Complex & Intelligent Systems (2021) 7:2103–2117
https://doi.org/10.1007/s40747-021-00325-w

ORIGINAL ARTICLE

End-to-end heart sound segmentation using deep convolutional recurrent network

Yao Chen[1,2] · Yanan Sun[1] · Jiancheng Lv[3] · Bijue Jia[1] · Xiaoming Huang[4]

Received: 12 October 2020 / Accepted: 4 March 2021 / Published online: 29 March 2021

Abstract

Heart sound segmentation (HSS) aims to detect the four stages (first sound, systole, second heart sound and diastole) from a heart cycle in a phonocardiogram (PCG), which is an essential step in automatic auscultation analysis. Traditional HSS methods need to manually extract the features before dealing with HSS tasks. These artificial features highly rely on extraction algorithms, which often result in poor performance due to the different operating environments. In addition, the high-dimension and frequency characteristics of audio also challenge the traditional methods in effectively addressing HSS tasks. This paper presents a novel end-to-end method based on convolutional long short-term memory (CLSTM), which directly uses audio recording as input to address HSS tasks. Particularly, the convolutional layers are designed to extract the meaningful features and perform the downsampling, and the LSTM layers are developed to conduct the sequence recognition. Both components collectively improve the robustness and adaptability in processing the HSS tasks. Furthermore, the proposed CLSTM algorithm is easily extended to other complex heart sound annotation tasks, as it does not need to extract the characteristics of corresponding tasks in advance. In addition, the proposed algorithm can also be regarded as a powerful feature extraction tool, which can be integrated into the existing models for HSS. Experimental results on real-world PCG datasets, through comparisons to peer competitors, demonstrate the outstanding performance of the proposed algorithm.

Keywords Heart sound segmentation · End-to-end heart sound segmentation · Deep convolutional recurrent network · Sequence tagging

Introduction

Heart disease is the leading cause of death worldwide [43,66]. Many types of heart disease can be diagnosed by auscultation, which is realized in practice by experienced physicians[24, 34]. Unfortunately, auscultation highly relies on clinical expertise, which often results in biased results due to the various diagnostic levels of different doctors. Thus, automatic heart sound analysis, which can automatically analyze heart sounds via algorithms, has recently become a popular research topic [33]. Generally, automatic heart sound analysis is composed of three steps. They are preprocessing, heart sound segmentation (HSS), and classification, among which HSS is widely recognized as the key step in automatic heart sound analysis [58].

Generally, the HSS aims to detect the four stages in one heart cycle from a phonocardiogram (PCG) [7,9,11,29,60], i.e., the first sound (denoted as S_1), the systole, the second heart sound (denoted as S_2) and the diastole. Specifically, S_1 is audible at the onset of mechanical systole, and S_2 occurs at the onset of mechanical diastole. For a glance, two examples of the PCG including the four states are illustrated in Fig. 1, where different colored areas symbolize different states of the heart cycle. HSS is a challenging task [31,33]. First, PCG recording is often populated by background noise in different environments, such as friction noise between the stethoscope and skin [59]. Second, a variety of other sounds, such as the sound of breathing, conversational voice, cardiac mur-

✉ Yanan Sun
ysun@scu.edu.cn

[1] College of Computer Science, Sichuan University, Chengdu 610065, China

[2] College of Computer Science, Panzhihua University, Panzhihua 617000, China

[3] State Key Laboratory of Hydraulics and Mountain River Engineering, College of Computer Science, Sichuan University, Chengdu 610065, China

[4] CETC Cyberspace Security Research Institute Co., Ltd., Chengdu 610041, China

mur, third sound, and fourth sounds, are often injected into the recording of normal sounds even without environmental noise [33]. These sounds collectively make the HSS very difficult to exactly identify the S_1 and the S_2. Finally, most of the PCGs are common with short recordings, which propose higher requirements for the HSS algorithms. This is because the short recording makes it difficult for the corresponding algorithm to find the patterns [7].

In the past few decades, many HSS methods have been proposed to address these issues. These methods can be divided into four categories: envelope-based methods, feature-based methods, probabilistic model-based methods, and machine learning methods. Among these methods, the hidden semi-Markov model (HSMM)-based algorithms and the deep recurrent neural network (DRNN) algorithms, which are from the third and fourth categories, respectively, have demonstrated their promising performance [3,7,11,18,23,31,37,42]. Generally, both HSMM-based and DRNN algorithms consider the HSS as a sequence tagging task [23] by assigning a categorical label to each member of a sequence of the observed values. Specifically, HSMM-based algorithms assume that the state of the heart is unknown, while stochastic output and heart sounds can be observed [13,52–54,59,63]. Recently, DRNN algorithms have also shown promising performance as the dominant algorithm among various machine learning approaches, achieving state-of-the-art results using proper metrics [39,47,55]. In principle, these methods are composed of two main steps in addressing HSS tasks: extracting features and tagging sequences. Both phases are important to the performance of the HSS tasks. A variety of feature extraction methods have been developed to extract the useful features of heart sounds in recent decades, such as the homomorphic envelope features, the energy envelope feature, the Hilbert envelope features, the wavelet envelope features, the spectral features, and the power spectral density (PSD) envelope features [4,8,10,26,38,41,44,45,47–49,51,56,65]. Figure 2 illustrates some envelope features that are often used in heart segmentation algorithms. The HSMM-based algorithms often adopt the four envelope features, and the DRNN algorithm takes the combination of the envelope features and the spectral features as their inputs [6,12,39,59]. After that, these features are processed by the above algorithms, and segmentation is finished by the sequence tagging models.

Previous researchers need feature extraction algorithms to process raw audio data for two reasons. On the one hand, the raw audio signal of heart sound is frequency data, with an important structure at many time scales [46]. Therefore, researchers usually avoid modeling raw audio. Some classical feature extraction algorithms, such as wavelet transform [10,47], can greatly reduce the redundancy information of the raw audio signal [30]. On the other hand, audio data are high-dimensional sequence data, and traditional methods cannot deal with them directly. For instance, the sampling frequency of common heart audio files is 2000 Hz, while the artificial envelope features are usually used at 50 Hz in the HSMM method and DRNN processing [33,54,59]. Overall, the ability of the feature extraction algorithm determines the final performance of HSS. However, these algorithms often have some limitations. First, most of these algorithms are designed within some context, such as specific data or a specific environment. These HSS methods require considerable time and manpower to verify and process these feature extraction algorithms [33]. In addition, the extracted features may obtain good results in some data but can lead to cliff-like falls in some environments [4,10,26,38,41,45,47,49,51]. Second, the feature algorithms for HSS are not easy to extend to other heart sound annotation tasks [8]. If we need to solve other heart sound tasks, such as locating the position of the heart murmur during segmentation, these feature algorithms will redesign or combine according to the new requirements [44,48,65]. To solve the above problems, we explored a method to implement HSS with the raw audio signal. For instance, in image processing, the convolution layer adjusts the parameters of the convolutional kernel through error backpropagation and can extract the effective features of the image [16]. Moreover, some deep fully convolutional networks are designed for processing and generating raw audio waveforms [46], which utilize various dilation factors that allow the receptive field to grow exponentially with depth and cover thousands of timesteps. In certain biological sequence signals, such as electrocardiograms, convolutional and long short-term memory networks, are used to realize automatic diagnosis of heart disease, which has the advantages of fewer computations and high accuracy [64]. These results inspired us to explore the power of the convolutional network and LSTM to extract efficient features and implement end-to-end segmentation. Unlike multistage training, end-to-end network training can learn global solutions and is more convenient and elegant; it is widely used in various domains [19,57]. To the best of our knowledge, there is currently no contribution of raw audio signals to label heart sounds in the literature.

In this study, the CLSTM algorithm is proposed based on convolutional neural networks [32] and long short-term memory (LSTM) neural networks [16] to efficiently solve HSS tasks. Figure 3 illustrates the differences between the traditional method and the CLSTM. The proposed CLSTM algorithm can directly use the original digital audio signal as input and has no limitations of the existing methods, which require the features extracted in advance. Therefore, the CLSTM is trained in an end-to-end manner. The contributions of the proposed algorithm are summarized as follows:

1. An end-to-end algorithm is proposed to address HSS tasks. In the proposed algorithm, the convolutional lay-

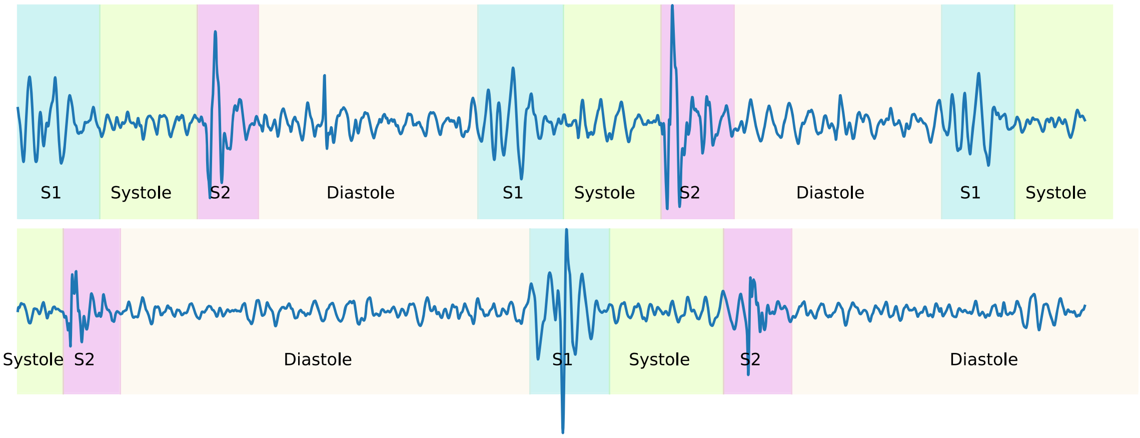

Fig. 1 Two examples of PCG signals. Each signal recording has many cardiac cycles (beats), and each heart sound beat consists of four states (S_1, systole, S_2 and diastole). The HSS task aims to determine the four states of the beat and identify the exact location of S_1 and S_2 from PCG recording

