Intelligent Medicine 3 (2023) 200–212

Contents lists available at ScienceDirect

Intelligent Medicine

journal homepage: www.elsevier.com/locate/imed

Research Article

An evolutionary ensemble learning for diagnosing COVID-19 via cough signals

Mohammad Hassan Tayarani Najaran*

University of Hertfordshire School of Physics Engineering and Computer Science, Hatfield, United Kindom

ARTICLE INFO

Keywords:
Optimization
Evolutionary algorithms
COVID-19

ABSTRACT

Objective The spread of the COVID-19 disease has caused great concern around the world and detecting the positive cases is crucial in curbing the pandemic. One of the symptoms of the disease is the dry cough it causes. It has previously been shown that cough signals can be used to identify a variety of diseases including tuberculosis, asthma, etc. In this paper, we proposed an algorithm to diagnose the COVID-19 disease via cough signals.

Methods The proposed algorithm was an ensemble scheme that consists of a number of base learners, where each base learner used a different feature extractor method, including statistical approaches and convolutional neural networks (CNNs) for automatic feature extraction. Features were extracted from the raw signal and some transforms performed it, including Fourier, wavelet, Hilbert-Huang, and short-term Fourier transforms. The outputs of these base-learners were aggregated via a weighted voting scheme, with the weights optimised via an evolutionary paradigm. This paper also proposed a memetic algorithm for training the CNNs in the base-learners, which combined the speed of gradient descent (GD) algorithms and global search space coverage of the evolutionary algorithms.

Results Experiments were performed on the proposed algorithm and different rival algorithms which included a number of CNN architectures in the literature and generic machine learning algorithms. The results suggested that the proposed algorithm achieves better performance compared to the existing algorithms in diagnosing COVID-19 via cough signals.

Conclusion COVID-19 may be diagnosed via cough signals and CNNs may be employed to process these signals and it may be further improved by the optimization of CNN architecture.

1. Introduction

As the COVID-19 pandemic is ravaging across the world, identifying the COVID-19 positive cases is a matter of crucial importance in the battle against the spread of the disease. Testing capacity, in terms of human and financial resources remains a challenge in many countries. In this respect, many machine learning techniques have been developed for the early identification of the cases [1] via faster and cheaper methods.

Human audio signals contain valuable information about many aspects of a person's condition. Previous studies have processed these signals to identify people's health condition. Examples of these include processing voice signal to identify one's personality [2], children psychiatry [3], depression [4], tuberculosis detection [5], asthmatic detection [6], sleep apnea detection [7], breathing rate measurement [8], stress detection [9], child pneumonia [10], pertussis [11], among others.

Using cough signals to identify COVID-19 cases has been the focus of a number of studies. One of the best efforts has used Mel Frequency Cepstral Coefficients (MFCCs) of cough signals to train a convolutional neutral network (CNN) architecture (ResNet50) to identify positive cases [12]. Cough signals collected via phone calls can be used to identify positive cases [13]. A deep learning architecture is proposed which consists of two sub-network components [14]. The first component captures hidden features and the second one identifies deeper temporal acoustic features of cough signals. A CNN with three feature extraction and three classification layers to identify positive cases has been proposed in [15]. MFCC features are used to build a model via symbolic recurrence quantification measures to diagnose the cases via cough signals [16]. A transfer learning algorithm has been proposed to overcome the current shortage of data in this domain [17-18].

Cough and breathing signals are collected and processed to identify the positive cases [19]. The authors show that even simple learning algorithms are capable of diagnosing the cases. It is shown that speech signals can also be used for the classification problem [20]. Other studies that use cough signals for diagnosing the disease include [21-22].

* Corresponding author: Mohammad Hassan Tayarani Najaran, University of Hertfordshire School of Physics Engineering and Computer Science, Hatfield, UK (Email: tayarani@yahoo.com).

https://doi.org/10.1016/j.imed.2023.01.001
Received 22 November 2022; Received in revised form 10 January 2023; Accepted 11 January 2023

An ensemble learning algorithm is proposed [23-24] to detect the disease via cough signals. A comparison between different classification algorithms in detecting COVID-19 is performed [25]. To diagnose the disease via cough signals, an automated extraction of time-frequency features is proposed [26].

In ensemble learning, multiple learning algorithms, called *base-learners* are combined to immprove overall pattern recognition performance. The idea behind ensemble learning techniques is that weaknesses of individual base-learners are balanced by the strength of others (sometimes interpreted as the "wisdom of crowd" in the machine learning context). There are several reasons why an ensemble of learners improves the performance [27–29]). First, by having a number of learning algorithm each building their own hypothesis mitigates the overfitting problem. Second, combining the output of a number of base-learners decreases the risk of getting stuck in local optima. Third, combining different models provides an extended view that increases the chance of reaching the optimal hypothesis that may be outside the ability of the single learners. Fourth, since an increased number of features results in an exponential increase in the size of the search space (known as the curse of dimensionality [30]), splitting the feature space into smaller pieces and using an ensemble of learners can mitigate this problem, too.

To reach good performance, the base learners should be diverse. There are different ways of achieving the required diversity. *Input manipulation* is to train the base learners via different subsets of training data [31]. *Manipulation of the learning algorithm* is performed by manipulating the way the base learners traverse the hypothesis space, so each learner is led to a different convergence path [32]. *Partitioning* refers to partitioning the dataset into smaller pieces and using each partition for a base-learner [33]. *Output manipulation* refers to the methods in which some binary classifiers are combined into a single multi-class classifier [34].

After the base-learner models have been created, the output of these algorithms needs to be aggregated. These approaches can be categorized as follows.

Weighting methods combine outputs using a weighted sum. Majority voting is the simplest method [35]. In regression problems, this is performed by averaging the output over the base-learners [36]. Some examples of these approaches include the weighted averaging based on the base-learners strength [37], Bayesian combination in which the weights are determined to maximize the probabilistic likelihood of the model based on the dataset [38] and weighted averaging based on the confidence of each base-learner [39].

Meta-learning methods use the output of the base-learners as input to the meta-learner which acts as the output layer. Meta-learners are trained to model how different base-learners perform on different subspaces. Example of these approaches include stacking [40] which models the performance of each base learner on the original dataset, weighted bagging [41], which uses a kernel density estimator to assign the weights based on their closeness to the target set, and mixture of experts which is based on the idea of divide-and-conquer to divide the space into a number of experts [42].

In this study, we propose an evolutionary ensemble classification (EEC) algorithm to diagnose COVID-19 cases via cough signals. The proposed algorithm achieves diversity by using different sets of features for each of the base-learners in the ensemble. The feature extractors used in this paper include statistical approaches and CNNs for automatic feature extraction from the audio signals. As well as extracting features from raw signals, the proposed method performs some transforms on the signals to extract frequency-domain features, including Fourier, wavelet, Hilbert-Huang, and short-term Fourier transforms. Each of the base-learners performs classification based on the features adopted via one of the feature extraction methods. Each base-learner uses different types of features.

The output of some transform functions, like the wavelet transform, are 2D signals, similar to images. Since CNNs are very powerful for automatically extracting features from images, these algorithms are adopted in this paper to process the transformed signals. An evolutionary algorithm is proposed in this paper to find the optimal CNN architecture for each feature set. The outputs of these base-learners are aggregated in an aggregation layer which uses a weighted averaging scheme. The weights of the base-learners in the voting scheme are found via an evolutionary training algorithm.

To train the CNNs, a memetic algorithm is proposed in this paper which is a mixture of a Gradient Descent (GD) algorithm and an evolutionary algorithm. GD algorithms typically are quick to reach a reasonable solution, but suffer from the problem of getting stuck in local optima. In contrast, the advantage of evolutionary algorithms is their global search, at the expense of larger computational effort. By combining the two, the proposed algorithm benefits from the speed of GD algorithms and the global search of evolutionary algorithms.

The proposed algorithm is different from the existing methods for COVID-19 diagnosis via cough signals in different aspects. First, this is the first research that tries to find the optimal CNN architecture for processing cough signals. This is different from the existing methods that use generic architectures that are not necessarily suitable for this particular problem. Second, the proposed combination of evolutionary and gradient-based training of CNNs for cough signal processing has not been presented in the literature. Third, extracting features from the spectrogram of cough signals via CNNs and processing them as images is novel. Forth, extracting this large number of features and incorporating them into a group of machine learning algorithms has not been done in the literature before. Finally, the proposed ensemble learning algorithm which is a combination of different machine learning algorithms and different features has not been addressed before.

This paper is organized as follows. Section 2.1 describes the CNN algorithms and architecture optimization, Section 2.2 explores the other base-learner algorithms and their aggregation, Section 3.2 reports the experimental results and Section 4 concludes the paper.

2. Methods

2.1. Data acquisition

In this paper, the COUGHVID [43] data are used to diagnose COVID-19 via cough signals. The dataset contains over 20,000 cough recordings which include data about age, gender, location and COVID-19 diagnosis. These data were collected via a web application on a server located at the cole Polytechnique Fdrale de Lausanne (EPFL), Switzerland. The authors then developed a cough detection software [44] which determines the degree of certainty to which a recording contains a cough. The dataset contains this as a probability between zero and one. We relied on these values to select the data and omitted the data records for which the probability of containing a cough signal is below 0.5. After preprocessing the data, 8,407 of the data records were selected for the training and classification purposes. Among these, 6,466 cases were healthy, 658 cases are COVID-19 and 1,283 cases are symptomatic. Some data were removed due to containing noise, including background sound, etc. Then the silent segments of the signals were removed and only the part of the signal containing the cough were used.

2.2. Optimizing convolutional neural networks architecture

Due to their outstanding performance and flexibility, CNNs have found numerous applications in a variety of areas in the pattern recognition fields. These networks are a type of feed-forward neural network that processes input signals and automatically extracts features for pattern recognition tasks. The architecture of CNNs is presented in Figure 1. The network consists of a number of convolution and pooling layers that are applied to the input image. The features are extracted by the network via these operators to be processed by a pattern recognition algorithm at the fully connected layers at the tail of the architecture. In this example in Figure 1, there are four groups of feature maps with one fully connected layer at the tail. A convolutional operator extracts features from the previous layer, like a filter. The convolution is performed by sliding

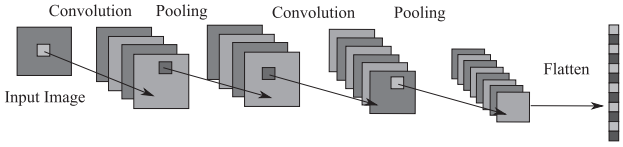

Figure 1. The architecture of CNNs, consisting of a combination of convolution and pooling operators.

Table 1 Tunable parameters of a CNN that are critical for pattern recognition performance

Convolutional operators	Pooling operators
Convolutional type	Pooling type
Filter width	Kernel width
Filter height	Kernel height
Stride width	Stride width
Stride height	Stride height
Connection weight	
Number of feature maps	

the filter, in the form of a matrix, over the images, thereby creating a feature map. The filter slides on the image with a step size called stride. At each step, the dot product of the pixels in the filters with the corresponding pixels in the image is calculated and stored in the pixels in the feature map. The convolutional operators have parameters that need to be tuned to achieve good performance; this process is called *learning*. The parameters of this operator are summarized in Table 1.

