PVC Detection Using a Convolutional Autoencoder and Random Forest Classifier

Max Gordon[†] and Cranos Williams
*Department of Electrical and Computer Engineering, North Carolina State University,
Raleigh, North Carolina 27607, USA*
[†]*E-mail: mjgordo3@ncsu.edu*
www.ncsu.edu

The accurate detection of premature ventricular contractions (PVCs) in patients is an important task in cardiac care for some patients. In some cases, the usefulness to physicians in detecting PVCs stems from their long-term correlations with dangerous heart conditions. In other cases their potential as a precursor to serious cardiac events may make their detection a useful early warning mechanism. In many of these applications, the long-term nature of the monitoring required and the infrequency of PVCs make manual observation for PVCs impractical. Existing methods of automated PVC detection suffer from drawbacks such as the need to use difficult to extract morphological features, domain-specific features, or large numbers of estimated parameters. In particular, systems using large numbers of trained parameters have the potential to require large amounts of training data and computation and may have issues generalizing due to their potential to overfit. To address some of these drawbacks, we developed a novel PVC detection algorithm based around a convolutional autoencoder to address these weaknesses and validated our method using the MIT-BIH arrhythmia database.

Keywords: Electrocardiogram; Premature Ventricular Contraction (PVC) Detection; Autoencoder.

1. Introduction

Electrocardiograms (ECGs) are a useful and noninvasive diagnostic and monitoring tool in cardiac care.[1] One significant application of ECGs in cardiology is their use in the monitoring and treatment of arrhythmias. Premature Ventricular Contractions (PVCs) are a common arrhythmic beat type that occurs commonly in many patients, including individuals with good cardiac health.[2] However, when they occur in large numbers or with high frequency in patients with other risk factors, PVCs can be associated with serious cardiac problems and may precede heart attacks or sudden cardiac death in rare cases.[2] As a result, the automated detection of PVCs in ECG records would allow information about their long-term frequency to be tracked over time, providing a new means to track the trends in a patient's cardiac health as well as potentially providing an early warning of events requiring swift medical attention.

There are several main categories of approaches to feature extraction for the automated detection of PVCs: 1) morphological and timing features extracted from the ECG signal[3–5] and 2) time-frequency features such as wavelet transforms of the ECG signal.[6,7] In addition to these two main approaches to PVC detection, there are methods utilizing other approaches to the connected problems of feature extraction and beat classification,[8] Markov models, independent component analysis,[9] and autoencoders.[10]

Geddes and Warner[3] used R-R interarrival time, QRS complex duration, and signal slope during several sections of the QRS complex as features in their detection system. They made classification decisions based on a manually constructed decision tree. This allowed for computationally simple evaluation of a QRS complex but sacrificed adaptibility and required heuristic tuning and domain specific knowledge of the PVC detection problem to adjust the classifier. Trahanias et al.[4] used a number of structural descriptors to create a syntactic description of the QRS complex. After this syntactic description was created, they used a normalized distance metric to form classes of QRS complexes, which were found to correspond to some clinically significant classes of heartbeats. However, this method did not lead to a direct and useful classification of the QRS complex. Zadeh et al.[5] used a total of 10 morphological features and 3 timing features extracted from the signal of a detected QRS complex. They compared several kinds of classifiers including MLP neural networks, RBF neural networks, probabilistic neural networks, and support vector machines. In addition to detecting PVCs, they used their classification system to identify non-PVC arrhythmias.

In all of these approaches, significant domain knowledge was used to determine feature sets and detection accuracy was dependent on the classification of different parts of the QRS complex for segmentation and measurement. It is desirable to avoid these issues by using a more general and robust method of feature extraction. Ham and Han[6] used two estimated linear prediction coefficients in combination with the mean squared value of the signal as features for classification. They used a fuzzy ARTMAP neural network to perform the classification. Lim[7] used a discrete wavelet transform with the Haar wavelet to generate a feature vector and used a fuzzy neural network for classification. While these approaches still require manual feature selection, the specific features extracted are less domain specific and do not require segmentation of the QRS complex to calculate.