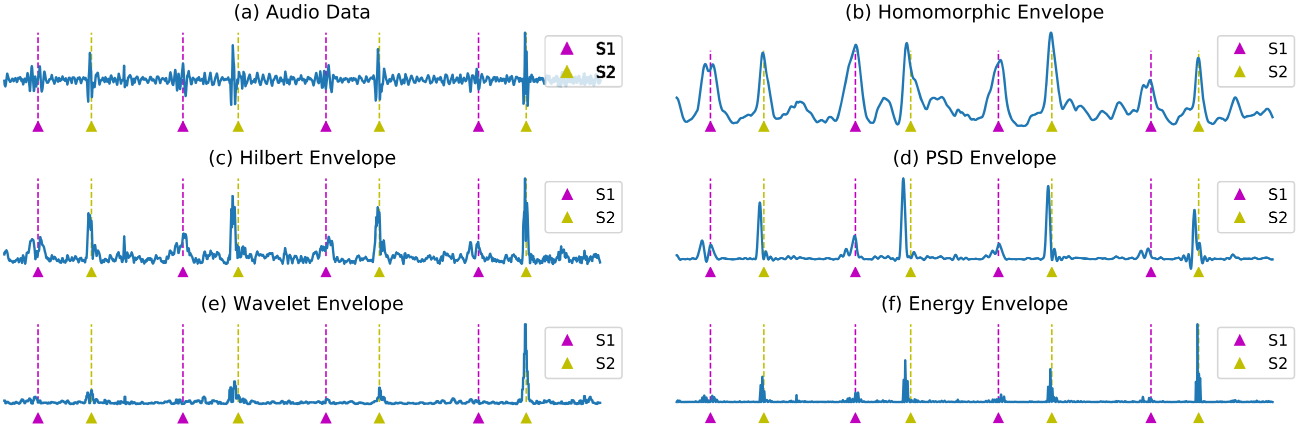

Fig. 2 The examples of envelope features. Extracting the location of S_1 and S_2 is more time dependent than amplitude dependent

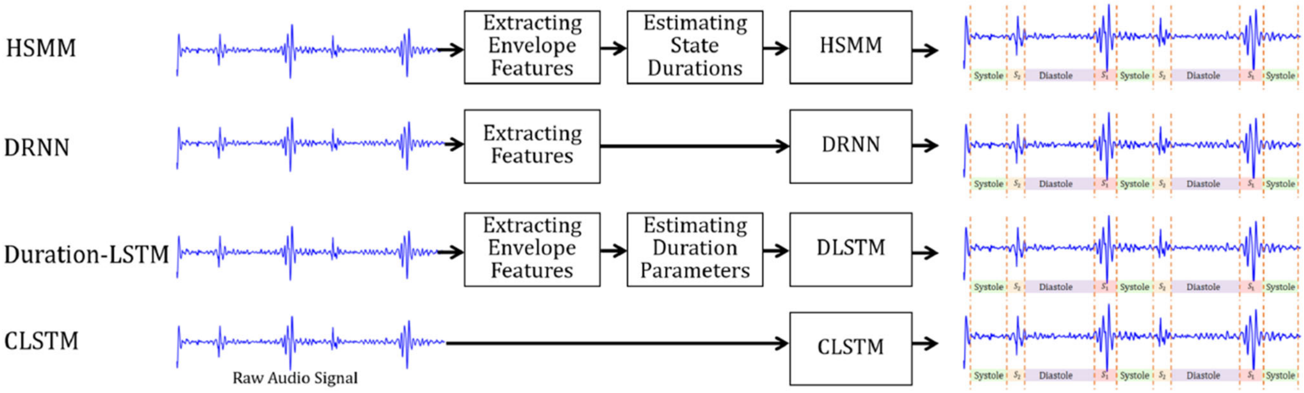
HSMM
Extracting Envelope Features
Estimating State Durations
HSMM
DRNN
Extracting Features
DRNN
Duration-LSTM
Extracting Envelope Features
Estimating Duration Parameters
DLSTM
CLSTM
CLSTM
Raw Audio Signal
Systole
Diastole

Fig. 3 The overall system design. The first three lines of the figure illustrate some design methods based on features extracted by other algorithms. The raw PCG recording must be extracted to obtain the features for the following DRNN-, duration-LSTM-, and HSMM-based algorithms [52–54,63]. The last line of the figure is our proposed method, which implements end-to-end segmentation and does not need to extract features

ers and the LSTM layers are reasonably integrated to deal with the raw acoustic data, where the convolutional layers play the role of extracting the feature and performing the downsampling, and the LSTM layers perform the sequence recognition task and feature extraction. Experimental comparisons have demonstrated the competitive performance of the proposed algorithm against state-of-the-art models.

2. To address the challenge of effectively processing high-dimensional PCG recordings, stacked convolutional layers are introduced into the proposed algorithm. Specifically, the temporal convolutional layers and LSTM layers in the model can collectively extract the features closely related to the data. These features achieve good tagging results using only two fully connected neural networks, which can also be used by traditional models.
3. The impact of the model parameters and the related key factors have been extensively investigated. Specifically, we explore the different sizes and numbers of convolutional kernels in combination with dropout and augmentation regularization and experimentally investigate the effect of the recording length and sampling rates, which can provide the guidelines of researchers in designing methods for similar tasks.

The remainder of this paper is organized as follows. The background related to the base knowledge of the proposed algorithm is introduced in the section. The next section documents the details of the proposed algorithm. To verify the performance of the proposed algorithm, the following sections show the experimental design and the experimental results, respectively. The last section provides the conclusions and our further work.

Background

In this section, the background of the LSTM and the convolutional neural networks is provided, which are the base work of the proposed CLSTM algorithm in this study. Please note that the temporal convolutional layers and the dilated convolutional layers serve as the background of the convolutional neural networks. This is because both are the main operations for processing the data having a high-dimensional signal. Please note that the recording data to be investigated in this paper are 1-D audio data.

Long short-term memory (LSTM) and BiLSTM

LSTM targets addressing the disadvantages of the vanilla recurrent neural network, such as gradient vanishing/exploration problems and hard training [16]. LSTM is often used to detect the state of sequential data, which can be naturally presented to segment heart sounds, which is principally a sequence tagging task [15,17,21,21,35,39,39] Through more complex nonlinear structures, LSTM can process and capture the long-term memory in sequential data. Specifically, its architecture uses purpose-built memory cells c_t to store information, which is beneficial to find and exploit the long-range context [47,55]. The units in the LSTM are mathematically formulated by Eq. (1):

$$\begin{cases} i_t = \sigma\left(\left[W_{xi} \quad W_{hi} \quad W_{ci} \quad 1\right]\left[x_t \quad h_{t-1} \quad c_{t-1} \quad b_i\right]^T\right) \\ f_t = \sigma\left(\left[W_{xf} \quad W_{hf} \quad W_{cf} \quad 1\right]\left[x_t \quad h_{t-1} \quad c_{t-1} \quad b_f\right]^T\right) \\ o_t = \sigma\left(\left[W_{xo} \quad W_{ho} \quad W_{co} \quad 1\right]\left[x_t \quad h_{t-1} \quad c_{t-1} \quad b_o\right]^T\right) \\ \tilde{c}_t = \tanh\left(\left[W_{xc} \quad W_{hc} \quad 1\right]\left[x_t \quad h_{t-1} \quad b_c\right]^T\right) \\ c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t \\ h_t = o_t \odot \tanh\left(c_t\right), \end{cases} \tag{1}$$

where i, f and o denote three gates (input, forget, output) that use sigmoid activation. c is the cell memory that is transformed with the activation. h_t is the output of LSTM at step t. In addition, $\odot$ denotes the elementwise multiplication operation, W_{jk} means the weight from the unit j to the unit k, b is the bias term, t refers to the time slot, and x is the input data. In Messner's study tasks, x is the stack of the envelope features as the BiLSTM input [12,14]. There have been multiple LSTM variants proposed for different purposes. In this paper, BiLSTM is used to capture the dependencies of features in two directions, which is widely used to process the annotation task as an effective version of LSTM. The BiLSTM computes the forward hidden sequence $\overrightarrow{h}$ and the backward hidden sequence $\overleftarrow{h}$ in both input directions for capturing bidirectional semantic dependencies [15,17]. The output z is computed by Eq. (2):

$$\begin{cases} \overrightarrow{h_t} = f\left(\left[W_{x\overrightarrow{h}} \quad W_{\overrightarrow{h}\overrightarrow{h}} \quad 1\right]\left[x_t \quad \overrightarrow{h_{t-1}} \quad b_{\overrightarrow{h}}\right]^T\right) \\ \overleftarrow{h_t} = f\left(\left[W_{x\overleftarrow{h}} \quad W_{\overleftarrow{h}\overleftarrow{h}} \quad 1\right]\left[x_t \quad \overleftarrow{h_{t-1}} \quad b_{\overleftarrow{h}}\right]^T\right) \\ z_t = \left[W_{z\overrightarrow{h}} \quad W_{z\overleftarrow{h}} \quad 1\right]\left[\overrightarrow{h_t} \quad \overleftarrow{h_t} \quad b_z\right]^T. \end{cases} \tag{2}$$

After that, the output sequence z is obtained through the forward $\overrightarrow{h}$ and backward hidden layer sequence $\overleftarrow{h}$. Usually, each step z_t can use the softmax function to classify for annotating the sequence.

