This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TBME.2018.2843258, IEEE Transactions on Biomedical Engineering

Heart Sound Segmentation - An Event Detection Approach using Deep Recurrent Neural Networks

Elmar Messner, *Student Member, IEEE,* Matthias Zöhrer, and Franz Pernkopf, *Senior Member, IEEE*

Abstract*—*Objective: **In this paper, we accurately detect the state-sequence *first heart sound (S1) - systole - second heart sound (S2) - diastole*, i.e. the positions of S1 and S2, in heart sound recordings. We propose an event detection approach, without explicitly incorporating a priori information of the state duration. This renders it also applicable to recordings with cardiac arrhythmia and extendable to the detection of extra heart sounds (third and fourth heart sound), heart murmurs, as well as other acoustic events. *Methods:* We use data from the 2016 PhysioNet/CinC Challenge, containing heart sound recordings and annotations of the heart sound states. From the recordings, we extract spectral and envelope features and investigate the performance of different deep recurrent neural network (DRNN) architectures to detect the state-sequence. We use virtual-adversarial training (VAT), dropout and data augmentation for regularization. *Results:* We compare our results with the state-of-the-art method and achieve an average score for the four events of the state-sequence of $\mathbf{F_1 \approx 96\%}$ on an independent test set. *Conclusion:* Our approach shows state-of-the-art performance carefully evaluated on the 2016 PhysioNet/CinC Challenge dataset. *Significance:* In this work, we introduce a new methodology for the segmentation of heart sounds, suggesting an event detection approach with DRNNs using spectral or envelope features.**

***Index Terms*—heart sound segmentation, acoustic event detection, deep recurrent neural networks, gated recurrent neural networks, bidirectional**

I. Introduction

COMPUTER-AIDED heart sound analysis can be considered as a twofold task: segmentation and subsequent classification. The accurate segmentation of the fundamental heart sounds, or more precisely of the state-sequence *first heart sound (S1) - systole - second heart sound (S2) - diastole*, is a challenging task. In heart sound recordings of healthy adults only S1 and S2 are present. However, extra heart sounds (third heart sound - S3 and fourth heart sound - S4) can occur during diastole, i.e. in the interval S2-S1, and heart murmurs during systole, i.e. in the interval (S1-S2), and/or diastole, as shown in Figure 1. Furthermore, the corruption by different noise sources (e.g. motion artefacts, ambient noise) and other body sounds (e.g. lung sounds, cough sounds) renders the segmentation even more challenging.

Elmar Messner, Matthias Zöhrer and Franz Pernkopf are with the Signal Processing and Speech Communication Laboratory, Graz University of Technology, Graz, Austria.

This work was supported by the Austrian Science Fund (FWF) under the project number P27803-N15. We acknowledge NVIDIA for providing GPU computing resources.

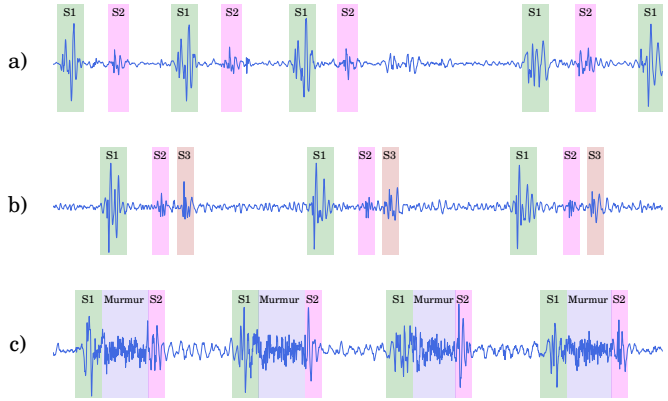


Fig. 1. Examples of heart sound recordings: a) cardiac arrhythmia, b) extra (third) heart sound S3, and c) heart murmur (mitral valve prolapse). The marked events are first (S1), second (S2) and third (S3) heart sounds and heart murmurs.

According to [1], existing heart sound segmentation methods are classified into four groups: Envelope-based methods [2]–[7], feature based methods [8]–[14], machine learning methods [15]–[21] and HMM methods [22]–[28]. The authors in [28] introduced a logistic regression hidden semi-Markov model (LR-HSMM) to predict the most likely sequence of states in the order of *S1 - systole - S2 - diastole*, using a priori information about expected durations of the heart sound states. In experiments they achieve an average F-score of $F_1 = 95.63\,\%$ on an independent test set. Due to the significant improvement in comparison to other reported methods in the literature, it is considered as the state-of-the-art method by the authors in [1]. A more extensive evaluation of the algorithm on the 2016 PhysioNet/CinC Challenge data [1] is presented in [29]. The authors report an average F-score of $F_1 = 98.5\,\%$ for segmenting S1 and systole intervals and $F_1 = 97.2\,\%$ for segmenting S2 and diastole intervals. They observe detection errors especially in the situations of long heart cycles and irregular sinus rhythm. Also, the authors in [21] point out that the LR-HSMM-method [28] may be unsuitable for the segmentation in recordings with cardiac arrhythmia. Their main objective is to investigate if S1 and S2 can be detected without using a priori information about the state duration. They propose a machine learning approach with a deep neural network (DNN) in combination with mel frequency cepstral coefficients (MFCCs)-features for S1 and S2 heart sound recognition. Using the K-means algorithm, they cluster the MFCC features into two groups to refine their representation and discriminative capability. The refined features are then fed

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TBME.2018.2843258, IEEE Transactions on Biomedical Engineering

to a DNN. In experiments with a relatively small dataset, the authors show that S1 and S2 can be detected with an accuracy of 91 %, outperforming well-known classifiers, such as K-nearest neighbor, Gaussian mixture models, logistic regression, and support vector machines.

Within this paper, we exploit spectral information and temporal dependencies of heart sounds for heart sound segmentation. To this end, we propose an acoustic event detection approach with deep recurrent neural networks (DRNNs) [30]–[32]. Recurrent neural networks (RNNs) are suitable to process sequential input of variable length, and learn temporal dependencies within the data [33], [34]. They are already used in heart sound *classification* [35]–[37], but, to the best of our knowledge, not introduced specifically for heart sound *segmentation*. Compared to the LR-HSMM-method [28], we do not directly incorporate a priori information about the state durations, because the model is capable of learning the temporal dependencies itself. Furthermore, we are flexible regarding the order of occurring states enabling to model additionally S3, S4 and heart murmurs. In acoustic event detection, sound events are usually detected by onsets and offsets, defining the beginning and ending of a particular event within an audio recording. It is differentiated between *polyphonic* and *monophonic* event scenarios: In the first case, multiple events can occur at the same time, whereas in the second case no overlapping events exist. Within this work, we consider heart sound segmentation as a *monophonic* event scenario, although heart sound recordings can be contaminated with body sounds and different noise sources, and therefore represent a *polyphonic* event scenario. DNNs show a significant boost in performance when applied to acoustic event detection. In particular, Gencoglu et al. [38] proposed a DNN architecture for acoustic event detection. Although DNNs are powerful network architectures, they do not model temporal context explicitly. To account for temporal structure long short term memory (LSTM) networks, i.e. DNNs capable of modeling temporal dependencies, have been applied to acoustic keyword spotting [39] and polyphonic sound event detection [40]. Performance in recognition comes at the expense of computational complexity and the amount of labeled data. LSTMs have a relatively high model complexity and parameter tuning is not always simple. A simplification of LSTMs are gated recurrent neural networks (GRNNs), which have less parameters, but achieve comparable performance. Due to this fact, we focus on GRNNs for the accurate segmentation of fundamental heart sounds, or more precisely of the state-sequence *S1 - systole - S2 - diastole*. GRNNs already show promising results for acoustic event detection [41]. To exploit future information as well, and not just information from the past, we also consider bidirectional recurrent neural networks [42], [43].

In particular, we extract spectral and envelope features from heart sounds and investigate the performance of different DRNN architectures to detect the state-sequence, i.e. acoustic events. We use data from the 2016 PhysioNet/CinC Challenge [1], containing heart sound recordings and annotations of the heart sound states. Our main contributions and results are:

- We compare different recurrent neural network architectures.
- We evaluate bidirectional gated recurrent neural networks (BiGRNNs) in combination with virtual adversarial training (VAT), dropout and data augmentation for regularization.
- We show state-of-the-art performance on the 2016 PhysioNet/CinC Challenge dataset.

The paper is structured as follows: In Section II, we discuss common DRNN architectures. In particular, we explain *vanilla* RNNs, LSTMs, GRNNs, and their implementations as bidirectional networks. We introduce virtual adversarial training (VAT), dropout and two data augmentation approaches for regularization in Section III. In Section IV, we introduce our processing framework for heart sound segmentation and show experimental results, including the comparison with the LR-HSMM-method [28]. Finally, we discuss our findings in Section V and conclude the paper in Section VI.

II. Recurrent Neural Network Architectures

RNNs are extensions of traditional feed forward neural networks [44]. They are able to process sequential input of variable length, and learn temporal dependencies within the data. Various RNN architectures exist, such as Elman networks [45], Jordan networks [46], or Hop field networks [47]. In this work, we focus on a more classical model, i.e. the *vanilla* RNN, two very popular architectures, i.e. LSTMs and GRNNs, and their bidirectional implementations.

