Case Studies in Data-Driven Verification of Dynamical Systems

Case Studies in Data-Driven Verification of Dynamical Systems*

Alexandar Kozarev
Metron, Inc.
San Diego, CA
kozareva@ca.metsci.com

John Quindlen and Jonathan How
Aerospace Controls Lab
MIT, Cambridge, MA
{quindlen,jhow}@mit.edu

Ufuk Topcu
Dept. of Aerospace Eng.
Univ of Texas, Austin, TX
utopcu@utexas.edu

ABSTRACT

We interpret several dynamical system verification questions, e.g., region of attraction and reachability analyses, as data classification problems. We discuss some of the tradeoffs between conventional optimization-based certificate constructions with certainty in the outcomes and this new data-driven approach with quantified confidence in the outcomes. The new methodology is aligned with emerging computing paradigms and has the potential to extend systematic verification to systems that do not necessarily admit closed-form models from certain specialized families. We demonstrate its effectiveness on a collection of both conventional and unconventional case studies including model reference adaptive control systems, nonlinear aircraft models, and reinforcement learning problems.

1. INTRODUCTION

Certification of safety-critical systems is the procedure of demonstrating compliance with often a large collection of criteria to an authority. For example, MilSpecs [1] and DO-178 [2] provide guidelines for flight control applications. While linear stability, performance and robustness metrics have been regarded as satisfactory certification criteria in these guidelines for relatively conventional applications, increasingly adaptable and autonomous systems necessitate new quantitative criteria.

Consider an example in which the system operates at a nominal condition x_{nom} (e.g., a trim condition of an aircraft) in a state space $X \subseteq \mathbb{R}^n$. For simplicity, assume that the system is stationary at this nominal condition x_{nom}. An analyst, who has identified a set G of perturbations to which the system may be subject while operating at x_{nom}, *queries* whether the system can recover (i.e., returns to x_{nom}) from all possible perturbed configurations in G. Given a mathematical model of the system, a number of methods seek answers to similar analysis queries. We consider two of them: extensive simulations (or testing) and deductive methods.

Extensive simulations emanating from initial conditions sampled from G constitute a *data-driven* approach which is widely applicable as long as there exist a simulation model and an oracle that labels the sampled initial conditions as "yes, the nominal condition is recovered from the sampled initial condition" or "no, it is not." Such simulations are informative yet unable to cover all possible perturbations.

If a model is given in a closed-form as differential (or difference) equations, then it may be possible to construct certificates that witness affirmative answers to the queries. Examples of such certificates include the so-called Lyapunov, barrier and storage functions. For example and quite roughly speaking, consider a smooth and positive definite function V that decreases along all trajectories—except for the stationary trajectory at x_{nom}—emanating from the set Ω in which V is less than or equal to 1 and attains its minimum value at x_{nom} [3]. If such a V exists, then an affirmative answer to the above query can be predicted by checking whether $G \subseteq \Omega$. Essentially, the boundary of the set Ω *separates* the states in X into two categories: the initial conditions from which the system is known to recover to x_{nom} and those initial conditions from which the system is *not known* to recover. Equipped with a certificate V and the corresponding set Ω, responding to the above query boils down to *predicting* to which side of the boundary of Ω the set G belongs.

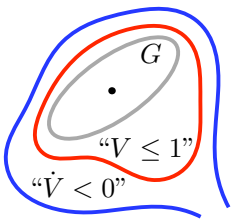

Figure 1: Schematic of a Lyapunov-type proof of G is contained in Ω.

While existence of certificates provides unambiguous answers to system analysis queries, their applicability is limited by whether such certificates can be automatically constructed for given system models. While system models expressed as differential equations that are linear or polynomial in the state variables—though only in modestly sized state spaces—are amenable to automated construction [4, 5, 6], deductive methods cannot be directly applied to a broad range of practically interesting models utilized in simulation-based analysis.

We showcase a data-driven approach to respond to a large collection of system analysis queries. This approach combines the applicability and scalability of simulation-based

*This work has been partly funded by the awards AFRL FA8650-15-C-2546, ONR N000141310778, ARO W911NF-15-1-0592, NSF 1550212 and DARPA W911NF-16-1-0001.