Temporal convolutional layer

The temporal convolutional layer is also known as the 1-D convolutional layer, which is widely used in image and video action recognition [28,32]. This is because the temporal convolutional layer can capture how features at lower levels change over time. The filters slide over the whole input sequence and help identify different features present in the

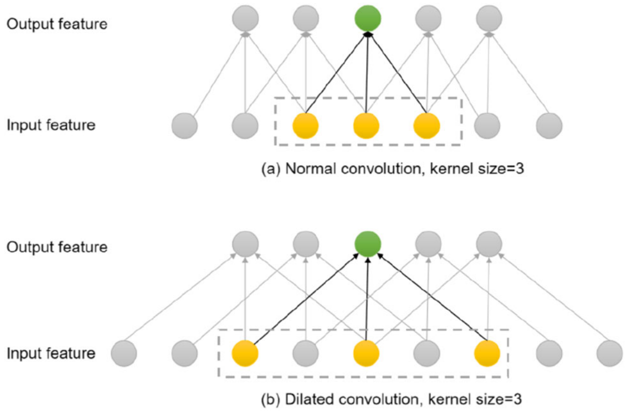
Output feature
Input feature
(a) Normal convolution, kernel size=3
Output feature
Input feature
(b) Dilated convolution, kernel size=3

Fig. 4 An example to demonstrate how dilated convolution works. Compared with normal convolution, the output of the same length has a capacity of more input

temporal convolutional layer. Feature maps are the output of one filter applied to the previous layer and can be regarded as the convolutional activation of the corresponding filter. In practice, the temporal convolutional layer is often followed by a pooling operation that is used to efficiently compute long-term time patterns. At each convolutional layer, the previous layer's feature maps are convoluted with learnable kernels and put through the activation functions to form the output feature map. The output is obtained by Eq. (3):

$$x_k^l = f\left(\sum_{i=1}^{N_{l-1}} \mathrm{conv1}D\left(w_{ik}^{l-1}, x_i^{l-1}\right) + b_k^l\right), \tag{3}$$

where x_k^l is defined as intermediate output through the activation function f, x_i^{l-1} is the output of the i neuron at layer $l-1$, w_{ik}^{l-1} is the kernel from the i neuron at layer $l-1$ to the k neuron at layer l and b is bias. conv() is 'invalid' 1-D convolutional without zero-padding.

Dilated convolution

Dilated convolution, which is achieved by the traditional operation with holes, has been previously used in various contexts, e.g., signal processing [1], waveNet [46], sound classification [67] and image segmentation [50]. The receptive field of the dilated convolution is the implicit area captured on the initial input by each input to the next layer in the convolutional neural network, which is an efficient method for increasing the receptive view of the network exponentially and linear parameter accretion. As shown in Fig. 4, the dilated convolution is similar to the traditional convolution where the filter is applied over an area but skips the input values with a certain step [46].

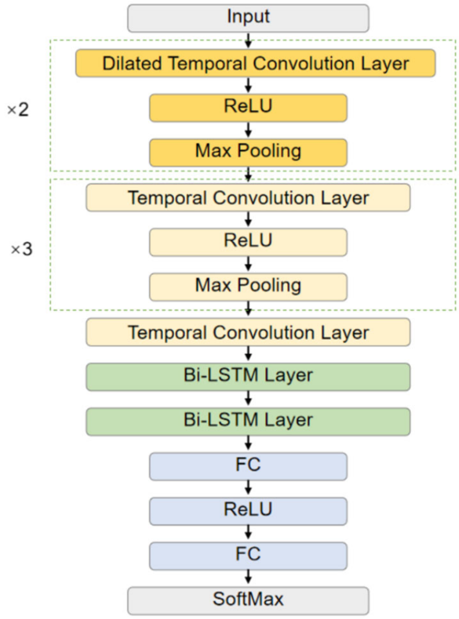
Input
×2
Dilated Temporal Convolution Layer
ReLU
Max Pooling
×3
Temporal Convolution Layer
ReLU
Max Pooling
Temporal Convolution Layer
Bi-LSTM Layer
Bi-LSTM Layer
FC
ReLU
FC
SoftMax

Fig. 5 The overall architecture of the proposed CLSTM algorithm

The proposed algorithm

In this section, the proposed CLSTM algorithm is detailed. Specifically, the architecture of CLSTM is elaborated first. Then the training details of CLSTM are documented. Finally, the output of the CLSTM middle layers is discussed, which is helpful for interpreting what features the proposed CLSTM algorithm has learned.

The CLSTM architecture

The ideas and principles of network model design are as follows. First, convolutional layers are utilized to extract the meaningful features of the raw recording data. Stacked convolutional layers and numerous feature maps can extract rich and effective features. As some studies have shown that the accuracy of segmentation is correlated with the length of input in models [6,12], the dilated temporal convolutional layer is introduced into this structure to effectively increase the receptive view. Second, the pooling layer is very important in this architecture, which is designed to halve the length of input and capture the valid features of high-dimensional audio recording. Third, the LSTM layers focus on the sequence tagging tasks, which have been sufficiently demonstrated to be effective in the HSS task [6,12,39]. Figure 5 shows the architecture of the proposed CLSTM algorithm.

CLSTM is composed of convolutional layers, LSTM layers, and fully connected layers. Specifically, the convolutional layers contain three parts. The first part is approximately two dilated temporal convolutional units, three temporal convolutional units, and one temporal convolutional layer. Each dilated temporal convolutional unit is composed of a dilated temporal convolutional layer, a rectified linear unit (ReLU) activation function, and a max-pooling layer. The temporal convolutional unit contains a temporal convolutional layer, a ReLU activation function, and a max-pooling layer. Separately, the temporal convolutional layers extract the features and deal with the 1-D data, which is often high-dimensional. These layers are stacked at the beginning of the model to effectively extract the features of the high-dimensional PCG recordings. In these temporal convolutional layers, k filters slide over the whole audio sequence to generate k feature maps. The max-pooling layer following every convolutional layer is designed to remove redundant information and downsample features to a fixed length. The pooling operation obtains the maximum output in the sequence neighborhood and is applied to reduce the complexity of each feature map and construct important features. The LSTM layers are two stacked BiLSTM layers, and the output sequence of one layer generates the input sequence for the next layer. The number of units for each BiLSTM layer is set to 128 in the proposed CLSTM algorithm. Finally, the fully connected layers are designed to concatenate the output of the BiLSTM layers for the softmax process. The softmax function is over all the predicted output annotation sequences to compute the probability of each state, and the negative log-likelihood function is used for training this neural network model [47,55].

Algorithm 1: CLSTM algorithm training

```
   Input: The raw PCG signal recording $X = (x_1, x_2, \ldots, x_U)$.
 1 Downsampling PCG recording to 1, 600 $Hz$, then processing the
   signal with a bandpass filter ;
 2 for number of training iterations do
 3     for k steps do
 4         Getting minibatch of $M$ fixed-length PCG recording
           $x^{(1)}, x^{(2)}, \ldots, x^{(M)}$;
 5         Extracting features using convolutional layers and
           downsampling to fixed length using pooling layers,
           $C = (c_1, c_2, \ldots, c_T)$ ;
 6         BiLSTM models the sequence and incorporates long-term
           sequential information to obtain the feature vectors
           $Z = (z_1, z_2, \ldots, z_T)$;
 7         The softmax layer generates the sequence of classification
           outputs $(\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_T)$:
               $\hat{y}_t = \mathrm{softmax}\,(g\,(z_t))$
 8         Computing the loss with respect to that sequence $L(\theta)$:
               $L(\theta) = -\frac{1}{T}\sum_{t=1}^{T} y_t \log\left(\hat{y}_t(\theta)\right)$
 9         Given $M$ samples, computing the overall loss:
               $L_{minibatch}(\theta) = \frac{1}{M}\sum_{m=1}^{M} L_m(\theta) + \frac{\lambda}{2}\|\theta\|_2^2$
10     end
11     Update the model by ascending its stochastic gradient
       $\nabla L_{minibatch}(\theta)$;
12 end
```

