Abnormal Data Classification Using Time-Frequenc Temporal Logic

Luan Viet Nguyen
University of Texas at Arlington, USA

James Kapinski
Toyota Technical Center

Xiaoqing Jin
Toyota Technical Center

Jyotirmoy V. Deshmukh
Toyota Technical Center

Ken Butts
Toyota Technical Center

Taylor T. Johnson
Vanderbilt University, USA

ABSTRACT

We present a technique to investigate abnormal behaviors of signals in both time and frequency domains using an extension of time-frequency logic that uses the continuous wavelet transform. Abnormal signal behaviors such as unexpected oscillations, called *hunting* behavior, can be challenging to capture in the time domain; however, these behaviors can be naturally captured in the time-frequency domain. We introduce the concept of parametric time-frequency logic and propose a parameter synthesis approach that can be used to classify hunting behavior. We perform a comparative analysis between the proposed algorithm, an approach based on support vector machines using linear classification, and a method that infers a signal temporal logic formula as a data classifier. We present experimental results based on data from a hydrogen fuel cell vehicle application and electrocardiogram data extracted from the MIT-BIH Arrhythmia Database.

1. INTRODUCTION

For the last decade, signal temporal logic (STL) [11] has been successfully extended and applied in many domains such as exploring requirements for closed-loop control systems [8], identifying oscillatory behaviors of biology systems [5], and formalizing and recognizing music melodies [7]. Recently, Kapinski et al. introduced a new signal library template for constructing formal requirements of automotive control applications using STL [10]. These requirements involve various control signal behaviors such as settling time, overshoot, and steady state errors. Although most of such control signal behaviors can be characterized in the time domain, some abnormal signal behaviors such as *hunting* (undesirable oscillations) or *spikes* (abrupt, momentary jumps in signal values) are challenging to capture without frequency information. In most practical control systems, hunting behaviors are considered undesirable, or at least not ideal, and care is taken to minimize or eliminate the behavior. In signal processing, hunting behavior can manifest around sharp transitions, as a result of compression artifacts; this occurs, for example, in image processing, resulting in ghostly bands near edges, or in audio compression, resulting in forward echo problems. In circuit design, a hunting behavior can be the unwanted oscillation of an output current or voltage, which may cause a significant rise in power consumption, temperature, electromagnetic radiation, or settling time [9]. Although some hunting behaviors can be defined loosely as an oscillation around a given average and can be well captured using STL, some modulated hunting signals are challenging to detect using only time domain information [10]. Because hunting signals relate to oscillatory properties, it is appropriate to investigate them using time-frequency analysis.

The first attempt to introduce a specification formalism for both time and frequency properties of a signal, called time-frequency logic (TFL), was proposed by Donzé and his collaborators [7]. There, a signal is preprocessed using a Short-Time Fourier Transform (STFT) [4] to generate a spectral signal that represents the evolution of the STFT coefficients at some particular frequency over time. The time-frequency predicates and arithmetic expressions constructed from this spectral signal are added into an STL formula to yield a TFL formula. TFL was originally applied to music, though it can be easily extended to other application domains. A key limitation of the approach using the STFT is the inherent trade-off required between resolution in the time domain and resolution in the frequency domain; it is difficult or impossible to obtain satisfactory resolution in both time and frequency using the STFT for the analysis. Such limitations can be overcome using the continuous wavelet transform (CWT).

In the following, we extend the notion of TFL by using the CWT to specify and check time-frequency properties of signals. We introduce the concept of parametric time-frequency logic (PTFL) and use it to perform parameter synthesis for the purpose of classifying hunting behavior. Previous efforts have focused on data classification of time-series signals using STL [2, 3, 8], but identifying some abnormal behaviors such as hunting requires both time and frequency information [10]. Moreover, existing classification methods require an extensive amount of data, and the inferred classifier is often difficult for engineers to interpret. In contrast, our proposed method using PTFL can efficiently classify abnormal behaviors with an interpretable data classifier and requires less data than existing techniques. We note that although the below presentation is focused on one behavior type, it is straightforward to extend the work to detect other abnormal behaviors such as noise, spikes, or

HSCC '17, April 18–20, 2017, Pittsburgh, PA, USA.

 ISBN 978-1-4503-4590-3/17/04…$15.00.
DOI: http://dx.doi.org/10.1145/3049797.3049809

other anomalous behavior, in the time-frequency domain. We evaluate the proposed algorithm by comparing the performance against two existing classification techniques: a traditional machine learning technique using a support vector machine with a linear kernel, and a method that infers STL formulae as data classifiers [3]. To perform the evaluation, we use data sets from two different domains, the automotive and medical domains.

