Reducing False Arrhythmia Alarms in the Intensive Care Unit

Andrew Ward, Daniel Miller, and Katarina Miller
Department of Electrical Engineering
Stanford University, Stanford, CA 94305
Email: {atward, danielrm, kmiller3}@stanford.edu

***Abstract*—Due to entrenched methodology and a tendency towards deep focus in domain-specific knowledge, health care operations is a field with open problems, yet relatively little previous work in applying modern machine learning methods. In particular, alarm fatigue in hospital intensive care units is a major issue that could greatly benefit from the application of even relatively simple and standard learning techniques. Previous work explores only very specific instances of and solutions to this problem. This paper explores the application of several standard methods to the problem of reducing false alarm rates in the ICU and discusses the relative advantages and drawbacks to each model. We present promising results in false alarm suppression, and identify possible extensions that could lead to further improvement in this application. Future work will include fine tuning the models, and combining our methods with more domain-specific medical knowledge.**

I. INTRODUCTION

In hospital intensive care units, high rates of false arrhythmia alarms result in the fatigue and desensitization of attending medical staff. This "false alarm fatigue" causes workers' response times to slow, and leads to detrimental decreases in the quality of patient care [1]. Excessive alarms can also lead to sleep deprivation and depressed immune systems among patients [2]. We consider in particular the alarms triggered by the five life threatening conditions of asystole, extreme bradycardia, extreme tachycardia, ventricular tachycardia, and ventricular flutter/fibrillation. These alarms are triggered by ECG and pulsatile waveforms recorded by monitoring equipment. The standard alarm triggering criteria are instantaneous thresholds on the predictor values. False alarm rates of up to 86% have been reported [3] in pediatric intensive care units. Such high rates suggest the potential for significant improvements by applying modern methods to identify and suppress false alarms in real time. Towards this end, we seek to develop a model to correctly identify and distinguish true and false alarms reported in the ICU.

We hypothesize that applying statistical and machine learning techniques can achieve significant improvements in the specificity of the alarm system, while maintaining the true alarm sensitivity. The input to our model is a set of waveform traces taken from ICU patients who triggered an alarm via the standard methods in practice. These traces extend 5 minutes previous to the alarm trigger, and are labeled according to whether the alarm was true or false. We extract relevant features from the waveform according to the literature, and use said features as the inputs to our classification models. The four classification models considered in this paper are (1) logistic regression, (2) support vector machine, (3) binary random forest, and (4) multi-class random forest. The goal of these models is to accurately predict whether the provided features indicate a true or false alarm. The output of our models fills this requirement by reporting the sensitivity and specificity (false-alarm suppression rate) as the primary metrics for each classification model.

II. RELATED WORK

Cropp [4] played recordings of critical and non-critical alarms for a sample of medical professionals and found they could correctly identify critical alarms only half the time. They posit this is due to an over-proliferation of alarms in the ICU, resulting in medical professionals paying less attention to the alarms. This work partly inspired the PhysioNet Challenge, and our collaboration with the Lucile Packard Children's Hospital. Hagerman [5] found that the increase in alarms increased patient stress, resulting in higher patient pulse amplitude in some types of patients. These papers show the importance of implementing a higher standard for triggering alarms.

Aboukhalil [1] investigated reducing false alarms with a database of more than 2000 records. With the input of doctors and nurses, they decided on viable ranges for each of the features they used based on the last 16-second window of the monitors, and graphed the true alarm and false alarm rate as a function of each parameter changing, without using methods such as stochastic gradient descent. They set many features equal to zero for different alarms, and limited their search space significantly. We chose to investigate what our models would find without too much a priori input, to contrast their work. They observed a false alarm suppression rate of 95% on the asystole alarm, with no true alarm suppression, but saw lower results on the ventricular tachycardia alarm, with a 38.7% false alarm suppression rate at the cost of a 4% true alarm suppression rate. This approach was based on medical knowledge, and produced strong results, but we feel that using advanced machine learning methods may ultimately produce better results, especially when both techniques are combined.