Training of CLSTM

Based on the conventions of the neural network community, the proposed CLSTM algorithm is trained by minibatch stochastic gradient descent (SGD), which offers computational and statistical efficiency in training. In CLSTM, the input of the network is a sequence of raw audio clips denoted as $X = (x_1, x_2, \ldots, x_U)$, and the sequence of classification outputs is $\hat{Y} = \left(\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_T\right)$, where U is the length of the input sequence and T is the length of the output sequence. The corresponding target is $Y = (y_1, y_2, \ldots, y_T)$, which is the sequence of one-hot encoding vectors. The middle features using convolutional layers and pooling layers are denoted as C. The relationship between U and T is:

$$T = \frac{U}{2^N}, \tag{4}$$

where N is the number of pooling layers. The pool operation reduces the length of the sequence, thus greatly reducing the model complexity. The details of the training strategy are shown in Algorithm 1. Specifically, line 1 shows the preprocessing of the PCG recording. Because the architecture of the proposed model has five pooling layers and the sampling rate of the target sequence is 50 Hz, the raw audio signal needs to be converted to 1600 Hz using a polyphase antialiasing filter, as suggested in [55]. The signal is filtered with a fourth-order Butterworth bandpass filter with cut-off frequencies at 25 Hz and 400 Hz [27]. As the majority of the frequency content of S_1 and S_2 is below 150 Hz [2], the main content of heart sounds is retained, and low- and high-frequency noise can be reduced after filtering. Lines 2–11 demonstrate the training details of training the proposed algorithm using the minibatch SGD. Particularly, line 4 shows the process of randomly obtaining clips from raw audio when training. The fixed-length raw audio clips are extracted from the raw recording using a random start position. The method of randomly obtaining clips is widely used in the sequence annotation and classification of biological data, which can greatly expand the diversity of training data to avoid overfitting. It is also an essential step to train the CLSTM algorithm. Line 5 shows how to extract the middle features C using convolutional layers and pooling layers. After several convolutional layers and pooling layers, the length of the middle feature is T and coincides with the target length. Line 6 shows the process of the middle feature

C using BiLSTM layers. BiLSTM can capture the long-term temporal dependence of the two directions. z_t is the output sequence of stacked BiLSTM layers, which is computed through the backward layer $\overrightarrow{h}$ and the forward layer $\overleftarrow{h}$. Line 7 illustrates the CLSTM algorithm to obtain the probability of the target tags, and g denotes the nonlinear function that outputs a vocabulary-sized vector in each time step. At each output time step t, the model makes a prediction $\hat{y}_t$, where $\hat{y}_t \in \{S_1, \text{Systole}, S_2, \text{Diastole}\}$. Line 8 shows how to obtain the sequence loss $L(\theta)$, and line 9 is the minibatch loss $L_{\text{minibatch}}(\theta)$, where λ denotes the hyperparameter that controls the strength of the penalty, $\|\boldsymbol{\theta}\|_2^2$ is an ℓ_2-norm regularization term, which can help avoid overfitting and improve the accuracy of deep learning models.

To efficiently utilize the audio data, we increase the depth of the networks by adding more convolutional layers and recurrent layers. However, it inevitably becomes more challenging to train the network using the gradient descent algorithm as the size and depth increase. In the training process, the loss value of the model can be small, and the prediction accuracy is high, but the prediction accuracy is lower in test data, which refers to the overfitting problem [62,68]. To address this issue, the dropout mask [25] and weight decay [36,61] are used in the model. The dropout works like the activation neurons stop working with a certain probability. This causes the model to not rely too much on the local features, thus reducing overfitting and improving the performance of the model. L2 regularization [5] is another commonly used method to deal with overfitting by decaying the weights, which can also help to improve the convergence of the model. Please note that some other mechanisms that are commonly used by other research, such as batch normalization [20] and adding the skip connections between layers [40], are not adapted in the proposed algorithm. This is because they cannot significantly improve the performance of the proposed CLSTM algorithm after our careful and extensive experimental investigation.

The output of the middle layers

We have noted that in the proposed CLSTM algorithm, the output of the temporal convolutional layers and the LSTM layers are very similar to the features extracted by traditional methods [26,38,45,56]. These layers can collectively extract the features closely related to the data, and using a large number of convolutional kernels, the model can be regarded as extracting the essential characteristics of more categories.

Figure 6 is an example of the output of the middle layers. These visualizations can provide insight into the internal representations for the convolutional layers and LSTM layers. As shown in Fig. 6a, the model input is a raw PCG recording, which samples at 1600 Hz. Figure 6b illustrates an output after the temporal convolutional layers, which is very similar to envelope features extracted from PCG signals. Each feature map can be regarded as the convolutional activation of the corresponding filter over the whole sequence. The output after the LSTM layers is illustrated in Fig. 6c, which can extract the local features efficiently and use both past and future input features to determine the labels of the segmentation. Generally, the number of feature maps can be regarded as the number of feature types extracted in these layers. Therefore, the combination of convolutional layers and LSTM layer methods can be more effective and powerful because the feature map has diversity.

Experimental design

Benchmark dataset

Based on the conventions of the HSS community [39,59], the Massachusetts Institute of Technology heart sound database (MITHSDB) [63] is employed as the benchmark dataset in this experiment. In particular, the MITHSDB dataset is a high-quality and rigorously validated standard database of heart sound signals obtained from a variety of healthy and pathological conditions [33]. In this dataset, there are synchronous 405 PCGs and 405 electrocardiography (ECG) recordings varying from $9 \sim 36$ s. Corresponding to the positions of the R-peak and the T wave-end in synchronous ECG recordings, accurate positions of S_1 and S_2 in the PCG are easily obtained. These positions are recognized as the gold standard of the HSS tasks. In addition, the MITHSDB dataset is sufficiently diverse compared to the other datasets. These PCG recordings were collected from 121 subjects and were grouped as follows: (1) normal control group: 117 recordings, (2) murmurs relating to mitral valve prolapse (MVP): 134 recordings, (3) innocent or benign murmurs group (benign): 118 recordings, (4) aortic disease (AD): 17 recordings, and (5) other miscellaneous pathological conditions (MPC): 23 recordings.

Segmentation annotation information is essential to the training and evaluation of the proposed algorithm. However, manual annotation is not an easy task in PCG, especially in this dataset, which has 405 recordings and 14,559 beats. To achieve this, the target label is obtained by the popular Springer method [59] and the error labels are manually revised. In the tagging sequence, the four stages are annotated, i.e., S_1, systole, S_2 sound, and diastole, as 0, 1, 2, and 3, respectively.

Evaluation metric

In this experiment, the F_1 score of locating S_1 and S_2 sound is used to evaluate the algorithm performance. As the four

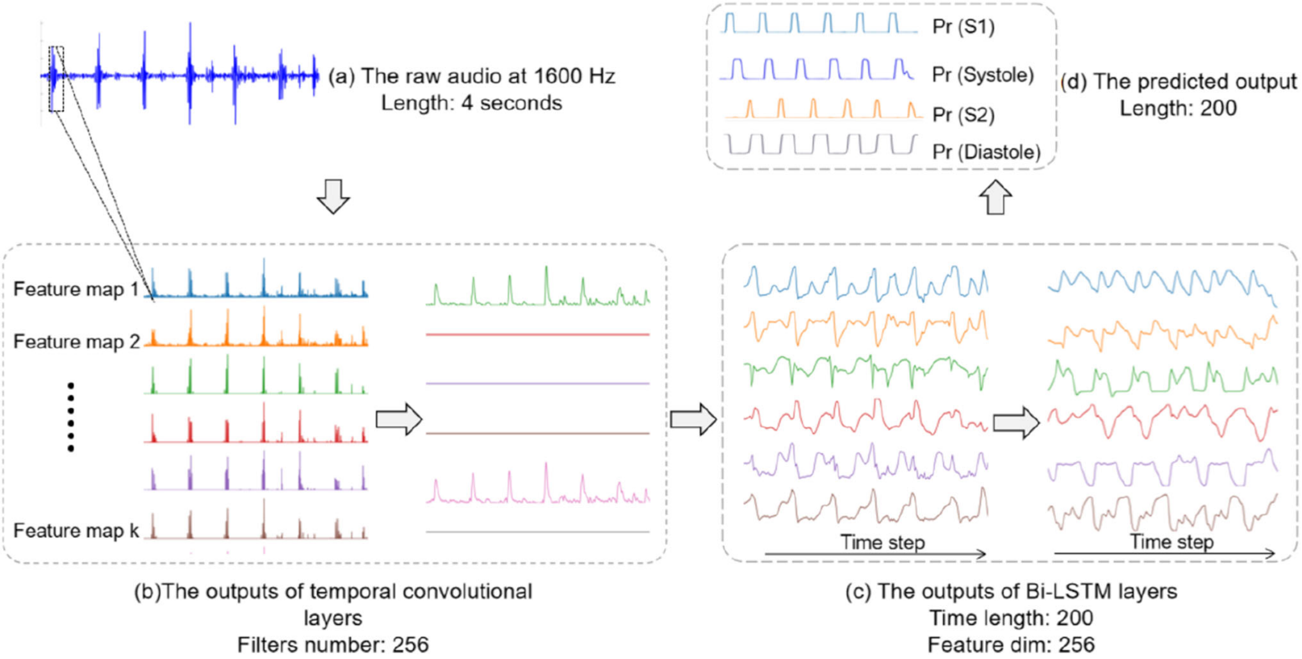

Fig. 6 The input, output, feature maps, and implementation details in the proposed CLSTM algorithm

