ECG R-R Interval Estimation

David Zeng
Electrical Engineering
Stanford University

December 7, 2015

Introduction

Magnetic resonance imaging (MRI) is a biomedical imaging modality that provides high resolution images and excellent soft tissue contrast [1]. This makes MRI a preferable imaging modality for many applications. One limitation of MRI is slow data acquisition. MRI acquires data in the Fourier domain one point at a time, usually until enough data has been collected to approximately satisfy Nyquist sampling conditions. Thus MRI scans are on the order of minutes to tens of minutes and this makes MRI particularly sensitive to motion. Existing methods can robustly correct small amounts of motion but correcting large motions is an active area of research [2].

In cardiac MRI, motion comes from cardiac and respiratory motion. Respiratory motion is relatively small and can be corrected. Cardiac motion during diastole is also relatively small and correctable and thus data is usually collected during this period (Figure 1). Current implementations have a fixed data acquisition duration as shown in Figure 1 and are triggered by the R-peak. Using the results of this project, we aim to modify this scheme to create a dynamic-length data acquisition sequence to maximize signal-to-noise ratio (SNR) efficiency (SNR efficiency $\propto \text{SNR}/\sqrt{T}$). If diastole is shorter than the acquisition length, we will end up collecting data during systole. This data will be too corrupted by motion to recover and all the data collected during the previous interval must be discarded. If diastole is longer than acquisition length, we will wasting time during diastole in which we could have acquired data, reducing SNR efficiency.

We would like to estimate diastole length but this is not an easy problem. Instead, since we are already collecting ECG data during the exam and diastole length is correlated to the R-R interval (the time between two consecutive R-peaks) [3, 4], we estimate the R-R interval.

This project first implements several classification approaches as a proof of concept. The first approach is a series of classification approaches using logistic regression and support vector machines (SVM) to predict if the next R-R interval will be sufficiently long to accommodate the data acquisition. These methods use an n-gram model of previous R-R intervals as input and output a prediction of the length of the next R-R interval. We then try a softmax classification approach to predict the largest integer number of imaging blocks (IMG, Figure 1) that will fit in the interval. We again use an n-gram model of previous R-R intervals for input and output a predicted maximum integer.

A nonlinear autoregressive neural network (NARNN) is then implemented to estimate the length of the next R-R interval. The input to the NARNN is the ECG time series.

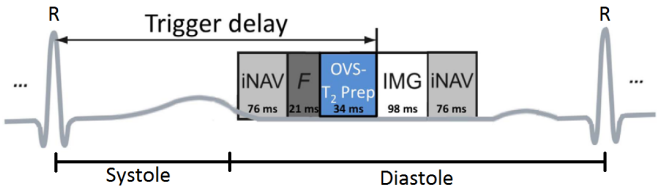


Figure 1: MRI data acquisition sequence and partially annotated ECG.

Related Work

There has been one recent paper on R-R interval prediction [5]. This paper uses autoregressive-moving average with exogenous terms (ARMAX) and artificial neural networks (ANN). They wavelet transform the time series, predict the wavelet coefficients, and then inverse wavelet transform the signal and compute the R-R interval. The results from the ARMAX are reasonable but damped compared to the range of the original signal and have about 20% error. The results from the ANN are much worse, often predicting negative R-R intervals, which are impossible. The ANN results have between 25% and 35% error.

Dataset and Features

The MIT-BIH Arrhythmia Database is used for this project [6]. The only annotations of the dataset are time markers for each R peak (Figure 2). Each ECG is sampled at 360 Hz and each recording is 30 minutes. There are 48 recordings for a total of 24 hours of ECG data and 3.12x10^7 time points.

For the classification methods, there were two approaches to create training and testing sets. One method divided each time series for training and testing in a 70:30 ratio respectively. Another method randomly divided the data files in a 70:30 ratio. The R-R intervals were then calculated from the annotations and grouped to construct the n-grams.

Alternatively, the time series itself can be used as a feature vector. The feature vector is the length of the longest R-R interval and the other intervals are zero-padded to this length. This implementation allows the QRS complexes to be roughly aligned.

For the neural network, no processing was performed on the time series. We divided the training and testing data by dividing each time series in a 70:30 ratio for input. This was prompted by the noticeably better classification results using this method. The expected output is a step function with height in seconds of the length of the next R-R interval. The output has this value for the duration of the current R-R interval.