One approach to avoid the challenges associated with engineering a problem-specific feature set is to use feature learning approaches such as independent component analysis or autoencoders to extract a feature set that is able to describe much of the information content of a signal in a low-dimensional latent space.[11] Yu and Chou[9] used independent component analysis to identify and extract a set of features, which were combined with QRS complex timing information to create the full feature set passed to their neural network classifier. Yang et al.[10] used a sparse autoencoder (SAE) to generate a feature vector for classification. This resulted in a large number of estimated network weights, which increased the computation and data required to train the network and increased the potential for overfitting.

The primary aims of this study are to develop a system for the detection of PVCs in ECG data that does not rely on manually selected features and has fewer parameters to be estimated than existing SAE methods. These improvements will reduce the possibility of overfitting and improve the generalization of the detection system. For this purpose, we used an autoencoder architecture based on convolutional layers to extract and select features for use in classifying beats. Our architecture is differentiated from existing convolutional autoencoders (CAEs)[12] by its multi-stage encoding process, which allows it to encode information about the frequency content of a signal at different points in time.

2. Methods

2.1. *Data Set and Implementation*

We used ECG records from the MIT-BIH arrhythmia database annotated with beat locations and types.[13] This database consists of 48 30-minute 2 channel ECG records sampled at 360 Hz. Only channel 1 of the ECG was used for PVC detection because in the MIT-BIH arrhythmia database this signal is a modified limb lead II, which has clearer signals for non-ectopic beats than the modified lead V1 available on channel 2. As much of the information content of a QRS complex is centered on the R peak, the ECG signals obtained from the database were segmented based on the annotated R peaks, with 89 samples before and 160 samples after each annotated R peak extracted for feature calculation. In application outside the MIT-BIH database, this means we assume the QRS complexes are reliably detected before being passed to our detection system. We then removed the mean from each segmented QRS complex to reduce the impact of baseline drift, variations in instrumentation, and differences across patients. The PVC detection system was implemented in Python using the Keras,[14] TensorFlow,[15] and scikit-learn[16] libraries.

2.2. *Proposed PVC Detection Method*

A convolutional autoencoder (CAE)[12] was used to extract and select features for classification automatically and in an unsupervised manner from ECG data annotated with beat locations. This reduced the need for domain-specific knowledge as compared to manual feature selection. Compared to a SAE, a CAE reduces the number of weights that need to be trained, increases the robustness of the features extracted when the window alignment of the beats being processed is variable, and takes advantage of the structure of the ECG signal in its architecture. We used a Random Forest Classifier to perform the final PVC detection due to its resistance to overfitting and its performance with the indistinct groupings of PVC and non-PVC beats. Our system architectures for training the CAE and Random Forest Classifier are shown in Figure 1, while our classification architecture is shown in Figure 2. Examples of normal beats and PVCs are given in Figure 3

2.2.1. *Feature Extraction*

An autoencoder is a neural network that encodes its input to a latent space representation attempts to decode this representation to recover the inputs.[17] In a CAE, the layers responsible for encoding and decoding the latent space are convolutional, using shared weights to kernels to extract features from their input. After the network has been trained, the encoding layers alone can be used to reduce the dimensionality of the input data for further processing.

In the proposed PVC detection method, two convolutional layers with linear activations were used to encode the input to the CAE. The first of these layers generated n kernels of