HSCC'16, April 12-14, 2016, Vienna, Austria

DOI: http://dx.doi.org/10.1145/2883817.2883846

analysis[1] with the end use of a certificate to conveniently predict answers to queries. Specifically, we interpret system analysis queries as *data classification* problems [7]. The constructed classifier—as appropriate sublevel sets of Lyapunov-like functions do—separates the set of initial conditions into two categories. It can be used to predict which category a priori unseen (sets of) initial conditions belong along with quantified confidence in the form of generalization errors.

We present a candidate setup for interpreting a sample of system analysis queries as data classification problems. This interpretation in turn opens up dynamical system verification to a wide range of methods from statistical learning theory [7]. We overview a particular classification method and demonstrate the effectiveness of the approach on conventional examples (for comparison with optimization-based proof construction techniques) as well as on systems including those for which the loop is closed by an adaptive controller and by a logic obtained through a prior reinforcement learning phase, hence with no closed-form models.

2. TECHNICAL APPROACH

We now discuss a candidate setup for data-driven verification. For ease of presentation, we focus on *region of attraction* analysis around a stationary trajectory of an autonomous system. That is, we are interested in characterizing the extent of the set of initial conditions from which the system approaches a fixed stationary trajectory.

2.1 Setup

Suppose that the system is represented by a finite collection S of signals (or sequences) $\phi(\cdot; x_0)$ over the time horizon $[0, T_{x_0}]$ and set $X \subseteq \mathbb{R}^n$ parametrized by the initial condition $x_0 \in X$. The signal $\phi(\cdot; x_0)$ may represent the evolution of the state, if evident from the underlying model, of a dynamical system from the initial state x_0. If the state of the system is not evident, then $\phi(\cdot; x_0)$ may simply represent the evolution of the variables in the model that can be monitored by an observer. The latter is often the case for software-based simulation models (e.g., those in Simulink).

Suppose that x_{nom} is a stationary signal, and we are interested in estimating the extent of the region of attraction $\{x \in \mathbb{R}^n : \phi(t; x) \to x_{\text{nom}} \text{ as } t \to \infty\}$ of x_{nom}. If x is the state in a closed-form model for the system, it may be possible to construct Lyapunov certificates that witness affirmative answers to the query. Roughly speaking, a Lyapunov function V that decreases along all nonstationary trajectories emanating from the set $\{x \in \mathbb{R}^n : V(x) \leq 1\}$ and attains its minimum value at x_{nom} [3]. Essentially, $\{x \in \mathbb{R}^n : V(x) = 1\}$ *separates* the states in X (Figure 1): the initial conditions from which the system is *known* to recover to x_{nom} and those from which the system is *not known* to recover. While the existence of certificates provides unambiguous answers, automated construction is possible only for limited family of models with a small-to-modest number of states.

In order to combine the scalability of simulations and the end-use flexibility of analytical certificates, we now interpret the system analysis query as a *data classification* problem. To this end, consider that we are equipped with an *oracle* that labels a given initial condition (more specifically the trajectory through the initial condition) as *converging* (i.e., the trajectory through the initial condition converges to $x = 0$) vs. *non-converging* (i.e., the trajectory through the initial condition does not converge to $x = 0$). Call the set of initial conditions labeled as *converging* as S_C and those labeled as *non-converging* as S_D. Given a collection of converging and non-converging initial conditions, computation of a classifier can be formulated as a supervised learning problem [7].

REMARK 1. *The collection $S_C \cup S_D$ of initial conditions (possibly in the space of observed variables) may be obtained through simulations or come from the monitoring of the actual system operation. The sets S_C and S_D may be extended by including finitely many points along the trajectories emanating from the initial conditions in the corresponding set.*

REMARK 2. *The construction of an oracle is beyond the scope of the paper. In most cases, an oracle will be based on an expert's judgment or a somewhat automated version of such judgment. For example, by limiting the execution over finite windows, one can label the executions that result in undesirable behavior within the window as -1 and all other executions as "practically" $+1$. We also refer the reader to [6] for preliminary ideas in the context of simulation-based proof construction.*