The pooling operators are similar to the convolutional operators in that they traverse over the images to process the data. The operator uses a matrix, called the kernel, to combine the data from the previous layer, by collecting the average or the maximum values within the kernel. This reduces the size of the representation, and thereby the amount of computation, the required memory, and the number of parameters. The parameters of this operator are summarized in Table 1.

The fully connected layer at the tail of the network which is connected to all the neurons in the previous layer performs the final classification.

The ordering of pooling and convolutional layers, operators and parameters are known as the CNN architecture and play a crucial role in the performance of the algorithm. There is not one CNN architecture that performs equally well for all problems. Therefore, when solving a pattern recognition problem, part of the solution will be to optimize the CNN architecture for the problem. This is usually achieved by trial and error, plus some expert knowledge of the CNN designer. In this paper, we adopt an evolutionary algorithm to find the best CNN architecture for COVID-19 diagnosis via cough signals. In the optimization process, the number and order of the pooling and convolutional operators, along with the numerical parameters of the architecture, (kernel, filter and stride size, pooling type, etc.) are optimized. Algorithm 1 is proposed in this paper to optimize the CNN architecture.

Algorithm 1 Evolutionary optimization of CNN architecture.

begin
  set the parameters m, n and l
  $\tau = 0$
1. initialize the population X^0
2. **while** not termination condition do
  begin
3. find the fitness of the individuals in X^τ
4. select the parent solutions
5. use crossover to generate offsprings and store in O^τ
6. apply mutation on O^τ with probability m
7. perform selection on $X^\tau \cup O^\tau$ and store in $X^{\tau+1}$
    $\tau = \tau + 1$
  end
8. **return** the best solution in X^τ
end

At first, in the proposed algorithm, the parameters of the algorithm, m the mutation rate, n the number of individuals in the population and l the maximum number of layers in the CNN are set. Here, l controls the complexity of the CNN in terms of the computational cost and performance. Although a larger number of layers in the CNN does not necessarily mean better performance, larger CNNs are usually capable of modelling more complex patterns. This, however, results in higher computational cost both in the training and classification phase. Therefore, a trade-off between the performance and computational cost should be performed. Based on our computational budget, we set $l = 8$ in this paper, that is the maximum number of convolutional, pooling and fully connected layers is be 8.

The population is initialized in step 1 of the algorithm, where n individuals X_i^0, $i = 1 \dots n$ are randomly generated. Each individual represents a CNN architecture. First, the number of layers in the individual is set to a random number between 2 and l. The minimum number of layers is 2 because the smallest CNN consists of a convolutional layer and a fully connected layer. By initialising individuals with different numbers of layers, the optimization process explores the space of the number of CNN layers. When the number of layers for an individual is determined, a convolutional operator is placed at the first layer and a fully connected layer is placed at the tail. For the layers between these two, either a pooling or convolutional operator is placed with a probability of 0.5.

In step 1, also the numerical parameters of the pooling and convolutional operators, presented in Table 1, are initialized at random. The evolutionary process of the proposed algorithm optimizes these values. To reduce the size of the search space, we use the same parameter sets for all convolutional layers. The connection weights are randomly initialized and then optimized within the training phase via GD. While evolutionary algorithms could also be used to optimize the weights, we do not expect them to perform well due to the large number of weights. The initial weights greatly affect the optimization process in gradient descent algorithms as this algorithm is prone to getting stuck in local optima. Therefore, the mean and standard deviation of the Gaussian distribution for random weight initialization are optimized optimize through the evolutionary process. After the population is initialized, the 'while' loop in step two of the algorithm runs the evolutionary algorithm until the termination condition is met. In this paper we set the maximum number of generations as the termination condition.

Step 3 evaluates the individuals in the population. Every time the fitness function is called, the entire dataset is partitioned randomly into train and validation data. The evaluation is performed by generating a CNN with the architecture encoded in the individual, training it (via GD) on the training data and testing on the validation data. The fitness of an architecture is defined as the average of TPR, PPV, TNR and ACC measures for the three classes of COVID-19, asymptomatic and healthy (Equations 3, 4, 5 and 6) on the validation data. These measures have been used as the fitness to make sure the optimization process considers all metrics. We also considered using the number of correctly classified cases as the fitness. However, because the data are imbalanced, such criterion tends to classify most of the cases into the healthy class. An average over these metrics helps to manage the issue. Note that here the CNN is trained and evaluated on different datasets (training and validation sets) to avoid contamination. It is important to evaluate an architecture on a different dataset than the training dataset to avoid overfitting of the architecture on the training data.

In step 4, the parent individuals are selected using tournament selection. The crossover operator is applied in step 5 to generate offspring. Figure 2 presents the way crossover is applied to the individuals. Since the individual CNNs in this representation can have different sizes, the crossover operator needs to take this into account. In the proposed method, a random crossover point in each of the individuals is chosen, and the offspring is created by inheriting one part from one parent and the other from another parent. The resulting offspring can have a size smaller or larger than its parents. This allows the evolutionary process

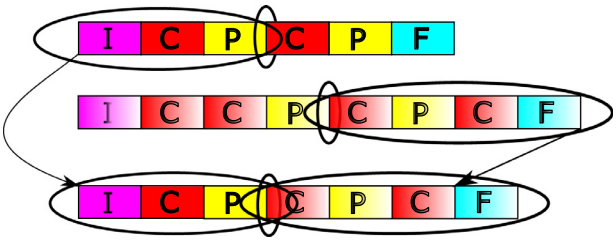

Figure 2. The crossover operator for the CNN architecture optimization.

to also optimize the number of layers. Because the maximum number of layers is l, if an offspring with the number of layers greater than this limit is generated, some of the operators are selected and removed randomly to satisfy the size limit.

In step 6 of the algorithm the mutation operator is applied. We devise three types of mutations in this paper as follows.

- An operator is chosen at random and its type is swapped (from to convolutional to pooling and vice versa).
- A pooling and a convolutional operator are chosen at random and their position is swapped.
- A layer is randomly inserted in or removed from the architecture. This mutation changes the number of layers in the architecture performing the search to find the optimal number of layers.

These mutation operators explore to find the best number of operators and their ordering within a CNN. In order to optimize the numerical parameters of the operators in Table 1, these parameters are are coded as a vector of numbers, where a single point crossover operator is used. The mutation is defined as randomly selecting a number in the vector and changing it to a random value.

2.3. Diagnosing COVID-19 via cough signals

Around 60% of COVID-19 cases show dry cough symptom [45]. Dry cough is a type of cough which is dry with no sputum or mucus [46]. In a dry cough, three phases is observed: the initial opening burst, the noisy airflow and the glottal closure [47]. The first phase is characterized by more energy at high frequencies, while in phase 2 less energy is observed at higher frequencies [48]. Wet cough on the other hand, is caused by inflammation and secretion in the lower airways, which sometimes can be triggered by upper respiratory inflammation. For wet coughs, more energy is presented in phase two at higher frequencies [48]. Differences between coughs of patients with COVID-19 infection and those of other patients can be used to diagnose the disease via cough signals.

2.3.1. Feature extraction techniques

In the literature, a variety of features have been used to analyse the signals and to identify COVID-19 cases. In some works, the statistical properties of the cough signals are used as features. These include Log energy, Zero crossing rate, Variance, Skewness, Kurtosis [49], Entropy, Formant frequencies, and Fundamental frequency [21]. Some work use frequency information of the signals to extract features including Mel spectrogram [17], short-term magnitude spectrograms [13], STFT, MFB, MFCC [50], and non-negative matrix factorisation (NMF) [51].

While using statistical features of the signals in the time-domain prove useful for many applications, there are important information in the frequency domain that can be exploited. In this respect, many mathematical transforms are used to extract frequency features from signals. The Fourier transform, which is widely used in many applications, identifies the frequency components of a signal. The Fourier transform tells what frequency components exist in a signal, but it does not provide time-frequency information. Short-term fourier transform (STFT) is designed to provide time-frequency information. The method works by segmenting the input signal into shorter pieces of shorter length and performing Fourier transform on each piece. This transform uses a fixed-size time window, therefore the time-frequency resolution is not always adequate, especially when the signal contains short bursts. The wavelet transform has been proposed to mitigate this by extracting the time-frequency information via variable-sized window. This property is similar to the STFT, with the advantage that it provides a more accurate representation for nonstationary signals with discontinuities, like cough signals. Selecting the optimal mother wavelet is critical for this method [52]. The Hilbert-Huang Transform (HHT) has been proposed for this purpose since it is suitable for the analysis of non-linear and non-stationary signals [53]. The HHT consists of two steps, the empirical mode decomposition (EMD) and Hilbert transform. HHT does not suffer from the uncertainty principle and it can provide high time and frequency resolutions at the same time.

The Mel Frequency Cepstral Coefficients (MFCC) are spectral properties of a signal that are extracted via a frequency transform of the log spectrum. MFCCs are designed in a way to have similar frequency resolution to that of human ear [54]; therefore, they are capable of representing the non-linear response of human auditory system to sound signals. This makes these coefficients successful in the recognition of audio signals.

While these mathematical transforms are useful tools to extract features from the signals, automatic feature extraction methods can be employed. Among the most successful automatic feature extraction methods are the CNNs. The output of many mathematical transforms like wavelet or HHT on one dimensional signals (like sound signals) are two dimensional signals, similar to images (and often visualized as such). These two dimensional signals contain valuable information about the properties of the signals. In this sense, CNNs can be used to automatically extract features from these images.

All these feature extraction methods are useful in finding good features for the classification of cough signals. Here, we propose taking the advantage of all these tools in a single model to improve performance in the classification task.

2.3.2. The proposed ensemble learning algorithm

We propose in this paper an algorithm which is an ensemble of a number of base learner algorithms. Each base learner is trained based on a distinct set of features, extracted via different algorithms. Therefore, using different sets of features for each base learner provides the required diversity among the learning algorithms. Figure 3 shows the structure of the proposed ensemble learning algorithm. In the following we describe the 9 base learners used in this paper, with their corresponding feature extraction methods.

The first base learner uses OpenSMILE (Open source Speech and Music Interpretation by Large-space Extraction) [55] to extract features from the raw cough signal. The software extracts 384 features including signal energy, Bark-spectra, Octave-spectra, PLP-CC, Pitch, Formants, LPC, Line Spectral Pairs, Spectral Shape description, etc. The features extracted by this software have been successfully used to analyse human voice signals [2–4]. To classify the cough signals via the OpenSMILLE features we used the matlab Support Vector Machine (SVM) implementation.

The second base learner in the ensemble performs Fourier transform on the cough signal and extracts the statistical features of the resulting signal. The statistical features used in this paper include the minimum, maximum, mean, median and the standard deviation of the first 20 Fourier paramters. These features have been widely used in the literature [56-57].

The third base learner uses the wavelet transform on each of the cough signals and extracts wavelet features, f_c which are calculated from the feature matrix F_M. These features have shown to provide promising results for cough signal classification [58].

