Conformal Predictions for Hybrid System State Classification*

Luca Bortolussi[1,4], Francesca Cairoli[1], Nicola Paoletti[2], and Scott D. Stoller[3]

[1] Department of Mathematics and Geosciences, Università di Trieste, Italy
[2] Department of Computer Science, Royal Holloway, University of London, UK
[3] Department of Computer Science, Stony Brook University, USA
[4] Modelling and Simulation Group, Saarland University, Germany

Abstract. Neural State Classification (NSC) [19] is a scalable method for the analysis of hybrid systems, which consists in learning a neural network-based classifier able to detect whether or not an unsafe state can be reached from a certain configuration of a hybrid system. NSC has very high accuracy, yet it is prone to prediction errors that can affect system safety. To overcome this limitation, we present a method, based on the theory of conformal prediction, that complements NSC predictions with statistically sound estimates of prediction uncertainty. This results in a principled criterion to reject potentially erroneous predictions *a priori*, i.e., without knowing the true reachability values. Our approach is highly efficient (with runtimes in the order of milliseconds) and effective, managing in our experiments to successfully reject almost all the wrong NSC predictions.

1 Introduction

Hybrid systems, i.e., systems characterized by the interaction between discrete (digital) and continuous (physical) components, are a central model for many cyber-physical system applications, from avionics to biomedical devices [1]. Formal verification of hybrid systems typically boils down to solving a hybrid automata (HA) reachability checking problem [13]: given a model $\mathcal{M}$ of the system expressed as an HA and a set of unsafe states U of $\mathcal{M}$, check whether U is reached in any (time-bounded) path from a set of initial states. HA reachability checking is undecidable in general [13], a difficulty that current HA reachability checking algorithms address by over-approximating the set of reachable states. These algorithms are computationally very expensive, and thus, usually limited to design-time (offline) analysis.

Motivated by the need to make HA reachability checking more efficient and suitable for online analysis, Phan et al. [19] recently proposed *Neural State Classifications (NSC)*, an approach for approximating reachability checking using deep neural networks (DNNs). Their work shows that it is possible to train,

* This material is based on work supported in part by NSF Grants CNS-1421893 and CCF-1414078 and ONR Grant N00014-15-1-2208.

using examples computed via suitable HA model checkers, DNN-based state classifiers that approximate the result of reachability checking with very high accuracy. For any state s of the HA, such a classifier labels s as *positive* if an unsafe state is reachable from s within a given time bound; otherwise, s is labeled as *negative*.

The key advantage of this approach is its efficiency. Neural state classifiers indeed run in constant time and space, because the computation is not directly affected by the size and complexity of the HA model or specification, but only by the complexity of the chosen DNN architecture.

The main drawback is that DNNs for NSC (like any other machine learning model) are subject to *classification errors*, the most important being false negatives, i.e., when the DNN classifies a state as negative while it is actually positive. While Phan et al.'s work allows estimation of the classification accuracy for a region of states (i.e., the probability that a state in the region is wrongly classified), it does not provide any indication about the reliability of single-point predictions, i.e., DNN predictions on individual HA states whose true reachability value is unknown. This limits the applicability of NSC for online analysis, where state classification errors can compromise the safety of the system. This is in contrast with methods like smoothed model checking [4], which leverages Gaussian Processes and Bayesian statistics to quantify uncertainty, but on the other side faces severe scalability issues as the dimension of the system increases.

The aim of this work is to equip NSC with rigorous methods for quantifying the reliability of single-point predictions. For this purpose, we investigate *Conformal Prediction (CP)* [22], a method that provides statistical guarantees on the predictions of machine learning models. Importantly, CP requires only very mild assumptions on the data (i.e., exchangeability, a weaker version of the independent and identically distributed assumption).

By applying CP, we estimate two statistically sound measures of NSC prediction uncertainty, *confidence* and *credibility*. Informally, the confidence of a prediction is the probability that a reachability prediction for an HA state s corresponds to the true reachability value of s. Credibility quantifies how a given state is likely to belong to the same distribution as the training data.

Using confidence and credibility, we show how to derive criteria for anomaly detection, that is, for rejecting NSC predictions that are likely to be erroneous. The key advantage of such an approach is that predictions are rejected on rigorous statistical grounds, and we show experimentally its superiority with respect to discrimination based on the DNN's class likelihood. Furthermore, computation of CP-based confidence and credibility is very efficient (approximately 3 ms in our experiments), which makes our method suitable for online analysis.

In summary, the main contributions of this paper are the following:

- We extend the framework of neural state classification with conformal prediction to quantify the reliability of NSC predictions.
- We derive criteria for anomaly detection based on CP to reject unreliable NSC predictions.

– We evaluate our method on three hybrid automata models showing that, with adequate choices of confidence and credibility thresholds, our method successfully rejects almost all prediction errors: over a total of 30,000 test samples, our method successfully rejected 43 out of 44 errors.

The paper is structured as follows. Sections 2 and 3 provide background on neural state classification and conformal prediction, respectively. In Section 4, we introduce our CP-based measures of prediction reliability. Results of the experimental evaluation are given in Section 5. Related work is discussed in Section 6. Section 7 offers concluding remarks.

2 Neural State Classification for Hybrid System Reachability

Neural state classification seeks to solve the *State Classification Problem* (SCP) [19], a generalization of the reachability checking problem for hybrid systems. Let $\mathbb{B} = \{0, 1\}$ be the set of Boolean values. Given an HA $\mathcal{M}$ with state space $S(\mathcal{M})$, time bound T, and set of unsafe states $U \subset S(\mathcal{M})$, the SCP problem is to find a *state classifier*, i.e., a function $F^* : S(\mathcal{M}) \to \mathbb{B}$ such that for all $s \in S(\mathcal{M})$, $F^*(s) = 1$ if $\mathcal{M} \models \mathsf{Reach}(U, s, T)$, i.e., if it is possible for $\mathcal{M}$, starting in s, to reach a state in U within time T; $F^*(s) = 0$ otherwise. A state $s \in S(\mathcal{M})$ is called *positive* if $F^*(s) = 1$. Otherwise, s is *negative*.