2.2 Classification

Naively, consider the classifier with an affine form $w^T x + b$ such that $w^T \bar{x} + b > 0$ for all initial conditions $\bar{x} \in S_C$ and $w^T \bar{x} + b < 0$ for all $\bar{x} \in S_D$. Then one can efficiently compute w and b that satisfy the conditions imposed at the given initial conditions, for example, by forming a linear program. While the utility of this naive approach vanishes if the data sets are not separable by the level sets of affine functions, richer parameterizations with nonlinear basis functions can be used instead. Given a vector $\Phi(x) \in \mathbb{R}^m$ of basis functions, note that the constraints $w^T \Phi(\bar{x}) + b > 0$ for all $\bar{x} \in S_C$ and $w^T \Phi(\bar{x}) + b < 0$ for all $\bar{x} \in S_D$ remain affine in the parameters w and b. Therefore, the search for a classifier of the form of $w^T \Phi(x) + b$ admits an encoding similar to an affine classifier. Note that depending on the expressivity of the basis functions in Φ, the hyperplane $\{x \in \mathbb{R}^n : w^T \Phi(x) + b = 0\}$ (for fixed w and b) may still be quite complicated. Due to this modest increase (note that typically $m > n$) in computational cost yet potentially enhanced expressivity, this transformation of the search for classifiers from the data space $\mathbb{R}^n$ to the feature space $\mathbb{R}^m$ of Φ is often used in the machine learning literature [7].

The support vector machine (SVM) algorithm computes an optimal, separating linear hyperplane in the feature space $\mathbb{R}^m$ to separate the sample data [7]. Suppose that the points in one set, say S_C, are labeled as -1 and those in the other set S_D as 1. Let $\bar{y}$ denote the label for $\bar{x} \in S_C \cup S_D$ and enumerate the points by $1, \ldots, N$. Then, the soft-margin SVM algorithm solves the following quadratic program for fixed integer $k \geq 1$ and positive constant $C > 0$:

$$\begin{array}{ll} \text{minimize}_{w,b} & \frac{1}{2}\|w\|^2 + C(\sum_{i=1}^N \xi_i)^k \\ \text{subject to} & y_i(w^T \Phi(x_i) + b) \geq 1 - \xi_i,\ i = 1, \cdots, N. \\ & \xi_i \geq 0,\ i = 1, \cdots, N \end{array} \tag{1}$$

Here, the non-negative slack variables ξ_i are introduced to accommodate possible inseparability of the dataset while minimizing the degree of misclassification.

[1]That is, conducting numerical simulations with a dynamical system model is typically computationally less demanding than constructing algebraic proof certificates using the same model.

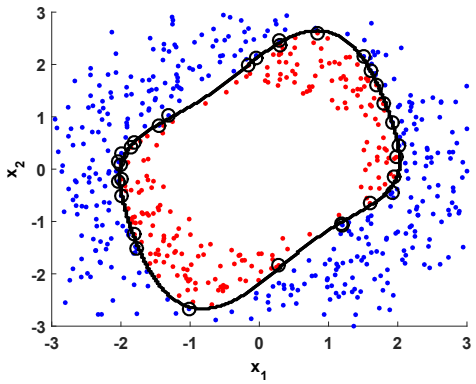

Figure 2: Data samples (S_C – red; S_D – blue), a classifier, and the contour of the decision surface with the support vectors circled.

If m is too large, then computing the map $w^T\Phi(x)+b$ (and solving for w using the formulation in (1)) becomes computationally demanding. In such cases, the so-called "kernel trick" is used to avoid the computation of the map $w^T\Phi(x)+b$. To this end, consider the Lagrangian dual of (1) for $k=1$ (for simplicity):

$$\begin{aligned} \text{maximize}_\alpha \quad & \textstyle\sum_{i=1}^N \alpha_i - \sum_{i,j=1}^N \alpha_i\alpha_j y_i y_j (\Phi(x_i)^T\Phi(x_j)) \\ \text{subject to} \quad & \textstyle\sum_{i=1}^N \alpha_i y_i = 0 \\ & 0 \le \alpha_i \le C \quad i=1,\cdots,N. \end{aligned} \tag{2}$$

Now define a kernel function $k:\mathbb{R}^n\times\mathbb{R}^n\to\mathbb{R}$ such that $k(x,\bar{x}) = \Phi(x)^T\Phi(\bar{x})$. The dual problem and introduction of the kernel function allow the solution of the dual problem without having to evaluate map Φ with potentially high dimensional feature space. For example, while $k(x,\bar{x}) = (x^T\bar{x}+1)^2$ is a compact representation of a kernel function, the corresponding basis function vector $\Phi(x_1,x_2) = [x_1^2, x_2^2, \sqrt{2}x_1x_2, \sqrt{2}x_1,\ x_2, 1]^T$ requires a larger representation.

