Heart Sound Classification via Sparse Coding

Bradley M Whitaker[1] and David V Anderson[1]

[1] Department of Electrical & Computer Engineering, Georgia Institute of Technology, Atlanta, GA, USA

Abstract

Introduction: The aim of the Physionet/CinC Challenge 2016 is to automatically classify heart sound recordings as normal or abnormal. The Challenge provides 3,153 labeled audio recordings taken from a single precordial location, as well as Springer's state-of-the-art beat segmentation algorithm.

Algorithm: Using Springer's segmentation algorithm, we divide each audio segment into an array of sub-second audio files corresponding to the four phases of the cardiac cycle. We take an N-point FFT of each audio segment and create five different data matrices: one for each sub-cycle (S1, Systole, S2, and Diastole), and one for a complete cardiac cycle. A column of the data matrix corresponds to the N-point FFT of one audio segment. Using sparse coding, we decompose the data matrix into a dictionary matrix and a sparse coefficient matrix. The dictionary matrix represents statistically important spectral features of the audio segments. The sparse coefficient matrix is a mapping that represents which features are used by each segment. Working in the sparse domain, we train support vector machines (SVMs) for each sub-cycle and for the complete cycle. We train a sixth SVM to combine the results from the preliminary SVMs into a single binary label for the entire sound recording.

Results: Our algorithm achieves a cross-validation score of 0.8652 (Se=0.8669 and Sp=0.8634). The best unofficial score when tested on a subset of the unknown challenge data is 0.812 (Se=0.825 and Sp=0.799).

Conclusions: We developed an algorithm to classify heart sound recordings as normal or pathological. Our results show that sparse coding is an effective way to define spectral features of the cardiac cycle and its sub-cycles for the purpose of classification. Further work will attempt to increase the sensitivity and specificity of the algorithm by exploring other classifiers while still working in the sparse domain.

1. Introduction

This work describes the solution of our entry in the 2016 Physionet/CinC Challenge. The goal of the Challenge was to accurately classify normal and abnormal heart sound recordings. The Challenge details can be found at `https://physionet.org/challenge/2016/`. The database we used in conjunction with the challenge is described in detail in [1].

The novelty of our solution and the focus of this paper relates to using sparse coding as a tool for performing unsupervised feature extraction. Using sparse coding in image and audio classification tasks is an active research area [2–9]. Our own previous work has found success in sparse-domain classification tasks using Support Vector Machines (SVMs) [10].

2. Algorithm

2.1. Audio Preprocessing

Prior to extracting features and learning a classifier, we preprocessed the audio data by segmenting the heart sounds and converting the data into the frequency domain. Fig. 1 offers a visual representation of the preprocessing steps.

Segmenting a PCG into periods is fundamental in the automated analysis of heard sounds [11]. As the first step of our algorithm we utilized Springer's state-of-the-art segmentation code, which was provided by the Challenge [12], to separate each audio file into five arrays of smaller audio segments. The first array contained a list of all S1 sounds present in the audio. The second, third, and fourth arrays contained all of the systole, S2, and diastole sounds, respectively. The fifth array contained copies of the full heart cycles, starting with S1.

The next step in preprocessing the audio was to convert each sound segment from the time domain to the frequency domain with an N-point FFT. (The value of N was determined by looking at the maximum lengths of the segmented heart sounds. We selected N to be 364, 1024, 324, 2048, and 3760 for S1, systole, S2, diastole, and the full cycle, respectively. At the $2\,\mathrm{kHz}$ sampling frequency of the provided PCGs, these N-values correspond to $182\,\mathrm{ms}$, $512\,\mathrm{ms}$, $162\,\mathrm{ms}$, $1.024\,\mathrm{s}$, and $1.88\,\mathrm{s}$, respectively.) After calculating the FFT of each segment, we discarded the phase information and half of the (symmetric) magnitude information to reduce computational complexity.

Computing in Cardiology 2016; VOL 43 ISSN: 2325-887X DOI:10.22489/CinC.2016.234-191