2. TIME-FREQUENCY LOGIC USING CWT

Although many control system behaviors can be naturally characterized in the time domain, there are some signal behaviors, such as hunting and spikes, that are challenging to capture without frequency information. This is especially true for non-stationary signals whose frequency components vary over time; for this class of signals, it is essential to analyze the signal properties in the time-frequency domain. STFT is a popular transformation that has been widely used in time-frequency analysis [4]. Using STFT to perform time-frequency analysis, a signal is partitioned into small segments (each segment is assumed to be stationary) whose lengths are equal to the width of a chosen window function. The window function is used to modulate the signal to emphasize the time instant associated with each segment. Unfortunately, the STSF provides a fixed time-frequency resolution so that it is not effective for signals that need to be analyzed with different time-frequency resolutions [14]. Moreover, it is difficult to choose a proper window function with an appropriate size that not only provides both desirable time and frequency resolutions but also does not violate the stationarity condition [14]. To overcome the limitation of the STFT, we use the CWT to analyze a signal in the time-frequency domain.

2.1 Continuous Wavelet Transform

The CWT of a signal $x(t)$ is formally defined as follows:

$$Wf(\zeta,\tau) = \int_{-\infty}^{+\infty} x(t)\psi^*_{\zeta,\tau}(t), \tag{1}$$

where $\psi^*_{\zeta,\tau}(t)$ is the complex conjugation of a basic wavelet function $\psi_{\zeta,\tau}(t)$ which is derived from a mother-wavelet function $\psi(t)$. This function has zero average in the time domain, i.e. $\int_{-\infty}^{+\infty}\psi(t)dt = 0$. Furthermore, a basic wavelet function $\psi_{\zeta,\tau}(t)$ can be written as:

$$\psi_{\zeta,\tau}(t) = \frac{1}{\sqrt{\zeta}}\psi\left(\frac{t-\tau}{\zeta}\right), \tag{2}$$

where $\zeta \in \mathbb{R}_{>0}$ is a scale parameter representing the width of the basic wavelet function, $\tau \in \mathbb{R}$ is a translation factor representing the location of the basic wavelet function, and $\frac{1}{\sqrt{\zeta}}$ is the energy normalization across different scales. Thus, the CWT maps an original signal to a function of ζ and τ that provides both time and frequency information. Note that the scale factor is inversely proportional to the frequency of a signal [14]. The CWT in Equation 1 measures the similarity between a basic wavelet function and a signal. Indeed, if a signal $x(t)$ has a frequency component f corresponding to a particular scale ζ of a wavelet function $\psi_{\zeta,\tau}(t)$, then the portion of $x(t)$ at some particular time interval where f exists will be similar to $\psi_{\zeta,\tau}(t)$. As a result, the CWT coefficients of $x(t)$ corresponding to f will be relatively large over this time interval. Moreover, the time-frequency energy density of the CWT is equivalent to the square norm of the CWT coefficients:

$$P_W f(\zeta,\tau) = |Wf(\zeta,\tau)|^2. \tag{3}$$

Time-frequency resolution. In contrast to the STFT, the CWT can either dilate or compress the window size of the wavelet function, and translate it along the time axis. The Heisenberg box [12] is a range of times and frequencies that indicates the accuracy of a time-frequency transformation. Although the area of the Heisenberg box does not change, the time and frequency resolutions can be varied depending on the value of ζ. As a result, the CWT can analyze all frequency components within a signal by considering appropriate scales of the mother-wavelet function. For instance, the CWT can use the wavelet function with a short duration and low scale for analyzing high frequency components, and vice versa. This advantage of the CWT allows us to efficiently analyze a signal that includes abnormal behaviors such as spikes and hunting.

2.2 Time-Frequency Logic

TFL is an extension of STL that can be used to specify both time and frequency properties of a signal [7]. In TFL, a signal predicate is defined over the signal representing the evolution of the STFT coefficient at a particular frequency over time. Given a pair (f,τ) of frequency and time, the STFT of a signal $x(t)$ is obtained by:

$$S_{f,\tau} = \int_{-\infty}^{+\infty} x(t)\psi_L(t-\tau)e^{-2i\pi ft}dt\,, \tag{4}$$