Finally, a classifier can be constructed from the solution of the dual problem in the following manner. At the optimal solution, a dual variable α_i is non-zero only when the corresponding primal constraint $y_i(w^T\Phi(x_i)+b) \ge 1-\xi_i$ holds with equality. Similarly, the slack variable ξ_i of the primal problem is zero when α_i is strictly less than then upper bound C [8]. The solution to the primal problem, i.e., a classier, can be constructed as

$$\sum_{j=1}^N \alpha_j y_j k(x_j,x) + b. \tag{3}$$

The elements x_j with non-zero α_j are the "support vectors". In practice, often only a small number of α_i's are non-zero and the classifier admits a simple representation. Figure 2 shows a trained classifier with the training data where the support vectors are emphasized.

2.3 Error Analysis

Once a classifier is constructed, it can be used to respond to verification inquiries. However, it is constructed based on only a finite set of randomly selected data points that represent the underlying dynamical system over a continuous (or partly continuous) state space. Therefore, it is critical to provide an assessment of how correctly it would predict the responses of verification inquiries that involve a priori unseen configurations of the system. Statistical learning theory offers a number of metrics to evaluate the potential errors in classification.

Generalization error describes a classifier's accuracy when applied to arbitrary test data. It represents the expected probability of misclassification and is commonly used as a performance measure for classification algorithms. We use two approaches to calculate the generalization error in the case studies in Section 3.

The first error metric is empirical: construct a grid of fixed spacing across the set of state or observed variables, determine the true class of each sample, and compare that to the class predicted by the classifier. The generalization error is taken as the fraction of samples for which the two differ out of all samples. The level of accuracy of this error metric depends on the granularity of the discrete grid.

The second error metric in based on so-called κ-fold cross-validation, which is a commonly practiced approximation to leave-one-out cross-validation [9, 10]. Leave-one-out procedure removes an element from the training set and attempts to classify the removed element with a classifier trained on the incomplete set. This procedure is repeated for each element and the fraction of misclassified samples estimates the expected probability of error. The complete leave-one-out cross-validation is computationally intensive with large data sets. κ-fold cross-validation is an abbreviated approach. The training data $S = S_c \cup S_D$ is separated into κ disjoint sets $S_1, S_2, \cdots, S_\kappa$. For each $i \in \{1,2,\cdots,\kappa\}$, a classifier is trained on $S \setminus S_i$ and tested on S_i. The accuracy of each iteration is the number of classification errors over the size of the training set. The generalization error is obtained by taking the average iteration accuracy.

REMARK 3. *The soft-margin SVM formulation allows misclassfication errors. On the other hand, depending on the application, false positives (e.g., classified as safe while not) and false negatives (e.g., classified as unsafe while safe) may not be equally critical. For example, for verification in safety-critical applications, one may prefer reduce the possibility of false positives at the expense of larger number of false negatives (i.e., safe yet possibly conservative classification). This effect can be achieved by using non-homogenous weights instead of the same weight C in the penalization of possible classification errors in (1)[11].*

3. CASE STUDIES

We now demonstrate the method on several case studies ranging from the commonly used Van der Pol oscillator to multi-dimensional aircraft controllers and reinforcement-learning-based controllers. These cases cover a wide range of dynamics and evaluation criteria. For instance, an unstable oscillator is judged as "safe" if the system reaches the stable equilibrium point while an adaptive control system is concerned with minimizing deviation from a separate reference trajectory. We use the Matlab Machine Learning library for the SVM implementation and a 3.4 GHz dual-core/8GB RAM Windows machine. The implementations for the case studies are available at `http://tinyurl.com/z8hyfrf`. In the case studies, we refer to certain Lyapunov-like functions that we either draw from literature (e.g., from [5, 6, 12]) or compute using sum-of-squares optimization for comparison of the results. We use the sampling method discussed in

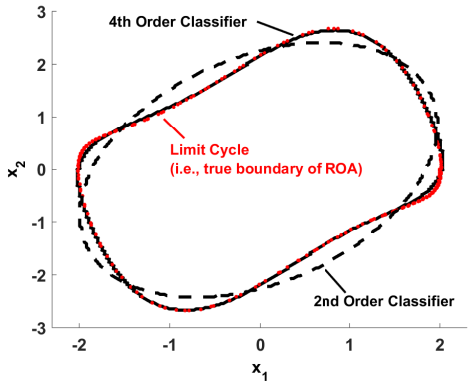

Figure 3: 2nd(dashed) and 4th(solid) order classifiers for Van der Pol dynamics.