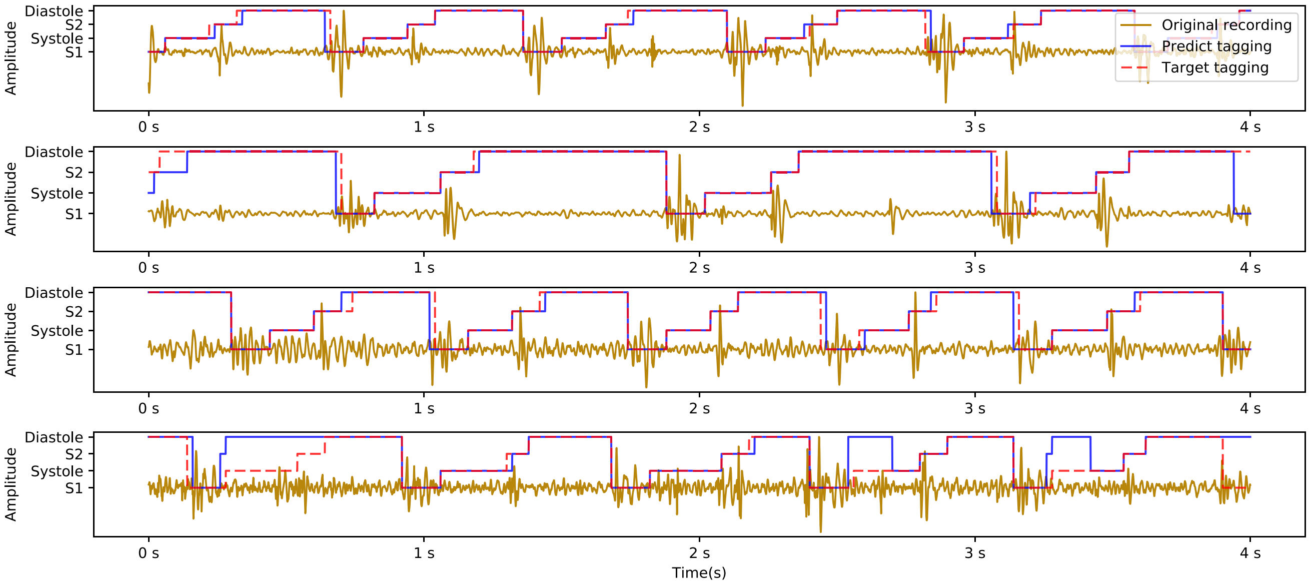

Fig. 7 The output of the proposed CLSTM algorithm. Specifically, the raw PCG recording is shown using the yellow line, and the target states and the predicted states are illustrated with the red dotted line and the blue line, respectively

stages in a heart cycle can be easily obtained through the positions of S_1 and S_2 in the PCG, this evaluation metric is effective for HSS tasks. In addition, the output of the model has a negligible chance of matching target labels at an exact time. Therefore, the tolerance of the windows is used to address this problem, and the prediction is considered correct just when it falls within these windows. The tolerance window is often defined as an absolute time in HSS tasks. For example, Springer set the tolerance window to 100 ms [59], Schmidt set to 60 ms [54], and Messner set to 40 ms [39]. In practice, the tolerance window is enough to find an exact location when the value is set to 100 ms, but the stricter standard is also used to measure the algorithm performance. Furthermore, the first and last 20% of the length are excluded from the tests, as the segmentation algorithm easily fails to identify the heart sounds at the beginning and end of the recordings [54].

A heart sound is identified as a true positive (TP) when the distance between the predicted position and the target position is less than the tolerance window; otherwise, it is defined as a false positive (FP). Accuracy is the most common measure of classification, but accuracy does not adequately reflect the results in the case of unbalanced samples. In the HSS tasks, precision or positive predictive value (P_+ or PPV), sensitivity (Se), and F_1 score are used to comprehensively measure the performance of the annotation. P_+ is the fraction of correct instances among the retrieved heart sounds, which are defined by Eq. (5). Se refers to how many positive examples in the sample have been predicted correctly, which is defined by Eq. (6). The F_1 score can be regarded as a weighted average of model accuracy and recall, having a maximum value of 1 and a minimum value of 0. The measure score is calculated using Se and P_+ as the intermediate quantities. F_1 score is defined by Eq. (7):

$$P_+ = \frac{\text{number of TP}}{\text{number of TP} + \text{number of FP}} \tag{5}$$

$$Se = \frac{\text{number of TP}}{\text{total number of } S_1 \text{ and } S_2} \tag{6}$$

$$F_1 = \frac{2 \times P_+ \times Se}{P_+ + Se}. \tag{7}$$

Peer competitors

In this study, three popular models are chosen as peer competitors for comparing the annotation performance: DRNN [39] and logistic regression hidden semi-Markov model (LR-HSMM) [59]. The LR-HSMM method addresses the problem of HSS within noise using HSMM and logistic regression for emission probability estimation and achieves state-of-the-art performance when the tolerant window is 100 ms. The DRNN method is a framework for heart sound segmentation using neural networks, which uses traditional artificial methods as the input of the BiLSTM model. DRNN also achieves the state-of-the-art model when the tolerant window is 40 ms. Two versions of the DRNN algorithm (i.e., BiLSTM and BiGRNN) are used in follow-up experiments. Duration-LSTM (LSTM) and duration-LSTM (BiLSTM) [6] are also compared with the proposed model. Similar to DRNN, duration-LSTM also uses envelope sequence features as input but incorporates duration parameters to model intrinsic sequence characteristics. Additionally, the convolutional part of our model is also used for comparison. Two versions of CNN with different numbers of convolutional layers are implemented with the same parameters and inputs in experiments. Based on the conventions, the four proven envelope features (homomorphic, Hilbert, wavelet, and the PSD envelope) are extracted from the raw PCG signal and used as input in all peer competitors [10,45,47,56,65]. These features and their combination have been proven to be the best artificial features for the HSS task [6,39,59].

Implementation details

All recordings are randomly divided into training and test sets according to the proportion of 70% and 30% and repeated 5 times. In CLSTM, the input is an audio file with a sample rate of 1600 Hz. The next part is the convolutional layers, which are contributed by two dilated temporal convolutional layers and four normal temporal convolutional layers. All convolutional layers have 256 feature maps, and the size of each feature map is set to 5. The pooling size of 2 is used for all max-pooling layers, which are located behind the convolutional layer. After 5 pooling layers, the length of the feature sequence is downsampled to 50 Hz, and this sampling rate is the time step of the output sequence. In addition, each BiLSTM layer has 256 feature dimensions. For all compared algorithms, the Adam optimizer [22] is used to train their respective weights. The configuration of the proposed CLSTM algorithm is presented in Fig. 6. The learning rate of CLSTM is set to 0.001, and the batch size is 64. All models are implemented and tested using Keras and PyTorch.

Experiment results

In this section, the experimental results of the proposed CLSTM algorithm against the chosen peer competitors are presented at the different tolerance windows specified. Furthermore, we also evaluate the sizes and numbers of the convolutional kernels in the proposed CLSTM, the comparisons of inputs of different sampling frequencies, the performance of varied input lengths, and convergence analysis to extensively demonstrate the effectiveness of our designs.

Overall results

The segmentation results of CLSTM and peer competitors are shown in Tables 1 and 2. These tables illustrate the final F_1 scores of the proposed CLSTM algorithm and the chosen peer competitors and the F_1 scores for S_1 and S_2 sounds. Specifically, Table 1 shows the experimental results of the tolerance window specified as 100 ms. The results of LR-HSMM achieve an average F_1 score of 95.63±0.85%, which comes from its seminal paper [59]. According to Messner's method [39], BiLSTM and BiGRNN are implemented and trained under the same conditions. The average F_1 score of BiLSTM is 94.12±0.42% and 94.46±0.42%, respectively. The result of duration-LSTM (LSTM) [6] is 94.82±0.49%, and duration-LSTM (BiLSTM) is 96.11±0.27%. The results of CNN are depicted in the following two lines. In the

Table 1 Segmentation results of CLSTM and peer competitors when the tolerance window = 100 ms

Method	Input	S_e	P_+	$F_1^{S_1}$	$F_1^{S_2}$	F_1
LR-HSMM[a]	Four envelope	95.34 ± 0.88	**95.92** ± 0.83	96.95 ± 0.90	94.29 ± 1.08	95.63 ± 0.85
DRNN (BiGRNN)	Four envelope	94.93 ± 0.34	93.33 ± 0.56	95.46 ± 0.42	92.78 ± 0.48	94.12 ± 0.42
DRNN (BiLSTM)	Four envelope	95.01 ± 0.43	93.91 ± 0.48	95.85 ± 039	93.06 ± 0.53	94.46 ± 0.42
Duration-LSTM (LSTM)	Envelope + (eHR+ eSys)	95.60 ± 0.49	94.19 ± 0.52	95.59 ± 0.38	94.09 ± 0.66	94.82 ± 0.49
Duration-LSTM (BiLSTM)	Envelope + (eHR+ eSys)	96.36 ± 0.32	95.88 ± 0.23	96.28 ± 0.24	**95.98** ± 0.48	96.11 ± 0.27
CNN (5-conv-layer)	Raw audio	**98.59** ± 1.24	55.55 ± 1.18	73.99 ± 2.79	74.12 ± 1.87	74.13 ± 0.96
CNN (14-conv-layer)	Raw audio	92.80 ± 1.74	73.24 ± 2.26	82.26 ± 1.23	81.25 ± 1.48	81.83 ± 1.29
CLSTM (no dilated layers)	Raw audio	96.68 ± 0.41	95.70 ± 0.36	96.89 ± 0.32	95.48 ± 0.44	96.16 ± 0.34
CLSTM (5-dilated layers)	Raw audio	96.47 ± 0.55	95.54 ± 1.31	96.69 ± 0.99	95.18 ± 0.90	95.87 ± 0.92
CLSTM (2-dilated layers)	Raw audio	96.78 ± 0.64	95.67 ± 0.85	**97.11** ± 0.71	95.54 ± 0.74	**96.18** ± 0.70