where $\psi_L(t)$ is a window function. A *spectral signal* $y(t) = |S_{f,t}|^2$ is the projection of the spectrogram of $x(t)$ on a particular frequency f. Such a signal can be incorporated in TFL formulae to form some interesting time-frequency specifications. We can see that a TFL formula is actually an STL formula in which the signal predicate is defined over $y(t)$ instead of $x(t)$. TFL has been used to formalize and recognize music melodies, where time-frequency requirements are simply specified as $\varphi \triangleq |S_{f_p,t}|^2 > \theta$, where f_p is the pitch frequency and θ is the STFT coefficient threshold [7]; however, the shortcomings of the STFT mentioned previously may reduce the ability of TFL to precisely specify and evaluate time-frequency properties of a signal. We extend TFL to use the CWT to obtain spectral signals from a given time-series signal. In effect, we construct a TFL formula based on the CWT coefficients of the spectral signals instead of the STFT coefficients. Because the CWT can appropriately use various scaling factors, ζ, to analyze all frequency components at different time intervals, it gives us an ability to study signals at flexible time-frequency resolutions.

Although the following presentation focuses on the classification of hunting behaviors, we note that the proposed approach using TFL and CWT can be used to capture other time-frequency specifications as well. For instance, consider the property: "*For some time in the future, the dominant frequency of the signal is ω for 5 time units, and the dominant frequency subsequently rises to twice of this value within 10-time units.*" Here, the *dominant frequency*, $f(t)$, of a signal $x(t)$ is defined as the frequency corresponding to the maximum magnitude frequency component of the signal at time t, as provided by a CWT. Such a time-frequency property can be written as a TFL formula, $\varphi \triangleq \Diamond(\Box_{[0,5]}(f = \omega) \wedge \Diamond_{[5,15]}(f = 2\omega))$. Then, the TFL formula φ can be evaluated as a normal STL formula using Breach[1] [6]. Consider another property such as "*At some time in the future the energy densities of the signal within a particular time inter-*

[1]Breach [6] is a tool that allows evaluation of STL and TFL formulae on signals.

val and a particular frequency bandwidth are always greater than some threshold value θ." This property can be specified as a TFL formula, $\phi \triangleq \Diamond\Box_{[t_1,t_2]}(z(f,t) > \theta)$, where $z(f,t)$ is a spectral signal that captures the minimum value of the CWT coefficients of a signal over some frequency bandwidth $[f_1, f_2]$.

Parametric Time-Frequency Logic. We introduce parametric time-frequency logic (PTFL), which is an extension of TFL where the parameters in TFL template formulae are symbolic parameters. Similar to the concept of parameter signal temporal logic (PSTL) introduced in [1], PTFL allows constants in intervals bounding the temporal operators and constant values in the predicates of PTFL formulae to be replaced with parameters.

The p parameters in a PTFL formula are classified into two sets:

(a) $\Upsilon = \{\tau_1, ..., \tau_{p_t}\}$ is a set of p_t *time* parameters occurring in the time intervals of the temporal operators, and

(b) $\Theta = \{\theta_1, ..., \theta_{p-p_t}\}$ is a set of $p - p_t$ *threshold* parameters occurring in the signal predicates.

For any fixed values of Υ and Θ, a PTFL formula $\varphi(\tau_1, \ldots, \tau_{p_t}, \theta_1, \ldots, \theta_{p-p_t})$ yields a TFL formula corresponding to the fixed values of the parameters. For instance, consider a PTFL formula $\varphi(\tau, \theta) \triangleq \Box_{[0,\tau]}(y(t) > \theta)$, where $y(t)$ is a spectral signal, τ and θ are time and threshold parameters, respectively. The formula $\varphi(5, 10)$ is defined as the TFL formula $\Box_{[0,5]}(y(t) > 10)$.

3. HUNTING CLASSIFICATION

In this section, we will describe three different approaches using PTFL and TFL to efficiently classify hunting behaviors in signals. Informally, a hunting behavior is an undesirable oscillation appearing within a signal over some time interval.

3.1 Parameter Synthesis Approach

We now propose a method to classify hunting behavior based on mining parameters of the following PTFL formula:

$$\varphi_h \triangleq \bigwedge_{i=1}^{m} \Diamond_{[0,\tau_i]}(Wf_i(t) > \theta_i). \tag{5}$$

Intuitively, this formula specifies that "*the energy densities of the given signal at particular frequencies are eventually greater than some threshold value*". Here, $Wf_i(t)$ is a spectral signal over time that captures the energy densities of the CWT of an original time-series signal $x(t)$ at a particular frequency $f_i \in F$. Note that F is a set of frequencies based on the scales of the CWT. Each spectral signal, $Wf_i(t)$, is the row vector of the matrix representing the energy densities of the CWT of $x(t)$; such a matrix is obtained using Equation 1 and Equation 3. Also, $\tau_i \in \Upsilon$ and $\theta_i \in \Theta$ denote a time and threshold parameter corresponding to each spectral signal $Wf_i(t)$. We note that the satisfaction value of the property φ_h monotonically increases in τ_i and decreases in θ_i. Because of monotonicity, we can exponentially reduce the search over the parameter space so that the synthesis procedure is efficient [8]. Figure 1 conceptually illustrates a spectral signal $Wf_i(t)$, and an instance of a hunting behavior that may occur within a signal. We say that a signal $x(t)$ contains hunting behavior if the property φ_h holds. Overall, the hunting classification problem can be written as follows.