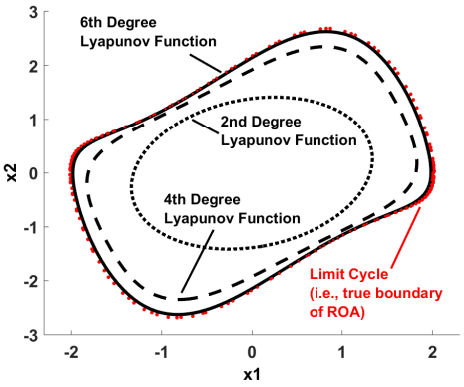

Figure 4: 2nd, 4th, and 6th order Lyapunov-based estimates [6] for Van der Pol dynamics.

[6] in the context of simulation-based search for polynomial Lyapunov functions. For ease of comparison, we often use polynomial kernels in classification unless noted otherwise.

3.1 Van der Pol Oscillator

The first test case is a Van der Pol oscillator $\dot{x}_1 = -x_2$ and $\dot{x}_2 = x_1 + (x_1^2 - 1)x_2$ with an unstable limit cycle and an asymptotically stable equilibrium at the origin. This problem has been widely used in previous nonlinear analysis and verification procedures [6, 13]. As such, the Van der Pol dynamics provide a baseline for data-driven verification as its region of attraction.

We trained two classifiers usingsimulated data (600 samples) with 2nd- and 4th-order polynomial kernels and both with homogenous misclassification costs. The results are shown in Figure 3. The 4th-order classifier separates the training data with 3 errors (2 false positives) and closely follows the limit cycle. The generalization error is calculated empirically as 0.48% on a test set of 75,000 samples. κ-fold cross-validation with 10 folds estimates the expected probability of error at 0.015. Average runtimes for sampling, training, and cross-validation in both cases were 27.78, 0.6, and 1.1 seconds respectively. The test set was generated in 14 minutes.

When compared to the results from traditional Lyapunov-based methods [6] shown in Figure 4, the classification-based results offer an improvement. The coverage provided by the estimate from the classifier using basis functions up to 4th-degree polynomials is comparable with that from the 6th-degree Lyapunov function. This example demonstrates the flexibility of the classification-based method, though it is critical to emphasize that this flexibility is at a cost. While the Lyapunov-based estimate comes with certainty, the classification-based estimate allows classification errors. Therefore, the results from the classification-based method must be interpreted with their quantified confidence.

3.2 Adaptive Controllers

We now apply the method to estimate the region of attraction for multiple systems with recently developed adaptive controllers [14] that challenge the conventional Lyapunov-based barrier certificate methods. First, consider the following second order uncertain system with unknown parameters W_1^*, W_2^* and initial conditions $x_1(0)$, $x_2(0)$:

$$\begin{aligned} \dot{x}_1 &= x_2, \\ \dot{x}_2 &= (-0.2 + W_1^*)x_1 + (-0.2 + W_2^*)x_2 + u. \end{aligned} \tag{4}$$

A concurrent learning model reference adaptive controller is used to estimate the unknown parameters and compute control inputs u to force the system to track a desired reference system $\dot{x}_m(t) = A_m x_m(t) + Br(t)$, where A_m is a Hurwitz matrix [14]. Because of online parameter estimation, the adaptive system has 4 states covering x_1, x_2, W_1^*, W_2^*. For ease of discussion, we fixed the initial values $x_1(0)$ and $x_2(0)$, resulting in a 2-dimensional problem over W_1^*, W_2^*. The success or failure of the system (i.e., the classification criterion) is based upon the ability of the controller to maintain $x_1(t)$ within ± 1.0 of $x_{m1}(t)$ for all $t \geq 0$.