A. Vanilla Recurrent Neural Networks

Figure 2 shows the flow-graph of an RNN unit. We consider a recurrent neural network with L layers, with $l \in \{1, ..., L-1\}$ indexing the hidden layers of the network. With the given input vector $\mathbf{x}_f^l$ and the previous recurrent hidden state vector $\mathbf{h}_{f-1}^l$, the sum between the dot product $\mathbf{W}_x^l \mathbf{x}_f^l$, the projected previous hidden state $\mathbf{W}_h^l \mathbf{h}_{f-1}^l$ and the bias term $\mathbf{b}_h^l$ is computed. $\mathbf{W}_x^l$ is the input weight matrix and $\mathbf{W}_h^l$ the hidden

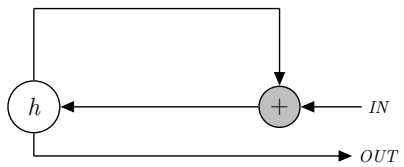


Fig. 2. Flow graph of a *vanilla* RNN unit. h denotes the activation.

weight matrix. A non-linear function g is applied to obtain the output $\mathbf{h}_f^l$, as shown in Equation (1). The output $\mathbf{h}_f^l$ is used as the input for the next layer $\mathbf{x}_f^{l+1}$. As shown in Equation (2), the output of the last hidden layer $\mathbf{h}_f^{L-1}$ is fed into the output layer. $\mathbf{W}_y$ is the output weight matrix and $\mathbf{b}_y$ the output bias term. A non-linear function m is applied to obtain the output $\mathbf{y}_f$.

$$\mathbf{h}_f^l = g(\mathbf{W}_x^l \mathbf{x}_f^l + \mathbf{W}_h^l \mathbf{h}_{f-1}^l + \mathbf{b}_h^l) \quad (1)$$

$$\mathbf{y}_f = m(\mathbf{W}_y \mathbf{h}_f^{L-1} + \mathbf{b}_y) \quad (2)$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TBME.2018.2843258, IEEE Transactions on Biomedical Engineering

Multiple RNN layers can be stacked, forming a deep recurrent neural network. They are trained via back-propagation through time using a differentiable cost function.

B. Long Short Term Memory Networks

LSTMs [48], [49] are temporal recurrent neural networks using memory cells to store temporal information. In contrast to RNNs, LSTMs have memory cells, which store or erase their content using *input* gates i or *forget* gates r. An additional *output* gate o is used to access this information. Figure 3 shows the flow-graph of an LSTM unit.

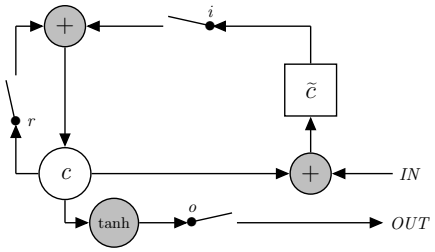


Fig. 3. Flow graph of a LSTM unit [31]. *i*, *r*, and *o* are the input, forget and output gates, respectively. *c* denote the memory cell and $\widetilde{c}$ the new memory cell content.

In Equations (3-8), the network is mathematically described. The input states $\mathbf{i}_f^l$ are calculated by applying a sigmoid function σ to the sum of the dot-product of the input weight matrix $\mathbf{W}_{xi}^l$ and the inputs $\mathbf{x}_f^l$, the projected previous hidden states $\mathbf{W}_{hi}^l\mathbf{h}_{f-1}^l$ and the bias vector $\mathbf{b}_i^l$ of layer l (cf. Equation 3). The forget states $\mathbf{r}_f^l$ (cf. Equation 4) and output states $\mathbf{o}_f^l$ (cf. Equation 5) are computed in a similar way, except for using individual *forget* matrices $\mathbf{W}_{xr}^l, \mathbf{W}_{hr}^l$ and the *forget* bias vector $\mathbf{b}_r^l$ and *output* matrices $\mathbf{W}_{xo}^l, \mathbf{W}_{ho}^l$ and the *output* bias vector $\mathbf{b}_o^l$, respectively. The new memory states $\tilde{\mathbf{c}}_f^l$ are obtained by applying a *tanh* activation function on the sum of the projected inputs $\mathbf{W}_{xc}^l\mathbf{x}_f^l$, previous hidden memory states $\mathbf{W}_{hc}^l\mathbf{h}_{f-1}^l$ and the bias vector $\mathbf{b}_c^l$ in Equation (6). The memory cell states $\mathbf{c}_f^l$ are updated by the previous memory states $\mathbf{c}_{f-1}^l$ and $\tilde{\mathbf{c}}_f^l$ (cf. Equation (7)), weighted with the forget states $\mathbf{r}_f^l$ and the input state $\mathbf{i}_f^l$, respectively ($\odot$ denotes an element-wise product). The outputs $\mathbf{h}_f^l$ are computed with the current memory states $\tanh(\mathbf{c}_f^l)$ and the output states $\mathbf{o}_f^l$ in Equation (8).

$$\mathbf{i}_f^l = \sigma(\mathbf{W}_{xi}^l\mathbf{x}_f^l + \mathbf{W}_{hi}^l\mathbf{h}_{f-1}^l + \mathbf{b}_i^l) \tag{3}$$

$$\mathbf{r}_f^l = \sigma(\mathbf{W}_{xr}^l\mathbf{x}_f^l + \mathbf{W}_{hr}^l\mathbf{h}_{f-1}^l + \mathbf{b}_r^l) \tag{4}$$

$$\mathbf{o}_f^l = \sigma(\mathbf{W}_{xo}^l\mathbf{x}_f^l + \mathbf{W}_{ho}^l\mathbf{h}_{f-1}^l + \mathbf{b}_o^l) \tag{5}$$

$$\tilde{\mathbf{c}}_f^l = \tanh(\mathbf{W}_{xc}^l\mathbf{x}_f^l + \mathbf{W}_{hc}^l\mathbf{h}_{f-1}^l + \mathbf{b}_c^l) \tag{6}$$

$$\mathbf{c}_f^l = \mathbf{r}_f^l \odot \mathbf{c}_{f-1}^l + \mathbf{i}_f^l \odot \tilde{\mathbf{c}}_f^l \tag{7}$$

$$\mathbf{h}_f^l = \mathbf{o}_f^l \odot \tanh(\mathbf{c}_f^l) \tag{8}$$

In classical RNNs, the hidden activation is overwritten at each time-step (cf. Equation 1). LSTMs are able to decide whether to keep or erase existing information with the help of their gates. If LSTMs detect important features from an input sequence at early stage, they easily carry this information over a long distance, hence, capturing potential long-distance dependencies.

C. Gated Recurrent Neural Networks

GRNNs [31], [32] are simplifications of LSTMs, achieving comparable performance, but having less parameters. Gated recurrent units have *reset*- and *update*-gates, coupling static and temporal information. This allows the network to learn temporal information. Whenever an important event happens, the *update*-gate z decides to renew the current state of the model. The network can forget the previously computed information by deleting the current state of the model with the *reset*-gate r. Figure 4 shows the flow graph of a gated recurrent unit.

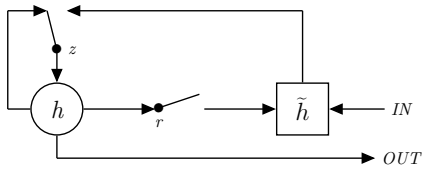


Fig. 4. Flow graph of a gated recurrent unit [31]. *r* and *z* denote the reset and update gates, and *h* and $\tilde{h}$ the activation and the candidate activation.