Best results are highlighted in bold

[a] LR-HSMM results are derived from the Springer's study [59]

Table 2 Segmentation results of CLSTM and peer competitors when the tolerance window = 40 ms

Method	Input	S_e	P_+	$F_1^{S_1}$	$F_1^{S_2}$	F_1
LR-HSMM[a]	Envelope	**95.5**	**94.6**	–	–	94.6
DRNN (BiGRNN)	Four envelope	93.13 ± 0.34	91.75 ± 0.56	94.58 ± 0.42	90.29 ± 0.55	92.44 ± 0.42
DRNN (BiLSTM)	Four envelope	93.51 ± 0.43	92.56 ± 0.50	95.07 ± 0.43	91.00 ± 0.58	93.03 ± 0.43
Duration-LSTM (LSTM)	Envelope + (eHR + eSys)	93.50 ± 0.24	92.47 ± 0.77	94.27 ± 0.39	91.67 ± 0.62	92.95 ± 0.50
Duration-LSTM (BiLSTM)	Envelope + (eHR + eSys)	94.87 ± 0.41	94.53 ± 0.43	95.58 ± 0.41	**93.86** ± 0.44	**94.69** ± 0.41
CNN (5-conv-layer)	Raw audio	86.13 ± 1.15	44.50 ± 1.82	58.10 ± 1.67	50.47 ± 2.33	59.35 ± 1.48
CNN (14-conv-layer)	Raw audio	89.22 ± 1.50	68.09 ± 2.71	58.10 ± 1.67	76.22 ± 2.07	76.02 ± 1.72
CLSTM (no dilated layers)	Raw audio	94.90 ± 0.41	93.85 ± 0.39	95.88 ± 0.33	92.90 ± 0.49	94.31 ± 0.36
CLSTM (5-dilated layers)	Raw audio	94.84 ± 0.63	94.01 ± 1.38	95.85 ± 1.01	92.94 ± 1.02	94.35 ± 0.99
CLSTM (2-dilated layers)	Raw audio	94.92 ± 0.65	93.92 ± 0.89	**96.09** ± 0.72	92.89 ± 0.80	94.37 ± 0.73

Best results are highlighted in bold

[a] LR-HSMM results are derived from the Messner's study [39]

task, the F_1 score of CNN (5 layers) is 74.13 ± 0.96%, and the version of convolutional layer14 is 81.83 ± 1.29%. The result of CLSTM having two dilated layers obtains an average F_1 score of 96.18 ± 0.70% on the same dataset. Experimental results demonstrate that the proposed CLSTM algorithm achieves the best score among the comparisons, which demonstrates the effectiveness of CLSTM. Please note that the chosen peer competitors, i.e., LR-HSMM, BiLSTM, BiGRNN, and duration-LSTM, cannot directly take effect on the raw data instead of using the artificial features. The proposed CLSTM algorithm is directly based on raw data, i.e., it is an end-to-end algorithm. In Table 2, the performance of the algorithm is evaluated by a smaller tolerance window. The final results of the CLSTM algorithm are 96.09%, 92.89%, and 94.37 ± 0.73% under more stringent conditions, and it also obtained the best results in locating the two heart sounds. It is clearly shown that the results of the proposed CLSTM algorithm are also the best among the comparisons. Please note that the F_1 scores of S_1 and S_2 are not provided in Table 2 because both were not reported in the corresponding paper.

To make the comparisons more intuitive, some final segmentation output of the CLSTM algorithm is depicted in Fig. 7. As seen in this figure, the raw PCG recording signals, target states, and estimated states are illustrated in different colors in these examples. The CLSTM algorithm accurately identifies four states of heart sounds and works well even in many locations that are difficult to manually distinguish. In the first three examples, the output label of CLSTM is equivalent to the target state except for a tiny time shift.In the fourth example, there was an error in annotations due to noise or murmur at some locations.

To further analyze the proposed algorithm to check where the proposed algorithm fails to tag the sequence in the test set, the confusion matrix of the estimated label is shown in Fig. 8. The performance of the proposed model can be observed from the figure, and some errors generated by the model can be explained based on this. For example, the most common labeling error is S_2 tagged to diastole and systole. The reason is that the duration of Systole and Diastole is relatively long, and S_1 and S_2 account for only a small part of a heart sound cycle. Considerable noise and murmurs occurring during this period affect the labeling of S_2.

Comparison of kernel sizes and number

The kernel sizes and the number of kernels are often viewed as important parameters affecting the performance of convolution-based models. Therefore, a series of experiments are illustrated to analyze their sensitivity to the proposed CLSTM algorithm, and these experiments are initiated by finding appropriate sizes and numbers of convolutional kernels on the same dataset. Figure 9 shows the results with varying numbers of convolutional kernels when

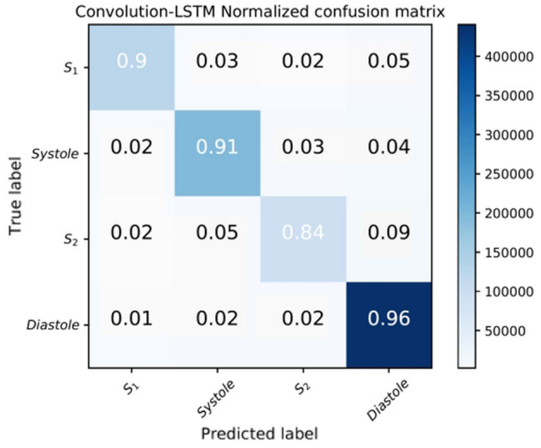

Fig. 8 The normalized confusion matrix of the proposed CLSTM algorithm

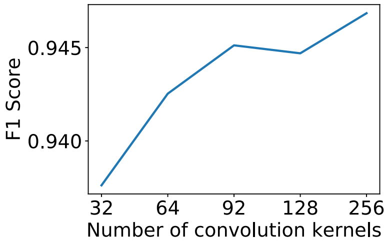

Fig. 9 Comparisons of the numbers of convolutional kernels, showing the F scores for a CLSTM with five convolutional layers using a {32, 64, 92, 128, 256} kernel per layer

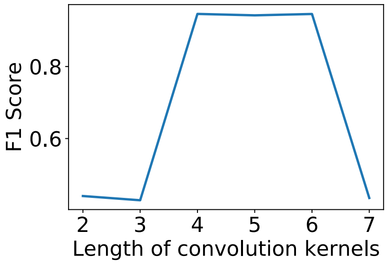

Fig. 10 Comparisons of the sizes of convolutional kernels, showing the F scores for a CLSTM using {2, 3, 4, 5, 6, 7} kernel lengths

the depth of the convolutional layers is 5. In principle, the results become better as the number of kernels increases. However, this also makes the network hard to train. When there are too many convolutional kernels, the network often fails to train in experiments. For this reason, an appropriate number of convolutional kernels ($\text{Number}_{\text{kernel}} = 256$) are used for the subsequent experiments.

Another exploration is about the sizes of the convolutional kernels. Please note that the convolutional kernel only considers the length because the PCG data in 1-D and the convolutional operation in this experiment is 1-D accordingly. The result is shown in Fig. 10, and the best score is achieved with kernel lengths of 6. In addition, the model would fail

Table 3 Comparison of different sample rates as input in CLSTM

Sample rate	S_e	P_+	$F_1^{S_1}$	$F_1^{S_2}$	F_1
200 Hz	86.78	86.87	90.72	82.93	86.83
400 Hz	91.47	89.88	93.07	88.27	90.67
800 Hz	**97.13**	93.32	95.68	94.70	95.19
1600 Hz	96.78	**95.67**	**97.11**	**95.54**	**96.18**

Best results are highlighted in bold

when the size of the convolutional kernels is too large or too small.

Comparison of input of different sampling frequencies

In this experiment, the impact of different sampling rates is explored for the proposed algorithm. The raw recordings in the MITHSDB are audio data sampled at 2000 Hz, which needs to be converted into an appropriate sampling rate as the input of the algorithm. When changing the sampling frequency of the input audio signal, the number of pooling layers must be changed to ensure that the output sampling rate is 50 Hz. It should be noted that the other parameters have not changed. Table 3 shows the results for CLSTM with different sample rates. As seen from this table, the best results are obtained with 1600 Hz, followed by 800 Hz and 400 Hz. 100 Hz is not listed because it loses audio details and is not enough to train the model.