- **Given** the following inputs:

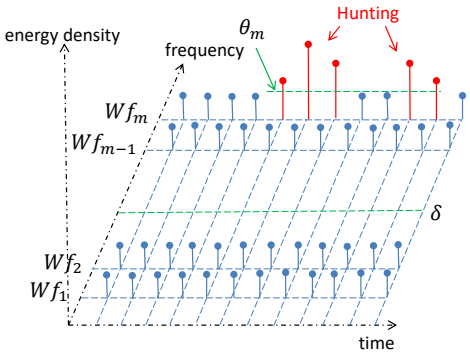

Figure 1: A sketch illustrates the hunting classification problem using time-frequency parameter synthesis. The set of spectral signals Wf_i is acquired from the CWT of an original time-series signal.

  - ☐ a set of labeled traces $\Psi \triangleq \{\Psi_\alpha, \Psi_\beta\}$, where Ψ_α and Ψ_β denote a set of training and testing traces, respectively. Moreover, we the notation $\Psi.B$ and $\Psi.G$ to respectively denote the set of traces with and without hunting behavior. Note that all traces in the training set exhibit hunting behavior, so that $\Psi_\alpha = \Psi_\alpha.B$
  - ☐ a cut-off frequency δ.
  - ☐ sets of parameters Υ, and Θ.
- **Find** values for Υ and Θ, such that:
  - ☐ $x_j(t) \models \varphi_h(\Upsilon, \Theta)$ for all $x_j(t) \in \Psi_\beta.B$.
  - ☐ $x_j(t) \not\models \varphi_h(\Upsilon, \Theta)$ for all $x_j(t) \in \Psi_\beta.G$.

We introduce the cut-off frequency δ to reduce the effort to exhaustively mine parameters over the entire time-frequency domain. It is essential for the control engineers to indicate that hunting behavior only occurs at some high-frequency region above δ.

Classification Algorithm. Next, we propose a heuristic to automatically obtain values for Υ and Θ that can be used to separate the hunting and non-hunting signals. An overview of the heuristic is described in Algorithm 1. The heuristic can be interpreted as follows.

Line 2 initializes a matrix Σ that represents the k m-dimensional spectral signals transformed from k original time-series signals in the training set using the CWT. We iterate over each trace in Ψ_α to construct sets of spectral signals $\{Wf_1(t), ..., Wf_m(t)\}$ using the CWT, and assign them to Σ. Next, we call the function TruncateParam to reduce the effort of exhaustively mining all parameters over the entire time-frequency domain. Here, Σ' represents the k n-dimensional $(n < m)$ matrix of Σ corresponding to the frequency range above δ. Next, we call the function HuntingParamSyn incorporated inside Breach to mine values for Υ and Θ. Then, we test the classifier with a given set of testing traces Ψ_β. The function Classifier checks the satisfaction of φ_h for each trace in Ψ_β, and returns the misclassification rate (MCR) value and the set of misclassified traces Ψ_m. The values of Υ, Θ and the set Ψ_m are then returned for further analysis. Furthermore, we can call EnhancedParam function to strengthen the values Υ and Θ and reduce the MCR value for the purpose of optimizing the classifier formula. Note that in the case studies, we do not use this function to evaluate the performance of the classifier to avoid the bias in our comparative analysis.