Equations (9-12), mathematically describe the network. Equation (9) starts with the output states $\mathbf{h}_f^l$, which are computed with a linear interpolation between past states $\mathbf{h}_{f-1}^l$ and current information $\tilde{\mathbf{h}}_f^l$, by using the *update*-states $\mathbf{z}_f^l$. The *update*-states $\mathbf{z}_f^l$ determine the update behavior of the units. According to Equation (10), they are computed as sigmoid function of the weighted input $\mathbf{x}_f^l$ and the past hidden states $\mathbf{h}_{f-1}^l$. $\mathbf{W}$ and $\mathbf{b}$ denote the weights and bias terms. In Equation (11) the states $\tilde{\mathbf{h}}_f^l$ are computed with a non-linear function g, applied to the affine transformed input and the previous hidden states $\mathbf{h}_{f-1}^l$. This is similar to Equation (1), only differing in the additional *reset*-state $\mathbf{r}_f^l$, which is element-wise multiplied with $\mathbf{h}_{f-1}^l$. In Equation (12), the reset state is computed with the current inputs $\mathbf{x}_f^l$ and the provided hidden states $\mathbf{h}_{f-1}^l$.

$$\mathbf{h}_f^l = (1 - \mathbf{z}_f^l) \odot \mathbf{h}_{f-1}^l + \mathbf{z}_f^l \odot \tilde{\mathbf{h}}_f^l \tag{9}$$

$$\mathbf{z}_f^l = \sigma(\mathbf{W}_{xz}^l\mathbf{x}_f^l + \mathbf{W}_{hz}^l\mathbf{h}_{f-1}^l + \mathbf{b}_z^l) \tag{10}$$

$$\tilde{\mathbf{h}}_f^l = g(\mathbf{W}_{xh}^l\mathbf{x}_f^l + \mathbf{W}_{hh}^l(\mathbf{r}_f^l \odot \mathbf{h}_{f-1}^l) + \mathbf{b}_h^l) \tag{11}$$

$$\mathbf{r}_f^l = \sigma(\mathbf{W}_{xr}^l\mathbf{x}_f^l + \mathbf{W}_{hr}^l\mathbf{h}_{f-1}^l + \mathbf{b}_r^l) \tag{12}$$

D. Bidirectional Recurrent Neural Networks

Conventional RNNs are limited to previous context, i.e. information in the past of a specific time frame. To make use of future context as well, their extension to bidirectional RNNs [42] can be used (see Figure 5). Bidirectional RNNs process data in both directions with two separate hidden layers, which are then fed into the same output layer.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TBME.2018.2843258, IEEE Transactions on Biomedical Engineering

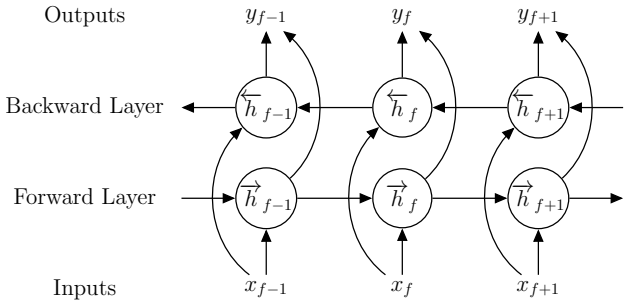


Fig. 5. Bidirectional recurrent neural network [50].

Equations (13-15) specify the network mathematically. The forward hidden sequence $\overrightarrow{\mathbf{h}}_f^l$ is computed by iterating the forward layer from $f = 1$ to F. The backward hidden sequence $\overleftarrow{\mathbf{h}}_f^l$ is computed by iterating the backward layer from $f = F$ to 1. $\mathbf{W}^l_{x\overrightarrow{h}}$, $\mathbf{W}^l_{x\overleftarrow{h}}$ are the input weight matrices, $\mathbf{W}^l_{\overrightarrow{h}\overrightarrow{h}}$, $\mathbf{W}^l_{\overleftarrow{h}\overleftarrow{h}}$ the hidden weight matrices, and $\mathbf{b}^l_{\overrightarrow{h}}$, $\mathbf{b}^l_{\overleftarrow{h}}$ the bias terms for the forward and backward hidden layer, respectively. Multiple bidirectional RNN layers can be stacked, forming a deep bidirectional RNN. Every hidden layer receives input from the previous forward and backward layers, i.e. $\overrightarrow{\mathbf{h}}_f^{l-1}$ and $\overleftarrow{\mathbf{h}}_f^{l-1}$. According to Equation (15), the output layer $\mathbf{y}_f$ is updated using the hidden activations $\overrightarrow{\mathbf{h}}_f^{L-1}$ and $\overleftarrow{\mathbf{h}}_f^{L-1}$ of the last hidden layer $L-1$. $\mathbf{W}_{\overrightarrow{h}y}$, $\mathbf{W}_{\overleftarrow{h}y}$ are the output weight matrices and $\mathbf{b}_y$ the output bias term.

BiRNNs can be combined with LSTMs or GRNNs, resulting in bidirectional long short term memory networks (BiLSTMs) or bidirectional gated recurrent neural networks (BiGRNNs) [50].

$$\overrightarrow{\mathbf{h}}_f^l = g(\mathbf{W}^l_{x\overrightarrow{h}}\mathbf{x}_f^l + \mathbf{W}^l_{\overrightarrow{h}\overrightarrow{h}}\overrightarrow{\mathbf{h}}_{f-1}^l + \mathbf{b}^l_{\overrightarrow{h}}) \tag{13}$$

$$\overleftarrow{\mathbf{h}}_f^l = g(\mathbf{W}^l_{x\overleftarrow{h}}\mathbf{x}_f^l + \mathbf{W}^l_{\overleftarrow{h}\overleftarrow{h}}\overleftarrow{\mathbf{h}}_{f-1}^l + \mathbf{b}^l_{\overleftarrow{h}}) \tag{14}$$

$$\mathbf{y}_f = m(\mathbf{W}_{\overrightarrow{h}y}\overrightarrow{\mathbf{h}}_f^{L-1} + \mathbf{W}_{\overleftarrow{h}y}\overleftarrow{\mathbf{h}}_f^{L-1} + \mathbf{b}_y) \tag{15}$$

III. Regularizers for RNNs

Deep neural networks usually require many training samples. The training set of the PhysioNet/CinC Challenge database [1] is limited to only 3153 heart sound recordings. We consider three different approaches for regularization to improve the ability of the model to generalize on test data, i.e. virtual adversarial training, dropout and data augmentation.

A. Virtual Adversarial Training

Virtual adversarial training (VAT) [51], [52] is a regularization method, which makes the model robust against adversarial perturbations [53], [54]. It promotes local smoothness of the posterior distribution $p(\mathbf{y}_f|\mathbf{x}_f)$ with respect to $\mathbf{x}_f$. The posterior distribution, or more precisely the softmax activation of the network output $\mathbf{h}_f^l$, should vary minimally for small, bounded perturbations of the input $\mathbf{x}_f$. The adversarial perturbation $\boldsymbol{\delta}_f$ is determined on frame-level by maximizing the Kullback-Leibler divergence KL-divergence $(\cdot||\cdot)$ of the posterior distribution for unperturbed and perturbed inputs, i.e.

$$\boldsymbol{\delta}_f = \underset{||\boldsymbol{\delta}||<\epsilon}{\arg\max}\, \mathrm{KL}(p(\mathbf{y}|\mathbf{x}_f)||p(\mathbf{y}|\mathbf{x}_f + \boldsymbol{\delta}), \tag{16}$$

where $\epsilon > 0$ limits the maximum perturbation, i.e. the noisy input $\mathbf{x}_f + \boldsymbol{\delta}$ lies within a radius ϵ around $\mathbf{x}_f$. The smaller the $\mathrm{KL}(p(\mathbf{y}|\mathbf{x}_f)||p(\mathbf{y}|\mathbf{x}_f + \boldsymbol{\delta}_f)$ the smoother is the posterior distribution around $\mathbf{x}_f$. Instead of maximizing the conditional likelihood $p(\mathbf{y}_f|\mathbf{x}_f)$ of the model during training, we maximize the regularized objective

$$\sum_f \log p(\mathbf{y}_f|\mathbf{x}_f) - \lambda \sum_f \mathrm{KL}(p(\mathbf{y}|\mathbf{x}_f)||p(\mathbf{y}|\mathbf{x}_f + \boldsymbol{\delta}_f)), \tag{17}$$

where the tradeoff parameter λ and the radius ϵ have to be selected on a validation set.

For further details regarding the implementation, we refer to [51]. In our experiments, we tune the number of iterations I_p, the radius ϵ and the tradeoff parameter λ.

B. Dropout

The idea of dropout is to randomly drop units from the neural network during training [55]. In this work, we consider input dropout applied on the hidden layers. Due to simplicity, we show dropout just for the *vanilla* RNN. Equations (18-20) describe the feed-forward operation of the network with dropout.

$$\mathbf{r}^l \sim Bernoulli(\mathbf{p}) \tag{18}$$

$$\tilde{\mathbf{x}}_f^l = \mathbf{r}^l \odot \mathbf{x}_f^l \tag{19}$$

$$\mathbf{h}_f^l = g(\mathbf{W}^l_x\tilde{\mathbf{x}}_f^l + \mathbf{W}^l_h\mathbf{h}_{f-1}^l + \mathbf{b}^l) \tag{20}$$

For any hidden layer $l \in \{1, ..., L-1\}$, $\mathbf{r}^l$ is a vector of independent Bernoulli random variables, each having a probability p of being 1, with $\mathbf{p} = [p, p, ..., p]^T$. The vector $\mathbf{r}^l$ is multiplied element-wise with the inputs of the layer $\mathbf{x}_f^l$, to create the thinned inputs $\tilde{\mathbf{x}}_f^l$. The thinned inputs are then used as inputs to the current layer. For training, the derivatives of the loss function are backpropagated through the sub-network. For testing, the network is used without dropout and the weights are scaled as $\mathbf{W}^l_{x,test} = p\mathbf{W}^l_x$.

C. Data Augmentation

We consider two approaches for data augmentation, i.e. *noise injection* and generating of additional training data with various *audio transformations*.