We generated a training dataset of 10000 different (W_1^*, W_2^*) pairs in roughly 2.9 hours. We trained a set of 8th-order polynomial classifiers. In order to observe the effect of non-homogenous weighting on the decrease in unsafe errors, we swept the false-negative to false-positive ratio from 1:1 to 100:1 and evaluated on an independent testing set of over 30000 data points that span the entire feasible (W_1^*, W_2^*) space. The average time required for training and testing the classifiers were 164.17 and 0.229 seconds, respectively. The results are shown in Figure 5 and in Table 1. Note that Figure 5 also includes an independently obtained barrier certificate [12], labeled as "Lyapunov".

Table 1: Error rates for the system in (4). The weighting ratio of false negatives to false positives is increased from 1:1 to 100:1.

Data Set	Error (frac.)	False Neg.	False Pos.
Training	0.0402	54 of 3000	357 of 7222
Testing	0.0332	90 of 13263	986 of 19098
Training	0.0826	844 of 3000	0 of 7222
Testing	0.0704	2275 of 13263	3 of 19098

As seen in Figure 5, the SVM classifiers are able to much less conservatively estimate the ROA than previous Lyapunov-based barrier certificate methods [6]. If the lack of false-positive errors is important, the cost weighting in Remark 3 offers a viable way of reducing unsafe errors, as seen for the 100:1 training and testing data. While the total number of false negatives increases, the weighting does reduce

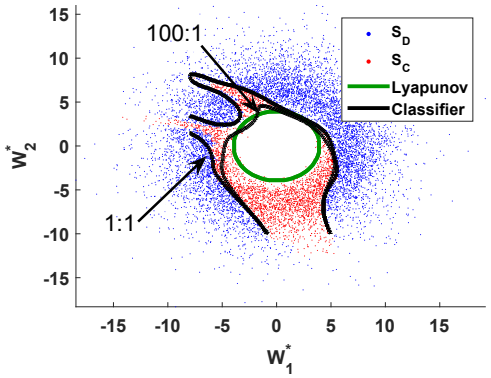

Figure 5: Classifiers for the system in (4) are compared against each other and the ROA estimate from a Lyapunov-based analysis.

probability of unsafe false positives to 1.57×10^{-4}.

As the number of variables considered increases, the sampling, training, and testing times will increase. We provide an additional, more complex example along with the implementation details at `http://tinyurl.com/z8hyfrf`.

3.3 Aircraft Reversionary Safety Controller

Another example is on an aircraft reversionary controller.[2] The reversionary control is intended as a safety check that takes over for the advanced controller in the case of failure. We are interested in estimating the set of perturbed states from which the reversionary controller can recover to a particular nominal condition. The underlying simulation model has 10 states and is highly nonlinear. While the simulations are performed on this 10-dimensional model, the verification inquiry (hence the classification) involves only a 3-dimensional subspace (i.e., the system behavior is monitored in these 3 dimensions).

We trained two 4th-order polynomial classifiers with 4000 and 16000 data points. As this reversionary controller is used as a safety-critical component of the flight control system, the classifier should feature a smaller rate of false-positive predictions (unsafe states labeled as safe) potentially at the cost of an increased number of false-negative predictions. To this end, we increased the weighting ratio of false positives to false negatives gradually from 1:1 to 100:1 with false positive errors going to zero.

Table 2: Error rates for the example in section 3.3 with 100:1 weight ratio.

Data Set	Error (%)	False Pos.	False Neg.
N = 4000	7.90	0 of 1604	316 of 2396
N = 16000	7.74	0 of 6497	1238 of 9503

Table 2 summarizes the results. A 10-fold cross-validation estimates the mean probability of error at 3.4×10^{-3} for the small set and 2.6×10^{-3} for the large test dataset. Observed runtimes for sampling, training, and cross-validation were 222, 13, 137 seconds with 4000 samples and 884, 174, 1818 seconds with 16000 samples. The weighted classifiers show better optimized estimates of the safety region by allowing no false-positives while correctly classifying a greater number of safe samples.

[2]Obtained based on personal communication with J. Schierman and M. Devore of Barron Associates.