The fourth base learner performs HH transforms on the signal and extracts statistical features. The statistical features include mean, entropy

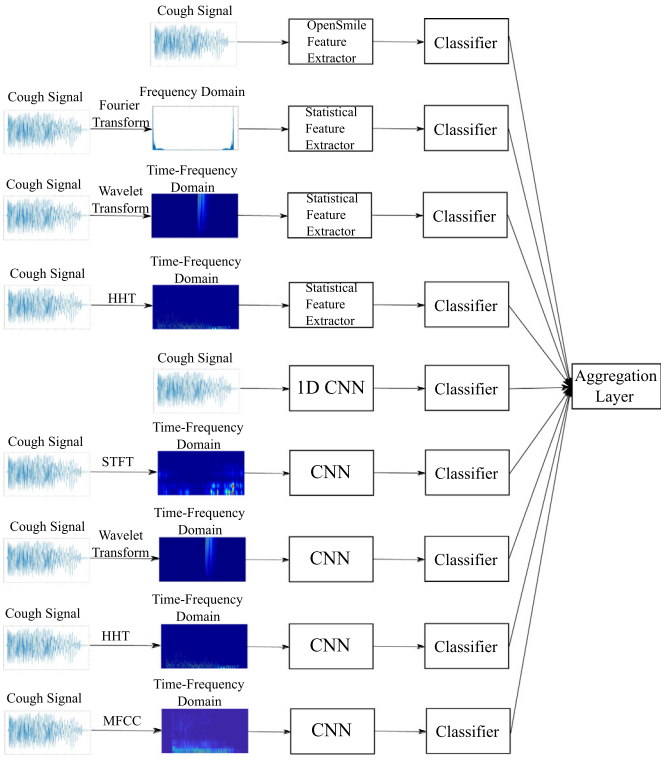

Figure 3. The structure of the proposed ensemble learning algorithm.

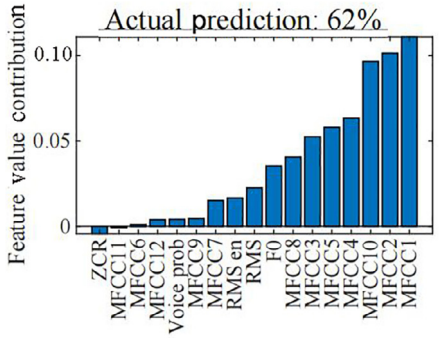

Figure 4. The Shapely values for a particular data record with a prediction of 0.62. The horizontal axis is the OpenSmile features and the vertical axis is the feature value contribution.

error, entropy estimation, histogram lower, histogram upper, RMS, kurtosis and skewness.

The fifth base learner is a 1-dimensional CNN which processes the cough signal and automatically extracts features. Usually, in order to use CNNs for 1D signals, the signal is converted from 1D to 2D and then the resulting signal is processed [59-60]. However, it has been shown that employing 1D CNNs can provide better outcome for 1D signals [61]. A detailed description of 1D CNNs and their applications can be found [62].

The sixth base learner in the ensemble uses STFT to convert the cough signal into frequency domain. The output of STFT is a 2D signal that contains time-frequency information about the signal. In this paper we treat the 2D signal as an image and use CNNs to automatically extract features and perform classification. Similarly, the 7th, 8th and 9th base learners use wavelet, HHT and MFCC transforms to produce the 2D images. These images are then processed via CNNs to automatically extract features for identification of coughs of COVID-19 patients.

As presented in Figure 3, the proposed ensemble learning algorithm uses different sets of feature extraction and feeds them to different classifier algorithms. For the CNN algorithms, are used to perform the classification at the tail of the CNNs. The connection weights of these fully connected layers are optimized via the gradient-based learning algorithm which is described later in the paper. For the statistical features, SVM is used to classify the data.

In the proposed ensemble algorithm, a variety of features are collected via different feature extraction methods. One way of using all these features is to concatenate all these feature vectors into a single vector and feed them into a single learning algorithm. Here, we propose an approach which consists of a number of base learners where each learner is trained based on one feature extraction method. There are several advantages in using such an approach. (1) Using a large number of features for a single learning algorithm usually results in suboptimal performance as the problem becomes too complex for the learning algorithm to manage (the curse of dimensionality). In this condition, a feature selection mechanism should be adopted to reduce the feature space. In the proposed method, each base learner is trained to classify the cases based on a smaller number of features. (2) Ensemble learning algorithms require diversity to reach improved performance [63]. Using different sets of features for each base learner provides the diversity which allows us to use the ensemble approach. (3) Because each of the learning algorithms can be trained and tested independently and in parallel, the proposed paradigm can be implemented in a distributed scheme. (4) In the proposed approach, the base learners are architecturally designed and trained via distinct set of features. This way, each base learner is tuned and adjusted to classify cases based on a particular feature set which results in having more specialized base learners. For instance, one learning algorithm is trained to discover the patterns in the wavelet coefficient domain. This base learner, is thus not required to recognize the patterns of other transformation signals like MFCC. This specifically is an advantage when CNNs are used to extract features. For example, features extracted from the wavelet transform are characteristically different from the features extracted from MFCC, but CNNs depend on convolution operators being uniformly applied over the input feature space. Using the same CNN to extract features from all the feature sets would therefore not be expected to yield good, as the characteristics of the features are distinct for each transform function and it is unlikely that a convolutional operator can be found that is optimal for all of them. Therefore, having different CNNs for different transform functions offers improved performance. (5) Because in the proposed method the feature space is segmented into smaller pieces, each base learner needs to identify the patterns in a smaller feature space. Therefore, the training process in each of the base learners is performed in a smaller search space with smaller number of local optima, resulting in a better chance of the algorithm in finding superior local optima.

2.3.3. *Memetic training of CNNs*

We propose a memetic algorithm to train the CNNs for diagnosing COVID-19 via cough signals. Memetic algorithms are a group of algorithms that use local search algorithms in evolutionary algorithms to improve the performance by managing premature convergence [64]. Gradient Descent (GD) algorithms are usually used for training CNNs due to their comparably fast convergence on this type of problem. But since GD algorithms perform local search, they suffer from the problem of getting stuck in local optima. Training CNNs involves the optimization of a considerably large number of learning parameters which include the connection weights, filter values, etc. For such a problem with the huge fitness landscape, there is a high probability that such local optima prevent GD algorithms from finding the global optimum. This paper proposes a memetic algorithm to train CNNs which is presented in Algorithm 2. Memetic algorithms are a mixture of local search and evolutionary algorithms to combine the advantage of both paradigms. The proposed memetic algorithm in this paper is a mixture of GD and an evolutionary algorithm. The GD algorithm performs the search to find a local optimum and the evolutionary process performs global search by performing operations on the local optima found by the GD algorithm.

Algorithm 2 The proposed memetic training algorithm.

```
begin
    set the parameters m, α, N and n
    τ = 0
1.  initialize N sub-populations Y_j^0, j = 1 . . . N
2.  while not termination condition do
    begin
3.    for j = 1 . . . N
      begin
4.      for all individuals in Y_j^τ
        begin
5.        partition the training data T into T_t and T_v
6.        train the individual with T_t via GD
7.        evaluates the individual with T_v
        end
8.      select the parent solutions from Y_j^τ
9.      generate offffsprings via crossover and store in O_j^τ
10.     perform mutation on O_j^τ with probability of m
11.     perform selection on Y_j^τ ∪ O_j^τ and store in Y_j^{τ+1}
      end
12.   perform migration with probability of α
      τ = τ + 1
    end
13. return the best solution in Y_τ
end
```

Each individual in the evolutionary optimization process is a set of the learning parameters of the CNNs which include the connection weights and filters. We devise a multi-population scheme in the proposed algorithm to increase diversity in the population and improve the performance of the algorithm. The population is divided into N sub-populations of equal size. This scheme allows each sub-population to search through the fitness landscape independently which helps the algorithm to explore wider area in the search space. This results in diversity among the individuals helping the algorithm to find better local optima.

In the proposed memetic algorithm, GD is used as the local search algorithm to train individual CNNs. The next step of the evolutionary process requires to assess the fitness of the individuals. The final training error of the CNN could be used as the fitness; however, this would likely result in overfitting to the training data. To avoid this, we partition the training data into T_t (train dataset) and T_v (validation dataset) in step 5 of the algorithm. CNNs are first trained using T_t, then T_v is used to evaluate the fitness of the individuals, that is, the learning parameters of the trained CNN, which are the connection weights and filters. The partitioning is performed by randomly inserting 3/4 of the training data into T_t and 1/4 of them into T_v.

The individuals in the population are trained in step 6. In order to train an individual (the learning parameters of a CNN), the GD algorithm starts from the learning parameters of the individual and performs descent until it reaches a local optimum. This improves the fitness of the individual via the GD algorithm. Note that in the first iteration of the evolutionary algorithm ($\tau = 0$), the individuals are initialized at random, so the GD algorithm starts from a random point in the fitness landscape. However, as the evolutionary algorithm progresses, the individuals that are the result of the crossover and mutation operators are further optimized via the GD algorithm. In other words, the individuals generated via crossover and mutation are used as the starting point for the GD optimization process.

The individuals are evaluated in step 7 of the algorithm. As mentioned before, the partition T_t is used to train the individuals and T_v is used to evaluate them. Since the data partitions T_t and T_v are generated by random splitting for each individual, each CNN is trained and evaluated based on different sets of data, leading to improved exploration of the search space and avoiding overfitting.

The parents are selected in step 8 of the algorithm. In step 9 of the algorithm the crossover operator is applied to the selected individuals to generate the new set of offsprings. Here, the crossover is only applied to the individuals that are in the same sub-population (this is performed via the *for* loop in step 3). The individuals are the set of the learning parameters of the CNNs. The learning parameters are the connection weights which are a number of matrices. The crossover in this paper is defined as a single point crossover on the matrices. In CNNs, the filters are also matrices and a similar paradigm is used for the crossover. The mutation operator is applied in step 10 of the algorithm which is defined as changing randomly 5% of the values in the matrices.

When the new population of individuals is generated via the crossover and mutation operators, the GD algorithm is applied to the newly generated individuals. By changing the individuals, the crossover and mutation operators move the individuals away from the local optima they have reached via the GD algorithm in previous steps. This helps the GD algorithm to escape from the local optima and perform again a search process to search other regions in the search space. The mutation operator increases the diversity in the population by randomly changing the individuals. The crossover operator performs a global search in the landscape by combining two individuals that have already reached the local optima. Performing the crossover between two local optima generates a solution that inherits properties from both local optima. When GD is performed on this new individual, a new local optimum is reached that, in expectation, is better than the previous local optima. A more in-depth analysis of this process is available elsewhere [65].

We have devised a migration operator in step 12 of the algorithm which randomly selects two individuals from two sub-populations and swaps them.