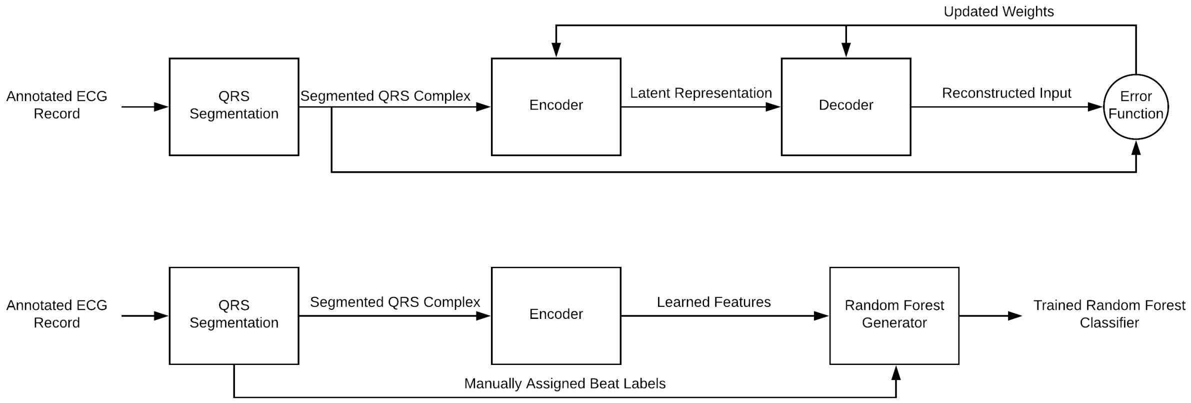

Fig. 1. CAE and Random Forest Training Architecture

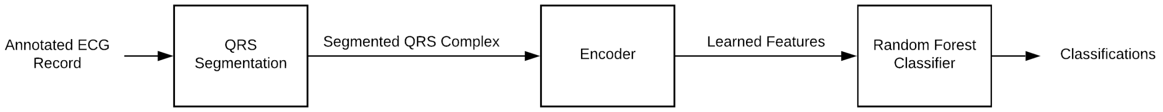

Fig. 2. Classification Architecture

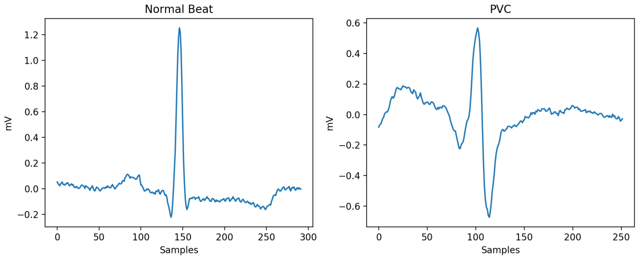

Fig. 3. Comparison of Normal Beat and PVC

order m to extract different features from the input. A stride length of k was used in this layer to downsample the input, reducing its dimensionality. The second convolutional layer generated a single kernel to compute a linear combination of the outputs of the previous layers kernels at each point. This second layer serves as a feature selection stage. As a result, each feature in the latent space representation of the input corresponds to a combination of all features extracted in the first layer from a continuous subset of the input. This provides information on the frequency components of the ECG signal most important for creating an accurate reconstruction of the original signal as well as some degree of temporal localization within the signal. This allows the encoded representation to contain distinct information about various stages in the progression of the QRS complex without the need to explicitly define and detect these stage, simplifying the PVC detection process in comparison to methods using morphological features of the QRS complex.

We used transposed convolutional layers to decode the latent space representation generated by the encoder. These layers have the same connectivity and dimensionality as the encoding layers but are reversed. This results in an output matching the dimensionality of the input to the CAE and allows us to train the network to replicate its inputs. In operation, only the encoding side of the network was used to generate the features used in classification. The resulting network architecture is shown in Figure 4.