Neural State Classification [19] offers an approximate solution to the SCP based on machine learning models, deep neural networks (DNNs) in particular. The NSC method is summarized in Figure 1. The state classifier is trained using supervised learning, where the training examples are derived by sampling the state space according to some distribution and labelling the sampled states with the corresponding reachability values. The latter are computed by invoking an oracle, e.g., an hybrid system model checker [11]. The approach can handle parametric HA, by encoding parameters as additional inputs to the classifier.

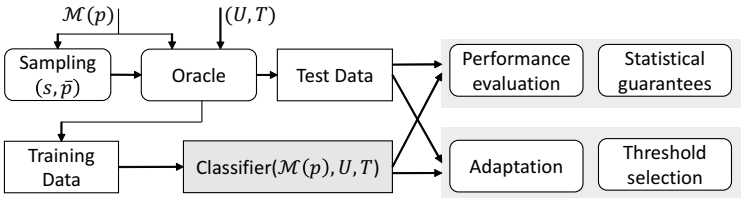

Fig. 1. Overview of the NSC approach (diagram from [19]).

NSC supports arbitrary state distributions. In [19], the following distributions and corresponding sampling methods are considered:

– Uniform sampling, where every state is equi-probable.

- Dynamics-aware sampling, which is based on the probability that a state is visited in any time-bounded evolution of the system, where such probabilities are estimated by performing isotropic random walks of the HA (i.e., by uniformly sampling the non-deterministic choices during the HA simulation).
- Balanced sampling, which seeks to draw a balanced number of positive and negative states. This is useful when U is a small portion of $S(\mathcal{M})$, in which case uniform sampling would produce imbalanced datasets with an insufficient number of positive samples, leading to classifiers with poor accuracy. For this purpose, in [19] the authors introduce a new method for the construction and simulation of reverse hybrid automata. Indeed, arbitrary numbers of positive samples can be generated by simulating the reverse HA starting from an unsafe state [19].

The performance of a trained classifier is evaluated by computing the empirical accuracy, rate of false positives (FPs), and rate of false negatives (FNs) using, as commonly done in supervised learning, test datasets of samples unseen during training. Inspired by statistical model checking [14], NSC also applies sequential hypothesis testing [23] to certify that a classifier meets prescribed accuracy, FN, or FP levels on unseen data, up to some given confidence level. Albeit useful, these kinds of statistical guarantees are, however, only applicable to regions of the HA state space, and as such, cannot be used to quantify the reliability of single-point predictions. The present paper aims to solve this very problem.

NSC includes two methods to reduce FNs: *threshold selection*, which adjusts the DNN's classification threshold to favor FPs over FNs, and a more advanced technique called *falsification-guided adaptation* that iteratively re-trains the classifier with false negatives found through adversarial sampling, i.e., by solving a non-linear optimization problem that maximizes the disagreement between DNN-predicted and true reachability values.

In [19], the authors applied NSC to six nonlinear hybrid system benchmarks, achieving an accuracy of 99.25% to 99.98%, and a false-negative rate of 0.0033 to 0, which was further reduced to 0.0015 to 0 by applying falsification-guided adaptation. While, with such performance, NSC can derive nearly perfect approximations of the HA reachability function, it does not provide a recipe for rejecting uncertain predictions *a priori*, i.e., without knowing the true reachability value. Our work extends NSC in this direction.

3 Conformal Prediction for Neural Networks

Conformal Prediction (CP) [22] is a flexible framework built on top of any traditional supervised machine learning model, called in CP the *underlying model*.

In this section, we describe CP in relation to a generic classification problem (of which NSC is an instance), where we denote with X the set of inputs and with $Y = \{y^1, \ldots, y^c\}$ the set of classification labels (or classes). The underlying classification model is a function $h : X \to [0,1]^c$ mapping inputs into a vector of class likelihoods, such that the class predicted by h corresponds to the class with the highest likelihood. For a generic input x_i, we will denote with y_i the

true label of x_i and with $\hat{y}_i$ the label predicted by h (i.e., the label with highest likelihood). Further, we will often use the notation x_* to indicate test points whose true label is unknown.

The interpretation of CP is two-fold. On one hand, conformal predictors output *prediction regions*, instead of single point predictions. In the case of classification, given a test point x_i and a significance level $\epsilon \in (0, 1)$, the prediction region of x_i, $\Gamma_i^\epsilon \subseteq Y$, is a set of labels guaranteed to contain the true label y_i with probability $1 - \epsilon$. We call this the *global* interpretation of CP.

On the other hand, given a prediction $\hat{y}_i \in Y$ for x_i, we can compute the minimum value of ϵ such that the prediction region Γ_i^ϵ contains only $\hat{y}_i$. The corresponding probability $1 - \epsilon$ is called the *confidence* of the predicted label $\hat{y}_i$. Along with the confidence, CP allows computing another measure, called *credibility*, which indicates how suitable the training data are for the current prediction. Therefore, CP complements each prediction, on a new input, with a measure of confidence and a measure of credibility. We call this the *point-wise* interpretation of CP.

Importantly, CP does not require prior probabilities, unlike Bayesian methods, but only that data is exchangeable (a weaker version of the classic i.i.d. assumption). We now provide a brief description of the method, but we refer to [22] for a detailed introduction.