2.3.4. Aggregation in the ensemble scheme

The last layer in the algorithm is the ensemble layer, in which a weighted voting paradigm is performed among the base learners. Part of the learning process is therefore to find optimal weights for each the base learners. In the proposed algorithm, first the base learners are trained via the training data. Then, the output of the ensemble scheme is calculated as,

$$C = \frac{\sum_{i=1}^{k} w_i c_i}{\sum_{i=1}^{k} w_i}, \tag{1}$$

where C is the aggregated output of the base learners, c_i is the predicted class of the i-th base learner, and k is the number of base learners. In this paper, the weight values of the base learners, w_i are optimized via a simple Genetic Algorithm (GA). In the initialization step of the evolutionary process, the individuals are generated. Each individual is a vector of real values and is initialized by setting to random numbers between 0 and 1, $w_i = R(0, 1)$, where $i = 1, \ldots, c$ and $R(.,.)$ is a uniform random number generator. The fitness of the individuals is measured based on the accuracy of the prediction of the ensemble learning algorithm. Algorithm 3 presents the way the fitness of the individuals is measured.

Algorithm 3 Weight optimization fitness function.

```
input the individual x
begin
1.  partition the training data T into 4 equal
      partitions T_j, j = 1 . . . 4
2.  S = 0
3.  for j = 1 → 4 do
    begin
4.    train the base learners with T − T_j
5.    test the base learners with T_j
6.    use the weights in x to aggregate the base learners
      and store the number of correctly classifified cases in s
7.    S = S + s
    end
8.  return S
end
```

To measure the fitness of the individuals, the voting weights of the base learners suggested by the individual x is used to classify the data and the accuracy of the classification process is returned as the fitness. To measure the accuracy of the classification, the training data is partitioned into found equal size partitions T_j, and in a four fold scheme, three quarters of the training data are used to train the base learners and one quarter is used as validation dataset to measure the accuracy of the classification. Note that the training and evaluation in the steps 4 and 5 of the algorithms do not need to be performed every time the fitness function is called. The base learners are trained and tested in the four-fold scheme. If the partitioning is performed the same way every time the fitness function is measured, the output of the base learners for each of the data records in the validation data set can be stored in a look-up table. Then, to measure the fitness of the individual x, the output of the base-learners is found via the table.

In the ensemble scheme, the weights of the voting system should be positive numbers. Thus, in the optimization process it is made sure that the weights do not become negative numbers.

The proposed algorithm improves the performance in two aspects. First, it takes advantage of a wide range of features from different transformation functions. Each of these feature extraction methods, like wavelet or MFCC, have their own advantage that can benefit the classification process. Second, because the output of the base learners is found via a voting scheme (the weights of which are optimized via the evolutionary paradigm), if any set of features, (for example OpenSMILE features) do not offer good performance and do not contribute in the classification process, the optimization process of the voting weights automatically reduces the weight of the base learner. This performs as an automatic feature set selection scheme giving higher weight to the more discriminating set of features.

3. Results

3.1. Interpretability

While there has been progress in designing accurate decision making algorithms, there has emerged the idea among the research community that the prediction accuracy is not the only objective and the intelligent algorithms should be designed in a way that it can be explained why and how a particular decision has been made. This is particularly relevant in medical applications, where a decision is made, based on a feature vector, about the health condition of a patient. If the learning algorithm reaches the decision that a particular case is healthy, it is very beneficial to verify the output and check if it is associated with the clinical features that are relevant to the disease, rather than some accidental correlation between irrelevant features and the target [66]. Conversely, if there is a case that has been identified as unhealthy by the machine learning, an explanation of why such decision has been made can be of help as this provides insights on what features have caused the disease and what therapeutic actions can be taken to reduce symptoms or to cure the condition.

In designing interpretable machine learning algorithms, it is assumed that the input features are interpretable by the receiver [67]. Despite all these, it is sometimes difficult to create interpretable machine learning algorithms, when the features are not clearly understandable. For example, some features like “blood pressure” and “blood sugar level” are easy to understand by expert. However, in many applications, there may be some features that are extracted from some transformations on some signals which are not trivial for the experts to interpret. Many of the features used in this paper belong to this group. For example, some features extracted from the wavelet transform of a signal are not much interpretable by a doctor of medicine.

Deep neural network algorithms can be presented as a number of functions as

$$f(x) = f_L \circ \ldots \circ f_1(x) \tag{2}$$

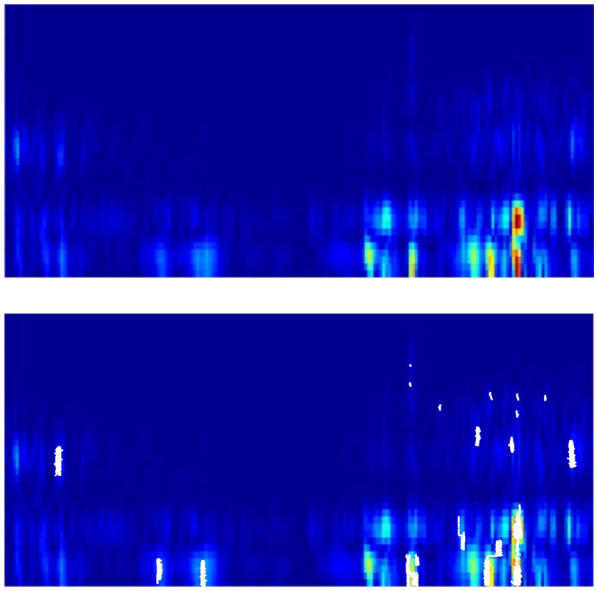

Figure 5. The spectrogram representation of the cough signal after the short term Fourier transform is applied.

where each layer applies a function on the output of the previous layer. Although having a large number of layers gives a high prediction power to the algorithms, the complexity of these systems adds to the complications of making the system interpretable. First, in these systems, some neurons are only activated for some data records, while some other neurons are activated for others. Such behavior causes an amalgamation of local and global effects which in turn makes finding the root point that expands to the prediction a hard task [67]. Second, the depth of these networks causes a phenomenon known as “shattered gradient”, where the gradient resembles white noise [68]. It has been shown that in worst case, the discontinuity of the gradient grows exponentially with the number of layers [69]. Third, the challenge arises when trying to find a root point for explanation, which is close to the data but not adversarial examples [70].

Although these challenges make the task of designing an interpretable CNN hard, we try our best in this paper to add interpretability to the proposed algorithm.

The proposed algorithm is an ensemble learning algorithm so each of the base learners should be interpreted individually. Then, after the individual ones are interpreted, the aggregation step in the ensemble is interpreted.

We start by discussing the interpretability of the first base learner which uses the features extracted via OpenSmile and the SVM algorithm to classify the data. It has been shown in the literature that SVM can be interpreted [71]. While the methods proposed specifically for SVMs can be used to interpret the algorithms, in this paper we use Shapley method [72] to interpret the algorithm. Figure 5 shows the Shapely values for a particular data record. As the data in this figure suggest, the feature MFCC1 contributes around 0.11 to the prediction for this data record compared to the average prediction for the whole dataset.

In [67], a method is proposed to make CNNs interpretable. In this method, the regions in the figure that contain the features responsible for the decision are detected and presented. Figure 5 shows the regions in the spectogram representation of the short Fourier transform of the cough signal that contain these features for a particular data record.

The experiments so far present how different base-learners in the ensemble can be interpreted. The aggregation step in the ensemble, as presented in Algorithm 1, generates a linear combination of the output of the base-learners. Such learning algorithms are easily intperpretable. More details of how these can be interpreted can be found in [72].

Table 2 The best architecture for each of the feature extraction CNNs

Transform	Architecture					
	Layer 1	Layer 2	Layer 3	Layer 4	Layer 5	Layer 6
STFT	Conv.	Pool.	Conv.	Conv.	Pool.	
	FS=5	KS=3	FS=5	FS=5	KS=6	
	NF=3	PT=M	NF=3	NF=4	PT=M	
	SS=3	SS=3	SS=6	SS=3	SS=6	
Wavelet	Conv.	Conv.	Pool.	Conv.	Pool.	Pool.
	FS=3	FS=5	KS=5	FS=4	KS=6	KS=6
	NF=3	NF=2	PT=M	NF=4	PT=M	PT=M
	SS=3	SS=3	SS=5	SS=3	SS=5	SS=5
HHT	Conv.	Conv.	Pool.	Pool.		
	FS=5	FS=3	KS=6	KS=5		
	NF=4	NF=3	PT=M	PT=A		
	SS=4	SS=3	SS=3	SS=6		
MFCC	Conv.	Pool.	Conv.	Pool.	Pool.	Conv.
	FS=6	KS=3	FS=5	KS=5	KS=3	FS=5
	NF=2	PT=A	NF=5	PT=A	PT=A	NF=5
	SS=6	SS=6	SS=6	SS=3	SS=3	SS=4

Conv: convolutional; Pool: pooling; FS: filter size; NF: number of filters; SS: stride size; KS: kernel size; PT: pooling type.

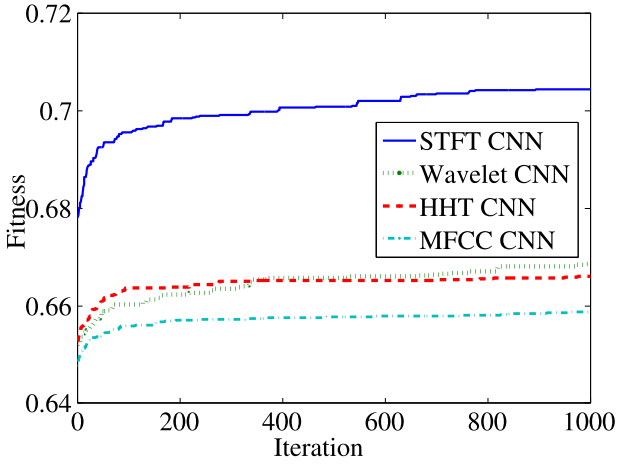

Figure 6. The optimization progress of the evolutionary CNN architecture design.

3.2. Experimental results

The experimental studies in this paper start with the optimization of CNN architecture for each feature set via Algorithm 1. In these experiments, the population size is set to $n = 20$, the maximum number of layers is set to $l = 9$, the mutation rate is set to 0.1 and a maximum number of 1000 generations is set as the termination condition. The CNNs have a fixed number of one input layer and two fully connected classification layers. The number and ordering of the convolutional and pooling operators as well as the parameters filter size, number of filters, stride size, kernel size and pooling type are optimized via the evolutionary paradigm. In all experiments, the signals are converted into 256×256 images. For each set of features, the CNN architecture is optimized independently.

To evaluate the performance of the learning algorithms three metrics are used in this paper which include the true positive rate (TPR, also known as sensitivity), the positive predictive rate (PPR, also known as precision), true negative rate (TNR, also known as specifity) and accuracy (ACC). These measures are defined as follows.

$$\mathrm{TPR} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} \tag{3}$$

$$\mathrm{PPV} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}} \tag{4}$$

$$\mathrm{TNR} = \frac{\mathrm{TN}}{\mathrm{TN} + \mathrm{FP}} \tag{5}$$

$$\mathrm{ACC} = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{TN} + \mathrm{FP} + \mathrm{FN}} \tag{6}$$

Table 2 reports the optimized CNN parameters for each 2D feature set. For example, the best CNN architecture for extracting features for STFT consists of 5 layers of convolutional and pooling operators with 2 fully connected layers at the tail. The first layer is a convolutional operator with the parameters filter size equal to 5, number of filters equal to 3 and stride size equal to 3. All experiments in this paper use the architectures reported in Table 2.