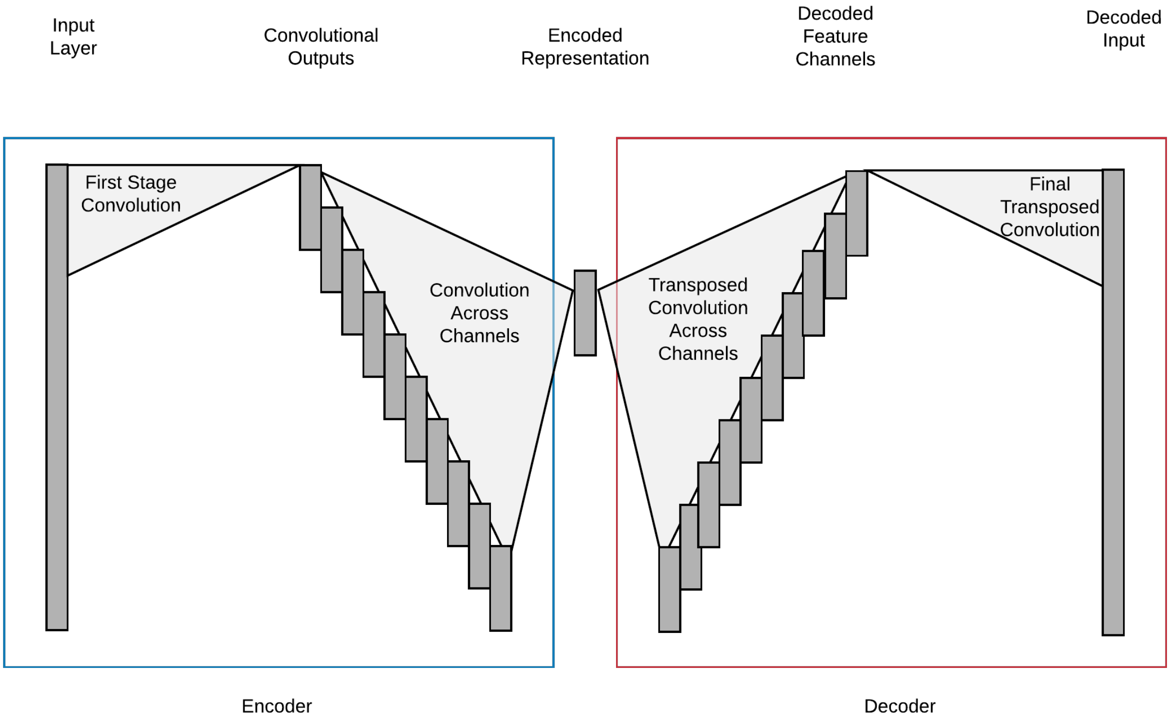

Fig. 4. Convolutional Autoencoder Architecture

For this application, the length of signal extracted around each beat even was 250 samples, with 89 samples before the annotation and 160 samples after the annotation. These values were selected because they were found to provide generally acceptable classification performance and allowed for a more direct comparison with the PVC detection system described by Yang et al.[10] An n value of 25 provided a sufficient number of base features for the following layer to perform feature selection on. An m value of 20 provided sufficiently complex filters to extract a wide range of characteristics from the signal. A k value of 10 allowed the final feature vector to be of dimension 25. This was found to provide sufficient segmentation of the input signal in time while also being of low enough dimensionality to allow for adequate classifier performance. The CAE was trained using an ADAM optimizer as described by Kingma and Ba[18] with a learning rate of 0.01 and a mean squared error loss function: $MSE = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y_i})^2$, where Y is the input to the autoencoder and $\hat{Y}$ is the output of the autoencoder.

2.2.2. *Classification*

We used a Random Forest Classifier as described by Breiman.[19] The random forest used in this detection system consisted of 10 decision trees with Gini impurity as their splitting criterion. Gini impurity is the probability that a randomly selected element in a set would be mislabeled if labeled at random.[19] For J classes with probability of selection p, the Gini impurity of a set is given by $I_G = 1 - \sum_{i=1}^{J} p_i^2$. The features used to split each node of the tree were randomly determined. The classifier also used bagging to avoid overfitting, using a set of training examples of the same size as the full dataset sampled without replacement as the training dataset for each random tree. The Random Forest Classifier was chosen due to its low number of parameters, its resistance to overfitting, and its ability to handle fuzzy group boundaries in comparison to support vector methods, neural networks, and other common classifiers.

3. Results

We evaluated our method with 3 tests. First, we tested its performance when provided with ample training data including samples from each record. Next, we added a randomized error to the R peak location used in segmentation to simulate inaccurate QRS detection. Finally, we provided our system with training data that included no beats from the records used for testing to evaluate its ability to generalize to new patients. Each of these tests was also performed using a SAE to provide context to the performance of the CAE. In addition to the testing we performed, we examined the number of estimated weights and the number of training epochs necessary for convergence in both the CAE and SAE architectures.

3.1. *Full Database Evaluation*

We evaluated the classification system using the MIT-BIH arrhythmia database. Half of the beats from each record were selected as training data and the remainder were used as testing data. This resulted in a training set consisting of 54,695 beats with 3,495 PVCs and a testing set consisting of 54,675 beats with 3,633 PVCs. The results of this testing are shown in Table 8 with information for each record. A SAE similar to one described by Yang et al.[10] was constructed, with the sparsity imposed by L1 regularization instead of the Kullback-Leibler divergence derived regularization described, to compare the feature extraction provided by the CAE to that provided by an existing alternative architecture. A comparison of the performance of these two architectures is provided in table Table 1 and Table 2. This evaluation demonstrates that the CAE provides similar performance to the SAE when ample training data is available, with a difference in overall accuracy of 0.2%. However, the PVC sensitivity of the CAE is 4.88% higher than that of the SAE, meaning that fewer PVCs are missed by the CAE. This is desirable given the relative rarity of PVCs, although the importance of sensitivity and specificity will need to be evaluated for individual applications.