Let $Z = X \times Y$. The main ingredients of CP are: a *nonconformity function* $f : Z \to \mathbb{R}$, a set of labelled examples $Z' \subseteq Z$, an underlying model h trained on (a subset of) Z', and a statistical test. The nonconformity function $f(z)$ measures the "strangeness" of an example $z = (x_i, y_i)$, i.e., the deviation between the label y_i and the corresponding prediction $h(x_i)$. A natural choice for f is $f(z) = \Delta(h(x_i), y_i)$, where Δ is a suitable distance [5]. As explained below, $f(z)$ is used to construct prediction regions in CP. In general, any function $f : Z \to \mathbb{R}$ will result in valid regions. However, a good nonconformity function, i.e. one that produces tight prediction regions, should give low scores to correctly predicted inputs, and large scores to misclassified inputs. See Section 3.2 for details about the nonconformity function definition.

3.1 CP Algorithm

Given a set of examples $Z' \subseteq Z$, a test input $x_* \in X$, and a significance level $\epsilon \in (0, 1)$, a conformal predictor computes a prediction region Γ_*^ϵ for x_* as follows.

1. Divide Z' into a training set Z_t, and calibration set Z_c. Let $q = |Z_c|$ be the size of the calibration set.
2. Train a model h using Z_t.
3. Define a nonconformity function $f((x_i, y_i)) = \Delta(h(x_i), y_i)$, i.e., choose a metric Δ to measure the distance between $h(x_i)$ and y_i (see Section 3.2).
4. Apply $f(z)$ to each example z in Z_c and sort the resulting nonconformity scores $\{\alpha = f(z) \mid z \in Z_c\}$ in descending order: $\alpha_1 \geq \cdots \geq \alpha_q$.

[5] The choice of Δ is not very important, as long as it is symmetric.

5. Compute the nonconformity scores $\alpha_*^j = f((x_*, y^j))$ for the test input x_* and each possible label $j \in \{1, \ldots, c\}$. Then, compute the smoothed p-value

$$p_*^j = \frac{|\{z_i \in Z_c : \alpha_i > \alpha_*^j\}|}{q+1} + \theta \frac{|\{z_i \in Z_c : \alpha_i = \alpha_*^j\}| + 1}{q+1}, \tag{1}$$

   where $\theta \in \mathcal{U}[0,1]$ is a tie-breaking random variable. Note that p_*^j represents the portion of calibration examples that are at least as nonconforming as the tentatively labelled test example (x_*, y^j).
6. Return the prediction region

$$\Gamma_*^\epsilon = \{y^j \in Y : p_*^j > \epsilon\}. \tag{2}$$

Note that steps 1–4 have to be performed only once, while 5–6 for every test point x_*[6].

The idea behind the above procedure is use a statistical test to check if (x_*, y^j) is particularly nonconforming compared to the calibration examples. The rationale is to estimate Q, the unknown distribution of $f(z)$, by applying $f(z)$ to calibration examples, then to compute α_*^j for every possible label y^j and test for the null hypothesis $\alpha_*^j \sim Q$. We reject the null hypothesis when the p-value associated to α_*^j is smaller than the significance level ϵ. That is, we do not include y^j in Γ_*^ϵ if it appears unlikely that $f((x_*, y^j)) \sim Q$. The prediction region therefore contains all the labels for which we could not reject the null hypothesis. This is an application of the Neyman-Pearson theory for hypothesis testing and confidence intervals [15].

Note that in Equation 1 by setting θ to a random value between 0 and 1, we compute a so-called smoothed p-value. The main difference between a standard p-value (where $\theta = 1$) and a smoothed p-value is that in the latter situation, we treat the borderline cases where $\alpha_i = \alpha^j$ more carefully. Instead of increasing the p-value by $\frac{1}{q}$ for each $\alpha_i = \alpha^j$, we increase it by a random amount between 0 and $\frac{1}{q}$. It has been proven that any smoothed conformal predictor is exactly valid, whereas a general conformal predictor is only conservative; see [22] for a complete treatment.

3.2 Nonconformity function

In general, the nonconformity function is a measurable function with type $f : Z \to \mathbb{R}$. A nonconformity function is well-defined if it assigns low scores to correctly predicted inputs and high scores to wrong predictions. It is typically based on the underlying machine learning model h, and defined by

$$f((x_i, y_i)) = \Delta(h(x_i), y_i),$$

[6] The approach we use is known in literature as inductive CP. The original CP approach, also called transductive CP, requires retraining the model for each new test sample and does not use a calibration set. See [18].

where Δ is some function that measures the prediction error of h. Recall that, for an input $x \in X$, the output of h is a vector of class likelihoods, which we denote by $h(x) = [P_h(y_1|x), \ldots, P_h(y_c|x)]$. For classification problems, a common choice for Δ is

$$\Delta(h(x_i), y_i) = 1 - P_h(y_i|x_i), \tag{3}$$

where $P_h(y_i|x_i)$ is the likelihood of class y_i when the model h is applied on x_i. Note that such defined Δ induces a well-defined nonconformity function. Indeed if h correctly predicts y_i for input x_i, then the corresponding likelihood $P_h(y_i|x_i)$ is high (the highest among all classes) and the resulting nonconformity score is low. The opposite holds when h does not predict y_i.

Using (3) also guarantees that the resulting p-values (see Equation 1) preserve the ordering of the class likelihoods predicted by model h. This means that, for example, the class with the lowest likelihood will also be the class with the smallest p-value and the class with the highest likelihood will result in the largest p-value. This property ensures that the prediction regions are consistent with the classification predicted by h.

In our experiments we use (3) as nonconformity function. Other functions designed specifically for neural networks have been proposed in [18]. However, our results showed no significant differences between the latter and (3).