Figure 6 shows the evolutionary progress of the evolutionary optimization of the CNN architecture. The graph shows the fitness of the best solution at each iteration out of a population of 20. The fitness is defined as percentage of the data records in T_{test} that are correctly classified. CNNs are trained with GD. The optimization process starts from a random CNN architecture and performs the evolutionary process. The performance of a random architecture is shown at iteration zero, which e.g. in the case of STFT yield 66.49% correct classification. As the architecture is optimized, the performance improves to 70.67%. Therefore, optimizing the architecture of the CNN resulted in a performance gain of roughly 4% in this case.

In this paper, we compare the performance of the proposed algorithm with five established algorithms: SVM, ANN, SAE, DBN and BLS. The features used for these algorithms were the OpenSMILE, Fourier, wavelet and HHT statistical features. The libSVM [73] algorithm was used for SVM and the BLS [74] was used for the BLS algorithms. The DeepLearn toolbox [75] was used to implement ANN, SAE and DBN. In order to implement the CNN algorithms, the respective MATLAB toolbox was employed. We used a polynomial kernel for SVM; for ANN 4 layers with 20 neurons at each layer were used. The number of feature nodes and the number of enhancement nodes in BLS are set to 100 and 2000 respectively. All other parameters in the algorithms used the default parameters of the employed software.

As explained in Section 2.1, the architecture of CNNs is optimized in this paper via an evolutionary process based on the entire dateset. To evaluate the algorithms, a four-fold scheme is used to avoid contamination of train with the test data. Table 3 presents the performance of the CNN algorithms in terms of the metrics when a random architecture is used, versus the optimized architecture via the proposed method. The data in these tables are averaged over 30 independent runs. As the data suggest, the proposed architecture optimization scheme improves the performance of the CNNs on all the experiments and metrics. A *t*-test analysis is also presented in this paper which suggests improvement with significance.

The idea in optimizing the architecture of CNNs in this paper is that there is no architecture that suits all the existing problems and for each problem there is a particular architecture that works better. In order to show how specifically design an architecture for a problem can improve the performance, Table 4 shows the performance of a random architecture (random here means an architecture made via step 1 of the Algorithm 1), the optimized architecture and some of the well-known CNNs in the literature. The performance is defined as the percentage of the cases that are correctly classified by the algorithm. The results are for four transform functions, STFT, Wavelet, HHT and MFCC.

In order to compare the optimized architecture with the state-of-the-art architecture, we use CAE-2 [76], TIRBM [77], PGBM + DN-1 [77], ScatNet-2 [78], RandNet-2 [79], LDANet-2 [79], SVM + RBF [80], SVM + Poly [80], NNet [80], SAAA-3 [80], SqweezNet (SQNet) [81], MobileNetV2 [82] and DBN-3 [80]. Clearly, all these architectures easily outperform a random architecture. However, the proposed optimization technique builds an architecture that performs better than the existing algorithms. This is because all these architectures are some generic architectures that are not particularly designed for this problem. For example, because of their specific properties, when processing wavelet

Table 3 Different algorithms in terms of performance, accuracy, true positive rate, positive predictive rate and true negative rate (%)

		Healthy				COVID-19				Symptomatic			
Algorithms	Performance	ACC	TPR	PPV	TNR	ACC	TPR	PPV	TNR	ACC	TPR	PPV	TNR
STFT Rnd	63.31	67.56	63.36	91.96	81.55	80.19	62.86	22.54	81.66	78.86	63.27	38.34	81.67
STFT Opt.	66.85	70.67	66.85	93.07	83.42	82.13	67.24	25.59	83.40	80.89	66.65	42.04	83.45
P-value	2.35e-3	2.8e-4	8.1e-3	2.8e-4	4.0e-3	7.8e-4	4.9e-4	1.5e-5	8.1e-3	1.5e-4	6.1e-3	8.2e-5	1.3e-4
Wavelet Rnd	59.41	64.06	59.28	90.80	79.99	77.99	59.55	19.83	79.55	76.75	59.93	34.80	79.78
Wavelet Opt.	61.89	66.24	61.86	91.50	80.85	79.50	61.63	21.61	81.01	78.04	62.18	36.95	80.90
P-value	3.42e-2	8.3e-3	1.4e-4	7.4e-5	2.2e-3	9.1e-4	3.6e-5	1.2e-3	6.0e-3	5.0e-5	6.7e-4	1.1e-5	3.7e-4
HHT Rnd	59.57	64.21	59.52	90.77	79.84	78.19	60.22	20.13	79.72	76.71	59.42	34.66	79.83
HHT Opt.	61.54	65.94	61.53	91.37	80.64	79.28	61.72	21.41	80.77	77.85	61.46	36.56	80.80
P-value	2.21e-3	2.5e-4	1.3e-3	3.4e-5	2.2e-3	1.2e-4	2.8e-4	1.1e-4	3.9e-5	1.3e-5	2.0e-3	4.0e-4	4.3e-5
MFCC Rnd	58.83	63.55	58.71	90.58	79.66	77.80	59.14	19.59	79.38	76.30	59.21	34.08	79.37
MFCC Opt.	60.53	65.12	60.53	91.14	80.39	78.63	60.07	20.48	80.21	77.29	60.70	35.67	80.28
P-value	4.21e-2	7.6e-4	2.8e-4	4.2e-2	2.2e-2	1.1e-5	1.3e-1	1.0e-4	4.0e-4	3.6e-4	1.0e-2	1.2e-4	8.0e-5

Table 4 The performance of different architectures (%)

Algorithm	STFT	Wavelet	HHT	MFCC
Random	63.31	59.41	59.57	58.83
Optimized	66.85	61.89	61.54	60.53
NNet	66.63	61.03	61.50	59.94
DBN-3	65.92	61.60	60.86	59.93
CAE-2	66.05	61.78	61.47	59.47
TIRBM	65.50	60.93	61.47	59.46
PGBM	64.65	60.91	60.36	59.48
ScatNet-2	65.45	60.13	61.33	59.93
RandNet-2	65.51	60.51	60.69	59.27
LDANet-2	65.56	61.24	61.52	59.52
SQNet	66.54	61.79	60.34	60.05
MBNet-2	64.98	61.68	61.18	59.89
GPNet	64.24	60.91	61.04	60.23

transform images, it is better to design a CNN architecture that is specifically suitable for this type of images. Table II in supplementary materials presents more details experimental analysis on the data in terms of different metrics.

Table 5 summarizes the experiments on different algorithms. For the CNN algorithms, the best architectures presented in Table 2 are used. The data are averaged over 30 independent runs. The first five algorithms are the established algorithms that we use here as a benchmark to compare our results against. Algorithms 6 to 15 correspond to the base learners. The last algorithm, EEC is the proposed Evolutionary Ensemble Classification algorithm. The first column represents the performance of each algorithm which is defined as the percentage of the cases that are correctly classified. The data for SFTF CNN, Wavelet CNN, HHT CNN AND MFCC CNN are for the experiments in which the proposed memetic learning scheme is used to train the CNNs. A comparison between the data in Tables 3 and 5 shows that the memetic learning improve the performance of these algorithms. EEC consistently achieves the best performance among all algorithms. It outperforms the base-learners because the ensemble scheme optimizes the weights that combine the output of the base-learners to achieve better performance than the base-learners individually. Among the established algorithms, the best performance is achieved by SVM, followed by ANN algorithm. Among the CNN algorithms it is STFT which achieves the best performance.

In this table, the PPV value for COVID-19 cases is smaller than other metrics. This is because a relatively smaller proportion (around 8%) of data belong to the COVID-19 cases. As per Equation 4, when the number of cases for a particular class is smaller than the other classes, its TP will be small, while its FP can be relatively large as there are more cases of the other classes to be misclassified to belong to this class. This is why PPV is relatively small for the COVID-19 and symptomatic cases, and larger for healthy cases.

Table 6 presents an analysis of the results in Table 5 using the Kruskal-Wallis [83] test. In this table, 'SS' is the sum of squares of each source, 'df' is the degree of freedom associated with each source, 'MS' is the mean squares (the ration SS/df) and 'Chi-square' is the ratio of mean squares. The results in Table 6 confirm that the null hypothesis that the samples are taken from the same mean can be rejected with a high probability significance level.

More detailed experiments are performed and the results are presented in Table 7.

4. Discussion

An ensemble learning algorithm was proposed in this paper to diagnose COVID-19 via cough signals. In this algorithm, a number of base learners are trained and aggregated in a weighted voting scheme. Each base learner is trained based on a particular set of features. The weights in the voting scheme are optimized via an evolutionary algorithm. The proposed paradigm has the advantage of effectively adjusting the weights in a way to select the feature sets which are more capable of correctly classifying the data. If any of the feature selection techniques do not provide adequate performance, it is automatically given a small weight thus it plays a less significant role.

Because there are many mathematical transforms that have been designed to find frequency information of signals, when it comes to signal processing, there usually exist a large number of feature extraction approaches. Historically, for a particular application, researchers normally use a specific feature extraction method to build a learning algorithm. Each of these feature extraction methods has their own characteristics.

The proposed algorithm combines the advantages of different feature extraction methods. It extracts features from the time-domain signals and uses a variety of transform functions including wavelet, Fourier, Hilbert-Huang, Walsh, and short-term Fourier. Two approaches for feature extraction are employed in this paper, first are the statistical features from the signals, and the second is using CNNs to extract features automatically. Because the output of many transform functions is a 2D signal, similar to an image, extracting the properties of the signals from these images can be successfully performed via CNN algorithms.

There are many generic CNN architectures in the literature that have been used for a variety of applications. Although these generic CNNs have proved successful, there is not a particular architecture that suits all the existing problems and the optimal decision is to finely tune the CNN architecture for the problem. This paper proposes an evolutionary paradigm to optimize the CNN architecture for each of the feature sets.

Traditionally, GD algorithms are used to train CNN algorithms. While due to their speed GD algorithms are very successful, these algorithms are prone to get stuck in local optima. To manage this, we propose a memetic algorithm that benefits from the speed of GD and the global search of evolutionary algorithms. This paradigm improves the chance of the learning algorithm to reach better local optima.