Performance variations with input length

Testing the impact of different input lengths on performance can guide the design of the model. Because the smallest length in the dataset is 9 s, we extracted the length of clips from 2 to 8 s. The performance of variations with input length evaluated by two tolerance windows is illustrated in Figs. 11 and 12. All input segments are extracted dynamically to avoid overfitting during the training phase. Although there are some fluctuations in the F_1 score, we can observe that the longer the input length is, the more accurate the annotation. This experimental result shows that the F_1 score is positively correlated with the length of the input. The reason is that a longer input length can obtain more global information and more accurately annotate these models.

Convergence analysis

The convergence of F_1 and loss value score is illustrated in Figs. 13 and 14. In Fig. 13, with the increase in training times, the accuracy of the model increases in the test set. After many epochs, the F_1 score of CLSTM becomes stable. DRNN (BiLSTM) and DRNN (BiGRU) easily converge, while CNN is the slowest. As shown in Fig. 14, the objective function value decreases first and changes significantly after several iterations. In addition, the convergence rate of DRNN (BiLSTM) and DRNN (BiGRU) is much faster than CLSTM and CNN.

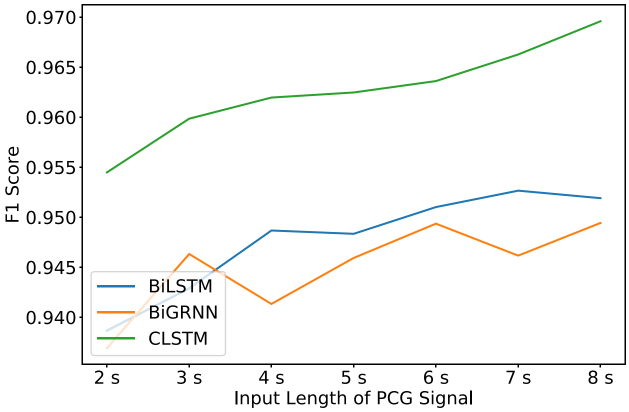

Fig. 11 The results using different input lengths (tolerance window = 100 ms)

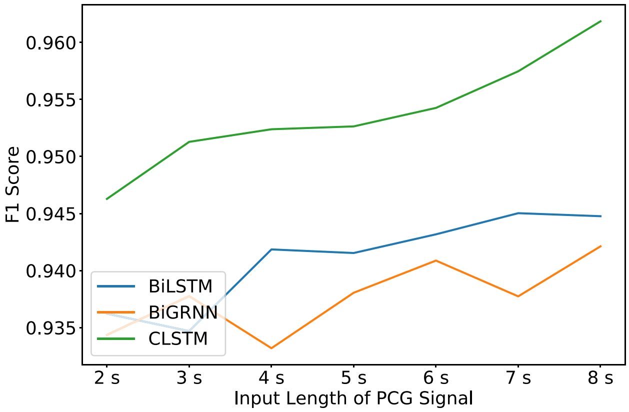

Fig. 12 The results using different input lengths (tolerance window = 40 ms)

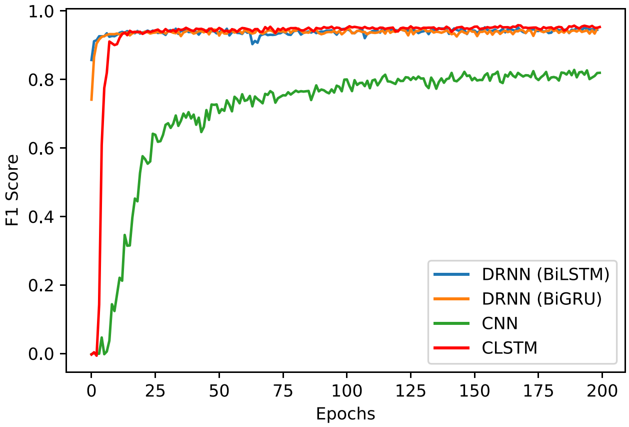

Fig. 13 The convergence of F_1 over test data

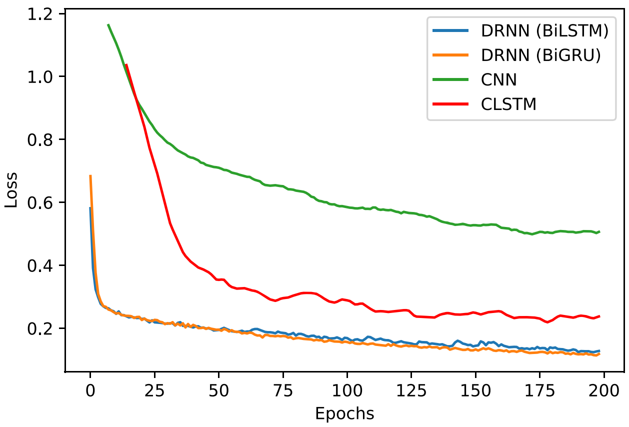

Fig. 14 The convergence of the loss function over test data

Conclusion

In this study, the CLSTM algorithm can deal with the HSS task effectively by the end-to-end method and directly estimate the four heart states from the raw audio signal. In CLSTM, the temporal convolutional layers are utilized to extract the meaningful features of the raw recording data, the pooling layers are used to perform downsampling, and the LSTM layers focus on capturing the long-term memory of the 1-D feature and sequence recognition task. As an end-to-end algorithm that combines the extracting features and tagging sequence in a model, the proposed CLSTM algorithm is good at processing high-dimensional audio data. CLSTM can be regarded as a feature extraction method used by other sequence models (e.g., LR-HSMM). The proposed algorithm is also flexible and can be extended to more annotations in the PCG. A series of experimental results was performed and demonstrated the promising performance of the proposed algorithm. In addition, a group of experiments was also designed to verify the robustness of the proposed algorithm in terms of parameter settings. In the future, we will continue exploring the benefits of the convolutional method approach to HSS and improve the performance of the deep neural network model through comprehensive use of the sequence of each stage.

Acknowledgements This study was funded by the State Key Program of National Science Foundation of China under Grant 61836006 and the National Natural Science Fund for Distinguished Young Scholar under Grant 61625204.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

References

1. Arneodo A, Grasseau G, Holschneider M (1989) Wavelet transform analysis of invariant measures of some dynamical systems. Springer, Berlin
2. Arnott PJ, Pfeiffer GW, Tavel ME (1984) Spectral analysis of heart sounds: relationships between some physical characteristics and frequency spectra of first and second heart sounds in normals and hypertensives. J Biomed Eng 6(2):121
3. Baranek HL, Lee HC, Cloutier G, Durand LG (1989) Automatic detection of sounds and murmurs in patients with Lonescu–Shiley aortic bioprostheses. Med Biol Eng Comput 27(5):449–455
4. Boutana D, Benidir M, Barkat B (2011) Segmentation and identification of some pathological phonocardiogram signals using time-frequency analysis. Iet Signal Process 5(6):1
5. Bühlmann P, Yu B, Bűhlmann P (2003) Boosting with the l 2 loss: regression and classification. J Am Stat Assoc 98(462):324–339
6. Chen Y, Lv J, Sun Y, et al (2020) Heart sound segmentation via Duration Long–Short Term Memory neural network. Applied Soft Computing 95:106540
7. Choi S, Cho SH, Park CW, Shin JH (2015) A novel cardiac spectral envelope extraction algorithm using a single-degree-of-freedom vibration model. Biomed Signal Process Control 18:169–173
8. Choi S, Jiang Z (2008) Comparison of envelope extraction algorithms for cardiac sound signal segmentation. Expert Syst Appl 34(2):1056–1069
9. Cozic M, Durand LG, Guardo R (1998) Development of a cardiac acoustic mapping system. Med Biol Eng Comput 36(4):431
10. Ergen B, Tatar Y, Gulcur HO (2012) Time-frequency analysis of phonocardiogram signals using wavelet transform: a comparative study. Comput Methods Biomech Biomed Eng 15(4):371–81
11. Feldman M, Braun S (1997) Description of free responses of sdof systems via the phase plane and Hilbert transform: the concepts of envelope and instantaneous frequency. In: Proceedings of SPIE, the international society for optical engineering, vol 3089. Society of Photo-Optical Instrumentation Engineers, pp 973–979
12. Fernando T, Ghaemmaghami H, Denman S, Sridharan S, Hussain N, Fookes C (2020) Heart sound segmentation using bidirectional lstms with attention. IEEE J Biomed Health Inform 24(6):1601–1609
13. Gamero LG, Watrous R (2003) Detection of the first and second heart sound using probabilistic models. In: International conference of the IEEE engineering in medicine and biology society
14. Gers FA, Schraudolph NN, Schmidhuber J (2003) rgen: learning precise timing with lstm recurrent networks. J Mach Learn Res 3(1):115–143
15. Ghosh S, Vinyals O, Strope B, et al (2016) Contextual lstm (clstm) models for large scale nlp tasks. arXiv preprint arXiv:1602.06291
16. Goodfellow I, Bengio Y, Courville A (2016) Deep learning. MIT Press. http://www.deeplearningbook.org