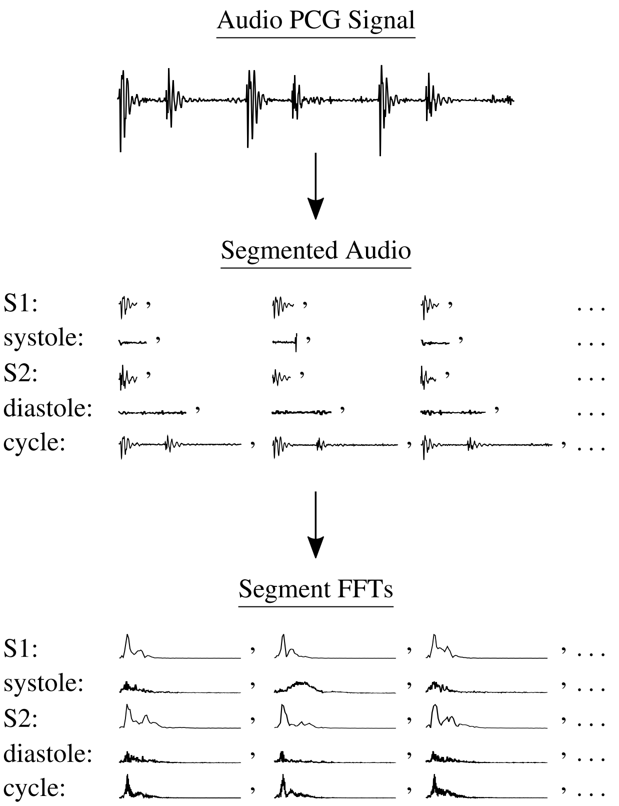

Figure 1. Visual representation of preprocessing applied to one PCG file. The preprocessing converts an audio file in the time domain into five arrays of frequency information, grouped by segmented heart sounds.

2.2. Sparse Coding as Unsupervised Feature Extraction

After preprocessing the PCG data, we randomly selected 1,000 of the 3,153 provided files from which to learn features. We used these training files to create five data matrices. The columns of the first data matrix were the preprocessed S1 segments. Likewise, the systole, S2, diastole, and full-cycle segments made up the columns for the other data matrices. We then applied sparse coding on these data matrices as a form of unsupervised feature extraction.

The goal of sparse coding is to decompose a data matrix ($\mathbf{Y}$) into the product of a dictionary matrix ($\mathbf{D}$) and a sparse coefficient matrix ($\mathbf{X}$):

$$\mathbf{Y} = \mathbf{D}\mathbf{X}. \tag{1}$$

Each column of $\mathbf{Y}$ represents a data sample, which in our case is the N-point FFT of a single subsegment of PCG audio. The dictionary matrix, $\mathbf{D}$, can be thought of as a set of commonly-occurring features learned from the training data. Fig. 2 gives a visual representation of Eq. 1.

Mathematically, performing this matrix decomposition

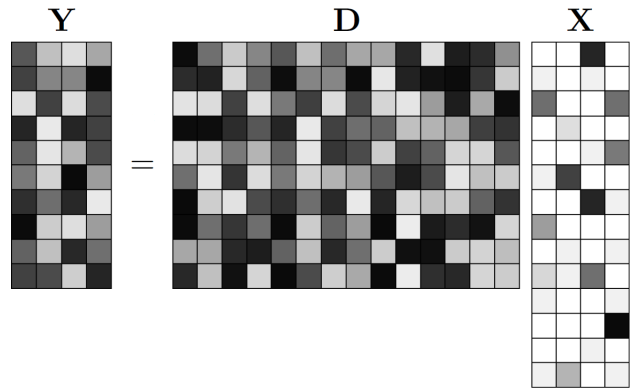

Figure 2. Visual representation of sparse coding. The feature matrix (left) is factored into the product of a dictionary matrix (center) and a sparse coefficient matrix (right).

corresponds to solving the following minimization problem [13, 14]:

$$\min_{\mathbf{D}\in\mathcal{C},\{\boldsymbol{x}_{\mathrm{m}}\}} \frac{1}{\mathrm{M}}\sum_{\mathrm{m}=1}^{\mathrm{M}} \frac{1}{2}\left\|\boldsymbol{y}_{\mathrm{m}} - \mathbf{D}\boldsymbol{x}_{\mathrm{m}}\right\|_2^2 + \lambda\left\|\boldsymbol{x}_{\mathrm{m}}\right\|_0. \tag{2}$$

In this equation, each $\boldsymbol{y}_{\mathrm{m}}$ corresponds to a column of $\mathbf{Y}$ and each $\boldsymbol{x}_{\mathrm{m}}$ corresponds to a column of $\mathbf{X}$. We jointly learn the dictionary matrix and the sparse coefficient vectors. We constrain the dictionary to $\mathcal{C}$, the set of matrices whose columns have ℓ_2-norm less than one. This prevents the dictionary from growing arbitrarily large, which would remove the effect of the ℓ_0 term in the objective function. The λ term is a fidelity-sparsity tradeoff parameter.

Unfortunately, the minimization program in Eq. 2 is a non-convex, NP-hard problem [15]. However, there are ways to approximate it and come up with workable solutions. One such method is to relax the ℓ_0-“norm” to the ℓ_1-norm and alternate solving for $\mathbf{D}$ and $\mathbf{X}$ while keeping the other constant. These relaxations result in the Alternating Minimization Algorithm, outlined in Alg. 1 [16, 17].