1) Noise Injection: Noise injection to the inputs of a neural network can be considered as a form of data augmentation [56]. The model should be capable to detect the heart sound sequence, although random noise is added to the inputs and also applied to the hidden units. The authors in [57] showed that noise injection can be very effective if the noise magnitude is carefully tuned. Dropout (see Section III-B) can be considered as a process of constructing new inputs by using a particular type of noise [56]. We add zero mean Gaussian noise to the inputs $\mathbf{x}_f$ and the hidden units during training. Standard deviation and noise level are tuned.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TBME.2018.2843258, IEEE Transactions on Biomedical Engineering

2) Audio Transformations: The best way to prevent overfitting is to train on more data. Therefore, we augment the training data by using various audio transformations from SoX [58], similar as in [59]. We consider the following two transformations to slightly modify the heart sound recordings:

- *Pitch*: Change the audio pitch without changing tempo.
- *Tempo*: Change the audio playback speed but not its pitch.

We provide an overview of the augmented training set in Table IV in Section IV-B2.

IV. Heart Sound Segmentation - Experiments

A. Audio Processing Framework

Figure 6 shows the basic steps of our heart sound segmentation framework. Given the raw audio data $\mathbf{x}_t = [x_1, \ldots, x_T]$, we extract a sequence of feature frames $\mathbf{x}_f \in \mathbb{R}^D$. D indicates the dimension of the feature vector and $f \in \{1, ..., F\}$ is the frame index, with F inidicating the number of frames.

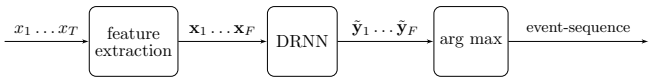


Fig. 6. Audio processing framework for heart sound segmentation with DRNNs.

We process the feature frames with a multi-label DRNN with a softmax output layer. The index of the maximum value (*arg max*) of the real-valued output vector $\tilde{\mathbf{y}}_f$ determines the event class per frame. This results in a sequence of frame labels as output. We group consecutive identical frame labels as one event.

B. Material

1) Heart Sound Database (Training Sets of the PhysioNet/CinC Challenge 2016): For the experiments within this section, we use heart sounds from the 2016 PhysioNet/CinC Challenge [1]. The dataset is a collection of several heart sound databases from different research groups, obtained in different real-world clinical and nonclinical environments. It contains recordings from normal subjects and pathological patients, which are grouped as follows: Normal control group (Normal), murmurs related to mitral valve prolapse (MVP), innocent or benign murmurs (Benign), aortic disease (AD), miscellaneous pathological conditions (MPC), coronary artery disease (CAD), mitral regurgitation (MR), aortic stenosis (AS) and pathological (Pathologic). The heart sounds were recorded at the four common recording locations: aortic area, pulmonic area, tricuspid area and mitral area. Due to the fact that the database is a collection of several small databases from different research groups, the recordings vary regarding several aspects: recording hardware, recording locations, data quality and patient types and methods for identifying gold standard diagnoses. For further details, we refer to [1].

The training set includes data from six databases, with a total of 3153 heart sound recordings from 764 subjects/patients (see Table I). The recordings are sampled with $f_s = 2\,\text{kHz}$ and vary in length between $5\,\text{s}$ and just over $120\,\text{s}$. The dataset is unbalanced, i.e. the number of normal recordings differ from that of abnormal recordings. Besides a binary diagnosis (-1=normal, 1=abnormal) for each heart sound recording, the challenge dataset further provides annotations for the heart sound states (*S1*, *systole*, *S2*, *diastole*). The annotations were generated with the LR-HSMM-based segmentation algorithm [28] (trained on PN-training-a) and further manually corrected. The annotations solely generated with the *segmentation algorithm* and those generated with the *segmentation algorithm and subsequent hand correction*, are accessible separately. In total 84426 beats were annotated in the PN-training set (after hand correction).

TABLE I
Summary of the dataset (Training data from the 2016 PhysioNet/CinC Challenge) [1].

Challenge set	# Patients	# Recordings	# Beats
PN-training-a	121	409	14559
PN-training-b	106	490	3353
PN-training-c	31	31	1808
PN-training-d	38	55	853
PN-training-e	356	2054	59593
PN-training-f	112	114	4260
Total	764	3153	84426

TABLE II
Summary of the dataset (Training data from the 2016 PhysioNet/CinC Challenge) [1] after excluding areas labeled as *noisy* (labels: *'(N', 'N)'*) and files marked as *unsure*.

Challenge set	# Recordings	# Beats
PN-training-a	392	14559
PN-training-b	368	3353
PN-training-c	27	1808
PN-training-d	52	853
PN-training-e	1926	59567
PN-training-f	109	4260
Total	2874	84400

Because the reference annotations for the four heart sound states were not available for heart sound recordings marked with *unsure* (=low signal quality), we excluded these recordings. We further excluded areas labeled as *noisy* (labels: *'(N', 'N)'*) by setting the respective areas of the signal to zero (*no signal*). Table II shows the resulting number of recordings and beats.

2) Training, Validation and Test Data: Due to the fact that the original test set from the PhysioNet/CinC Challenge 2016 is not publicly available so far, we generated a new test-, validation- and training-set out of the original PhysioNet (PN)-training set (see Section IV-B1). In the test set, we put exclusively PN-training-a and some recordings from PN-training-b and PN-training-e. For the recordings from PN-training-b and PN-training-e, we ensured their exclusivity in terms of subject affiliation, i.e. each subject is either only in the training set or the test set. We selected all recordings from the same subject with increasing 'Subject ID' (for PN-training-b) and increasing 'Raw record' name (for PN-training-e). This additional information is provided by the online appendix of the database. The resulting test set contains 764 recordings with 21116 beats

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TBME.2018.2843258, IEEE Transactions on Biomedical Engineering

TABLE III
SUMMARY OF THE TEST, VALIDATION AND TRAINING SET. THE ASSIGNED NUMBER OF RECORDINGS (#R.) AND BEATS (#B.) ARE REPORTED. THE RECORDINGS ARE GROUPED AS FOLLOWS: NORMAL CONTROL GROUP (NORMAL), MURMURS RELATED TO MITRAL VALVE PROLAPSE (MVP), INNOCENT OR BENIGN MURMURS (BENIGN), AORTIC DISEASE (AD), MISCELLANEOUS PATHOLOGICAL CONDITIONS (MPC), CORONARY ARTERY DISEASE (CAD), MITRAL REGURGITATION (MR), AORTIC STENOSIS (AS) AND PATHOLOGICAL (PATHOLOGIC).

Dataset	Challenge set	Normal		MVP		Benign		AD		MPC		CAD		MR		AS		Pathologic		**Total**	
		#R.	#B.	#R.	#B.	#R.	#B.	#R.	#B.	#R.	#B.	#R.	#B.	#R.	#B.	#R.	#B.	#R.	#B.	#R.	#B.
Test	PN-training-a	116	4419	126	4583	114	4200	13	425	23	932	-	-	-	-	-	-	-	-	392	14559
	PN-training-b	135	1278	-	-	-	-	-	-	-	-	29	238	-	-	-	-	-	-	164	1516
	PN-training-e	205	5018	-	-	-	-	-	-	-	-	3	23	-	-	-	-	-	-	208	5041
	Total	456	10715	126	4583	114	4200	13	425	23	932	32	261	-	-	-	-	-	-	**764**	**21116**
Validation	PN-training-b	12	100	-	-	-	-	-	-	-	-	5	49	-	-	-	-	-	-	17	149
	PN-training-c	1	23	-	-	-	-	-	-	-	-	-	-	3	125	1	18	-	-	5	166
	PN-training-d	3	32	-	-	-	-	-	-	-	-	-	-	-	-	-	-	2	26	5	58
	PN-training-e	156	4987	-	-	-	-	-	-	-	-	15	344	-	-	-	-	-	-	171	5331
	PN-training-f	7	269	-	-	-	-	-	-	-	-	-	-	-	-	-	-	5	162	12	431
	Total	179	5411	-	-	-	-	-	-	-	-	20	393	3	125	1	18	7	188	**210**	**6135**
Training	PN-training-b	148	1313	-	-	-	-	-	-	-	-	39	375	-	-	-	-	-	-	187	1688
	PN-training-c	6	340	-	-	-	-	-	-	-	-	-	-	9	772	7	530	-	-	22	1642
	PN-training-d	23	302	-	-	-	-	-	-	-	-	-	-	-	-	-	-	24	493	47	795
	PN-training-e	1419	46564	-	-	-	-	-	-	-	-	128	2631	-	-	-	-	-	-	1547	49195
	PN-training-f	71	2820	-	-	-	-	-	-	-	-	-	-	-	-	-	-	26	1009	97	3829
	Total	1667	51339	-	-	-	-	-	-	-	-	167	3006	9	772	7	530	50	1502	**1900**	**57149**

in total. From the residual recordings, we randomly selected 210 recordings (6135 beats) for the validation set and 1900 recordings (57149 beats) for the training set. Details about the splitting are shown in Table III.

In Section III-C, we introduce two transformations for data augmentation, pitch shifting and temporal stretching/compressing. We modify the recordings from the training set with a pitch shift of $\pm$ *a semitone*, i.e. a fundamental frequency of 50 Hz varies with approximately $\pm$3 Hz. We modify the time-scale of the recordings with $\pm$10 %. In total, we get an augmented dataset consisting of 9500 recordings and 285745 beats, as shown in Table IV.