3.2. *Timing Disturbance Evaluation*

As QRS detection is necessary to the identification and segmentation of potential PVCs for processing by a PVC detection system, this property makes resistance to small shifts in the

Table 1. MIT-BIH Full Database Comparative Evaluation

Architecture	Correct	PVC Sensitivity	PVC Specificity
CAE	98.43	85.64	98.90
SAE	98.23	80.76	99.07

Table 2. Full Database CAE and SAE Confusion Matrices

	CAE		SAE	
	True Normal	True PVC	True Normal	True PVC
Detected Normal	50483	299	50565	489
Detected PVC	559	3334	477	3144

precise placement of the annotation within the beat desirable. We evaluated this robustness by applying a random shift of up to 36 samples to each beat, corresponding to a detection error of up to 100 milliseconds. The results of this testing on the CAE are shown in Table 9 with information for each record, while a comparison of the performance of the CAE and SAE architectures under these conditions is presented in Table 3 and Table 4. This shows that the CAE suffers a 0.83% reduction in PVC sensitivity as a result of this shifting, while the SAE suffers a 4.26% reduction in PVC sensitivity. This results in a total sensitivity improvement for the CAE of 8.43% relative to the SAE under these conditions.

Table 3. MIT-BIH Full Database Disturbed

Architecture	Correct	PVC Sensitivity	PVC Specificity
CAE	97.60	84.93	98.42
SAE	97.17	76.50	97.66

Table 4. Full Database CAE and SAE Disturbed Confusion Matrices

	CAE		SAE	
	True Normal	True PVC	True Normal	True PVC
Detected Normal	50542	810	50708	1217
Detected PVC	501	2823	335	2416

3.3. *Cross-Patient Training Evaluation*

In an applied setting, it may not always be practical to obtain annotated training data from a patient to train any monitoring system. As a result, system performance when trained only using data obtained from other individuals is potentially important to the practical utility of any PVC detection method. We evaluated this performance metric by training both PVC detection systems using all beats in two ECG records and testing on all beats in four ECG records. All such combinations of records 116, 208, 210, 221, 228, and 233 in the MIT-BIH database were used to evaluate model generalization. We chose this subset of the MIT-BIH database because testing all combinations of records in the entire dataset is impractical and because it was selected as representative of the database by Ham and Han.[6] The averages of these results are given in Table 5, while Table 6 provides confusion matrices of the aggregated results. These show that the CAE provides 1.01% higher overall accuracy and 4.71% higher PVC sensitivity than the SAE. This meets our expectation that a reduced number of trained weights in the autoencoder would improve performance with reduced amounts of training data as well as improve the ability of the detection system to generalize to new data.

Table 5. MIT-BIH Restricted Training Cross-Validation

Architecture	Correct	PVC Sensitivity	PVC Specificity
CAE	87.80	86.56	88.09
SAE	86.79	81.85	87.91

Table 6. Cross-Validation CAE and SAE Confusion Matrices

	CAE		SAE	
	True Normal	True PVC	True Normal	True PVC
Detected Normal	111721	3874	111499	5234
Detected PVC	15109	24956	15331	23596

3.4. *Estimated Parameters and Convergence*

Our convolutional autoencoder architecture used 83.43% fewer network weights due to the weight sharing inherent in convolutional networks. For the 54695 example training set used in 3.1 and 3.2, this resulted in a decrease in the number of training epochs necessary for convergence from 5 to 1.