3.3 Confidence and credibility

We now describe the measures of confidence and credibility, which are point-wise measures, i.e., derived from individual predictions.

Let us first notice that the regions Γ^ϵ for different ϵ values are nested: when $\epsilon_1 \geq \epsilon_2$, we have that $\Gamma^{\epsilon_1} \subseteq \Gamma^{\epsilon_2}$. Indeed, for an input x_*, if we choose an ϵ lower than the p-values of all the classes ($\epsilon < \min_{j=1,\ldots,c}\ p_*^j$), then the region Γ^ϵ will necessarily contain all the class labels. On the opposite, as ϵ increases, fewer and fewer classes will have their p-value higher than ϵ, until the region becomes empty (when $\epsilon \geq \max_{j=1,\ldots,c}\ p_*^j$).

The *confidence* of a point $x_* \in X$, $1 - \gamma_*$, is a measure of how likely our prediction for x_* is compared to all other possible classifications (according to the calibration set). It is computed as one minus the smallest value of ϵ for which the conformal region is a single label, i.e. the second largest p-value γ_*:

$$1 - \gamma_* = \sup\{1 - \epsilon : |\Gamma_*^\epsilon| = 1\}.$$

The *credibility*, c, is an indicator of how suitable the training data are to classify that example. In practice, it is the smallest ϵ for which the prediction region is empty, i.e. the highest p-value according to the calibration set.

$$c_* = \inf\{\epsilon : |\Gamma_*^\epsilon| = 0\}.$$

A high confidence, $1 - \gamma_*$, means that there is no likely alternative to the point prediction, whereas a low credibility means that even the point prediction is

unlikely. Therefore, if c_* is close to zero, the test example x_* is not representative of the data set.

If we consider γ_*, i.e., one minus the confidence, and c_*, the credibility, we obtain the range I_* of ϵ values for which we are sure that the corresponding prediction region contains a single label: $I_* = [\gamma_*, c_*) \subseteq [0, 1]$. We stress that the class contained in the singleton prediction region corresponds to the model prediction $\hat{y}_*$. This is a consequence of the chosen nonconformity function (3), by which the ordering of class likelihoods is preserved in the corresponding p-values (as discussed in Section 3.2).

Confidence and Credibility in binary classification. When $Y = \{0, 1\}$, as in NSC, the conformal classifier outputs, for each input point x_*, two probabilities: p_*^0 and p_*^1. Suppose $p_*^1 > p_*^0$ (the same reasoning applies if $p_*^0 > p_*^1$), which implies that the predicted class is 1. We define confidence as $1 - p_*^0$, and credibility as p_*^1. We call the interval $I_* = [p_*^0, p_*^1)$ the *confidence-credibility interval*. It contains all values of ϵ for which we are sure that the prediction region contains a single label (in this case, $\Gamma^\epsilon = \{1\}$, $\forall \epsilon \in I_*$).

4 Measures of Prediction Reliability

Confidence and credibility can be used as uncertainty metrics. They measure how much a prediction $h(x)$, made by the underlying model, can be trusted. We will leverage both the global and the point-wise interpretations of CP in order to generate a statistically valid acceptance criterion. The following measures and acceptance criterion are described in relation to a test set $X_* \subseteq X$ of unseen input points, i.e., whose true label is unknown. Let $K = |X_*|$ be the size of the test set. Moreover, we will assume the case of binary classification, which is the one relevant for NSC.

4.1 Global evaluation

Recall the global interpretation of CP: given a significance level ϵ, constant along X_*, the conformal classifier produces regions Γ^ϵ for each test input $x_* \in X_*$ that guarantee a global error probability of ϵ across the entire test set X_*. We say that the CP algorithm makes an error, at point x_*, if the prediction set at this point does not contain the true label. The most interesting prediction regions are those containing only a single class label, referred to as singleton regions, since empty and double ($\Gamma^\epsilon = \{0, 1\}$) regions have little actionable information. A singleton region containing the output prediction of h makes an error, i.e., Γ^ϵ contains the wrong label, if that point is misclassified by h. An empty prediction region for x at significance level ϵ is equivalent to the case that x has credibility less than ϵ (low credibility) in the point-wise interpretation of CP, whereas a double region for x corresponds to having confidence smaller than $1 - \epsilon$ (low confidence) in the point-wise interpretation.

4.2 Acceptance criterion

The p-values returned by the CP algorithm can be interpreted as anomaly measures. In binary classification, the two p-values of a test point x_*, p_*^0 and p_*^1 (see Equation 1), coincide with γ_* and c_*, respectively. The rationale behind our acceptance criterion is that every unseen point x_* is required to have both values of confidence, $1-\gamma$, and credibility, c, sufficiently high in order to accept the classification made by h with a particular certainty level α. The derivation of α is shown later in this section.

Our acceptance criterion works as follows. First, a value for the significance level ϵ, fixed along the entire test set, has to be chosen. As discussed in the previous section, ϵ represents the global error probability that we are willing to accept. The next step is to apply the conformal algorithm and obtain a confidence-credibility interval, I_*, for each test point $x_* \in X_*$. We accept the prediction of model h for x_* if and only if $\epsilon \in I_*$, i.e., if $\gamma_* \leq \epsilon < c_*$. Note that the latter condition implies that we only accept singleton prediction regions, i.e., such that $|\Gamma_*^\epsilon| = 1$ (see Section 3.3). Otherwise, if credibility is smaller than ϵ or confidence is smaller than $1-\epsilon$, we reject the prediction of h for x_*. In other words, these uncertainty measures indicate if a prediction is trustworthy or not. As explained below, the certainty level α is determined by the chosen ϵ and the ratio of rejected points.