TABLE IV
AUGMENTED TRAINING SET.

Effect	Parameters	# Recordings	# Beats
Clean		1900	57149
Pitch	+semitone	1900	57149
Pitch	-semitone	1900	57149
Tempo	+10%	1900	57149
Tempo	-10%	1900	57149
Total		9500	285745

3) Labeling: Based on the hand corrected annotations, we generated the labeling for the state sequence *first heart sound (S1) - systole - second heart sound (S2) - diastole*. Due to the shift of 20 ms in our frame-wise processing framework (see Section IV-C), we generated a label for each frame from the annotation information. In addition to the state labeling, we further added the label *no signal*, for areas with absent signal due to zero-padding. Figure 7 shows an example of a phonocardiogram (PCG) with the five labels.

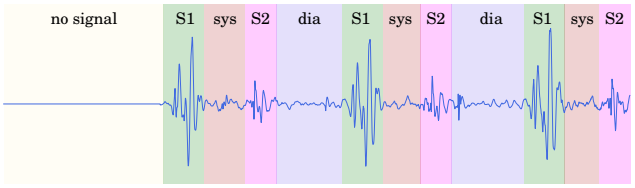


Fig. 7. Example of a phonocardiogram (PCG) showing the five possible labels: *no signal*, *first heart sound (S1)*, *systole (sys)*, *second heart sound (S2)*, and *diastole (dia)*.

C. Feature Extraction

We resampled the heart sound recordings to a sampling frequency of $f_s = 1$ kHz and removed DC offset with a high-pass filter with a cutoff frequency of $f_c = 10$ Hz (relevant for PN-training e). We zero-padded the recordings according to the longest one in each set.

For the spectral features, we preprocessed all recordings with a STFT using a Hamming window with window-size 80 ms ($\widehat{=}$ 80 samples) and 75 % overlap ($\widehat{=}$ frame-shifts of 20 ms or 20 samples). To exploit the spectral information of the heart sounds, we consider the following two types of spectral features:

- *Spectrogram*: We extract 41-bin log magnitude spectrograms.
- *Mel Frequency Cepstral Coefficients (MFCCs)*: MFCCs are used as features in various acoustic pattern recognition tasks, including heart sound classification [1] and heart sound segmentation [21].
  We extract 20 static coefficients, 20 delta coefficients (Δ) and 20 acceleration coefficients (Δ^2), resulting in a 60-bin vector per frame. We use 20 mel bands within a frequency range of 0-500 Hz. With a width of 9 frames, we calculate the delta and acceleration coefficients.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TBME.2018.2843258, IEEE Transactions on Biomedical Engineering

Furthermore, similar as for the LR-HSMM-method [28], we extract feature vectors for 20 ms-frames with the following features:

- *Envelope features* [28]: Homomorphic envelope, Hilbert envelope, wavelet envelope and power spectral density.

All features were normalized to zero-mean unit variance using the training corpus.

D. Evaluation Metrics

We evaluate the results event-based. We define an event as correctly detected if its temporal position overlaps with the one of an identically labeled event in the hand annotated ground truth. For temporal onset and offset, we allow a tolerance of ± 40 ms ($\widehat{=}$ $\pm$two frame-shifts of 20 ms), respectively. We determine for all heart sound recordings:

- True positives (*TP*): Events, where system output and ground truth have a temporal overlap;
- False positives (*FP*): The ground truth indicates no event that the system outputs;
- False negatives (*FN*): The ground truth indicates an event that is not detected by the system;
- Substitutions (*S*): Events in the automatic segmentation with correct temporal position, but incorrect class label;
- Insertions (*I*): False positives minus the number of substitutions;
- Deletions (*D*): False negatives minus the number of substitutions;
- Reference states (*N*): Number of events in the ground truth.

We evaluate the performance of the segmentation algorithm using Precision (Eqn. 21), Sensitivity (Equation 22), F-score (Equation 23) and Error Rate (Equation 24). The Error Rate should be small, while the F-score should be large. For a more detailed description of the metrics, we refer to [60].

$$P_+ = \frac{TP}{TP + FP} \tag{21}$$

$$Se = \frac{TP}{TP + FN} \tag{22}$$

$$F_1 = 2 \cdot \frac{P_+ \cdot Se}{P_+ + Se} \tag{23}$$

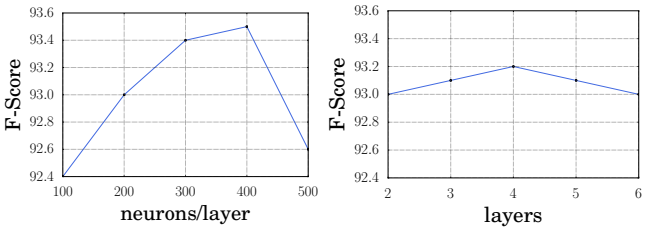
$$ER = \frac{S + I + D}{N} \tag{24}$$

E. Experiments and Results

For the experiments[1], we built a single multi-label classification system. We initialize the models with orthogonal weights [61] and use a softmax output gate as output layer. For optimizing the *cross-entropy error (CEE)* objective, we use ADAM [62]. We perform early stopping, where we train each model for 200 epochs and use the parameter setting that causes the smallest validation error for the evaluation of the model. The reported scores are the average values over the events *S1*, *systole*, *S2* and *diastole*. In addition to the average values, we report the scores for each event independently on the test set for the best setup. The reported scores are results of the validation set, except for the evaluation of the final setup on the test set in Section IV-E6.

[1]We conducted experiments using Python with Theano, and CUDA for GPU computing.

1) Comparison of GRNN network size: We initiate our experiments with finding an appropriate network size by using GRNNs and MFCC features. We use rectifier activations for the gated recurrent units. Figure 8 shows the results with varying number of neurons per hidden layer and varying number of hidden layers per model. For a 2-hidden layer GRNN, we achieved the best score of $F_1 = 93.5\,\%$ with 400 neurons/layer. For a GRNN with hidden layers of 200 neurons, we achieved the best score of $F_1 = 93.2\%$ with 4 hidden layers. Due to the small difference regarding the F-score, we choose a network size in favor of faster training. For the subsequent experiments, we fix the model size to 2 hidden layers, and 200 neurons per layer.



(a) Comparing network size (b) Comparing network depth

Fig. 8. Comparison of network size: (a) shows the F-scores for a GRNN using $\{2, ..., 6\}$ hidden layer of 200 neurons. (b) shows the F-score for a GRNN with two hidden layer using $\{100, 200, ..., 500\}$ neurons per layer.

2) Comparison of GRNN activation functions: Table V shows the results for different activation functions. In particular, we use *sigmoid, tanh* and *rectifier* non-linearities. Again, we use (2 hidden layer, 200 neurons per layer) GRNNs and MFCC features. Rectifier functions achieve the best average score, i.e. $F_1 = 93.0\,\%$. This is consistent with the literature [63].

TABLE V
COMPARING DIFFERENT ACTIVATION FUNCTIONS USING GRNNS.

Model	Features	Activation	$P_+(\%)$	$Se(\%)$	$F_1(\%)$	ER
GRNN	MFCCs	sigmoid	91.4	92.1	91.7	0.17
GRNN	MFCCs	tanh	92.4	93.3	92.8	0.15
GRNN	MFCCs	rectifier	92.2	93.8	**93.0**	0.14

3) Comparison of RNN architectures: Table VII shows the results for different RNN architectures. We compare different models, i.e. RNNs, LSTMs, GRNNs, and their bidirectional versions, using MFCC features. The model size for the conventional models is 2 hidden layers and 200 neurons per layer. For the bidirectional models, we use 2 hidden layers and 100 neurons for the forward and backward layers, respectively. The BiLSTM slightly outperforms the other models, by achieving an average F-Score of $F_1 = 94.1\,\%$. Due to the small difference between the BiLSTM and the BiGRNN, we choose the less complex BiGRNN for the subsequent experiments.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TBME.2018.2843258, IEEE Transactions on Biomedical Engineering

TABLE VII
COMPARISON OF DIFFERENT RECURRENT NEURAL NETWORKS ARCHITECTURES USING MFCC FEATURES.

Model	Features	$P_+(\%)$	$Se(\%)$	$F_1(\%)$	ER
RNN	MFCCs	90.2	93.1	91.6	0.17
LSTM	MFCCs	91.8	93.3	92.5	0.15
GRNN	MFCCs	92.2	93.8	93.0	0.14
BiRNN	MFCCs	91.8	94.5	93.1	0.14
BiLSTM	MFCCs	93.5	94.8	94.1	0.12
BiGRNN	MFCCs	92.8	94.5	**93.7**	0.13

4) Comparison of BiGRNN input features: Table VIII shows the results for BiGRNNs with MFCCs, spectrograms, envelope features, and their combinations. Best results are obtained with spectrograms, envelope features, and their combination. The envelope features already show promising results in combination with the LR-HSMM-method. For this reason, and with the assumption that spectrograms render the segmentation more robust against artefacts, we use the combination of spectrogram and envelope features for the subsequent experiments.

