene classification using multi-channel convolutional neural network, in Pacific Rim conference on multimedia, 2018, pp. 14-23.

[14] Rocha B et al: An open access database for the evaluation of respiratory sound classification algorithms, Physiological measurement, 2019; 40(3):035001.

[15] Jakovljevic N and Loncar-Turukalo T: Hidden markov model based respiratory sound classification, in Proc. BHI, 2017, pp. 39-43.

[16] Serbes G, Ulukaya S, Kahya Y: An automated lung sound preprocessing and classification system based onspectral analysis methods, in Proc. BHI, 2017, pp. 45-49.

[17] Ma Y et al: A smart digital stethoscope for detecting respiratory disease using bi-resnet deep learning algorithm, in Proc. BioCAS, 2019, pp. 1-4.