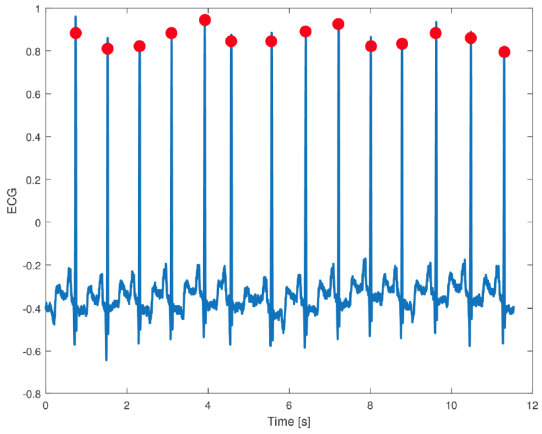


Figure 2: Example dataset time series with R-peaks annotated.

Methods

Our first approaches are classification learning algorithms using n-gram models. An n-gram model assumes that a process is Markov on its n previous states. This means that each feature vector x is the n previous R-R intervals. Each method classifies if the next time series will be greater than 600 ms. We arrive at this number by calculating that the minimum data acquisition time is 305 ms [7] and according to [3, 4], diastole is 53.4% or 57.9% of each R-R interval.

The first approach is a linear classifier for a logistic regression [8]. A linear classifier means that the input to our hypothesis function can be modeled as $\theta^T x$ and we will solve for θ. We usually augment x such that $x_0 = 1$ and $\theta \in \mathbb{R}^{n+1}$.

For logistic regression, our hypothesis is $h_\theta(x) = g(\theta^T x) = \frac{q}{1+e^{-\theta^T x}}$. Where $g(z) = \frac{1}{1+e^{-z}}$ is called the logistic or sigmoid function. Let us assume that $P(y = 1|x; \theta) = h_\theta(x)$ and $P(y = 0|x; \theta) = 1 - h_\theta(x)$. Furthermore, assume we have m training samples and let $(x^{(i)}, y^{(i)})$ be the i-th feature vector-output pair. Then we want to maximize the likelihood of

$$L(\theta) = \prod_{i=1}^{m} p(y^{(i)}|x^{(i)}; \theta) = \prod_{i=1}^{m} (h_\theta(x^{(i)})))^{y^{(i)}} (1 - h_\theta(x^{(i)})))^{1-y^{(i)}}.$$

For a more tractable problem, we can take the log likelihood and still maintain same solution.

$$\ell(\theta) = \log L(\theta) = \sum_{i=1}^{m} y^{(i)} \log h(x^{(i)} + (1 - y^{(i)}) \log(1 - h(x^{(i)}).$$

We then find the gradient and Hessian with respect to θ and use Newton's method to solve for $\theta = \arg\max \ell(\theta)$. To calculate a prediction y from an input x and the θ we have just found, we choose the maximum likelihood label of $P(y|x;\theta)$.

Another approach for classification is using a support vector machine (SVM)[8]. A support vector machine finds a linear separation of two sets to maximize the margin between the two sets. Let γ be the margin between the two sets and let w, b be parameters of the separating hyperplane. Then we want

$$\begin{aligned} &\max_{\gamma,w,b} \frac{\gamma}{\|w\|} \\ \text{s.t.} \quad &y^{(i)}(w^T x^{(i)} + b) \geq \gamma, \quad i = 1, \ldots, m \\ &\|w\| = 1. \end{aligned}$$

The problem is not convex but we can equivalently reformulate it to be.

$$\begin{aligned} &\min_{w,b} \frac{1}{2}\|w\|^2 \\ \text{s.t.} \quad &y^{(i)}(w^T x^{(i)} + b) \geq 1, \quad i = 1, \ldots, m \end{aligned}$$

However, this problem is infeasible for many real datasets because data is not always linearly separable. Even if it is linearly separable, the solution may not result in the lowest generalization error. We can relax and regularize the problem by instead solving for

$$\begin{aligned} &\min_{w,b} \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{m} \xi_i \\ \text{s.t.} \quad &y^{(i)}(w^T x^{(i)} + b) \geq 1 - \xi_i, \quad i = 1, \ldots, m \\ &\xi_i \geq 0, \quad i = 1, \ldots, m. \end{aligned}$$

This allows some points to be misclassified but with a penalty ξ_i. The parameter C controls the relative weighting of the size of the margin and number of misclassified points. This optimization problem is solved using any modern quadratic problem solver. In this project, the SVM is solved for by the LIBLINEAR library [9].