3.2 Decision Tree Approach

An approach based on decision trees to classify time series data using STL formulae was implemented in the tool

Algorithm 1 Hunting Classification Using Parameter Synthesis

```
1   function HuntingClassification(Ψ_α, Ψ_β, δ)
      Σ ← 0
3     for each trace x_j(t) ∈ Ψ_α, j ≤ k
        Σ(j,:,:) ← Wf_1(t), ..., Wf_m(t) ← CWT(x_j(t))
5     end for
      Σ' ← TruncateParam(δ, Σ)
7     Υ, Θ ← HuntingParamSyn(Σ')
      MCR, Ψ_m ← Classifier(Υ, Θ, Ψ_β)
9     return Υ, Θ, Ψ_m
    end function
11  function EnhancedParam(Ψ_m, Ψ_α, Ψ_β, δ)
      if Ψ_m.B ≠ ∅ then
13      Ψ'_α ← Ψ_α ∪ Ψ_m.B
        HuntingClassification(Ψ'_α, Ψ_β, δ)
15    end if
    end function
```

DT4STL [3]. That method uses a parameterized procedure to infer STL formulae from labeled data. Given a two-class training data and a set of PSTL templates, a decision tree for classification is recursively built such that each node of a tree is associated with a simple formula, selected from the given PSTL templates. The parameter synthesis is then conducted to find the STL formula that yields the best split for the data at each node. This technique can be used to automatically construct classifiers based on STL formula, but to achieve a low MCR value, the inferred STL formulae may be long and not easily interpretable by engineers. In this section, we apply this approach to classify hunting versus non-hunting signals. Instead of inferring an STL formula, we intend to infer a TFL formula as a data classifier. Thus, we transform original time series data into a collection of time-frequency data (spectral signals).

We assume that control engineers initially designate the frequency threshold separating hunting versus non-hunting behavior. A hunting behavior is specified as any oscillatory behavior occurring at frequencies above some specified cut-off frequency δ. Thus, the time-frequency profile of a hunting signal at some frequency component $f > \delta$ contains larger values for the CWT coefficients compared to those of non-hunting signals. So we define the spectral signal WThcoef based on the CWT coefficients of the signal in a high-frequency region such that:

$$\mathsf{WThcoef}(t) = \max_{\zeta \in [\frac{f_c}{T_s F_{max}}, \frac{f_c}{T_s \delta})} P_W f(\zeta, t), \tag{6}$$

where f_c is a center frequency associated with the mother-wavelet function, F_{max} is the maximum frequency that appears in the CWT, and T_s is the sampling period. We use such a spectral signal as an input for the DT4STL to infer a simple TFL formula. Note that in this scenario, the inferred TFL formula captures the non-hunting behavior of a signal.

3.3 Support Vector Machine Approach

Next, we present another approach that can solve the problem of hunting classification: linear classification using support vector machines (SVM) [15]. A linear SVM is a set of hyperplanes or decision boundaries that can correctly separate data into two classes. The general form of hyperplanes is $\langle w \cdot x \rangle + b = 0$, where w is a normal to the hyperplane, and $\frac{b}{||w||}$ is the perpendicular distance from the hyperplane to the origin. The sign of the linear discriminant function $f(x) \triangleq \langle w \cdot x \rangle + b$ determines on which side of the decision boundary the test data point is located. The distance from the decision boundary to the closest data point determines the *margin* of the linear classifier. Suppose that we have a set of n labeled training data $(x_i, c_i), ..., (x_n, c_n)$ where $x_i \in \mathbb{R}^d$ and $c_i \in \{1, -1\}$, the constrained optimization problem of linear classification using SVM is written as:

$$\begin{aligned} \underset{w,b}{\text{minimize}} \quad & \frac{1}{2}||w||^2 + C\sum_{i=1}^{n} \zeta_i \\ \text{subject to} \quad & c_i(\langle w \cdot x_i \rangle + b) \geq 1 - \zeta_i,\ i = 1, \ldots, n \\ & \zeta_i \geq 0. \end{aligned} \tag{7}$$

Here, ζ is a slack variable. If $0 < \zeta \leq 1$, the data point lies somewhere between the margin and the correct side of hyperplane, and the data point is misclassified if $\zeta > 1$. C is a regularization parameter that defines the trade-off between errors of the SVM on training data and margin maximization. A large value of C results in the low possibility of misclassified training data points, because the optimization in Equation 7 will choose a narrow margin hyperplane that correctly separates training data points as much as possible. In contrast, a small value of C will result in a large margin hyperplane, but it may yield a better result in terms of correctly separating testing data points. Due to space limitation, we will not discuss the formal optimization problem solved to obtain the SVM, but refer interested readers to [15]. In this work, instead of applying the linear SVM directly to original time series signals, we need to preprocess them to yield a corresponding set of time-frequency features. For each time-series signal $x(t)$, we collect a real-valued vector $W^{max} \triangleq [Wf_1^{max}, ..., Wf_m^{max}]$ such that each element $Wf_i^{max} \in W^{max}$ is the maximum value of a spectral signal $Wf_i(t)$. Such a vector will be used as a time-frequency feature to design the SVM.