Zong [6] also investigated how to reduce false alarms in the ICU. They assessed ABP signal quality, and if an ECG was available, adjusted the signal quality based on the ECG. They then adjusted the blood pressure data if the signal quality was above a threshold, and accepted or rejected the alarm based on this combination of data in the last 15-seconds of the monitors. They performed a thresholding model on this combination of data, manually tuning the thresholds. They eliminated nearly all false alarms on their test set, while suppressing only a few true alarms. They also did not use stochastic gradient descent but used their medical knowledge to decide on the algorithms and criteria for thresholding. We hope that our methods using machine learning algorithms will produce stronger results informed by the data.

Behar [7] analyzed QRS waves in the data and used Support Vector Machines with a Gaussian (nonlinear) kernel to classify the data. This work inspired us to investigate SVMs with nonlinear kernels. Their results are very impressive, but they, like us, see the least-beneficial results on ventricular flutter.

Li [8] analyzed ventricular flutter and ventricular tachycardia alarms using machine learning methods specifically for these alarms. They used signal processing to extract ventricular flutter rhythms from the data. They applied feature extraction using genetic learning to choose the features that minimized the RMSE of multivariate logistic regression. They then applied an SVM to classify each alarm. In contrast to their work, we did not use genetic learning, but we did apply regularized logistic regression to see how the algorithm worked with fewer features.

III. DATASET AND FEATURES

A. Source Data and Preprocessing

Our training set is from the 2015 PhysioNet/Computing in Cardiology Challenge. The dataset consists of records from 750 life-threatening arrhythmia alarms, sampled at random, from four different hospitals. For each alarm, there are 5-minute traces from up to five monitoring waveforms, including two ECG waveforms and at least one pulsatile waveform (photoplethysmogram (PPG or PLETH) and/or arterial blood pressure (ABP)). The data also includes a label indicating the alarm's True/False status.

These samples have been filtered and resampled to remove some noise, but still suffer from non-removable noise sources. For instance, the alarms were detected using four different software/hardware systems in each of the four hospitals, and system parameters were tuned on an individual patient basis. Neither of these factors are included in the provided dataset.

B. Feature Extraction

A key component of this project is extracting features from the time-series data traces. We cannot simply train our models on the entire time series; the points are highly correlated, and there would be too many features compared to our sample size, leading to a high probability of over-fitting the training data. Instead, we train our models on specific features of the time series that are indicative of the arrhythmia events the alarms are trying to detect. We determine such features both intuitively, and by referencing the literature.

The most important and relevant aspect of these traces is the heartbeat, so we focus on features related to heartbeats for all traces. To simplify the heartbeat extraction, we, like [1], only considered the 16 second window before the alarm was triggered (which will include any possible arrhythmias, since an alarm must trigger less than 10 seconds after an arrhythmia). For the ECG pulses, we segmented the data and, for each segment, calculated the average and standard deviation and looked for the R-waveform spikes that represented the heartbeats. Then, using these extracted heartbeats, we calculated all of the features shown in Table I. For the ABP and PPG (PLETH) waveforms, we again extracted the heartbeats by marking at what times the blood pressure rose quickly and significantly. We did this with the assistance of code provided by the PhysioNet Challenge [9]. Then, we used these heartbeat times to calculate the features shown in Table I for these traces. The signal-quality index was given by the provided code.