The next approach is a softmax regression again using n-gram models [8]. We want to classify imaging blocks we can fit during diastole. From Figure 1, we see that each imaging block is approximately 100ms so we form bins of [<600 ms, 600-760 ms, 760-920 ms, >920ms]. To implement softmax regression, we first determine the empirical probability of each label. Let $p(y = i|x;\theta) = \phi_i = \frac{\exp(\theta_i^T x)}{\sum_{j=1^k} \exp(\theta_j^T x)}$. Like before, θ is a parameter that we will solve for by maximizing the conditional probability. We solve for θ by maximizing the log likelihood

$$\arg\max_{\theta} \ell(\theta) = \sum_{i=1}^{m} \log p(y^{(i)}|x^{(i)};\theta) = \sum_{i=1}^{m} \log \prod_{l=1}^{k} \Big(\frac{\exp(\theta_l^T x)}{\sum_{j=1^k} \exp(\theta_l^T x)} \Big)^{1\{y^{(i)}=l\}}.$$

Now that we have learned the parameters θ, for a given input x, we can compute the conditional probability $p(y = i|x;\theta)$ and choose the maximum likelihood label. Softmax was calculated using the MATLAB (Mathworks, Natick, MA) function `mnrfit`.

The final approach is a nonlinear autoregressive neural network [10]. A neural network is made up of neurons. Let p be a vector of inputs to a neuron and q the output. $q = f_\theta(p)$ where θ is a set of parameters to be found and f is nonlinear in p. We connect many layers of neurons in a feed-forward network to successively combine outputs of each layer such that the i-th neuron of the k-th layer outputs $q_i^{(k)} = f_{\theta_{i,k}}(q_1^{(k-1)}, \ldots, q_n^{(k-1)})$. To solve for the parameters θ in a network with ℓ layers, we minimize mean square error $\sum(y^{(i)} - f_{\theta_{i,\ell}}(q_1^{(\ell-1)}, \ldots, q_n^{(\ell-1)}))^2$.

In this project, we use the Levenberg-Marquardt algorithm to solve the problem and use the MATLAB neural network framework to implement the neural network. We choose delays of 4 and 20 hidden layers of 4 neurons each and one output layer of 1 neuron (Figure 3).

The output of the NARNN is quite noisy because the expected output is a series of step functions and the trained response usually overshoots the steady-state level. Thus we must process the signal to extract a final value. We first filter the signal with a 5-tap low-pass filter $\frac{1}{5}[1, 1, 1, 1, 1]$. We then take the arithmetic mean of the filtered signal from 0.50 to 0.85 of each normalized interval. We choose this interval because the signal is approximately steady during this time.

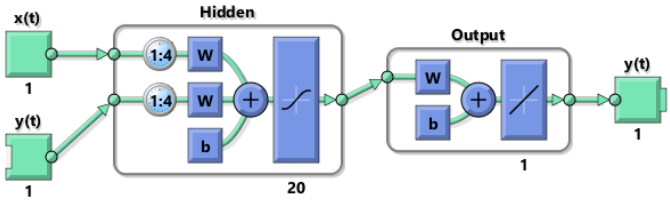


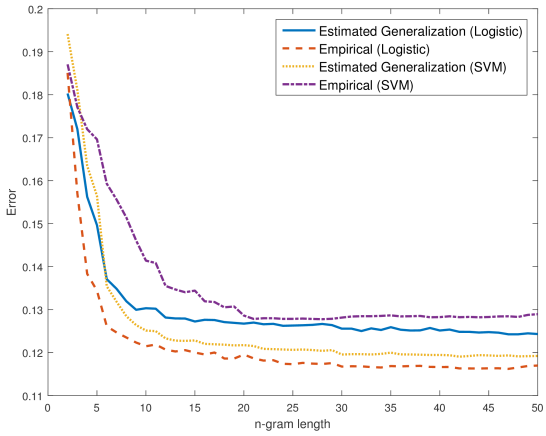
Figure 3: NARNN topology.