4. CASE STUDIES

In this section, we evaluate the capabilities of three different methods to classify hunting behavior for two case studies. The first case study is based on data from an air compressor motor speed (ACMS) system in a fuel cell (FC) vehicle application. The second case study is based on electrocardiogram (ECG) data. In both examples, we apply the Morlet CWT [12] to the time-series signals.

4.1 ACMS Data

The ACMS system uses a compressor to regulate the air intake of a hydrogen FC vehicle. An FC stack uses a mixture of air and hydrogen to generate electrical power for the vehicle. Accurate control of the compressor which translates to control of the quantities of hydrogen and oxygen (air) is required to achieve good performance and proper operation from the FC stack. Also, the water balance (moisture level) within the stack needs to be carefully regulated, which requires regulation of the air pressure at the inlet of the stack. The task of the ACMS system is to regulate air flow and air pressure delivered to the inlet of the FC stack.

We consider ACMS data from an FC vehicle application. Specifics of the data, such as units and descriptions of the measured quantities are omitted here for proprietary reasons. The ACMS data are partitioned into a collection of traces that are 100 seconds in length and are labeled as either good (the trace does not exhibit hunting behavior) or bad (the trace does exhibit hunting behavior). The ACMS

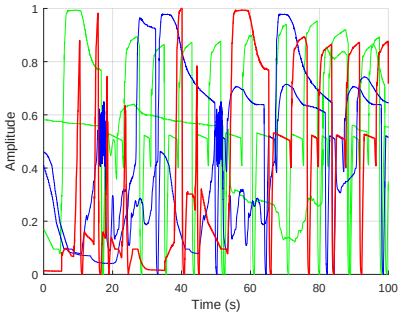

Figure 2: The classified testing data of the ACMS signals using parameter synthesis approach.

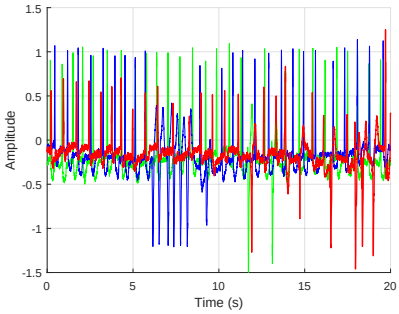

Figure 3: The classified testing data of the ECG signals using parameter synthesis approach.

data has a sampling period of 0.02 seconds. We note that the same training data is used for all of the evaluations, though the parameter synthesis approach only uses the bad traces. In this experiment, we use the training data including 50 total traces, in which 30 traces are labeled as good and the others are labeled as bad. We also use the same testing data including 10 good traces and 10 bad traces for all of the evaluations.

Parameter Synthesis. We now illustrate the performance of the classification heuristic shown in Algorithm 1 to classify hunting behavior for the ACMS signals. Because we do not know the frequency range where a hunting behavior may occur, we exhaustively mine all parameters $\tau_i \in \Upsilon$ and $\theta_i \in \Theta$. We choose the maximum frequency of the CWT as $F_{max} = 25$Hz. Here, the Algorithm 1 will search for the best $\theta_i \in [0, 1]$ and $\tau_i \in [0, 100]$ such that all spectral signals transformed from original time-series traces in the training data satisfy φ_h. We then use Breach with the optimized parameters of φ_h to classify good versus bad traces in the testing set.

Figure 2 shows the experimental results of classifying abnormal ACMS signals, using the function HuntingClassification. In the figure, we only show five representative signals in which good traces correctly classified are shown in green, and bad traces correctly classified are shown in blue. The one good trace that is misclassified is shown in red. The total running time of the classification process is approximately 3 minutes.

Decision Tree Approach. Next, we utilize the DT4STL toolbox to infer TFL formulae that can be used to classify hunting behavior for the ACMS data.

We preprocess the training data to yield the corresponding set of spectral signals WThcoef with $\delta = 15$Hz and $F_{max} = 25$Hz. We then run the DT4STL toolbox with this set of spectral signals using 2-fold cross-validation. As a result, we obtain the two following TFL formulae:

$$\varphi_{h1} \triangleq \square_{[37.4,98.2)}(\mathsf{WThcoef} < 0.0435)$$
$$\varphi_{h2} \triangleq \square_{[1.29,91.3)}(\mathsf{WThcoef} < 0.0394).$$

The procedure takes approximately 75 seconds to infer each formula. Using Breach, we then evaluate those formulae with the set of testing data. The formula φ_{h1} gives us all misclassified traces that are bad traces with the MCR value being equal to 25%. On the other hand, the formula φ_{h2} results in one misclassified trace, which is a bad trace.