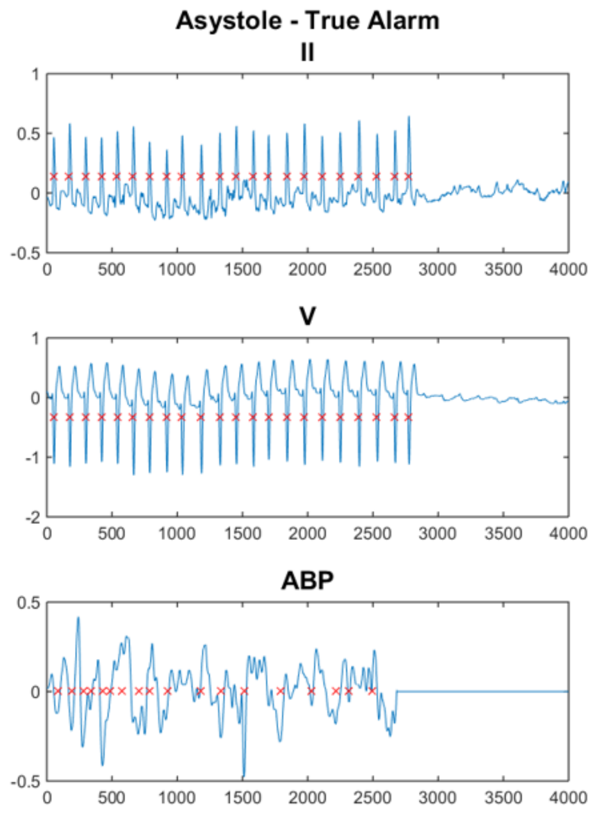

Fig. 1. An example of a patient with a true asystole alarm. The red X's represent the heartbeats extracted from the data, from which the features were created. This data is relatively clean; for the most part, the heartbeat positions agree across the ECG (II, V) and ABP traces.

An interesting challenge with this dataset was the asymmetry of the data. As seen in Figures 1 and 2, not all of the traces contained the same waveforms. So, when constructing our feature vectors, there were empty values for the features taken from nonexistent traces. Since the traces provided for a patient bore no indication of the true/false alarm status of that patient, we filled in these features by searching through the training set for traces that had similar features to the given patient. Once we found the closest match based on the features shared by both patients, we replaced the nonexistent trace data from the first patient from the existing trace data from the most similar patient.

While this method works for traces that are not provided, it does not help with provided traces for which it is impossible to extract a heartbeat. Consider Figure 2. This patient obviously has a heartbeat, as evidenced by the PPG trace; so, the

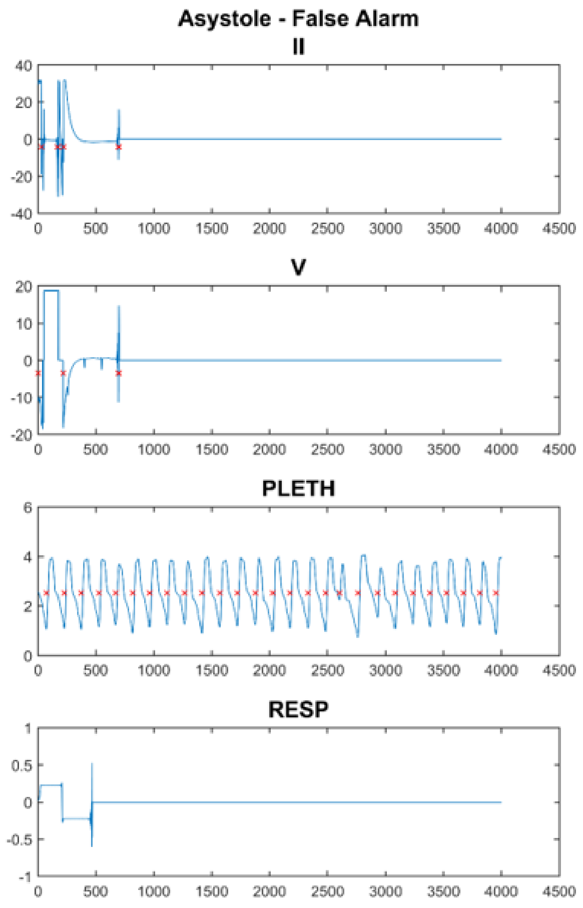

Fig. 2. An example of a patient with a false asystole alarm. The patient's heartbeat is visible on the PPG (PLETH) signal, but the ECG and RESP traces have zeroed for an unknown reason.

ECG traces should be treated as having not been provided. Ideally, we would have a normal, healthy patient with which to compare the PPG features, and replace the noisy ECG features with healthy ones. However, our dataset only provides us with traces that set off arrhythmia alarms; there are no healthy, non-noisy patient data. This was a limitation of the dataset provided by this project, but could be overcome in a real-world setting. In the literature, this limitation was overcome through data pre-processing, which looked for the presence of other waveforms if one waveform went to 0. We attempted to account for this later through different learning techniques.

IV. METHODS

A. Classification Models

We investigated four models to classify our data. We are motivated to try different models both to learn more about the data and because we may find that different alarms are classified more accurately with different models.