We now discuss how to derive α. With the acceptance criterion introduced above, we are sure to accept only singleton prediction regions, rejecting points with non-informative regions (empty and double regions). Since ϵ gives the error probability in relation to any test point (which might or might not be accepted), it gives no guarantees on the error of accepted predictions alone. For this purpose, we provide a revised error probability estimate, $\hat{\epsilon}$, for accepted predictions only, i.e., that does not consider the rejected points. The certainty level α that we seek to obtain is defined as $1-\hat{\epsilon}$.

To compute $\hat{\epsilon}$, we follow the approach of [16]. Given a significance level ϵ, let $P^\epsilon(e)$, $P^\epsilon(s)$ and $P^\epsilon(d)$ be respectively the fraction of empty, single and double prediction regions observed on a test set with K examples ($P^\epsilon(e) + P^\epsilon(s) + P^\epsilon(d) = 1$). Overall, the expected number of errors is $E = \epsilon K$. Since double predictions are never erroneous (they always contain the true label) and empty predictions are always erroneous (they never do), we can rewrite the expected number of errors as:

$$\epsilon K = \hat{\epsilon} \cdot K P_\epsilon(s) + K P_\epsilon(e) \Rightarrow \hat{\epsilon} = \frac{\epsilon - P_\epsilon(e)}{P_\epsilon(s)}. \tag{4}$$

Thus, $\hat{\epsilon}$ represents the expected error rate over the $K \cdot P_\epsilon(s)$ singleton predictions. In other words, $\hat{\epsilon}$ is the error probability on accepted predictions.

5 Experimental Evaluation

To evaluate the proposed method for NSC with CP-based anomaly detection, an experimental evaluation was conducted on a selection of the hybrid-system

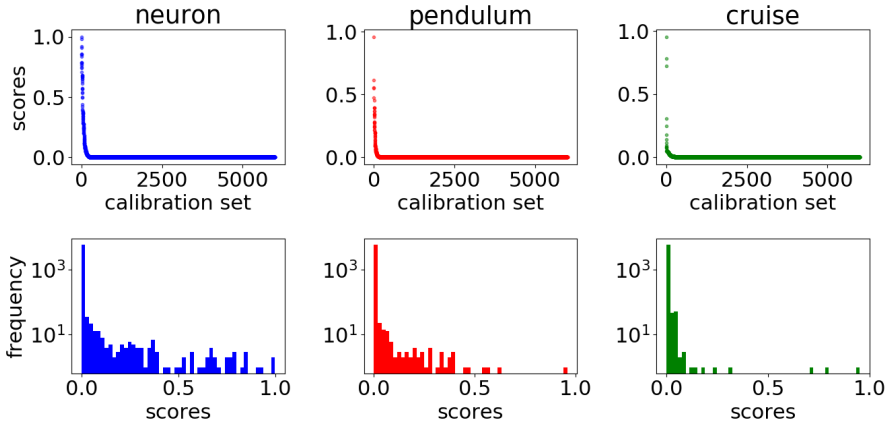

Fig. 2. Calibration scores $\alpha_1 \geq \cdots \geq \alpha_q$ for the neuron (left), pendulum (center) and cruise (right) models for a calibration set size of $q = |Z_c| = 6,000$. Histograms in the second row show the distributions of the calibration scores on a log-scale.

case studies considered in NSC (see [19] for details): a model of the spiking neuron action potential [9], the classic inverted pendulum on a cart, and a cruise controller [9].

Experimental settings. We consider the same settings used in NSC for training sigmoid DNNs [19]. NSC neural networks were learned using MATLAB's `train` function, with the Levenberg-Marquardt backpropagation algorithm optimizing the mean square error loss function, and the Nguyen-Widrow initialization method for the NN layers. The classifier is a DNN with 3 hidden layers, each consisting of 10 neurons with the Tan-Sigmoid activation function, and an output layer with 1 neuron with the Log-Sigmoid activation function. With such DNN architecture, the only output of the underlying model is the likelihood of class 1, which we denote with o^1, that is, the likelihood that a hybrid automaton state is positive, i.e., leads to a safety violation. The likelihood of class 0 is given by $o^0 = 1 - o^1$.

We consider training datasets of $14,000$ samples and calibration sets of $q = |Z_c| = 6,000$ samples. Training of the DNNs is very fast, taking 2 to 7 seconds. The test set contains $10,000$ points. The CP algorithm was implemented in Python. Computation of confidence and credibility is very efficient, and takes around 30 seconds for the entire test set, approximately 3 ms per point.

5.1 Calibration scores

We conduct a detailed analysis of the distribution of calibration scores, which depends both on the case study at hand and on the underlying model. The DNNs trained for NSC approximate the output of reachability checking with

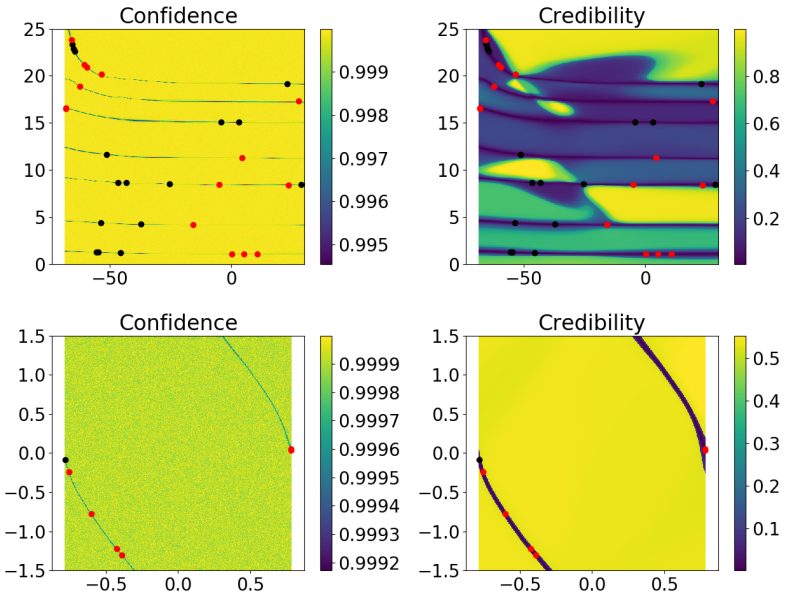

Fig. 3. Landscape of confidence (left) and credibility (right) values along the entire state space of the two-dimensional case-studies: spiking neuron (top) and inverted pendulum (bottom). Red dots indicate false-negatives, black dots false-positives.