17. Graves A (2012) Supervised sequence labelling with recurrent neural networks. Springer, Berlin
18. Huang NE, Zheng S, Long SR, Wu MC, Shih HH, Zheng Q, Yen NC, Chi CT, Liu HH (1998) The empirical mode decomposition and the Hilbert spectrum for nonlinear and non-stationary time series analysis. Proc Math Phys Eng Sci 454(1971):903–995
19. Phan H, Andreotti F, Cooray N, et al (2019) SeqSleepNet: end-to-end hierarchical recurrent neural network for sequence-to-sequence automatic sleep staging[J]. IEEE Transactions on Neural Systems and Rehabilitation Engineering 27(3): 400–410
20. Jia B, Lv J, Liu D (2019) Deep learning-based automatic downbeat tracking: a brief review[J]. Multimedia Systems 25(6): 617–638
21. Jia B, Lv J, Liu D (2019) Deep learning-based automatic downbeat tracking: a brief review. Multimed Syst 1–22
22. Kingma DP, Ba J (2014) Adam: aa method for stochastic optimization. arXiv:1412.6980
23. Kirbas I, Peker M (2017) Signal detection based on empirical mode decomposition and Teager–Kaiser energy operator and its application to p and s wave arrival time detection in seismic signal analysis. Neural Comput Appl 28(10):3035–3045
24. Kochanek KD, Murphy SL, Xu J, Arias E (2015) Mortality in the united states, 2013. Nchs Data Brief 168(168):1–8
25. Krizhevsky A, Sutskever I, Hinton G (2012) Imagenet classification with deep convolutional neural networks. In: NIPS
26. Kumar D, Carvalho P, Antunes M, Paiva RP, Henriques J (2011) Noise detection during heart sound recording using periodicity signatures. In: International conference of the IEEE engineering in medicine biology society, pp 3119–3123
27. Latif S, Usman M, Rana R, Qadir J (2018) Phonocardiographic sensing using deep learning for abnormal heartbeat detection. IEEE Sens J 18(22):9393–9400
28. Lea C, Vidal R, Reiter A, Hager GD (2016) Temporal convolutional networks: a unified approach to action segmentation, pp 47–54
29. Leatham A (1975) Auscultation of the heart and phonocardiography. Churchill Livingstone, London
30. Lehner RJ, Rangayyan RM (1987) A three-channel microcomputer system for segmentation and characterization of the phonocardiogram. IEEE Trans Bio-med Eng 34(6):485–9
31. Liang H, Lukkarinen S, Hartimo I (1997) Heart sound segmentation algorithm based on heart sound envelogram. In: Computers in cardiology. IEEE, pp 105–108
32. Lin S, Jia K, Yeung DY, Shi BE (2015) Human action recognition using factorized spatio-temporal convolutional networks. In: IEEE international conference on computer vision
33. Liu C (2016) An open access database for the evaluation of heart sound algorithms. Physiol Meas 37(12):2181–2213
34. Liu Q, Wu X, Ma X (2018) An automatic segmentation method for heart sounds. Biomed Eng Online 17(1):106
35. Lv J, Zhang Y (2005) An improved backpropagation algorithm using absolute error function. Lect Notes Comput Sci 3496:585–590
36. MacKay DJC (1992) A practical Bayesian framework for backpropagation networks. Adv Neural Inf Process Syst 4(3):448–472
37. Maglogiannis I, Loukis E, Zafiropoulos E, Stasis A (2009) Support vectors machine-based identification of heart valve diseases using heart sounds. Comput Methods Progr Biomed 95(1):47–61
38. Messer SR, Agzarian J, Abbott D (2001) Optimal wavelet denoising for phonocardiograms. Microelectron J 32(12):931–941
39. Messner E, Zöhrer M, Pernkopf F (2018) Heart sound segmentation—an event detection approach using deep recurrent neural networks. IEEE Trans Biomed Eng PP:1. https://doi.org/10.1109/TBME.2018.2843258
40. Ming T, Zhang X (2017) Speech enhancement based on deep neural networks with skip connections. In: 2017 IEEE international conference on acoustics, speech and signal processing (ICASSP)
41. Moukadem A, Dieterlen A, Hueber N, Brandt C (2013) A robust heart sounds segmentation module based on s-transform. Biomed Signal Process Control 8(3):273–281
42. Moukadem A, Dieterlen A, Hueber N, et al (2011) Localization of heart sounds based on S-transform and radial basis function neural network[C]//15th Nordic-Baltic Conference on Biomedical Engineering and Medical Physics (NBC 2011). Springer, Berlin, Heidelberg. pp 168–171
43. Mozaffarian D, Benjamin EJ, Go AS, Arnett DK, Blaha MJ, Cushman M, Das SR, de Ferranti S, Després JP, Fullerton HJ et al (2016) Executive summary: heart disease and stroke statistics-2016 update: a report from the American heart association. Circulation 133(4):447
44. Naseri H, Homaeinezhad MR (2013) Detection and boundary identification of phonocardiogram sounds using an expert frequency-energy based metric. Ann Biomed Eng 41(2):279–292
45. Navin GC, Palaniappan R, Swaminathan S (2005) Classification of homomorphic segmented phonocardiogram signals using grow and learn network. In: International conference of the engineering in medicine and biology society
46. Oord A, Dieleman S, Zen H, et al (2016) Wavenet: A generative model for raw audio[J]. arXiv preprint arXiv:1609.03499
47. Oskiper T, Watrous R (2002) Detection of the first heart sound using a time-delay neural network. Comput Cardiol 29(2):537–540
48. Papadaniil CD, Hadjileontiadis LJ (2014) Efficient heart sound segmentation and extraction using ensemble empirical mode decomposition and kurtosis features. IEEE J Biomed Health Inform 18(4):1138–1152
49. Patidar S, Pachori RB, Garg N (2015) Automatic diagnosis of septal defects based on tunable-q wavelet transform of cardiac sound signals. Expert Syst Appl 42(7):3315–3326
50. Qiao Z, Cui Z, Niu X, Geng S, Yu Q (2017) Image segmentation with pyramid dilated convolution based on resnet and u-net. In: International conference on neural information processing
51. Quiceno-Manrique AF, Godino-Llorente JI, Blanco-Velasco M, Castellanos-Dominguez G (2010) Selection of dynamic features based on time-frequency representations for heart murmur detection from phonocardiographic signals. Ann Biomed Eng 38(1):118–137
52. Ricke AD, Povinelli RJ, Johnson MT (2005) Automatic segmentation of heart sound signals using hidden Markov models. In: Computers in cardiology, pp 953–956
53. Sainath TN, Vinyals O, Senior A, Sak H (2015) Convolutional, long short-term memory, fully connected deep neural networks. In: IEEE international conference on acoustics, speech and signal processing, pp 4580–4584
54. Schmidt SE, Holsthansen C, Graff C, Toft E, Struijk JJ (2010) Segmentation of heart sound recordings by a duration-dependent hidden Markov model. Physiol Meas 31(4):513–29
55. Sepehri AA, Gharehbaghi A, Dutoit T, Kocharian A (2010) Kiani: a novel method for pediatric heart sound segmentation without using the ecg. Comput Methods Progr Biomed 99(1):43–48
56. Sharma H, Sharma KK, Bhagat OL (2015) Respiratory rate extraction from single-lead ecg using homomorphic filtering. Comput Biol Med 59:80–86
57. Smirnov E A, Timoshenko D M, Andrianov S N (2014) Comparison of regularization methods for imagenet classification with deep convolutional neural networks. Aasri Procedia 6:89–94
58. Springer D B, Tarassenko L, Clifford G D (2015) Logistic regression-HSMM-based heart sound segmentation. IEEE Transactions on Biomedical Engineering 63(4): 822–832
59. Springer DB, Tarassenko L, Clifford GD (2016) Logistic regression-hsmm-based heart sound segmentation. IEEE transactions on bio-medical engineering 63(4):822–832

60. Sun S (2015) An innovative intelligent system based on automatic diagnostic feature extraction for diagnosing heart diseases. Knowl Based Syst 75(C):224–238
61. Sun Y, Xue B, Zhang M, et al. Evolving deep convolutional neural networks for image classification[J]. IEEE Transactions on Evolutionary Computation, 2019, 24(2): 394–407
62. Sun Y, Xue B, Zhang M, Yen GG (2018) A new two-stage evolutionary algorithm for many-objective optimization. IEEE Trans Evolut Comput PP(99):1
63. Syed Hassan Z (2003) Mit automated auscultation system. Massachusetts Institute of Technology
64. Tan JH, Hagiwara Y, Pang W, Lim I, Oh SL, Adam M, San Tan R, Chen M, Acharya UR (2018) Application of stacked convolutional and long short-term memory network for accurate identification of cad ecg signals. Comput Biol Med 19–26
65. Wang H, Hu Y, Liu L, Wang Y, Zhang J (2010) Heart sound analysis based on autoregressive power spectral density. In: International conference on signal processing systems, pp V2-582–V2-586
66. WHO (2017) Cardiovascular diseases (cvds). sponsored by WHO. https://www.who.int/en/news-room/fact-sheets/detail/cardiovascular-diseases-(cvds)
67. Zhang X, Zou Y, Wei S (2017) Dilated convolution neural network with leakyrelu for environmental sound classification. In: 2017 22nd international conference on digital signal processing (DSP)
68. Zhou XJ, Lv JC, Zhao MH, Zhang H (2010) Advances in the genetics of anti-glomerular basement membrane disease. Am J Nephrol 32(5):482–490