To apply the first three models, we first split our data by alarm type. We looked at the data that triggered each alarm separately with sets for asystole, extreme bradycardia,

Features	Traces Used
Low heart rate High heart rate averaged over 16 beats Max heart rate Highest voltage difference between two heartbeats Number of heartbeats in 16 seconds	PPG, ABP, ECG1, and ECG2
Signal-quality index	PPG and ABP

Table I. The 22 features extracted from the various traces.

extreme tachycardia, ventricular tachycardia, and ventricular flutter/fibrillation. All patients were analyzed only with other patients who triggered the same alarm. The response variable for each patient was 0 or 1, with 1 meaning the alarm was correct and the patient needed emergency medical attention and 0 meaning the alarm was false and immediate attention was not necessary. On these five datasets, we used 5-fold cross validation on the training set to tune the hyperparameters when necessary.

First, we ran a logistic regression model using the features described above. We chose this model to run first as it is simple to implement and provides a baseline of the performance of our features. We use the formula:

$$h_\theta(x) = \frac{1}{1+e^{-\theta^T x}} \quad (1)$$

Logistic regression uses gradient descent to maximize the likelihood function:

$$L(\theta) = \Pi_{i=1}^{m}(h+\theta(x^{(i)})^{y^{(i)}}(1-h_\theta(x^{(i)}))^{(y^{(i)})} \quad (2)$$

We also ran logistic regression with lasso regularization with tuning, which minimizes:

$$L(\theta) = \Pi_{i=1}^{m}(h+\theta(x^{(i)})^{y^{(i)}}(1-h_\theta(x^{(i)}))^{(y^{(i)})}+\beta||\theta||_1 \quad (3)$$

We chose this because it encourages different values of θ_i to go to 0, which was inspired by our literature review where papers such as [1] set different feature weights to 0 in their model. However, our feature removal is informed by the data.

Next, we ran an SVM model with a polynomial kernel whose degree was chosen through cross-validation. We chose this model because it runs quickly even with large feature vectors. The SVM model maximizes the geometric margin of the featrues extracted to a higher dimension with polynomial terms with a w vector, subject to the magnitude of w being 1. An SVM is generated by solving the following convex optimization problem:

$$\begin{array}{ll} \min & \frac{1}{2}||w||^2 \\ \text{s.t.} & y^{(i)}(w^T x^{(i)}+b) \geq 1, \quad i=1,...,m \end{array} \quad (4)$$

The SVM problem is usually solved as the Lagrange dual of the problem shown below, potentially allowing for a kernel replacement for increased computation speed.

$$\begin{aligned} \max_{\alpha} \quad & \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} y^{(i)} y^{(j)} \alpha_i \alpha_j \langle x^{(i)} x^{(j)} \rangle \\ \text{s.t.} \quad & \alpha_i \geq 0, \quad i = 1, ..., m \\ & \sum_{i=1} \alpha_i y^{(i)} = 0 \end{aligned} \tag{5}$$

If the higher-dimensional features are linearly separable, SVM will generate a w vector such that the data is perfectly separated. However, there are some cases in which this is undesirable. Sometimes, an outlier can skew the fit and create a line that will perform poorly on test data. Therefore, we used cross-validation to tune the SVM with regularization. This changes the optimization problem to as below, allowing the margin to be less than 1, if it minimizes the overall error.

$$\begin{aligned} \min \quad & \frac{1}{2} ||w||^2 + C \sum_{i=1}^{m} \xi_i \\ \text{s.t.} \quad & y^{(i)} (w^T x^{(i)} + b) \geq 1 - \xi_i, \quad i = 1, ..., m \\ & \xi_i \geq 0, \quad i = 1, ..., m \end{aligned} \tag{6}$$

We used 5-fold cross validation to tune the parameters C and ξ, as well as the degree of the polynomial kernel, using the score function of the competition as our metric for the tuning, shown in Equation 7. This score function penalizes false negatives much more harshly than false positives.

$$Score = \frac{TP + TN}{TP + TN + FP + 5 \cdot FN} \tag{7}$$

Third, we ran a boosted classification model. Boosted tree models work by generating many random samples of the data by sampling with replacement in a process called bagging. Then, a tree is trained on each sample of the data using a random sample of the features with replacement. The test data is then evaluated on each tree, and the result is averaged. This process reduces variance as the models are trained using different samples of the training sets. We used this model in order to see how a tree-based model compared to the other chosen models, and as a stepping stone to a random forest.