SVM Approach. We apply the SVM method to classify normal versus abnormal ACMS data. We first transform all of the traces in the training data into sets of time-frequency features. Next, we run the linear SVM to learn the decision boundaries that separate data as either good or bad. Finally, we predict the testing data from the learned decision boundaries with different values of the SVM classifier margin C.

The MCR of the hunting classification for the ACMS data using SVM is 10% with $C = 10$ and reduces to 5% with $C = 100$. In this case, a larger value of C gives a better result for the classification. Moreover, the classification process takes only 0.393 seconds.

4.2 ECG Data

An electrocardiogram (ECG) test is a noninvasive procedure used to monitor the electrical activities of a heart via a collection of electrodes attached to the patient's skin. A doctor can read an ECG output signal to diagnose abnormal structure or function of the patient's heart. A normal ECG signal includes three signals: (a) the P wave representing the depolarization or contraction of the atrium (b) the QRS complex (the R wave) indicating the ventricular depolarization and (c) the T wave describing the ventricular repolarization. The distance between two consecutive R peaks is considered as a heartbeat. A healthy patient has a resting normal heartbeat (frequency) from 60 to 100 beats per minute (bpm).

In this paper, we focus on classifying the ECG signal that may contain a ventricular tachycardia (VT), a very fast heart rhythm arising in the ventricles that may cause a sudden heart failure. VT is defined as a sequence of three or more ventricular beats with the frequency varying from 110 to 250 bpm. Thus, a VT can be considered as a hunting behavior in an ECG signal. We conduct our classification approaches on the MIT-BIH Arrhythmia ECG Database. These data contain a variety of ECG signals collected from patients 23 to 89 years of age, including patients who experience ventricular arrhythmia [13]. We transform ECG signals 20 seconds in duration (provided at a sampling period of 0.0028 secs.) to spectral signals using the Morlet CWT. Here, the maximum frequency of the CWT is $F_{max} = 4.5$Hz ($\sim$ 270 bpm). For all of the evaluations, we use the same training data including 20 bad traces (the traces do contain a VT) and 40 good traces (the traces do not contain a VT), and the same testing data including 10 good traces and 10 bad traces.

Parameter Synthesis. In this scenario, we only mine the parameters for 20 bad traces in the training dataset. Here, we will search for the best $\theta_i \in [0, 5]$ and $\tau_i \in [0, 20]$. Figure 3 shows the experimental results of using the function HuntingClassification to classify abnormal ECG signals that contain VT. Here, we only show three signals for illustration. The approach results in one (5%) misclassified (red)

	PS	DT4STL	SVM
Interpretation of data classifier	◯	△	×
Computation time	×	×	◯
Bad behavior localization	◯	◯	×
Low misclassification rate	△	△	◯

Table 1: The comparison between parameter synthesis (PS) using PTFL, DT4STL toolbox using TFL, and linear SVM in classifying abnormal signals, where ◯, △, × respectively denote good, ok, bad.

trace, which is a bad trace. The total running time of the classification process is approximately 1 minute.

Decision Tree Approach. Next, we utilize the DT4STL toolbox to classify hunting behavior for the ECG data. We first preprocess the training data to yield the corresponding set of spectral signals WThcoef with $\delta = 1.5$Hz. Then, we run the DT4STL toolbox with this set of spectral signals using 2-fold cross-validation. As a result, we obtain two following TFL formulae:

$$\phi_{h1} \triangleq \Box_{[1.73,17.3)}(\mathsf{WThcoef} < 3.16)$$
$$\phi_{h2} \triangleq \Box_{[2.36,20)}(\mathsf{WThcoef} < 3.21).$$

The procedure takes approximately 105 seconds to infer each formula. We then use Breach to evaluate these formulae with a set of spectral data acquired from the CWT of 10 good traces and 10 bad traces in the testing data. The MCR values of using ϕ_{h1} and ϕ_{h2} to classify these data are both equal to 5% (but misclassified traces are different).

SVM Approach. Finally, we apply the SVM approach to classify hunting in the ECG data. Note that we use the same training and testing data used for the other methods. The hunting classification of the ECG data using an SVM results in a 5% MCR for all values of C (the one misclassified trace is a bad trace), and the classification procedure takes 0.3 seconds.