TABLE VIII
COMPARISON OF MFCCS, SPECTROGRAM, AND ENVELOPE FEATURES.

Model	Features	$P_+(\%)$	$Se(\%)$	$F_1(\%)$	ER
BiGRNN	MFCCs	92.8	94.5	93.7	0.13
BiGRNN	Spectrogram	95.0	95.7	95.4	0.09
BiGRNN	Envelope	95.0	95.9	95.4	0.10
BiGRNN	MFCCs + Envelope	93.7	94.6	94.2	0.12
BiGRNN	Spectrogram + Envelope	94.9	95.8	**95.4**	0.09

5) Comparison of different regularizers: Table IX shows the results using a BiGRNN with different regularization approaches. For dropout, we dropped units in the hidden layers during training with a probability of $p = 0.1$. This value achieved the best results among $p \in \{0.1, 0.5, 0.7, 0.9\}$. For VAT, we used the parameter setting of $\lambda = 0.1$, $\epsilon = 0.1$, and $I_p = 1$. For noise injection, we added zero mean Gaussian noise with standard deviation $\sigma = 0.025$ and magnitude $m = 0.25$. For data augmentation with audio transformations, we used the augmented training set for training (see Table IV). All regularization methods, except for data augmentation with audio transformations, improved the F-score. In particular, with dropout, we achieve the best result of $F_1 = 96.1\,\%$.

TABLE IX
COMPARISON OF DIFFERENT REGULARIZATION METHODS.

Model	Regularizer	Parameters	$P_+(\%)$	$Se(\%)$	$F_1(\%)$	ER
BiGRNN	-	-	94.9	95.8	95.4	0.09
BiGRNN	VAT	λ = 0.1, ϵ = 0.1, I_p = 1	95.3	96.1	95.7	0.09
BiGRNN	Dropout	p = 0.1	95.8	96.3	**96.1**	0.08
BiGRNN	Noise Injection	σ = 0.025, m = 0.25	95.4	95.8	95.7	0.09
BiGRNN	Audio Transformations	-	95.0	95.7	95.4	0.09

6) Evaluation of the final setup on the test set: Table X shows the results for the best setup (i.e. BiGRNN, 2 hidden layers, 200 neurons per layer, rectifier activations, spectrogram+envelope features, dropout regularization) evaluated on the test set. In addition to the metrics from the previous sections, we report in detail the numbers of reference states N_{ref} (ground truth), system states N_{sys} (BiGRNN-method), true positives N_{TP}, false negatives N_{FN} and false positives N_{FP} for each event, respectively.

TABLE X
DETAILED RESULTS PER EVENT EVALUATED WITH THE FINAL SETUP ON THE TEST SET.

Event	N_{ref}	N_{sys}	N_{TP}	N_{FN}	N_{FP}	$P_+(\%)$	$Se(\%)$	$F_1(\%)$	ER
S1	21115	21271	20659	456	612	97.1	97.8	97.5	0.05
Systole	21200	21453	20267	933	1186	94.5	95.6	95.0	0.10
S2	21073	21229	20102	971	1127	94.7	95.4	95.0	0.10
Diastole	21385	21758	20283	1102	1475	93.2	94.8	94.0	0.12
Average						94.9	95.9	95.4	0.09

TABLE VI
COMPARISON OF OUR BIGRNN WITH THE LR-HSMM-METHOD [28], EVALUATED ON 744 RECORDINGS FROM THE TEST SET.

Challange set	Disease	#Recordings	#Beats	BiGRNN				LR-HSMM			
				$P_+(\%)$	$Se(\%)$	$F_1(\%)$	ER	$P_+(\%)$	$Se(\%)$	$F_1(\%)$	ER
PN-training-a	Normal	112	4270	95.6	96.7	96.1	0.08	96.4	96.0	96.2	0.08
	MVP	119	4402	93.1	93.8	93.4	0.13	91.2	91.6	91.4	0.17
	Benign	113	4163	96.4	97.2	96.8	0.06	96.8	96.8	96.8	0.06
	AD	13	425	86.6	93.2	89.7	0.21	95.5	95.5	95.5	0.09
	MPC	23	932	89.6	92.7	91.1	0.18	91.3	91.8	91.6	0.17
	All	380	14192	94.4	95.6	**95.0**	0.10	94.5	94.6	**94.6**	0.11
PN-training-b	Normal	132	1256	92.8	94.4	93.6	0.13	96.8	96.1	96.4	0.07
	CAD	29	238	79.9	82.8	81.3	0.38	94.6	93.5	94.0	0.12
	All	161	1494	90.6	92.5	**91.6**	0.17	96.5	95.7	**96.1**	0.08
PN-training-e	Normal	201	4981	98.6	98.6	98.6	0.03	98.5	98.3	98.4	0.03
	CAD	2	19	68.1	69.4	68.7	0.64	85.9	85.9	85.9	0.28
	All	203	5000	98.5	98.5	**98.5**	0.03	98.4	98.2	**98.3**	0.03
All		744	20686	95.1	96.1	**95.6**	0.09	95.6	95.5	**95.6**	0.09

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TBME.2018.2843258, IEEE Transactions on Biomedical Engineering

7) Comparison with the LR-HSMM-method: For this experiment, we remove recordings from the training and test set containing areas with no signal, because the LR-HSMM is limited to the detection of four events in the order of *S1 - systole - S2 - diastole*. This results in 1810 recordings for the training set and 744 recordings for the test set.

For the LR-HSMM, we preprocess the recordings with resampling to $f_s = 1\,\text{kHz}$ and high-pass filtering with a cutoff frequency of $f_c = 10\,\text{Hz}$ (cf. Section IV-C). We process the audio signals with frames of 20 ms. We train the LR-HSMM by using the four feature types provided: homomorphic envelogram, Hilbert envelope, wavelet envelope, and power spectral density (PSD) [28].

Table VI shows the results achieved with the BiGRNN (final setup) compared with the LR-HSMM method.

Figure 9 shows nine examples of automatically segmented heart sound recordings (snippets of four seconds each). In each subfigure, we show the hand annotated ground truth (GT), the segmentation with the LR-HSMM method and the segmentation with the BiGRNN. We show five recordings from *PN-Training-a* (Figure 9a to 9e), two recording from *PN-Training-b* (Figure 9f and 9g) and two recordings from *PN-Training-e* (Figure 9h and 9i). For the visualization, we normalized each heart sound recording according to its maximum amplitude.

V. Discussion

In our experiments, we compare *vanilla* RNNs, LSTMs, GRNNs, and their bidirectional implementations, with BiGRNNs outperforming the rest. In subsequent experiments, we find the final setup using spectrogram and envelope features with a regularized BiGRNN. The network consists of 2 hidden layers with 200 neurons each and rectifier activations (except for the last layer). Regularization with *dropout* achieves the best result. Data augmentation with *audio transformations* does not result in any improvement.

In Section IV-E7, we compare our proposed method with the state-of-the-art, the LR-HSMM-method. The BiGRNN-method performs on par with the LR-HSMM-method with an overall F-score of $F_1 = 95.6\,\%$ (cf. Table VI). We have to remark that this is not a completely fair comparison, because the ground truth annotations, although trained on less data (i.e. PN-training-a) and manually corrected, were generated with the LR-HSMM-method. This may introduce bias towards the LR-HSMM-method. Furthermore, the hand annotated ground truth is not always correct (cf. Figure 9a and 9h), also being in favor of the LR-HSMM-method and in general causing distortion in the scores.

Table VI shows detailed results for the test data in terms of PN-training sets and diseases. We observe that the BiGRNN-method outperforms the LR-HSMM-method for PN-training-a and PN-training-e, but performs worse for PN-training-b. Regarding the diseases in PN-training-a, only for MVP the