In our implementation of Alg. 1, we chose to update the dictionary (Line 8) using gradient descent, following the method reported in [8]. Line 5 of the algorithm is known in the literature as ‘basis pursuit denoising’ [18]. This is a well-studied problem, and we chose to solve it using the software package `l1_ls`, developed by Koh, et. al. [19].

The intuition behind using sparse coding as a feature extraction tool is that each column of the learned coefficient matrix defines how much of each dictionary element (feature) is needed to reconstruct the respective column of the data matrix. Because the coefficient vectors are constrained to be sparse, most coefficients will be zero. Ideally, the trained dictionary will have some elements that correspond to normal heart sounds and other elements that correspond to abnormal heart sounds.

Algorithm 1 Alternating Minimization.

Require: Signals $\{\boldsymbol{y}_{\mathrm{m}} \in \mathbb{R}^{\mathrm{N}}\}_{\mathrm{m}=1,\ldots,\mathrm{M}}$, initial dictionary $\mathbf{D_0} \in \mathcal{C}$, regularization term λ, number of iterations K

Initialize $\mathbf{D} \leftarrow \mathbf{D_0}$
for $\mathrm{k} = 1, \ldots, \mathrm{K}$ **do**
  for several $\mathrm{m} \in \{1, \ldots, \mathrm{M}\}$ (*in parallel*) **do**
    Calculate coefficient vectors:
    $$\boldsymbol{x}_{\mathrm{m}} = \arg\min_{\boldsymbol{x}} \frac{1}{2} \|\boldsymbol{y}_{\mathrm{m}} - \mathbf{D}\boldsymbol{x}\|_2^2 + \lambda \|\boldsymbol{x}\|_1$$
  end for
  Update dictionary:
  $$\mathbf{D} = \arg\min_{\mathbf{D}\in\mathcal{C}} \frac{\mathbf{1}}{\mathrm{M}} \sum_{\mathrm{m}=\mathbf{1}}^{\mathrm{M}} \frac{\mathbf{1}}{\mathbf{2}} \|\boldsymbol{y}_{\mathrm{m}} - \mathbf{D}\boldsymbol{x}_{\mathrm{m}}\|_{\mathbf{2}}^{\mathbf{2}}$$
end for
return $\mathbf{D}$

Applying sparse coding on the data matrices resulted in five different dictionaries. Each dictionary represents commonly-occurring spectral features present in the preprocessed S1, systole, S2, diastole, and full-cycle segments. Using these dictionaries, we computed sparse coefficient vectors for each segment of each file. We then averaged the coefficient vectors in each file. After this step, each PCG signal is represented by five sparse coefficient vectors.

2.3. Classification

We used the coefficient vectors from the unused 2,153 files to learn cross-validated SVM classifiers for each segment type (S1, systole, S2, diastole, and full cycle). We used the `libsvm` software package to learn the SVMs [20], we chose to train the SVMs using a first-order polynomial kernel, and we tuned the other SVM parameters using the modified cuckoo search algorithm [21]. We used the soft-margin output scores from these SVMs to train a final SVM that classified the PCG file to a single binary label.

Learning the five sparse coding dictionaries and the six SVMs was done offline. When we receive a new PCG file to test, we follow the same preprocessing procedure and then learn the sparse coefficient vectors using the appropriate dictionaries. After averaging the sparse vectors across the file, we generate five soft-margin scores using the segment-specific SVMs. These five scores are combined into a single label using the sixth SVM, resulting in a final answer for the new file.

3. Results

The best cross-validation score for the 2,153 files used to train the SVMs was 0.8652. The sensitivity was 0.8669 and the specificity was 0.8634. Previous attempts had cross-validation or holdout scores ranging from 0.82 to 0.86, but when tested on a subset of the unknown challenge data produced a high score of 0.812 (Se=0.825, Sp=0.799).

The average running time of our algorithm on the training set used 11.4% of the quota ($\approx 1.71 \times 10^{10}$ instructions). This takes about 3 seconds to run using MATLAB R2016a on a quad-core i7 processor clocked at 3.4 GHz with 16 GB RAM. The maximum running time used 41.0% of the quota.

4. Conclusion

The main contribution of this paper is to show that sparse coding can be used as a tool for unsupervised feature extraction. These features could possibly be combined with other features that are known to work well in heart sound classification tasks. In addition, the features could be used by different types of classifiers (not limited to SVMs) to improve accuracy.

Acknowledgments

In addition to thanking the Challenge organizers, the authors wish to thank Paul Chamberlain, a medical student at Baylor College of Medicine, for helpful conversations in conjunction with this paper.

This material is based upon work supported by the National Science Foundation Graduate Research Fellowship under Grant No. DGE-1148903.