very high accuracy. Therefore, the scores $\alpha_1, \ldots, \alpha_q$ are close to zero for most of the points in Z_c (see Fig. 2). Recall that the p-values of an unseen test point x_* count the number of calibration scores greater than that of x_*. Credibility is the p-value associated with the class predicted by h, for which we expect a small score and therefore a high p-value. On the contrary, γ is the p-value associated to the other (non-predicted) class, for which we expect a larger score. However, given the high accuracy of h, the number of large calibration scores, i.e., scores significantly greater than zero, is very small. Therefore, the fraction of calibration scores determining γ is not very sensitive to changes in the value of the nonconformity score of x_*, α_*. On the contrary, credibility is extremely sensitive to small changes in α_*. In general, the sensitivity of confidence w.r.t. α_* increases as the accuracy of h decreases, and vice versa for credibility.

5.2 Performance evaluation

Figures 3 and 4 show the landscapes of confidence and credibility for the three case studies. Notice that both measures are able to detect the input regions with higher uncertainty, i.e., regions where misclassification occurs. However, given the high accuracy of our DNNs, credibility results in an extremely sensitive measure, as previously discussed. Indeed, we observe drastic drops in credibility values even for regions that return correct predictions. In these areas the DNN

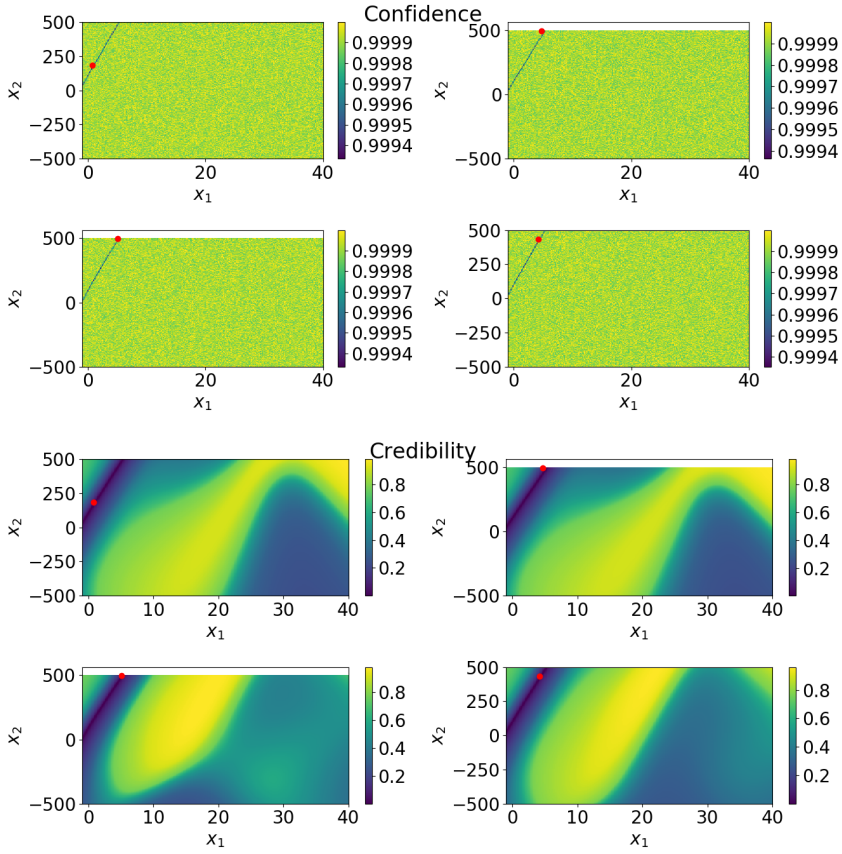

Fig. 4. The cruise controller model has a four-dimensional input space. Four points were misclassified by the DNN, and they all have coordinate $x_4 = 5.0$. The figure shows two dimensional sections ((x_1, x_2)-plane) at the x_3 coordinates of the four misclassified points and with $x_4 = 5.0$. The confidence landscapes are on top; the credibility landscapes are below them. Red dots indicate false-negatives, black dots false-positives.

is classifying properly but with lower accuracy with respect to areas with higher credibility. Confidence values, on the other hand, span in an extremely narrow interval close to 100%.

5.3 Benefit of conformal predictions

The key advantage of our approach is that predictions are rejected on rigorous statistical grounds. We experimentally compare it with a naive approach based on the DNN output.