5. DISCUSSION

In this section, we discuss the trade-offs related to the three classification approaches presented above to classify normal versus abnormal signals. Table 1 shows an aggregate performance evaluation between the approaches in four different categories, including (a) the ability to interpret the structure and parameters used to define the classifier, (b) the computation time, (c) the capacity to localize where bad behavior occurs in a signal, and (d) the ability to correctly classify normal versus abnormal signals. Although the linear SVM can classify abnormal signals much faster and more accurately than the parameter synthesis and the decision tree approaches, the main drawback of this method is that it cannot reveal where the bad behavior occurs within a signal. We found that the decision tree approach can infer specifications that accurately classify data as either good or bad; however, it is not easy to interpret the inferred formula unless the user has some expertise about the input data. If a dataset is not homogeneous (i.e., both normal and abnormal signals are very different from each other), the DT4STL toolbox may infer a complicated formula that cannot be easily interpreted. The parameter synthesis using PTFL and the decision tree approach using TFL have similar performance except the former provides a clearer intuition about the classifier, as the temporal logic formula that results is usually simpler for the PTFL case. Overall, we conclude that a traditional machine learning technique such as the linear SVM is the best choice if the only goal is to classify data as either good or bad, and the most important thing is to select a proper feature on which to base the classification algorithm. Otherwise, if the designer additionally wishes to both understand the meaning of a data classifier and automatically localize where abnormal behaviors occur within a signal, we conclude that the parameter synthesis approach is the best option, as a simple temporal logic formula that defines the classifier results from the analysis.

6. ACKNOWLEDGMENTS

The material presented in this paper is based upon work supported by the National Science Foundation (NSF) under grant numbers CNS 1464311, EPCN 1509804, and SHF 1527398, the Air Force Research Laboratory (AFRL) through contract number FA8750-15-1-0105, and the Air Force Office of Scientific Research (AFOSR) under contract numbers FA9550-15-1-0258 and FA9550-16-1-0246. The U.S. government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of AFRL, AFOSR, or NSF. The authors sincerely appreciate Jared Farnsworth and the Fuel Cell group at Toyota Technical Center for their help in obtaining and understanding the ACMS data.

7. REFERENCES

[1] E. Asarin, A. Donzé, O. Maler, and D. Nickovic. Parametric identification of temporal properties. In *Runtime Verification*, pages 147–160. Springer, 2011.

[2] E. Bartocci, L. Bortolussi, and G. Sanguinetti. Data-driven statistical learning of temporal logic properties. In *International Conference on Formal Modeling and Analysis of Timed Systems*, pages 23–37. Springer, 2014.

[3] G. Bombara, C.-I. Vasile, F. Penedo, H. Yasuoka, and C. Belta. A decision tree approach to data classification using signal temporal logic. In *Proceedings of the 19th international conference on Hybrid systems: computation and control*. ACM, 2016.

[4] L. Cohen. *Time-frequency analysis*, volume 299. Prentice hall, 1995.

[5] P. Dluhoš, L. Brim, and D. Šafránek. On expressing and monitoring oscillatory dynamics. *arXiv preprint arXiv:1208.3853*, 2012.

[6] A. Donzé. Breach, a toolbox for verification and parameter synthesis of hybrid systems. In *Computer Aided Verification*, pages 167–170. Springer, 2010.

[7] A. Donzé, O. Maler, E. Bartocci, D. Nickovic, R. Grosu, and S. Smolka. On temporal logic and signal processing. In *Automated Technology for Verification and Analysis*, pages 92–106. Springer, 2012.

[8] X. Jin, A. Donzé, J. V. Deshmukh, and S. A. Seshia. Mining requirements from closed-loop control models. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 34(11):1704–1717, 2015.

[9] H. W. Johnson, M. Graham, et al. *High-speed digital design: a handbook of black magic*, volume 1. Prentice Hall Upper Saddle River, NJ, 1993.

[10] J. Kapinski, X. Jin, J. Deshmukh, A. Donze, T. Yamaguchi, H. Ito, T. Kaga, S. Kobuna, and S. Seshia. ST-Lib: A library for specifying and classifying model behaviors. 2016.

[11] O. Maler and D. Nickovic. Monitoring temporal properties of continuous signals. In *Formal Techniques, Modelling and Analysis of Timed and Fault-Tolerant Systems*, pages 152–166. Springer, 2004.

[12] S. Mallat. *A wavelet tour of signal processing*. Academic press, 1999.

[13] G. B. Moody and R. G. Mark. The impact of the mit-bih arrhythmia database. *IEEE Engineering in Medicine and Biology Magazine*, 20(3):45–50, 2001.

[14] R. Polikar. The wavelet tutorial. 1996.

[15] B. Scholkopf and A. J. Smola. *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press, 2001.