3.4 Reinforcement Learning-Based Controller

We now consider an example in which a two-link robot, the *acrobot*, is controlled by a logic from a prior reinforcement learning step using Sarsa(λ) [15]. The controller determines the torque applied at the second joint which is most efficient at achieving the goal of raising the end of the second link a full link length above the top of the first link, simulating a gymnast swinging on a high bar. The system dynamics [15] are

$$\begin{aligned}
\ddot{\theta}_1 &= -d_1^{-1}(d_2\ddot{\theta}_2 + \phi_1) \\
\ddot{\theta}_2 &= (\frac{5}{4} - \frac{d_2^2}{d_1})^{-1}(\tau + \frac{d_2}{d_1}\phi_1 - \frac{1}{2}\dot{\theta}_1^2 \sin\theta_2 - \phi_2) \\
d_1 &= \frac{7}{2} + \cos\theta_2 \\
d_2 &= \frac{5}{4} + \frac{1}{2}\cos\theta_2 \\
\phi_1 &= -\frac{1}{2}\dot{\theta}_2 \sin\theta_2(\dot{\theta}_2 - 2\dot{\theta}_1) + \frac{3}{2}g\cos(\theta_1 - \pi/2) + \phi_2 \\
\phi_2 &= \frac{1}{2}g\cos(\theta_1 + \theta_2 - \pi/2),
\end{aligned} \tag{5}$$

where g is gravity. The velocities are limited to $\dot{\theta}_1 \in [-4\pi, 4\pi]$ and $\dot{\theta}_2 \in [-9\pi, 9\pi]$ while the positions are not bounded. The learning process refines an eligibility trace and converges to a sequence of torque selections ($\tau \in \{-1, 0, 1\}$) that guides the controller toward its goal from the current state.

The controller is trained from a single initial condition over a finite grid of the state space and the control inputs applied at points other than these finitely many grid points are determined through interpolation. At the initial conditions, the robot is stationary with its links fully extended as a pendulum, $(\theta_1, \theta_2) = (\dot{\theta}_1, \dot{\theta}_2) = (0, 0)$. Hence, the question of interest here analogous to stability is for which initial conditions will the trained controller still accomplish its objective. The criterion for a successful sample is that the robot achieves the goal in less than 1000 simulation steps (in integration with a uniform increment of 0.05 units of time) after which the episode terminates as a failed attempt. Initial conditions are sampled from the state subspace: $\theta_1 \in [-\pi/2, \pi/2]$, $\theta_2 \in [-\pi/2, \pi/2]$, $\dot{\theta}_1 \in [-2\pi, 2\pi]$, and $\dot{\theta}_2 \in [-2\pi, 2\pi]$.

Table 3 shows the observed classifier results with 4000 samples using 4th order polynomial kernel. The expected probability of error is estimated at 0.075 by 10-fold cross-validation. Runtimes for initial learning, SVM training, and cross-validation are 208, 3.6, and 26.5 seconds respectively.

Table 3: Error rates for the example in section 3.4.

Data Set	Error (%)	False Pos.	False Neg.
Training	0.75	0 of 134	30 of 3866
Testing	1.35	12 of 118	42 of 3882

4. CRITIQUE

The data-driven verification approach connects the classical systems analysis questions to a vast range of concepts and techniques: (i) The method requires only a notion of *observable variables* and an *oracle that generates labels*. Hence,

it can be directly applied on datasets without even a model or with high-fidelity simulation models for a range of system analysis questions, including safety, reachability, and more sophisticated temporal logic verification [16]. (ii) Statistical learning literature offers metrics to quantify the possibility of error in predictions (i.e., our confidence in the results) with respect to the amount of data used in classification. It remains open to understand the utility of these metrics in informing system designers. (iii) Regardless of the system representation, classification can be formulated as convex optimization. Rigorous sensitivity analysis and different weights used in the underlying optimization can guide subsequent, informative data collection (or generation) and help adjust the relative levels of prediction errors to meet certain criteria, e.g., *false positives* may be more tolerable than *false negatives* in safety-critical systems.

Though the method we presented makes, for the first time (to the best of our knowledge), connections between deductive dynamical system verification and statistical learning theory, it has similarities with several other methods discussed in the literature. The closest connection is perhaps with statistical model checking [17] of finite-state models. Smoothed model checking [18], a variation on statistical model checking, fits Gaussian process models to sampled data based on probabilistic results. Simulations—or signal-based view—have also been central in the verification techniques in [12, 19, 20, 21] and in the complementing view of falsification [22]. Finally, unlike Monte-Carlo-based approaches, the proposed method computes an algebraic map that can be used for future predictions of set membership and inclusion (e.g., prediction of the label for a new, unseen set of initial conditions).