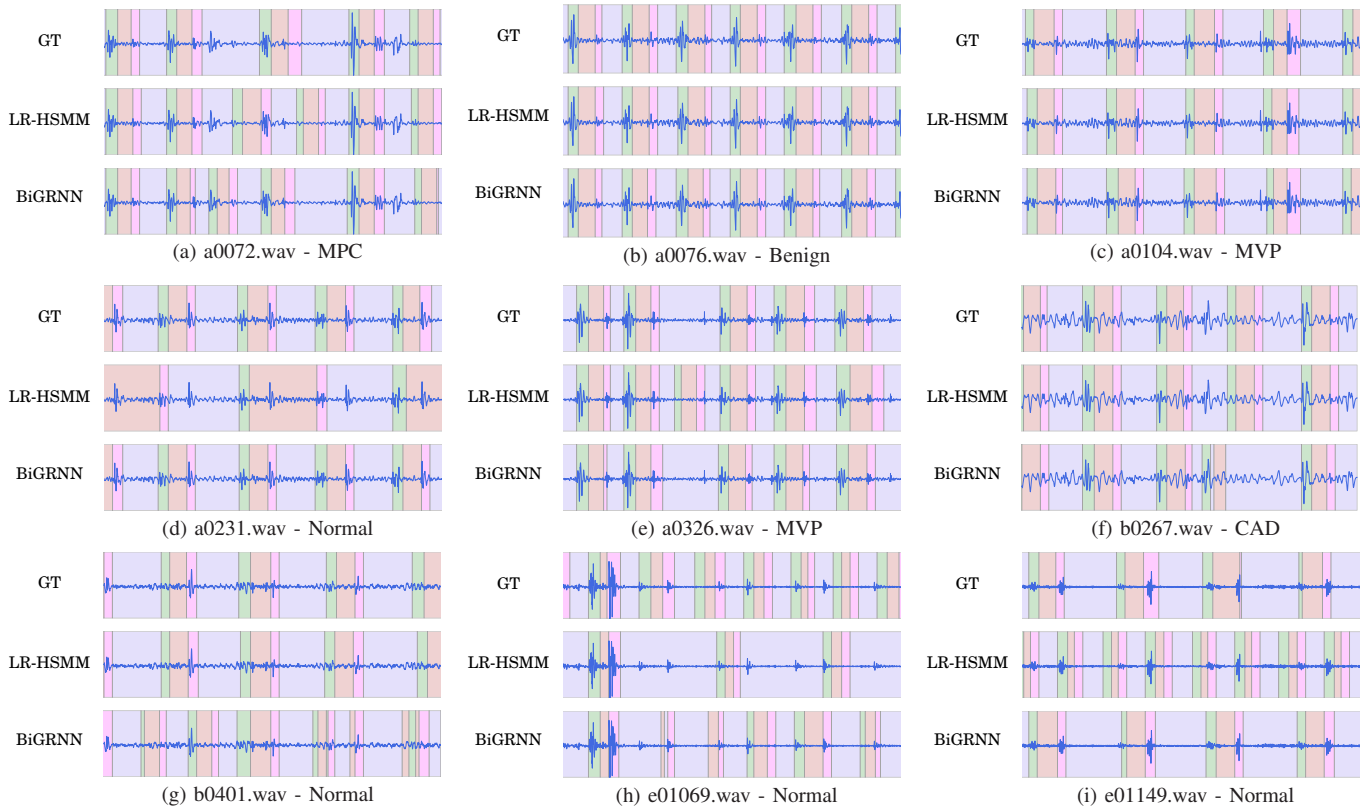


Fig. 9. Legend: *S1*; *systole*; *S2*; *diastole*.
Examples of automatically segmented heart sound recordings (snippets of four seconds each) from the test set. In each subfigure, the first plot corresponds to the hand annotated ground truth (GT), the second to the logistic regression hidden semi-Markov model based (LR-HSMM) method and the third to the proposed method from this paper (BiGRNN).

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TBME.2018.2843258, IEEE Transactions on Biomedical Engineering

BiGRNN-method is outperforming the LR-HSMM-method, and for benign murmurs (Benign) both methods perform on par. For PN-training-b, the LR-HSMM-method outperforms the BiGRNN-method for normal and CAD recordings. For normal recordings of PN-training-e, the BiGRNN-method outperforms the LR-HSMM-method. The LR-HSMM-method is distinctly better than the BiGRNN-method for the two recordings of CAD in PN-training-e.

The 2016 PhysioNet/CinC Challenge data does not provide any labeling for cardiac arrhythmia. According to [64], mitral valve prolapse is a source of arrhythmias. We refer to the results reported for MVP, with the BiGRNN-method ($F_1 = 93.4\,\%$) outperforming the LR-HSMM-method ($F_1 = 91.4\,\%$). Moreover, we visually inspected all test recordings of MVP and found 20 recordings with cardiac arrhythmia. On this selected set of recordings the BiGRNN-method ($F_1 = 87.2\,\%$) outperforms the LR-HSMM-method ($F_1 = 75.7\,\%$). An example for cardiac arrhythmia for MVP is shown in Figure 9e.

The examples in Figure 9 illustrate some observations for both segmentation methods, and also for the ground truth annotations. Figure 9b and 9c show examples, where both methods perform well. In Figure 9d, we observe that the LR-HSMM-method skips every second *S2*, and detects every second *S1* as *S2*. In contrast to this, in Figure 9i the LR-HSMM-method detects too many events, i.e. *S2* is always detected as *S1* and in between an additional state-sequence *S1 - systole - S2 - diastole* is detected. Figure 9g shows some segmentation errors for the BiGRNN method. Figure 9a, 9h and 9f are examples, where the ground truth labeling is partially incorrect. In Figure 9h, we further observe that both methods achieve partially incorrect segmentation results. Figure 9e shows an example for the failure of the LR-HSMM method for irregularity of the temporal occurrence of the events.

In our experiments, the proposed BiGRNN-method achieves performance on par with the LR-HSMM-method. We successfully show state-of-the-art performance without directly incorporating a priori information of the state durations. The proposed method is easily extendable to the detection of extra heart sounds (third and fourth heart sound), heart murmurs, as well as other acoustic events. However, this would require appropriate training data, i.e. heart sound recordings containing the additional events and their proper labeling. In a practical sense, our method features further advantages. Without preprocessing, it can easily handle absence of the signal, noise and irregularity of the temporal occurrence of the events (like in cardiac arrhythmia).

VI. Conclusion

In this paper, we introduce an event detection approach with deep recurrent neural networks (DRNNs) for heart sound segmentation, i.e. the detection of the state-sequence *first heart sound (S1) - systole - second heart sound (S2) - diastole*. We carefully conduct experiments with heart sound recordings from the 2016 Physionet/CinC Challenge and compare the proposed method with the state-of-the-art by reporting event based metrics and visualizing examples of segmented heart sound recordings.

In particular, we trained a BiGRNN on heart sound recordings and appropriate labeling of the state-sequences. In our final setup, we use spectrogram and envelope features and dropout for regularization. We obtain an event-based F-score of $F_1 = 95.6\,\%$, evaluated on an independent test set. The state-of-the-art, the logistic regression hidden semi-Markov model based heart sound segmentation method, achieves the same score. This result is however biased, since it has been used to annotate the dataset. Furthermore, the ground truth for the heart sound segmentation is partially incorrect. Nevertheless, we show that the proposed method achieves state-of-the-art performance, although we do not explicitly incorporate a priori information about the state-durations. Furthermore, the proposed method shows advantages regarding practical aspects. In particular, it can handle absence of the signal, noise and cardiac arrhythmia, i.e. irregularity of the temporal occurrence of the events.

The proposed method represents a general solution for the detection of different kinds of events in heart sound recordings. The method is easily extendable to the detection of extra heart sounds (third and fourth heart sound), heart murmurs, as well as other acoustic events. This, however, requires appropriate training data with *thorough* labeling of the events and further experiments.

References

[1] C. Liu *et al.*, "An open access database for the evaluation of heart sound algorithms," *Physiological Measurement*, vol. 37, no. 12, p. 2181, 2016.

[2] H. Liang *et al.*, "Heart sound segmentation algorithm based on heart sound envelogram," in *Computers in Cardiology*. IEEE, 1997, pp. 105–108.

[3] A. Moukadem *et al.*, "A robust heart sounds segmentation module based on S-transform," *Biomedical Signal Processing and Control*, vol. 8, no. 3, pp. 273–281, 2013.

[4] S. Sun *et al.*, "Automatic moment segmentation and peak detection analysis of heart sound pattern via short-time modified Hilbert transform," *Computer Methods and Programs in Biomedicine*, vol. 114, no. 3, pp. 219–230, 2014.

[5] S. Choi and Z. Jiang, "Comparison of envelope extraction algorithms for cardiac sound signal segmentation," *Expert Systems with Applications*, vol. 34, no. 2, pp. 1056–1069, 2008.

[6] Z. Yan *et al.*, "The moment segmentation analysis of heart sound pattern," *Computer Methods and Programs in Biomedicine*, vol. 98, no. 2, pp. 140–150, 2010.

[7] S. Ari *et al.*, "A robust heart sound segmentation algorithm for commonly occurring heart valve diseases," *Journal of Medical Engineering & Technology*, vol. 32, no. 6, pp. 456–465, 2008.

[8] H. Naseri and M. Homaeinezhad, "Detection and boundary identification of phonocardiogram sounds using an expert frequency-energy based metric," *Annals of Biomedical Engineering*, vol. 41, no. 2, pp. 279–292, 2013.

[9] D. Kumar *et al.*, "Detection of S1 and S2 heart sounds by high frequency signatures," in *Proceedings of the 28th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC'06)*. IEEE, 2006, pp. 1410–1416.

[10] V. N. Varghees and K. Ramachandran, "A novel heart sound activity detection framework for automated heart sound analysis," *Biomedical Signal Processing and Control*, vol. 13, pp. 174–188, 2014.

[11] J. Pedrosa *et al.*, "Automatic heart sound segmentation and murmur detection in pediatric phonocardiograms," in *Proceedings of the 36th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC'14)*. IEEE, 2014, pp. 2294–2297.

[12] J. Vepa *et al.*, "Segmentation of heart sounds using simplicity features and timing information," in *Proceedings of the 33th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP'08)*. IEEE, 2008, pp. 469–472.