There are still some open questions that have not been targeted in this paper and can be studied in future work. To manage the local

Table 5 Different algorithms in terms of performance, accuracy, true positive rate, positive predictive rate and true negative rate (%)

		Healthy				COVID-19				Symptomatic			
Algorithm	Performance	ACC	TPR	PPV	TNR	ACC	TPR	PPV	TNR	ACC	TPR	PPV	TNR
SAE	61.33	65.81	61.41	91.29	80.47	79.14	61.47	21.23	80.64	77.68	60.77	36.22	80.73
ANN	68.22	71.88	68.25	93.42	83.99	82.83	67.91	26.61	84.10	81.72	68.23	43.68	84.15
SVM	69.45	72.96	69.45	93.78	84.65	83.54	69.08	27.80	84.77	82.38	69.58	45.01	84.69
DBN	66.06	70.05	66.16	92.85	83.03	81.67	66.15	24.82	82.98	80.41	65.55	41.10	83.08
BLS	67.63	71.34	67.53	93.38	84.04	82.59	67.99	26.31	83.83	81.31	67.89	42.90	83.72
OpenSMILE	68.20	71.85	68.15	93.49	84.19	82.92	68.36	26.81	84.15	81.61	68.32	43.49	84.01
Fourier Stat.	68.91	72.48	68.86	93.68	84.51	83.24	69.45	27.45	84.41	82.10	68.85	44.42	84.48
Wavelet Stat.	65.25	69.27	65.31	92.54	82.46	81.27	65.05	24.14	82.64	79.95	64.99	40.28	82.64
HHT Stat.	67.60	71.34	67.62	93.27	83.75	82.50	67.77	26.15	83.75	81.35	67.41	42.93	83.86
1D CNN	62.55	66.90	62.66	91.67	81.02	79.83	61.87	21.98	81.36	78.36	62.30	37.44	81.25
STFT CNN	70.98	74.34	71.03	94.18	85.38	84.41	70.76	29.40	85.57	83.20	70.81	46.67	85.43
Wavelet CNN	66.19	70.11	66.19	92.91	83.17	81.70	65.93	24.81	83.04	80.56	66.29	41.44	83.13
HHT CNN	64.63	68.73	64.70	92.37	82.18	80.81	64.42	23.51	82.20	79.71	64.37	39.80	82.47
MFCC CNN	63.68	67.85	63.68	92.06	81.71	80.40	63.55	22.90	81.83	79.10	63.68	38.76	81.88
EEC	73.55	76.54	73.47	94.88	86.80	85.71	73.74	32.06	86.73	84.84	73.85	50.23	86.82

Table 6 The ANOVA and Kruskal-Wallis test on the data for different classes and measures

Groups	ANOVA						Kruskal-Wallis test					
	Source	SS	df	MS	F	Prob>F	Source	SS	df	MS	Chi-sq	P>Chi-sq
ACC	Columns	3.37e-01	14	2.41e-02	5.64e+03	0	Columns	7.50e+06	14	5.36e+05	444	7.63e-86
Healthy	Error	1.86e-03	435	4.27e-06			Error	8.96e+04 7.59e+06	435	2.06e+02		
	Total	3.39e-01	449				Total		449			
ACC	Columns	1.28e-01	14	9.17e-03	9.39e+02	1.21e-314	Columns	7.29e+06	14	5.21e+05	431	3.50e-83
COVID-19	Error	4.25e-03	435	9.77e-06			Error	3.03e+05	435	6.96e+02		
	Total	1 33e-01	449				Total	7 59e+06	449			
ACC	Columns	1.46e-01	14	1.04e-02	1.15e+03	0	Columns	7.32e+06	14	5.23e+05	433	1.48e-83
Sympt.	Error	3.94e-03	435	9.06e-06			Error	2.73e+05 7 59e+06	435	6.27e+02		
	Total	1 50e-01	449				Total		449			
TPR	Columns	4.25e-01	14	3.04e-02	2.51e+03	0	Columns	7.45e+06	14	5.32e+05	441	3.55e-85
Healthy	Error	5.25e-03	435	1.21e-05			Error	1.43e+05 7.59e+06	435	3.29e+02		
	Total	4.30e-01	449				Total		449			
TPR	Columns	4.66e-01	14	3.33e-02	7.10e+01	8.46e-103	Columns	5.17e+06	14	3.69e+05	306	8.36e-57
COVID-19	Error	2.04e-01	435	4.69e-04			Error	2.43e+06	435	5.58e+03		
	Total	6.70e-01	449				Total	7.59e+06	449			
TPR	Columns	4.83e-01	14	3.45e-02	1.61e+02	7.67e-162	Columns	6.29e+06	14	4.49e+05	372	1.15e-70
Sympt.	Error	9.34e-02	435	2.15e-04			Error	1.31e+06 7.59e+06	435	3.01e+03		
	Total	5.76e-01	449				Total		449			
PPV	Columns	3.85e-02	14	2.75e-03	2.57e+02	3.51e-200	Columns	6.68e+06	14	4.77e+05	395	1.57e-75
Healthy	Error	4.65e-03	435	1.07e-05			Error	9.16e+05 7.59e+06	435	2.11e+03		
	Total	4.31e-02	449				Total		449			
PPV	Columns	3 44e-01	14	2 46e-02	6 18e+02	9 09e-277	Columns	7 13e+06	14	5 09e+05	422	3 68e-81
COVID-19	Error	1.73e-02	435	3.97e-05			Error	4.65e+05 7.59e+06	435	1.07e+03		
	Total	3.61e-01	449				Total		449			
PPV	Columns	5.51e-01	14	3.93e-02	1.10e+03	0	Columns	7.31e+06	14	5.22e+05	432	1.86e-83
Sympt.	Error	1.55e-02	435	3.56e-05			Error	2.81e+05 7.59e+06	435	6.46e+02		
	Total	5.66e-01	449				Total		449			
TNR	Columns	1.18e-01	14	8.46e-03	1.07e+02	6.78e-131	Columns	5.72e+06	14	4.09e+05	338	1.11e-63
Healthy	Error	3.45e-02	435	7.92e-05			Error	1.87e+06 7.59e+06	435	4.30e+03		
	Total	1.53e-01	449				Total		449			
TNR	Columns	1.10e-01	14	7.83e-03	6.66e+02	1.95e-283	Columns	7.21e+06	14	5.15e+05	426	3.79e-82
COVID-19	Error	5.12e-03	435	1.18e-05			Error	3.86e+05 7.59e+06	435	8.87e+02		
	Total	1.15e-01	449				Total		449			
TNR	Columns	1.06e-01	14	7.59e-03	5.91e+02	1.20e-272	Columns	7.12e+06	14	5.08e+05	421	5.25e-81
Sympt.	Error	5.59e-03	435	1.29e-05			Error	4.77e+05 7.59e+06	435	1.10e+03		
	Total	1.12e-01	449				Total		449			

SS is the sum of squares, and df is the degrees of freedom. MS is the mean squared error. F is the ratio of the mean squared errors, Prob > F is the P-value that represents the significance of the differences between column means. P > chi-sq indicates the P-value that all the data samples come from the same distribution. It is called chi-sq because in Kruskal Wallis test, the p-value measures the significance of the chi-square statistic.

optima problem in the training process of CNNs we proposed the memetic scheme. However, it is a well-known concept that the design of optimization algorithms can be improved when gaining a better understanding of the fitness landscape of the optimization problem [84–86]. Therefore, we believe that studying the fitness landscape of the training phase of CNNs presents a promising future line of research. Such understanding can help to better understand the structure of the fitness landscape, help in providing insight into the ways in which the local optima problem can be managed, and increase the chance of finding the global optimum.

This paper employed a wide range of features, including statistical and CNN-extracted features from different transforms on the signal. Further improvement of the classification performance could be achieved through a smart feature selection scheme that narrows down the large set of features. This is specifically true for the features extracted via CNNs, as they are found automatically and do not undergo a selection process. Future research could conduct a feature analysis to find an improved set of features among the wide range of potential features that can be extracted using the rich toolbox provided by signal processing techniques, e.g. by using filter and wrapper techniques. Identifying

Table 7 Different algorithms in terms of performance, accuracy, true positive rate, positive predictive rate and true negative rate (%)