5. REFERENCES

[1] Department of Defense. *Department of Defense Standard Practice System Safety*, May 2012.

[2] Vance Hilderman and Tony Baghi. *Avionics certification: a complete guide to DO-178 (software), DO-254 (hardware)*. Avionics Communications, 2007.

[3] Hassan K. Khalil. *Nonlinear Systems*. Prentice Hall, 3rd edition edition, 2002.

[4] Stephen Boyd, Laurent El Ghaoui, E. Feron, and V. Balakrishnan. *Linear Matrix Inequalities in System and Control Theory*, volume 15 of *Studies in Applied Mathematics*. Society for Industrial and Applied Mathematics, 1994.

[5] Pablo A. Parrilo. *Structured Semidefinite Programs and Semialgebraic Geometry Methods in Robustness and Optimization*. PhD thesis, California Institute of Technology, 2000.

[6] Ufuk Topcu, Andrew Packard, and Peter Seiler. Local stability analysis using simulations and sum-of-squares programming. *Automatica*, 44(10):2669–2675, 2008.

[7] Vladimir Naumovich Vapnik and Vlamimir Vapnik. *Statistical learning theory*. Wiley New York, 1998.

[8] B Scholkopf. Statistical learning and kernel methods. Technical report, Microsoft Research, 2000.

[9] Davide Anguita, Alessandro Ghio, Sandro Ridella, and Dario Sterpi. K-fold cross validation for error rate estimate in support vector machines. In *International Conference on Data Mining (DMIN)*, pages 291–297, 2009.

[10] Ron Kohavi et al. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *International Joint Conference on Artificial Intelligence*, volume 2, pages 1137–1145, 1995.

[11] Xulei Yang, Qing Song, and Yue Wang. A weighted support vector machine for data classification. *International Journal of Pattern Recognition and Artificial Intelligence*, 21(05):961–976, 2007.

[12] James Kapinski, Jyotirmoy Deshmukh, Sriram Sankaranarayanan, and Nikos Arechiga. Simulation-guided lyapunov analysis for hybrid dynamical systems. In *Hybrid Systems: Computation and Control*, 2014.

[13] Ufuk Topcu, Andrew K. Packard, Peter Seiler, and Gary J. Balas. Robust region-of-attraction estimation. *IEEE Transactions on Automatic Control*, 55(1):137–142, January 2010.

[14] Girish Chowdhary, Hassan A. Kingravi, Jonathan How, and Patricio A. Vela. Bayesian nonparametric adaptive control using gaussian processes. *IEEE Transactions on Neural Networks and Learning Systems*, 26(3):537–550, March 2015.

[15] A. Barto and R. Sutton. *Reinforcement Learning: An Introduction*. MIT Press, 1998.

[16] Christel Baier and Joost-Pieter Katoen. *Principles of Model Checking*. MIT Press, 2008.

[17] Koushik Sen, Mahesh Viswanathan, and Gul Agha. Statistical model checking of black-box probabilistic systems. In *Computer Aided Verification*, pages 202–215, 2004.

[18] Luca Bortolussi, Dimitrios Milios, and Guido Sanguinetti. Smoothed model checking for uncertain continuous time markov chains. Technical report, University of Trieste/University of Edinburgh, 2014.

[19] Alexandre Donze and Oded Maler. Robust satisfaction of temporal logic over real-valued signals. In *Formal Modeling and Analysis of Timed Systems*, pages 92–106. 2010.

[20] Zhenqi Huang and Sayan Mitra. Computing bounded reach sets from sampled simulation traces. In *Hybrid Systems: Computation and Control*, pages 291–294, 2012.

[21] Yi Deng, Akshay Rajhans, and A Agung Julius. Strong: A trajectory-based verification toolbox for hybrid systems. In *Quantitative Evaluation of Systems*, pages 165–168. 2013.

[22] Sriram Sankaranarayanan and Georgios Fainekos. Falsification of temporal properties of hybrid systems using the cross-entropy method. In *Hybrid Systems: Computation and Control*, pages 125–134, 2012.