Table 7. Network Weights

Architecture	Estimated Weights
CAE	1702
SAE	10270

Table 8. MIT-BIH Full Database CAE Performance

Record	Beats	Normal	PVC	Correct	Sensitivity	Specificity
100	1135	1134	1	100.000	100.000	100.000
101	931	931	0	100.000	—	100.000
102	1092	1090	2	100.000	100.000	100.000
103	1040	1040	0	99.904	—	99.904
104	1113	1112	1	100.000	100.000	100.000
105	1285	1272	13	95.642	46.154	96.148
106	1012	673	339	96.542	89.676	100.000
107	1067	1020	47	99.438	87.234	100.000
108	880	873	7	99.432	28.571	100.000
109	1264	1242	22	81.487	77.273	81.562
111	1061	1061	0	99.811	—	99.811
112	1268	1268	0	100.000	—	100.000
113	896	896	0	100.000	—	100.000
114	938	936	2	100.000	100.000	100.000
115	975	975	0	100.000	—	100.000
116	1204	1158	46	99.917	97.826	100.000
117	766	766	0	100.000	—	100.000
118	1138	1130	8	99.385	25.000	99.912
119	992	747	245	100.000	100.000	100.000
121	930	929	1	99.785	0.000	99.892
122	1236	1236	0	100.000	—	100.000
123	758	756	2	100.000	100.000	100.000
124	808	789	19	98.886	52.632	100.000
200	1299	817	482	97.614	93.568	100.000
201	980	860	120	99.388	95.833	99.884
202	1066	1064	2	99.812	50.000	99.906
203	1489	1283	206	97.851	90.777	98.987
205	1326	1280	46	99.623	89.130	100.000
207	929	925	4	87.836	100.000	87.784
208	1476	1024	452	97.900	98.894	97.461
209	1501	1501	0	100.000	—	100.000
210	1323	1212	111	97.279	68.468	99.917
212	1372	1372	0	100.000	—	100.000
213	1624	1517	107	98.153	93.458	98.484
214	1129	1006	123	97.874	81.301	99.901
215	1680	1598	82	98.452	68.293	100.000
217	1103	1037	66	99.547	95.455	99.807
219	1076	1044	32	99.257	75.000	100.000
220	1022	1022	0	100.000	—	100.000
221	1212	1051	161	99.917	99.379	100.000
222	1240	1240	0	100.000	—	100.000
223	1301	985	316	96.772	88.291	99.492
228	1025	877	148	98.829	91.892	100.000
230	1127	1126	1	99.379	100.000	99.378
231	784	784	0	100.000	—	100.000
232	889	889	0	100.000	—	100.000
233	1538	1122	416	98.635	96.154	99.554
234	1375	1372	3	99.709	0.000	99.927
Total	54675	51042	3633	98.548	91.412	99.056

Table 9. MIT-BIH Full Database CAE Disturbed Performance

Record	Beats	Normal	PVC	Correct	Sensitivity	Specificity
100	1135	1134	1	99.912	0.000	100.000
101	931	931	0	100.000	—	100.000
102	1092	1090	2	99.817	50.000	99.908
103	1040	1040	0	100.000	—	100.000
104	1113	1112	1	100.000	100.000	100.000
105	1285	1272	13	93.541	23.077	94.261
106	1012	673	339	89.526	68.732	100.000
107	1067	1020	47	99.157	80.851	100.000
108	880	873	7	99.091	28.571	99.656
109	1264	1242	22	78.006	45.455	78.583
111	1061	1061	0	100.000	—	100.000
112	1268	1268	0	100.000	—	100.000
113	896	896	0	100.000	—	100.000
114	938	936	2	100.000	100.000	100.000
115	975	975	0	100.000	—	100.000
116	1204	1158	46	99.917	97.826	100.000
117	766	766	0	100.000	—	100.000
118	1138	1130	8	99.297	25.000	99.823
119	992	747	245	99.899	99.592	100.000
121	930	929	1	99.785	0.000	99.892
122	1236	1236	0	100.000	—	100.000
123	758	756	2	100.000	100.000	100.000
124	808	789	19	98.020	15.789	100.000
200	1299	817	482	95.766	88.589	100.000
201	980	860	120	97.449	79.167	100.000
202	1066	1064	2	99.906	50.000	100.000
203	1489	1283	206	95.433	76.214	98.519
205	1327	1281	46	99.171	76.087	100.000
207	929	925	4	95.048	100.000	95.027
208	1476	1024	452	96.206	95.354	96.582
209	1501	1501	0	100.000	—	100.000
210	1323	1212	111	93.878	28.829	99.835
212	1372	1372	0	99.927	—	99.927
213	1624	1517	107	97.845	85.047	98.748
214	1129	1006	123	93.711	47.154	99.404
215	1680	1598	82	96.845	35.366	100.000
217	1103	1037	66	98.368	84.848	99.229
219	1076	1044	32	98.792	84.375	99.234
220	1022	1022	0	100.000	—	100.000
221	1212	1051	161	99.752	98.137	100.000
222	1240	1240	0	99.919	—	99.919
223	1301	985	316	87.855	50.949	99.695
228	1025	877	148	97.268	81.081	100.000
230	1127	1126	1	99.734	100.000	99.734
231	784	784	0	100.000	—	100.000
232	889	889	0	100.000	—	100.000
233	1538	1122	416	95.904	85.817	99.643
234	1375	1372	3	99.782	33.333	99.927
Total	54676	51043	3633	97.608	77.814	99.017