In order to further decrease variance, we limited the number of features and depth of tree used with a random forest model. This works especially well on datasets with highly correlated data, allowing the model to see how each feature effects the data. We also were able to evaluate which features were most important to the tree. Figure 3 shows the decrease in Gini impurity (a measure of inaccuracy, desired to be small) in the tree after each feature was split, added up among all the trees. We can see that, for the ventricular tachycardia alarm, the ECG1 high heart rate, PPG low heart rate, and ABP max RR diff resulted in the highest change in Gini impurity.

Last, we ran a multiclass boosted classification model on all of the data with the additional feature of which alarm

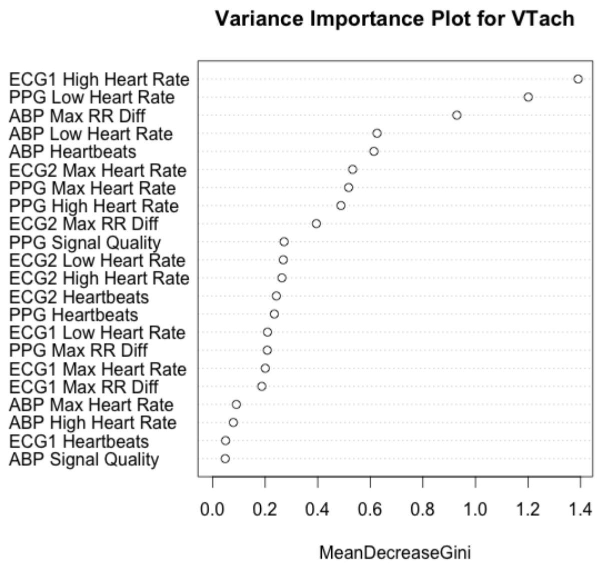

Fig. 3. A plot showing aggregate change in Gini coefficient for each feature.

was triggered. We used 10 categorical response variables. The numbers 1 through 5 each corresponded to a specific alarm. For the alarm that went off for a patient, the response variable is the corresponding number (1 through 5) multiplied by $+1$ if the alarm is correct and -1 if the alarm is incorrect. The multiclass random forest model allowed us to analyze if using the data from all patients is more informative than only using the data from patients with a specific alarm and see the most important features across the whole dataset.

V. EXPERIMENT / RESULTS / DISCUSSION

A. Preliminary Results: Basic Feature Set

Table II shows the number of traces present for each alarm type, and the baseline false alarm rates of our entire dataset. As shown, in most cases, the alarms were wrong. The ventricular flutter/fibrillation alarm performs the worst, with only one in ten alarms indicating a life-threatening event. We assume a baseline sensitivity of 1, i.e. that all life-threatening events are detected, resulting in an alarm.

Alarm Type	Samples	# True	# False	FA Rate
Asystole	122	22	100	0.8197
Bradycardia	89	46	43	0.4831
Tachycardia	140	131	9	0.0643
Ventricular Tachycardia	341	89	252	0.7390
Ventricular Flutter/Fibrillation	58	6	52	0.8966

Table II. Baseline false alarm rates for the provided PhysioNet Challenge data.

B. Numerical Results

For each model, we report the model sensitivity (true positive rate) and specificity (true negative rate). A high sensitivity indicates that true alarms are correctly identified as such (rather than being suppressed), and a high specificity corresponds to a reduced false alarm rate, or the *false alarm* *suppression rate*. Our goal is therefore to maximize specificity, while maintaining a perfect sensitivity of 1. We expected to achieve a significant decrease in false alarm rates by applying trained statistical models to this problem. This assumption has now been validated by the results given below in Tables III, IV, V and VI.

Alarm Type	Sensitivity	Specificity
Asystole	0.8444	0.58
Bradycardia	0.7476	0.6346
Tachycardia	0.3	0.8942
Ventricular Tachycardia	0.9394	0.0333
Ventricular Flutter/Fibrillation	0.8258	0.5667

Table III. Sensitivity and specificity for the split logistic model.