[13] C. D. Papadaniil and L. J. Hadjileontiadis, "Efficient heart sound segmentation and extraction using ensemble empirical mode decomposition and kurtosis features," *IEEE Journal of Biomedical and Health Informatics*, vol. 18, no. 4, pp. 1138–1152, 2014.

[14] A. Gharehbaghi *et al.*, "An automatic tool for pediatric heart sounds segmentation," in *Computing in Cardiology*. IEEE, 2011, pp. 37–40.

[15] T. Oskiper and R. Watrous, "Detection of the first heart sound using a time-delay neural network," in *Computers in Cardiology*. IEEE, 2002, pp. 537–540.

[16] A. A. Sepehri *et al.*, "A novel method for pediatric heart sound segmentation without using the ECG," *Computer Methods and Programs in Biomedicine*, vol. 99, no. 1, pp. 43–48, 2010.

[17] T. Chen *et al.*, "Intelligent heartsound diagnostics on a cellphone using a hands-free kit." in *AAAI Spring Symposium: Artificial Intelligence for Development*, 2010.

[18] C. N. Gupta *et al.*, "Neural network classification of homomorphic segmented heart sounds," *Applied Soft Computing*, vol. 7, no. 1, pp. 286–297, 2007.

[19] H. Tang *et al.*, "Segmentation of heart sounds based on dynamic clustering," *Biomedical Signal Processing and Control*, vol. 7, no. 5, pp. 509–516, 2012.

[20] S. Rajan *et al.*, "Unsupervised and uncued segmentation of the fundamental heart sounds in phonocardiograms using a time-scale representation," in *Proceedings of the 28th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC'06)*. IEEE, 2006, pp. 3732–3735.

[21] T.-E. Chen *et al.*, "S1 and S2 heart sound recognition using deep neural networks," *IEEE Transactions on Biomedical Engineering*, vol. 64, no. 2, pp. 372–380, 2017.

[22] L. Gamero and R. Watrous, "Detection of the first and second heart sound using probabilistic models," in *Proceedings of the 25th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC'03)*, vol. 3. IEEE, 2003, pp. 2877–2880.

[23] A. D. Ricke *et al.*, "Automatic segmentation of heart sound signals using hidden Markov models," in *Computers in Cardiology*. IEEE, 2005, pp. 953–956.

[24] D. Gill *et al.*, "Detection and identification of heart sounds using homomorphic envelogram and self-organizing probabilistic model," in *Computers in Cardiology*. IEEE, 2005, pp. 957–960.

[25] P. Sedighian *et al.*, "Pediatric heart sound segmentation using hidden Markov model," in *Proceedings of the 36th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC'14)*. IEEE, 2014, pp. 5490–5493.

[26] A. Castro *et al.*, "Heart sound segmentation of pediatric auscultations using wavelet analysis," in *Proceedings of the 35th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC'13)*. IEEE, 2013, pp. 3909–3912.

[27] S. Schmidt *et al.*, "Segmentation of heart sound recordings by a duration-dependent hidden Markov model," *Physiological Measurement*, vol. 31, no. 4, p. 513, 2010.

[28] D. B. Springer *et al.*, "Logistic regression-hsmm-based heart sound segmentation," *IEEE Transactions on Biomedical Engineering*, vol. 63, no. 4, pp. 822–832, 2016.

[29] C. Liu *et al.*, "Performance of an open-source heart sound segmentation algorithm on eight independent databases," *Physiological measurement*, vol. 38, no. 8, p. 1730, 2017.

[30] K. Cho *et al.*, "On the properties of neural machine translation: Encoder-decoder approaches," *CoRR*, vol. abs/1409.1259, 2014.

[31] J. Chung *et al.*, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *CoRR*, vol. abs/1412.3555, 2014.

[32] ——, "Gated feedback recurrent neural networks," *CoRR*, vol. abs/1502.02367, 2015.

[33] I. Sutskever *et al.*, "Sequence to sequence learning with neural networks," in *Advances in Neural Information Processing Systems 27*, 2014, pp. 3104–3112.

[34] A. Graves *et al.*, "Hybrid speech recognition with deep bidirectional LSTM," in *IEEE Workshop on Automatic Speech Recognition and Understanding*, 2013, pp. 273–278.

[35] T. c. I. Yang and H. Hsieh, "Classification of acoustic physiological signals based on deep learning neural networks with augmented features," in *Computing in Cardiology Conference (CinC)*, Sept 2016, pp. 569–572.

[36] C. Thomae and A. Dominik, "Using deep gated RNN with a convolutional front end for end-to-end classification of heart sound," in *Computing in Cardiology Conference (CinC)*, Sept 2016, pp. 625–628.

[37] J. van der Westhuizen and J. Lasenby, "Bayesian LSTMs in medicine," *arXiv preprint arXiv:1706.01242*, 2017.

[38] O. Gencoglu *et al.*, "Recognition of acoustic events using deep neural networks," in *Proceedings of the 22nd European Signal Processing Conference*, 2014, pp. 506–510.

[39] G. Chen *et al.*, "Query-by-example keyword spotting using long short-term memory networks," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP'15)*, 2015, pp. 5236–5240.

[40] G. Parascandolo *et al.*, "Recurrent neural networks for polyphonic sound event detection in real life recordings," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP'16)*, 2016, pp. 6440–6444.

[41] M. Zöhrer and F. Pernkopf, "Gated recurrent networks applied to acoustic scene classification and acoustic event detection," *IEEE AASP Challenge: Detection and Classification of Acoustic Scenes and Events*, 2016.

[42] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE Transactions on Signal Processing*, vol. 45, no. 11, pp. 2673–2681, 1997.

[43] E. Messner *et al.*, "Crackle and breathing phase detection in lung sounds with deep bidirectional gated recurrent neural networks," in *Proceedings of the 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC'18)*. IEEE, 2018.

[44] D. E. Rumelhart *et al.*, "Neurocomputing: Foundations of research," J. A. Anderson and E. Rosenfeld, Eds. Cambridge, MA, USA: MIT Press, 1988, ch. Learning Internal Representations by Error Propagation, pp. 673–695.

[45] J. L. Elman, "Finding structure in time," *Cognitive Science*, vol. 14, no. 2, pp. 179–211, 1990. [Online]. Available: http://groups.lis.illinois.edu/amag/langev/paper/elman90findingStructure.html

[46] M. I. Jordan, "Attractor dynamics and parallelism in a connectionist sequential machine," in *Proceedings of the Eighth Annual Conference of the Cognitive Science Society*. Hillsdale, NJ: Erlbaum, 1986, pp. 531–546.

[47] J. J. Hopfield and D. W. Tank, "Computing with neural circuits: A model," *Science*, vol. 233, pp. 624–633, 1986.

[48] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[49] A. Graves *et al.*, "Speech recognition with deep recurrent neural networks," in *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP'13)*, 2013, pp. 6645–6649.

[50] ——, "Hybrid speech recognition with deep bidirectional LSTM," in *IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*. IEEE, 2013, pp. 273–278.

[51] T. Miyato *et al.*, "Distributional smoothing by virtual adversarial examples." *CoRR*, vol. abs/1507.00677, 2015.

[52] M. Ratajczak *et al.*, "Virtual adversarial training applied to neural higher-order factors for phone classification." in *INTERSPEECH*, 2016, pp. 2756–2760.

[53] A. Makhzani *et al.*, "Adversarial autoencoders," *CoRR*, vol. abs/1511.05644, 2015.

[54] I. Goodfellow *et al.*, "Generative adversarial nets," in *Advances in Neural Information Processing Systems 27*, 2014, pp. 2672–2680.

[55] N. Srivastava *et al.*, "Dropout: A simple way to prevent neural networks from overfitting." *Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[56] I. Goodfellow *et al.*, *Deep Learning*. MIT Press, 2016, http://www.deeplearningbook.org.

[57] B. Poole *et al.*, "Analyzing noise in autoencoders and deep networks," *arXiv preprint arXiv:1406.1831*, 2014.

[58] "Sound exchange," http://sox.sourceforge.net, accessed: 2017-07-05.

[59] C. Thomae and A. Dominik, "Using deep gated RNN with a convolutional front end for end-to-end classification of heart sound," in *Computing in Cardiology Conference (CinC)*. IEEE, 2016, pp. 625–628.

[60] A. Mesaros *et al.*, "Metrics for polyphonic sound event detection," *Applied Sciences*, vol. 6, no. 6, p. 162, 2016.

[61] A. M. Saxe *et al.*, "Exact solutions to the nonlinear dynamics of learning in deep linear neural networks," *International Conference of Learning Representations (ICLR)*, 2014.

[62] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *CoRR*, vol. abs/1412.6980, 2014.

[63] X. Glorot *et al.*, "Deep sparse rectifier neural networks," in *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2011, pp. 315–323.

[64] E. Van der Wall and M. Schalij, "Mitral valve prolapse: a source of arrhythmias?" *The international journal of cardiovascular imaging*, vol. 26, no. 2, pp. 147–149, 2010.