Results and Discussion

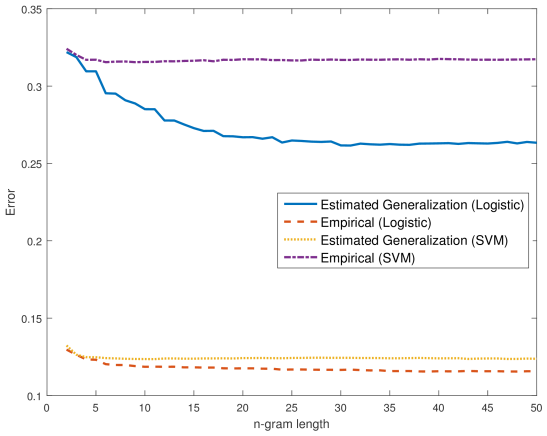
The results for the classification problems are shown in Figure 4. The results are from the average of 10 trials. We see that as n-gram length increases, the errors asymptotically reach the values shown in Figure 4c). A 25-gram looks like a reasonable input feature length as a tradeoff of error and computation. Further analyzing Figure 4c), we notice that by training on the dataset split by time series, both logistic regression and SVM perform significantly better than the 'yes' hypothesis. In contrast, training on the dataset split by data file resulted in much worse or only slightly better performance.

This significant difference in performance is interesting because intuitively, by randomly dividing the data using either method, a representative pattern should have emerged. One possible explanation is that certain files have unique arrhythmic patterns that are not seen in the other files and thus if these patterns are not in the training set, the learned model performs poorly.

Time series were also used as a feature vector for both logistic regression and SVM. The results are shown in Figure 4d). From the results, we can see that we are barely doing better than the yes hypothesis. This shows that the time series in these models add very little information.



a)



b)

	Logistic Regression	SVM	Logistic Regression	SVM
Estimated Generalization	12.4%	11.9%	26.3%	12.4%
Empirical Error	11.7%	12.9%	11.6%	31.7%
Yes Hypothesis	20.0%		13.8%	
	Time series split		Data file split	

c)

	Logistic Regression	SVM
Estimated Generalization	18.5%	18.8%
Empirical Error	17.1%	17.6%
Yes Hypothesis	20.0%	

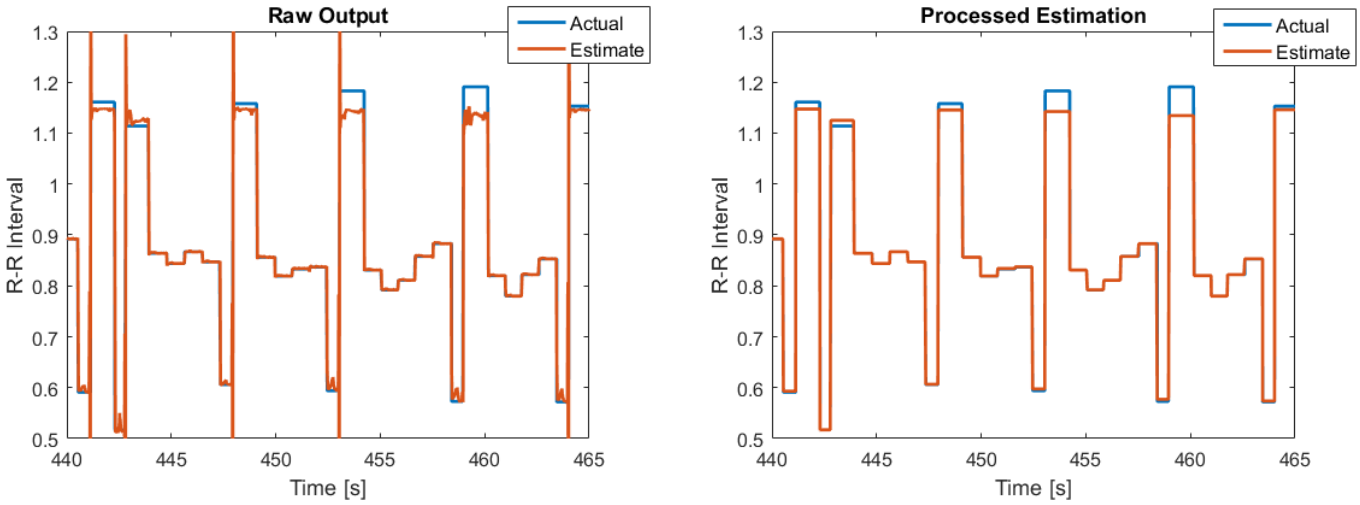
d)

Figure 4: Classification results. a) Error when training data is split by time series. b) Error when training data is split by data file. c) Tabulated asymptotic error results. d) Errors from using time series data as feature vector.