4. Discussion

We developed a system for the detection of PVCs in ECG data annotated with beat locations using a CAE. This provided comparable performance to a SAE architecture for the task with reduced training time due to its reduced number of parameters. The CAE provided improvements in the resilience of the PVC detection system to beat detection timing variance and improved detection performance when trained using ECG records from different patients.

Some limitations of this approach to PVC detection include the computational complexity of representation learning methods as compared to manual feature engineering and the lack of direct and unambiguous physical or medical significance for the features extracted by the system. There is also no guarantee that homologous features will be generated by training on different ECG data, which precludes the possibility of retraining the convolutional autoencoder without also retraining the final classifier.

The relatively low number of parameters in our model make it well suited to implementation on the limited hardware available in an applied setting while not relying on potentially unreliable QRS segmentation or features that are difficult to measure or compute in real time. In addition to its advantage in computational expense, the improvement provided by our autoencoder architecture in cross-patient generalization is of significant importance in the application of a PVC detection system to real patients, where it may be impractical or impossible to obtain a sufficient amount of expert-annotated training data.

Based on the performance of this system, we envision the extension of our CAE architecture to facilitate the detection of other arrhythmias in ECG data. Another potential avenue for future work with this autoencoder architecture is to take advantage of its small number of trained parameters to allow the model to be retrained on the spot based on a subset of available annotated ECG records most similar to a sample of the ECG data from the current patient.

References

1. H. J. L. Marriott, G. S. Wagner and D. G. Strauss, *Marriott's Practical electrocardiography* (Wolters Kluwer Health / Lippincott Williams & Wilkins, Philadelphia, 2014).
2. C. L. Stanfield and W. J. Germann, *Principles of human physiology.* (Pearson Benjamin Cummings, San Francisco; London, 2008).
3. J. S. Geddes and H. R. Warner, *Computers and Biomedical Research* **4**, 493 (1971).
4. P. Trahanias, E. Skordalakis and G. Papaconstantinou, *Pattern recognition letters* **9**, 13 (1989).
5. A. E. Zadeh, A. Khazaee and V. Ranaee, *Computer Methods and Programs in Biomedicine* **99**, 179 (August 2010).
6. F. M. Ham and S. Han, *IEEE Transactions on Biomedical Engineering* **43**, 425 (1996).
7. J. Lim, *IEEE Transactions on Neural Networks* **20**, 522 (March 2009).
8. W. Gersch, P. Lilly and E. Dong Jr, *Computers and Biomedical Research* **8**, 370 (1975).
9. S. Yu and K. Chou, *Expert Systems with Applications* **34**, 2841 (May 2008).
10. J. Yang, Y. Bai, G. Li, M. Liu and X. Liu, *Bio-Medical Materials and Engineering* **26**, S1549 (August 2015).
11. Y. Bengio, A. Courville and P. Vincent, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **35**, 1798 (August 2013).
12. X. Guo, X. Liu, E. Zhu and J. Yin, Deep clustering with convolutional autoencodersOctober 2017.

13. G. B. Moody and R. G. Mark, *IEEE engineering in medicine and biology magazine: the quarterly magazine of the Engineering in Medicine & Biology Society* **20**, 45 (June 2001).
14. F. Chollet *et al.*, Keras `https://keras.io`, (2015).
15. M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu and X. Zheng, TensorFlow: Large-scale machine learning on heterogeneous systems (2015), Software available from tensorflow.org.
16. F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot and E. Duchesnay, *Journal of Machine Learning Research* **12**, 2825 (2011).
17. C.-Y. Liou, W.-C. Cheng, J.-W. Liou and D.-R. Liou, *Neurocomputing* **139**, 84 (September 2014).
18. D. P. Kingma and J. Ba, Adam: A Method for Stochastic Optimization, in *International Conference on Learning Representations*, December 2014.
19. L. Breiman, *Machine Learning* **45**, 5 (October 2001).