We define the naive uncertainty metric as the difference between the likelihoods of the two classes, that is, $|o^0 - o^1|$. Intuitively, small differences should

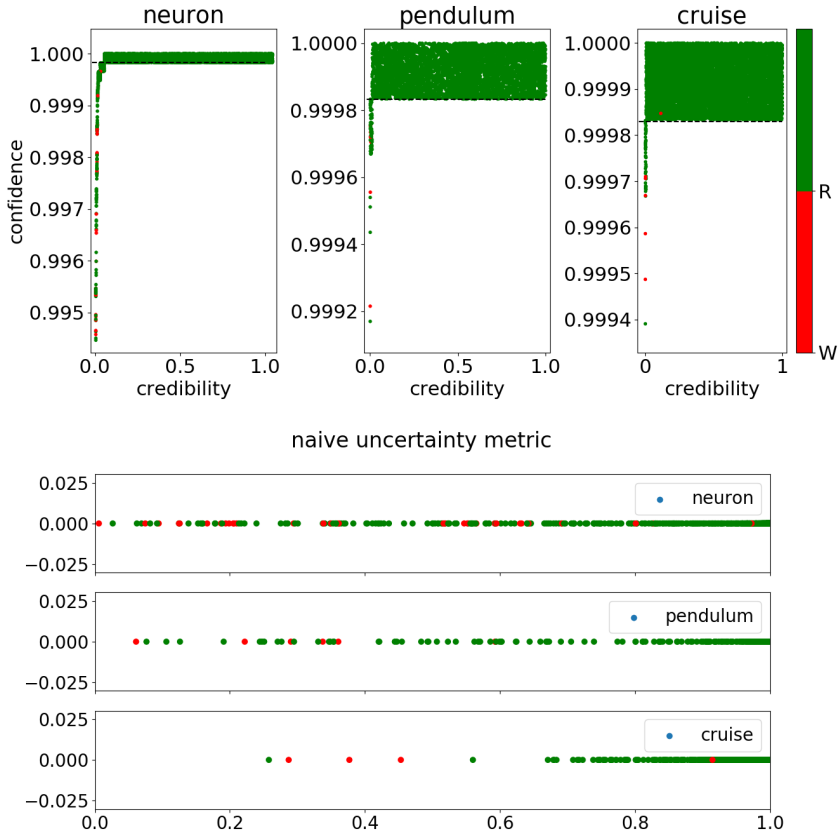

Fig. 5. Experimental superiority of conformal predictions over naive discrimination based on the DNN class likelihood. **Top**: Confidence-credibility pairs for the test datasets. The horizontal dashed line indicates the empirical and qualitative choice of ϵ. **Bottom**: Values of the naive uncertainty metric for the test datasets. In both cases (top and bottom) the true test labels were used to check the performances of the uncertainty metrics a posteriori. Green dots indicate properly classified points, red dots misclassified points.

indicate uncertain predictions. Although this simple approach does not provide any statistical guarantee, we may still look for a rejection threshold that allows us to reject the misclassified examples and keep the overall rejection rate low. However, Figure 5 (bottom) shows that this naive metric is not sufficiently discriminative, especially for the spiking neuron model. This supports our claim that a more principled method to measure uncertainty and define rejection criteria is needed. On the contrary, Figure 5 (top) shows that the values of confidence-credibility pairs for misclassified points are easily separated from the majority of properly classified points. Furthermore, the distribution of points in the

Model	ϵ	$P_\epsilon(e) + P_\epsilon(d)$	α	# accepted errors
neuron	0.000175	4.78%	0.9998	0/31
pendulum	0.000167	0.84%	0.9983	0/7
cruise	0.000170	0.55%	0.9998	1/6

Table 1. For each case study, a significance level ϵ was chosen qualitatively from Figure 5 (top), ignoring the colors, as we should not know the true labels of test points. We computed the fractions of empty, single and double prediction regions occurring along the entire test set. The sum $P_\epsilon(e) + P_\epsilon(d)$ gives the ratio of points rejected. $\alpha = 1 - \hat{\epsilon}$ is the statistical certainty level for accepted points/predictions. The last column counts how many errors, among all the errors made by the classifier in the test set, were not rejected by our criterion.

confidence-credibility plane helps us choose the proper value for ϵ, which leads to a statistically significant measure of uncertainty.

Table 1 summarizes the experimental performance of our rejection criterion on the three hybrid automata models. Setting an adequate threshold is very important. We choose the value of $1 - \epsilon$ that better distinguishes between points with low confidence and points with high confidence, shown with horizontal lines in Figure 5 (top). We successfully reject almost all prediction errors, with an overall rejection rate between 0.5% and 5%. The certainty value α is always greater than 99.83%, which demonstrates that our approach only accepts predictions that have very small probability of being incorrect.

6 Related Work

Even though research on reachability checking of hybrid systems [13, 1] has produced effective verification algorithms and tools [10, 7, 11], comparably little has been done to make these algorithms efficient for online analysis. Existing approaches are limited to restricted classes of models [8], or require handcrafted optimization of the HA's derivatives [2], or are efficient only for low-dimensional systems and simple dynamics [21]. NSC [19] (introduced in Section 2) overcomes these limitations because, by employing machine learning models, it is fully automated and its performance is not affected by the model size or complexity.

Applications of machine learning in verification include parameter synthesis of stochastic systems [5], techniques for inferring temporal logic specifications from examples [3], synthesis of invariants for program verification [12], and reachability checking of Markov decision processes [6].

A related approach to NSC is smoothed model checking [4], where Gaussian processes [20] are used to approximate the satisfaction function of stochastic models, i.e., mapping model parameters into the satisfaction probability of some specification. Smoothed model checking leverages Bayesian statistics to quantify prediction uncertainty, but faces scalability issues as the dimension of the system increases. On the contrary, our method for quantifying the reliability of NSC

predictions is very efficient, because its performance is nearly equivalent to that of the underlying machine learning model[7].