Alarm Type	Sensitivity	Specificity
Asystole	0.86	0.4
Bradycardia	0.6833	0.7378
Tachycardia	0.1	0.9085
Ventricular Tachycardia	0.9051	0.1588
Ventricular Flutter/Fibrillation	0.7855	0.1

Table IV. Sensitivity and specificity for the split SVM model.

Alarm Type	Sensitivity	Specificity
Asystole	0.96	0.37
Bradycardia	0.8611	0.8244
Tachycardia	0.1	1
Ventricular Tachycardia	0.9246	0.2346
Ventricular Flutter/Fibrillation	0.96	0

Table V. Sensitivity and specificity for the split random forest model. $mtry = \sqrt{p}$ features used in each split.

Alarm Type	Sensitivity	Specificity	Misclassification Rate
Asystole	0.5106	0.6619	0
Bradycardia	0.5968	0.8153	0.0077
Tachycardia	0.5556	0.7859	0.0222
Ventricular Tachycardia	0.5133	0.8120	0
Ventricular Flutter/Fib.	0.5593	0.7330	0.0222

Table VI. Sensitivity and specificity for the multiclass random forest model. The misclassification rate indicates whether each alarm was incorrectly identified as a different alarm. This is very low, and should ideally be 0, since the alarm type is used as a model feature.

C. Parameter Selection

Our initial results utilized only untuned models. By using cross-validation for parameter selection, we are able to achieve more balanced results. Figure 4 shows the results for a tuned vs. untuned SVM model. Our SVM model used a polynomial kernel with 3 parameters: cost, degree, and scale. These parameters were evaluated for each point on a 3-dimensional grid where $cost \in \{.01, .1, 1, 10, 100\}$, $degree \in \{1, 2, 3\}$, and $scale \in \{1\}$. For each set of parameter values, the 5-fold cross-validated PhysioNet score was computed according to Equation 7. The parameter combination yielding the largest cross-validated score was chosen.

Because of the asymmetry in the weighting of FN, tuning on the PhysioNet score has the effect of emphasizing, or ”caring about” sensitivity more than it does about specificity. This is basically a soft version of our goal to maximize specificity, while constrained to maintain sensitivity.

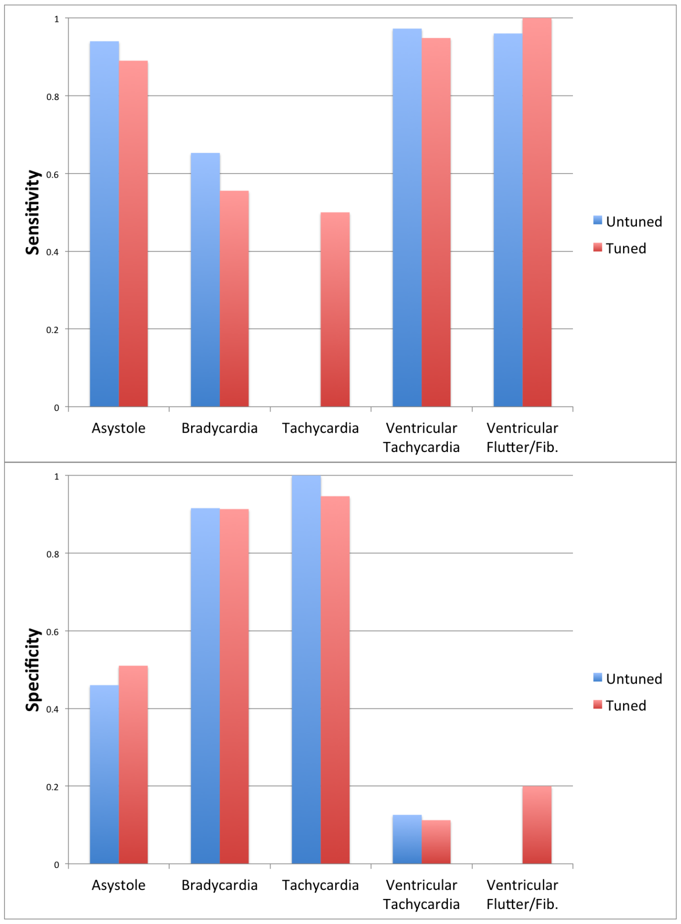

Fig. 4. Tuned vs Untuned SVM model sensitivity and specificity. Tuned by 5-fold cross validation on a grid search of polynomial kernel parameters, using the PhysioNet score, or a weighted metric of TP/TN as the tuning metric.