		Healthy				COVID-19				Symptomatic			
Algorithms	Performance	ACC	TPR	PPV	TNR	ACC	TPR	PPV	TNR	ACC	TPR	PPV	TNR
STFT, Random	63.31	67.56	63.36	91.96	81.55	80.19	62.86	22.54	81.66	78.86	63.27	38.34	81.67
STFT, Optimized	66.85	70.67	66.85	93.07	83.42	82.13	67.24	25.59	83.40	80.89	66.65	42.04	83.45
Wavelet, Random	59.41	64.06	59.28	90.80	79.99	77.99	59.55	19.83	79.55	76.75	59.93	34.80	79.78
Wavelet, Optimized	61.89	66.24	61.86	91.50	80.85	79.50	61.63	21.61	81.01	78.04	62.18	36.95	80.90
HHT, Random	59.57	64.21	59.52	90.77	79.84	78.19	60.22	20.13	79.72	76.71	59.42	34.66	79.83
HHT, Optimized	61.54	65.94	61.53	91.37	80.64	79.28	61.72	21.41	80.77	77.85	61.46	36.56	80.80
MFCC, Random	58.83	63.55	58.71	90.58	79.66	77.80	59.14	19.59	79.38	76.30	59.21	34.08	79.37
MFCC, Optimized	60.53	65.12	60.53	91.14	80.39	78.63	60.07	20.48	80.21	77.29	60.70	35.67	80.28
STFT, NNet	66.63	70.45	66.56	93.04	83.40	81.97	67.04	25.35	83.24	80.83	66.74	41.95	83.36
Wavelet, NNet	61.03	65.51	60.99	91.26	80.54	79.07	61.48	21.17	80.56	77.49	61.00	35.99	80.46
HHT, NNet	61.50	65.94	61.42	91.50	80.99	79.20	61.29	21.26	80.72	77.85	61.98	36.66	80.71
MFCC, NNet	59.94	64.56	59.91	90.91	80.04	78.32	60.30	20.26	79.85	77.00	59.90	35.13	80.08
STFT, DBN-3	65.92	69.84	65.88	92.83	83.04	81.58	65.93	24.68	82.91	80.42	66.11	41.18	82.99
Wavelet, DBN-3	61.60	66.04	61.68	91.36	80.56	79.29	61.35	21.35	80.82	77.85	61.26	36.54	80.84
HHT, DBN-3	60.86	65.40	60.89	91.21	80.44	78.85	60.92	20.85	80.37	77.46	60.66	35.88	80.48
MFCC, DBN-3	59.93	64.46	59.78	90.90	80.06	78.49	60.32	20.41	80.03	76.90	60.46	35.09	79.86
STFT, CAE-2	66.05	70.00	66.07	92.87	83.09	81.69	66.12	24.84	83.01	80.39	65.85	41.10	83.01
Wavelet, CAE-2	61.78	66.27	61.87	91.53	80.93	79.25	60.51	21.14	80.84	78.03	61.93	36.90	80.93
HHT, CAE-2	61.47	65.88	61.43	91.40	80.72	79.20	61.69	21.33	80.69	77.84	61.51	36.56	80.78
MFCC, CAE-2	59.47	64.13	59.41	90.75	79.83	78.11	60.06	20.03	79.64	76.69	59.41	34.63	79.80
STFT, TIRBM	65.50	69.43	65.40	92.71	82.86	81.43	65.65	24.45	82.77	80.12	65.88	40.67	82.69
Wavelet, TIRBM	60.93	65.45	60.99	91.17	80.30	78.94	60.86	20.93	80.48	77.46	60.62	35.88	80.49
HHT, TIRBM	61.47	65.93	61.42	91.50	80.98	79.17	61.94	21.36	80.63	77.82	61.44	36.52	80.77
MFCC, TIRBM	59.46	64.19	59.65	90.57	79.31	78.13	58.53	19.74	79.79	76.61	58.98	34.44	79.78
STFT, PGBM	64.65	68.76	64.71	92.39	82.25	80.88	64.71	23.64	82.25	79.66	64.29	39.71	82.43
Wavelet, PGBM	60.91	65.43	60.92	91.21	80.43	78.95	61.21	21.01	80.46	77.43	60.64	35.84	80.45
HHT, PGBM	60.36	64.87	60.35	90.93	79.94	78.70	61.27	20.78	80.18	77.14	59.91	35.32	80.24
MFCC, PGBM	59.48	64.17	59.60	90.59	79.38	78.22	59.27	19.97	79.83	76.55	58.93	34.36	79.73
STFT, ScatNet-2	65.45	69.45	65.52	92.60	82.56	81.43	64.97	24.31	82.83	80.01	65.34	40.41	82.65
Wavelet, ScatNet-2	60.13	64.70	60.11	90.92	80.00	78.46	60.29	20.38	80.00	77.09	60.12	35.29	80.15
HHT, ScatNet-2	61.33	65.76	61.23	91.41	80.82	79.23	61.68	21.36	80.72	77.68	61.63	36.35	80.56
MFCC, ScatNet-2	59.93	64.50	59.89	90.85	79.89	78.44	59.46	20.20	80.05	76.91	60.37	35.09	79.88
STFT, RandNet-2	65.51	69.49	65.54	92.63	82.62	81.44	65.97	24.51	82.75	80.09	65.07	40.51	82.79
Wavelet, RandNet-2	60.51	65.05	60.50	91.06	80.21	78.60	60.56	20.56	80.13	77.37	60.53	35.74	80.40
HHT, RandNet-2	60.69	65.19	60.64	91.15	80.37	78.90	60.54	20.83	80.46	77.28	61.02	35.70	80.21
MFCC, RandNet-2	59.27	64.01	59.28	90.71	79.76	78.05	58.87	19.73	79.68	76.46	59.36	34.32	79.54
STFT, LDANet-2	65.56	69.57	65.59	92.72	82.83	81.48	65.63	24.50	82.83	80.05	65.32	40.49	82.71
Wavelet, LDANet-2	61.24	65.68	61.24	91.27	80.48	79.15	61.44	21.24	80.65	77.64	61.09	36.20	80.62
HHT, LDANet-2	61.52	65.97	61.44	91.52	81.04	79.30	61.97	21.48	80.77	77.78	61.67	36.50	80.68
MFCC, LDANet-2	59.52	64.16	59.50	90.71	79.68	78.17	59.38	19.94	79.76	76.69	59.61	34.67	79.77
STFT, SQNet	66.54	70.39	66.55	92.95	83.19	81.92	66.74	25.24	83.21	80.74	66.30	41.75	83.34
Wavelet, SQNet	61.79	66.22	61.84	91.48	80.82	79.43	61.54	21.52	80.95	77.91	61.59	36.67	80.85
HHT, SQNet	60.34	64.97	60.30	91.17	80.54	78.56	60.17	20.44	80.12	77.13	60.58	35.43	80.12
MFCC, SQNet	60.05	64.62	59.99	90.93	80.06	78.55	60.85	20.57	80.05	76.92	59.91	35.02	79.98
STFT, MBNet-2	64.98	69.02	65.07	92.41	82.19	81.09	64.43	23.82	82.50	79.85	64.82	40.10	82.56
Wavelet, MBNet-2	61.68	66.09	61.72	91.40	80.64	79.41	61.32	21.46	80.94	77.85	61.62	36.59	80.77
HHT, MBNet-2	61.18	65.61	61.10	91.31	80.63	79.13	61.04	21.14	80.66	77.61	61.62	36.26	80.50
MFCC, MBNet-2	59.89	64.52	59.89	90.87	79.95	78.35	60.11	20.25	79.89	76.92	59.80	35.00	80.00
STFT, GPNet	64.24	68.39	64.17	92.42	82.46	80.65	64.30	23.31	82.04	79.45	64.61	39.43	82.12
Wavelet, GPNet	60.91	65.51	61.02	91.23	80.46	78.83	60.72	20.80	80.37	77.47	60.41	35.86	80.54
HHT, GPNet	61.04	65.51	61.01	91.24	80.49	78.98	61.17	21.02	80.49	77.58	61.09	36.13	80.55
MFCC, GPNet	60.23	64.79	60.22	90.94	80.00	78.65	60.54	20.60	80.19	77.01	60.08	35.18	80.06

COVID-19 via cough signals is in its infancy, so there is still ample work to be done to find what features are more discriminating for this task.

We employed evolutionary algorithms to optimize the architecture of CNNs and in the training phase of these algorithms. The fitness in the optimization process suffers from uncertainty (known as approximation uncertainty [87]). For example, for the optimization of the CNN architecture, in order to calculate the fitness, the CNN with the architecture is trained and tested on the data and the accuracy is measured as the fitness. Because the training and testing process is not deterministic, the measured fitness suffers from uncertainty. This uncertainty affects the optimization process and should be managed.

Conflicts of interest statement

The authors declare that there are no conflicts of interest.

Funding

This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

Author contributions

Mohammad Hassan Tayarani Najaran designed the study, did the literature search and drafted the manuscript.

Supplementary materials

Supplementary material associated with this article can be found, in the online version, at doi:10.1016/j.imed.2023.01.001.

References

[1] Tayarani NMH. Applications of artificial intelligence in battling against covid-19: a literature review. Chaos Soliton Fractal 2021;142:110338. doi:10.1016/j.chaos.2020.110338.

[2] Tayarani M, Esposito A, Vinciarelli A. What an “ehm” leaks about you: mapping fillers into personality traits with quantum evolutionary feature selection algorithms. IEEE Trans Affect Comput 2019:1. doi:10.1109/TAFFC.2019.2930695.

[3] Roffo G, Vo DB, Tayarani M, et al. Automating the administration and analysis of psychiatric tests: the case of attachment in school age children. In: Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems. New York, NY, USA: Association for Computing Machinery; 2019.

[4] Scibelli F, Roffo G, Tayarani M, et al. Depression speaks: automatic discrimination between depressed and non-depressed speakers based on nonverbal speech features. In: 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP); 2018. doi:10.1109/ICASSP.2018.8461858.

[5] Miranda IDS, Diacon AH, Niesler TR. A comparative study of features for acoustic cough detection using deep architectures. In: 2019 41st Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC); 2019.

[6] Yadav S, Keerthana M, Gope D, et al. Analysis of acoustic features for speech sound based classification of asthmatic and healthy subjects. Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). 2020.

[7] Simply RM, Dafna E, Zigel Y. Obstructive sleep apnea (osa) classification using analysis of breathing sounds during speech. In: 2018 26th European Signal Processing Conference (EUSIPCO); 2018.

[8] Mohamed I, Routray A. Automatic measurement of speech breathing rate. 27th European Signal Processing Conference (EUSIPCO). 2019.

[9] Partila P, Tovarek J, Rozhon J, et al. Human stress detection from the speech in danger situation Mobile Multimedia/Image Processing, Security, and Applications 2019. International Society for Optics and Photonics; 2019.

[10] Abeyratne UR, Swarnkar V, Setyati A, et al. Cough sound analysis can rapidly diagnose childhood pneumonia. Ann Biomed Eng 2013;41(11):2448–62. doi:10.1007/s10439-013-0836-0.

[11] Pramono RX, Imtiaz SA, Rodriguez-Villegas E. A cough-based algorithm for automatic diagnosis of pertussis. PLoS One 2016;11(9):e0162128. doi:10.1371/journal.pone.0162128.

[12] Laguarta J, Hueto F, Subirana B. Covid-19 artificial intelligence diagnosis using only cough recordings. IEEE Open J Eng Med Biol 2020;1:275–81. doi:10.1109/OJEMB.2020.3026928.

[13] Bagad P, Dalmia A, Doshi J, et al. Cough against covid: evidence of covid-19 signature in cough sounds. 2020. doi:10.48550/arXiv.2009.08790.

[14] Pal A, Sankarasubbu M. Pay attention to the cough: early diagnosis of covid-19 using interpretable symptoms embeddings with cough sound signal processing. arXiv:201002417. 2020.

[15] Bansal V, Pahwa G, Kannan N. Cough classification for covid-19 based on audio mfcc features using convolutional neural networks. In: 2020 IEEE International Conference on Computing, Power and Communication Technologies (GUCON). IEEE; 2020.

[16] Mouawad P, Dubnov T, Dubnov S. Robust detection of covid-19 in cough sounds. SN Comput Sci 2021;2(1):34. doi:10.1007/s42979-020-00422-6.

[17] Imran A, Posokhova I, Qureshi HN, et al. Ai4covid-19: ai enabled preliminary diagnosis for covid-19 from cough samples via an app. Inf Med Unlock 2020;20:100378. doi:10.1016/j.imu.2020.100378.

[18] Agbley BLY, Li J, Haq A, et al. Wavelet-based cough signal decomposition for multimodal classification. In: 2020 17th International Computer Conference on Wavelet Active Media Technology and Information Processing (ICCWAMTIP); 2020.

[19] Brown C, Chauhan J, Grammenos A, et al. Exploring automatic diagnosis of covid-19 from crowdsourced respiratory sound data. Association for Computing Machinery; 2020.

[20] Ritwik KVS, Kalluri SB, Vijayasenan D. Covid-19 patient detection from telephone quality speech data 2020:2011.04299. https://doi.org/10.48550/arXiv.2011.04299.

[21] Chaudhari G, Jiang X, Fakhry A, et al. Virufy: global applicability of crowdsourced and clinical datasets for ai detection of covid-19 from cough. 2020. doi:10.48550/arXiv.2011.13320.

[22] Dunne R, Morris T, Harper S. High accuracy classification of covid-19 coughs using mel-frequency cepstral coefficients and a convolutional neural network with a use case for smart home devices. 2020. doi:10.21203/rs.3.rs-63796/v1.

[23] Mohammed EA, Keyhani M, Sanati-Nezhad A, et al. An ensemble learning approach to digital corona virus preliminary screening from cough sounds. Sci Rep 2021;11(1):1540. doi:10.1038/s41598-021-95042-2.

[24] Chowdhury NK, Kabir MA, Rahman MM, Islam SMS. Machine learning for detecting covid-19 from cough sounds: an ensemble-based mcdm method. Comput Biol Med 2022;145:105405.

[25] Vrindavanam J, Srinath R, Shankar HH, et al. Machine learning based covid-19 cough classification models - a comparative analysis. In: 2021 5th International Conference on Computing Methodologies and Communication (ICCMC); 2021. doi:10.1109/ICCMC51019.2021.9418358.

[26] Tena A, Clari F, Solsona F. Automated detection of covid-19 cough. Biomed Signal Process Control 2022;71:103175. doi:10.1016/j.bspc.2021.103175.

[27] Dietterich TG. An experimental comparison of three methods for constructing ensembles of decision trees: bagging, boosting, and randomization. Mach Learn 2000;40(2):139–57.

[28] Dietterich TG, et al. Ensemble learning. The handbook of brain theory and neural networks. Arbib MA 2002.