References

[1] Liu C, Springer D, Li Q, Moody B, Juan RA, Chorro FJ, Castells F, Roig JM, Silva I, Johnson AE, Syed Z, Schmidt SE, Papadaniil CD, Hadjileontiadis L, Naseri H, Moukadem A, Dieterlen A, Brandt C, Tang H, Samieinasab M, Samieinasab MR, Sameni R, Mark RG, Clifford GD. An open access database for the evaluation of heart sound algorithms. Physiological Measurement 2016;37(9).

[2] Huang K, Aviyente S. Sparse representation for signal classification. In Advances in neural information processing systems. 2006; 609–616.

[3] Mairal J, Leordeanu M, Bach F, Hebert M, Ponce J. Discriminative sparse image models for class-specific edge detection and image interpretation. In Forsyth D, Torr P, Zisserman A (eds.), Computer Vision ECCV 2008, volume 5304 of Lecture Notes in Computer Science. Springer Berlin Heidelberg. ISBN 978-3-540-88689-1, 2008; 43–56.

[4] Wright J, Yang AY, Ganesh A, Sastry SS, Ma Y. Robust face recognition via sparse representation. IEEE Transactions on Pattern Analysis and Machine Intelligence Feb 2009;31(2):210–227. ISSN 0162-8828.

[5] Elad M, Figueiredo MA, Ma Y. On the role of sparse and redundant representations in image processing. Proceedings of the IEEE 2010;98(6):972–982.

[6] Kavukcuoglu K, Ranzato M, LeCun Y. Fast inference in sparse coding algorithms with applications to object recognition. arXiv preprint arXiv10103467 2010;.

[7] Wright J, Ma Y, Mairal J, Sapiro G, Huang T, Yan S. Sparse representation for computer vision and pattern recognition. Proceedings of the IEEE June 2010;98(6):1031–1044. ISSN 0018-9219.

[8] Charles AS, Olshausen BA, Rozell CJ. Learning sparse codes for hyperspectral imagery. Selected Topics in Signal Processing IEEE Journal of 2011;5(5):963–978. ISSN 1932-4553.

[9] Lee K, Hyung Z, Nam J. Acoustic scene classification using sparse feature learning and event-based pooling. In Applications of Signal Processing to Audio and Acoustics (WASPAA), 2013 IEEE Workshop on. IEEE, 2013; 1–4.

[10] Whitaker B, Carroll B, Anderson D. Sparse decomposition of audio spectrograms for automated disease detection in chickens. In Proceedings of the IEEE GlobalSIP. IEEE, 2014; 1122–1126.

[11] Schmidt SE, Holst-Hansen C, Graff C, Toft E, Struijk JJ. Segmentation of heart sound recordings by a duration-dependent hidden markov model. Physiological Measurement 2010;31(4):513.

[12] Springer DB, Tarassenko L, Clifford GD. Logistic regression-hsmm-based heart sound segmentation. IEEE Transactions on Biomedical Engineering April 2016; 63(4):822–832. ISSN 0018-9294.

[13] Olshausen BA, Field DJ. Sparse coding with an overcomplete basis set: A strategy employed by v1? Vision research 1997;37(23):3311–3325. ISSN 0042-6989.

[14] Tosic I, Frossard P. Dictionary learning. Signal Processing Magazine IEEE March 2011;28(2):27–38. ISSN 1053-5888.

[15] Tillmann AM. On the computational intractability of exact and approximate dictionary learning. IEEE Signal Processing Letters Jan 2015;22(1):45–49. ISSN 1070-9908.

[16] Mairal J, Bach F, Ponce J. Sparse modeling for image and vision processing. Foundations and Trends in Computer Graphics and Vision 2014;8(2-3):85–283. ISSN 1572-2740.

[17] Lee H, Battle A, Raina R, Ng AY. Efficient sparse coding algorithms. In Schölkopf B, Platt J, Hoffman T (eds.), Advances in Neural Information Processing Systems 19. MIT Press, 2007; 801–808.

[18] Chen SS, Donoho DL, Saunders MA. Atomic decomposition by basis pursuit. SIAM journal on scientific computing 1998;20(1):33–61.

[19] Koh K, Kim SJ, Boyd S. An interior-point method for large-scale l1-regularized logistic regression. Journal of Machine learning research 2007;8(7):1519–1555. Software available at `https://stanford.edu/~boyd/l1_ls/`.

[20] Chang CC, Lin CJ. LIBSVM: A library for support vector machines. ACM Transactions on Intelligent Systems and Technology 2011;2:27:1–27:27. Software available at `www.csie.ntu.edu.tw/~cjlin/libsvm`.

[21] Walton S, Hassan O, Morgan K, Brown M. Modified cuckoo search: A new gradient free optimisation algorithm. Chaos Solitons Fractals 2011;44(9):710 – 718. ISSN 0960-0779.