VI. CONCLUSION / FUTURE WORK

A. Discussion

Our final results show that significant improvements in false alarm suppression can be made as compared to the basic thresholding model used in hospitals. This project was limited by the quality of data we had available, as all data had triggered an alarm of some sort, so we had no ”control” data to work with. We were limited by the size of the dataset, particularly for alarms with fewer than 30 true positives. Split random forest achieved the highest sensitivities on most alarms, likely due to the correlation of features, and therefore should continue to be investigated. We are optimistic that, with data from a wider array of patients, the data can be pre-processed to produce even stronger results with all alarms. In addition, further work could be done with a larger dataset to tune the parameters to a sensitivity of 1, while maximizing the specificity.

B. Acknowledgements and Post-CS 229

This project will continue after CS 229 with David Scheinker, the Director of Systems Design and Collaborative Research at Stanford LPCH, who provided the inspiration for this project, and Professor Nicholas Bambos in the Stanford Management Science and Engineering department.

REFERENCES

[1] A. Aboukhalil, L. Nielsen, M. Saeed, R. G. Mark, and G. D. Clifford, "Reducing false alarm rates for critical arrhythmias using the arterial blood pressure waveform," *Journal of biomedical informatics*, vol. 41, no. 3, pp. 442–451, 2008.

[2] M.-C. Chambrin *et al.*, "Alarms in the intensive care unit: how can the number of false alarms be reduced?" *CRITICAL CARE-LONDON-*, vol. 5, no. 4, pp. 184–188, 2001.

[3] S. T. Lawless, "Crying wolf: false alarms in a pediatric intensive care unit." *Critical care medicine*, vol. 22, no. 6, pp. 981–985, 1994.

[4] A. J. Cropp, L. A. Woods, D. Raney, and D. L. Bredle, "Name that tone. the proliferation of alarms in the intensive care unit." *CHEST Journal*, vol. 105, no. 4, pp. 1217–1220, 1994.

[5] I. Hagerman, G. Rasmanis, V. Blomkvist, R. Ulrich, C. A. Eriksen, and T. Theorell, "Influence of intensive coronary care acoustics on the quality of care and physiological state of patients," *International journal of cardiology*, vol. 98, no. 2, pp. 267–270, 2005.

[6] W. Zong, G. Moody, and R. Mark, "Reduction of false arterial blood pressure alarms using signal quality assessement and relationships between the electrocardiogram and arterial blood pressure," *Medical and Biological Engineering and Computing*, vol. 42, no. 5, pp. 698–706, 2004.

[7] J. Behar, J. Oster, Q. Li, and G. D. Clifford, "Ecg signal quality during arrhythmia and its application to false alarm reduction," *Biomedical Engineering, IEEE Transactions on*, vol. 60, no. 6, pp. 1660–1666, 2013.

[8] Q. Li, C. Rajagopalan, and G. D. Clifford, "Ventricular fibrillation and tachycardia classification using a machine learning approach," *Biomedical Engineering, IEEE Transactions on*, vol. 61, no. 6, pp. 1607–1613, 2014.

[9] A. L. Goldberger, L. A. N. Amaral, L. Glass, J. M. Hausdorff, P. C. Ivanov, R. G. Mark, J. E. Mietus, G. B. Moody, C.-K. Peng, and H. E. Stanley, "PhysioBank, PhysioToolkit, and PhysioNet: Components of a new research resource for complex physiologic signals," *Circulation*, vol. 101, no. 23, pp. e215–e220, 2000 (June 13), circulation Electronic Pages: http://circ.ahajournals.org/cgi/content/full/101/23/e215 PMID:1085218; doi: 10.1161/01.CIR.101.23.e215.