[29] Polikar R. Ensemble based systems in decision making. IEEE Circuits Syst Mag 2006;6(3):21–45.

[30] Verleysen M, François D. The curse of dimensionality in data mining and time series prediction. International work-conference on artificial neural networks. Springer; 2005.

[31] Prodromidis AL, Stolfo SJ. Cost complexity-based pruning of ensemble classifiers. Knowl Inf Syst 2001;3(4):449–69. doi:10.1007/PL00011678.

[32] Brown G, Wyatt JL, Tino P, et al. Managing diversity in regression ensembles. J Mach Learn Res 2005;6(9):1621–50. doi:10.1007/s10846-005-9023-3.

[33] Chawla NV, Hall LO, Bowyer KW, et al. Learning ensembles from bites: a scalable and accurate approach. J Mach Learn Res 2004;5:421–51. doi:10.1023/B:JINT.0000026082.76722.ee.

[34] Dietterich TG, Bakiri G. Solving multiclass learning problems via error-correcting output codes. J Artif Intell Res 1994;2:263–86.

[35] Mao S, Jiao L, Xiong L, et al. Weighted classifier ensemble based on quadratic form. Pattern Recognit 2015;48(5):1688–706.

[36] Mendes-Moreira JA, Soares C, Jorge AM, et al. Ensemble approaches for regression: a survey. ACM Comput Surv 2012;45(1). doi:10.1145/2379776.2379786.

[37] Opitz DW, Shavlik JW. Actively searching for an effective neural network ensemble. Conn Sci 1996;8(3–4):337–54.

[38] Duan Q, Ajami NK, Gao X, et al. Multi-model ensemble hydrologic prediction using bayesian model averaging. Adv Water Resour 2007;30(5):1371–86. doi:10.1016/j.advwatres.2006.11.014.

[39] Schapire RE, Singer Y. Improved boosting algorithms using confidence-rated predictions. Mach Learn 1999;37(3):297–336. doi:10.1007/s10994-005-1123-6.

[40] Wolpert DH. Stacked generalization. Neural Netw 1992;5(2):241–59.

[41] Shieh AD, Kamm DF. Ensembles of one class support vector machines. In: Multiple Classifier Systems. Berlin, Heidelberg: Springer Berlin Heidelberg; 2009:181–190.

[42] Jacobs RA, Jordan MI, Nowlan SJ, et al. Adaptive mixtures of local experts. Neural Comput 1991;3(1):79–87. doi:10.1162/neco.1991.3.1.79. PMID: 31141872

[43] Orlandic L, Teijeiro T, Atienza D. The coughvid crowdsourcing dataset: a corpus for the study of large-scale cough analysis algorithms. Sci Data 2021;8(1):156. doi:10.1038/s41597-021-00937-4.

[44] COUGHVID. Codes, Available from https://c4science.ch/diffusion/10770/.

[45] Andrews M, Areekal B, Rajesh K, et al. First confirmed case of covid-19 infection in india: a case report. Indian J Med Res 2020;151(5):490–2. doi:10.4103/ijmr.IJMR_2131_20.

[46] Murata A, Taniguchi Y, Hashimoto Y, et al. Discrimination of productive and non-productive cough by sound analysis. Internal Med 1998;37(9):732–5. doi:10.2169/internalmedicine.37.732.

[47] Thorpe W, Kurver M, King G, et al. Acoustic analysis of cough. The Seventh Australian and New Zealand Intelligent Information Systems Conference, 2001. IEEE; 2001:391–394.

[48] Chatrzarrin H, Arcelus A, Goubran R, et al. Feature extraction for the differentiation of dry and wet cough sounds 2011 IEEE international symposium on medical measurements and applications. IEEE; 2011.

[49] Infante C, Chamberlain D, Fletcher R, et al. Use of cough sounds for diagnosis and screening of pulmonary disease. IEEE; 2017. p. 1–10.

[50] Miranda ID, Diacon AH, Niesler TR. A comparative study of features for acoustic cough detection using deep architectures. In: 2019 41st Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC). IEEE; 2019.

[51] You M, Wang H, Liu Z, et al. Novel feature extraction method for cough detection using nmf. IET Signal Proc 2017;11(5):515–20. doi:10.1049/iet-spr.2016.0341.

[52] Castro B, Kogan D, Geva AB. Ecg feature extraction using optimal mother wavelet. In: 21st IEEE Convention of the Electrical and Electronic Engineers in Israel. Proceedings (Cat. No.00EX377); 2000. doi:10.1109/EEEI.2000.924422.

[53] Gasca MV, Bueno-Lopez M, Molinas M, et al. Time-frequency analysis for nonlinear and non-stationary signals using hht: a mode mixing separation technique. IEEE Lat Am Trans 2018;16(4):1091–8. doi:10.1109/TLA.2018.8362142.

[54] Tokuda K, Kobayashi T, Masuko T, et al. Mel-generalized cepstral analysis-a unified approach to speech spectral estimation Third International Conference on Spoken Language Processing; 1994.

[55] OpenSMILE. open speech and music interpretation by large-space extraction. Available from https://www.audeering.com/opensmile/.

[56] El Ayadi M, Kamel MS, Karray F. Survey on speech emotion recognition: features, classification schemes, and databases. Pattern Recognit 2011;44(3):572–87. doi:10.1016/j.patcog.2010.09.020.

[57] Wang K, An N, Li BN, et al. Speech emotion recognition using fourier parameters. IEEE Trans Affect Comput 2015;6(1):69–75. doi:10.1109/TAFFC.2015.2392101.

[58] Kosasih K, Abeyratne UR, Swarnkar V, et al. Wavelet augmented cough analysis for rapid childhood pneumonia diagnosis. IEEE Trans Biomed Eng 2015;62(4):1185–94. doi:10.1109/TBME.2014.2381214.

[59] Lu C, Wang Z, Zhou B. Intelligent fault diagnosis of rolling bearing using hierarchical convolutional network based health state classification. Adv Eng Inf 2017;32:139–51. doi:10.1016/j.aei.2017.02.005.

[60] Ding X, He Q. Energy-fluctuated multiscale feature learning with deep convnet for intelligent spindle bearing fault diagnosis. IEEE Trans Instrum Meas 2017;66(8):1926–35. doi:10.1109/TIM.2017.2674738.

[61] Kiranyaz S, Ince T, Hamila R, et al. Convolutional neural networks for patient-specific ecg classification. 2015 37th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC); 2015.

[62] Kiranyaz S, Avci O, Abdeljaber O, et al. 1D convolutional neural networks and applications: a survey. Mech Syst Signal. Process 2021;151:107398. doi:10.1016/j.ymssp.2020.107398.

[63] Deng H, Runger G, Tuv E, et al. A time series forest for classification and feature extraction. Inf Sci (Ny) 2013;239:142–53. doi:10.1016/j.ins.2013.02.030.
[64] Aziz M, Tayarani-N MH, Meybodi MR. A two-objective memetic approach for the node localization problem in wireless sensor networks. Genet Programm Evolvab Mach 2016;17(4):321–58.
[65] Tayarani-N MH, Prgel-Bennett A. Anatomy of the fitness landscape for dense graph–colouring problem. Swarm Evol Comput 2015;22:47–65.
[66] Lapuschkin S, Wäldchen S, Binder A, et al. Unmasking clever hans predictors and assessing what machines really learn. Nat Commun 2019;10(1):1–8. doi:10.1038/s41467-019-08987-4.
[67] Samek W, Montavon G, Lapuschkin S, et al. Explaining deep neural networks and beyond: a review of methods and applications. Proc IEEE 2021;109(3):247–78. doi:10.1109/JPROC.2021.3060483.
[68] Balduzzi D, Frean M, Leary L, et al. The shattered gradients problem: If resnets are the answer, then what is the question? International Conference on Machine Learning PMLR; 2017.
[69] Montufar GF, Pascanu R, Cho K, et al. On the number of linear regions of deep neural networks. Adv Neural Inf Process Syst 2014;27. doi:10.5114/ceji.2014.42200.
[70] Goodfellow IJ, Shlens J, Szegedy C. Explaining and harnessing adversarial examples. arXiv:14126572. 2014.
[71] Martin-Barragan B, Lillo R, Romo J. Interpretable support vector machines for functional data. Eur J Oper Res 2014;232(1):146–55. doi:10.1016/j.ejor.2012.08.017. https://www.sciencedirect.com/science/article/pii/S0377221712006406.
[72] Molnar C. Interpretable machine learning. Lulu com; 2020.
[73] Chang CC, Lin CJ. Libsvm: a library for support vector machines. ACM Trans Intell Syst Technol (TIST) 2011;2(3):1–27.
[74] Chen CP, Liu Z. Broad learning system: an effective and efficient incremental learning system without the need for deep architecture. IEEE Trans Neural Netw Learn Syst 2017;29(1):10–24.
[75] Deeplearn toolbox. 2015. Available from https://github.com/rasmusbergpalm/DeepLearnToolbox.
[76] Rifai S, Vincent P, Muller X, et al. Contractive auto-encoders: explicit invariance during feature extraction Icml; 2011.
[77] Sohn K, Lee H. Learning invariant representations with local transformations; 2012. doi:1048550/arXiv12066418.
[78] Bruna J, Mallat S. Invariant scattering convolution networks. IEEE Trans Pattern Anal Mach Intell 2013;35(8):1872–86. doi:10.1109/TPAMI.2012.230.
[79] Chan T, Jia K, Gao S, et al. Pcanet: a simple deep learning baseline for image classification? IEEE Trans Image Process 2015;24(12):5017–32. doi:10.1109/TIP.2015.2475625.
[80] Larochelle H, Erhan D, Courville A, et al. An empirical evaluation of deep architectures on problems with many factors of variation. Proceedings of the 24th international conference on Machine learning; 2007.
[81] Iandola FN, Moskewicz MW, Ashraf K, et al. Squeezenet: alexnet-level accuracy with 50x fewer parameters and <1mb model size. 2016. doi:10.48550/arXiv.1602.07360.
[82] Sandler M, Howard A, Zhu M, et al. Mobilenetv2: inverted residuals and linear bottlenecks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR); 2018.
[83] Sheskin DJ. Handbook of parametric and nonparametric statistical procedures. crc Press; 2020.
[84] Tayarani-N M, Prügel-Bennett A. On the landscape of combinatorial optimization problems. IEEE Trans Evol Comput 2014;18(3):420–34. doi:10.1109/TEVC.2013.2281502.
[85] Prugel-Bennett A, Tayarani-Najaran M. Maximum satisfiability: anatomy of the fitness landscape for a hard combinatorial optimization problem. IEEE Trans Evol Comput 2012;16(3):319–38. doi:10.1109/TEVC.2011.2163638.
[86] Najaran MHT. How to exploit fitness landscape properties of timetabling problem: a new operator for quantum evolutionary algorithm. Expert Syst Appl 2021;168:114211. doi:10.1016/j.eswa.2020.114211.
[87] Yaochu J, Branke J. Evolutionary optimization in uncertain environments-a survey. IEEE Trans Evol Comput 2005;9(3):303–17. doi:10.1109/TEVC.2005.846356.