Softmax has an asymptotic generalization error of 33.5% and an empirical error of 30.3%. The empirical frequency of each bin is [10.6%, 26.2%, 35.2%, 27.9%]. These results are quite poor in this context and by examining the actual frequency of each bin, we see that almost 90% of the data are in three bins. Both errors are close to 33% so we are not performing much better than randomly assigning a label from one of these three bins. Such poor performance

from this algorithm given its n-gram input feature vectors is not unreasonable because we are almost asking for a true regression.

A representative output from the NARNN is shown in Figure 5a). Figure 5b) shows the output after processing and we see that the network performs quite well. There are some cases when the NARNN fails, usually when the R-R interval jumps to a large value. This might be a result of the dampening during training, trying to filter out noise. The output is a series of step functions by implementation and thus has very high frequency transitions. It makes sense that the NARNN may be filtering out these particularly large discontinuities. Mean L^2 error is 0.21 ms, mean L^1 error is 3.71 ms, and mean square error is 22.8 ms^2. [5] does not specify how it calculated its error but this project performs better by all common metrics.



a) b)

Figure 5: NARNN output. a) Raw NARNN output. b) Processed NARNN output.

Conclusion

The classification approaches did not perform that well. Only the division of data by time series produced results significantly better than the 'yes' hypothesis. In the context of this project, this is fine because they were more for exploration of the capabilities of the methods than for accurate prediction.

Nevertheless, there is still an interesting application of the classification approach in cardiac MRI. ECG is a complex setup that requires several wires and significant patient interaction. This can increase patient apprehension in an already stressful environment as well as require more time in the scanner, decreasing scanner utilization. An alternative approach is to use a plethysmograph as the cardiac trigger, which only involves a clip on the patient's index finger. The plethysmograph waveform is much simpler than an ECG and each peak is a constant delay behind an R-peak. Thus, although our feature vector is simple, it is applicable in this situation. A further extension could be to train a NARNN on the n-gram feature vectors or to repeat this project on the plethysmograph time series.

The NARNN has small errors and produces results that are accurate on the order of milliseconds. Our MRI acquisition sequence can be adjusted by units of 5 ms so the results of this project are accurate enough to make practical improvements in SNR efficiency. This allows us to reduce scan time and improve the patient experience with no tradeoff in image quality.

References

[1] Nishimura D.G., Principles of Magnetic Resonance Imaging. Stanford University. 2010.

[2] Wang H. and Amini A., "Cardiac motion and deformation recovery from MRI: a review." Medical Imaging, IEEE Transactions 2012:31(2):487-503.

[3] Bombardini et al., Diastolic time-frequency relation in the stress echo lab: filling timing and flow at different heart rates. Cardiovascular Ultrasound 2008;6(15).

[4] Singh K, Systolic and diastolic ratio and rate pressure product in anemia. Cardiology 2013;24(6):521-523.

[5] German-Sallo Z. and Calin C., "RR interval prediction in ECG signals." Electrical and Power Engineering (EPE), 2014 International Conference and Exposition on. IEEE, 2014.

[6] Goldberger et al., "Physiobank, physiotoolkit, and physionet components of a new research resource for complex physiologic signals." Circulation 2000:101(23):e215-e220.

[7] Luo J. et al., "Combined outer volume suppression and T2 preparation sequence for coronary angiography." Magnetic Resonance in Medicine 2014.

[8] Ng A.Y., CS229 Lecture Notes, 2012.

[9] Fan R.E. et al., LIBLINEAR: A library for large linear classification Journal of Machine Learning Research 2008:9:1871-1874.

[10] Dietz D-K. S., "Autoregressive Neural Network Processes." 2010.