In Bayesian approaches to uncertainty/confidence estimation, one has to assume a prior distribution, which is often chosen arbitrarily. However, in order to guarantee accurate confidence values, the correct priors must be known. In fact, if the prior is incorrect, the confidence values have no theoretical base. The CP framework instead provides confidence information based only on the standard i.i.d. or exchangeability assumption. Avoiding Bayesian assumptions makes CP conclusions more robust to different underlying data distributions. In [17] the authors show empirically that the performance of CP is close to Bayes when the prior is known to be correct. Unlike Bayes, the CP method still gives accurate confidence values even when different data distributions are considered.

7 Conclusion

We applied the theory of conformal predictions to endow the neural state classification approach with a criterion to reject unreliable predictions, predictions that can lead safety-critical state classification errors. Our criterion leverages two statistically sound measures of uncertainty, i.e., confidence and credibility. By accepting only predictions that satisfy specific confidence and credibility thresholds, our criterion is conservative and allows making safe choices with respect to any state classifier, independently of the classifier's accuracy. In the experiments, our criterion successfully rejected almost all classification errors, and doing so very efficiently, with an average runtime of 3 ms per sample.

In future work, we will investigate automated methods to derive the rejection thresholds for confidence and credibility, and how to use this approach in an active learning framework to improve accuracy, reduce false negatives, and reduce the rejection rate.

References

1. Alur, R.: Formal verification of hybrid systems. In: 2011 Proceedings of the Ninth ACM International Conference on Embedded Software (EMSOFT). pp. 273–278. IEEE (2011)
2. Bak, S., Johnson, T.T., Caccamo, M., Sha, L.: Real-time reachability for verified simplex design. In: Real-Time Systems Symposium (RTSS), 2014 IEEE. pp. 138–148. IEEE (2014)
3. Bartocci, E., Bortolussi, L., Sanguinetti, G.: Data-driven statistical learning of temporal logic properties. In: International Conference on Formal Modeling and Analysis of Timed Systems. pp. 23–37. Springer (2014)
4. Bortolussi, L., Milios, D., Sanguinetti, G.: Smoothed model checking for uncertain continuous-time Markov chains. Information and Computation **247**, 235–253 (2016)

[7] Our approach reduces to computing two p-values. Each p-value is derived by computing a nonconformity score, which requires one execution of the underlying model, and one search over the array of calibration scores.

5. Bortolussi, L., Silvetti, S.: Bayesian statistical parameter synthesis for linear temporal properties of stochastic models. In: TACAS. pp. 396–413 (2018)
6. Brázdil, T., Chatterjee, K., Chmelík, M., Forejt, V., Křetínskỳ, J., Kwiatkowska, M., Parker, D., Ujma, M.: Verification of markov decision processes using learning algorithms. In: International Symposium on Automated Technology for Verification and Analysis. pp. 98–114. Springer (2014)
7. Chen, X., Ábrahám, E., Sankaranarayanan, S.: Flow*: An analyzer for non-linear hybrid systems. In: International Conference on Computer Aided Verification. pp. 258–263. Springer (2013)
8. Chen, X., Sankaranarayanan, S.: Model predictive real-time monitoring of linear systems. In: Real-Time Systems Symposium (RTSS), 2017 IEEE. pp. 297–306. IEEE (2017)
9. Chen, X., Schupp, S., Makhlouf, I.B., Ábrahám, E., Frehse, G., Kowalewski, S.: A benchmark suite for hybrid systems reachability analysis. In: NASA Formal Methods Symposium. pp. 408–414. Springer (2015)
10. Frehse, G., Le Guernic, C., Donzé, A., Cotton, S., Ray, R., Lebeltel, O., Ripado, R., Girard, A., Dang, T., Maler, O.: SpaceEx: Scalable verification of hybrid systems. In: International Conference on Computer Aided Verification. pp. 379–395. Springer (2011)
11. Gao, S., Kong, S., Clarke, E.M.: dReal: An SMT solver for nonlinear theories over the reals. In: International Conference on Automated Deduction. pp. 208–214. Springer (2013)
12. Garg, P., Neider, D., Madhusudan, P., Roth, D.: Learning invariants using decision trees and implication counterexamples. ACM Sigplan Notices **51**(1), 499–512 (2016)
13. Henzinger, T.A., Kopke, P.W., Puri, A., Varaiya, P.: What's decidable about hybrid automata? Journal of computer and system sciences **57**(1), 94–124 (1998)
14. Legay, A., Delahaye, B., Bensalem, S.: Statistical Model Checking: An Overview. RV **10**, 122–135 (2010)
15. Lehmann, E.L., Romano, J.P.: Testing statistical hypotheses. Springer Science & Business Media (2006)
16. Linusson, H., Johansson, U., Boström, H., Löfström, T.: Reliable confidence predictions using conformal prediction. In: Pacific-Asia Conference on Knowledge Discovery and Data Mining. pp. 77–88. Springer (2016)
17. Melluish, T., Saunders, C., Nouretdinov, I., Vovk, V.: The typicalness framework: a comparison with the bayesian approach. University of London, Royal Holloway (2001)
18. Papadopoulos, H.: Inductive conformal prediction: Theory and application to neural networks. In: Tools in artificial intelligence. InTech (2008)
19. Phan, D., Paoletti, N., Zhang, T., Grosu, R., Smolka, S.A., Stoller, S.D.: Neural state classification for hybrid systems. In: Automated Technology for Verification and Analysis. LNCS, vol. 11138, pp. 422–440 (2018)
20. Rasmussen, C.E., Williams, C.K.: Gaussian processes for machine learning, vol. 1. MIT press Cambridge (2006)
21. Sauter, G., Dierks, H., Fränzle, M., Hansen, M.R.: Lightweight hybrid model checking facilitating online prediction of temporal properties. In: Proceedings of the 21st Nordic Workshop on Programming Theory. pp. 20–22 (2009)
22. Vovk, V., Gammerman, A., Shafer, G.: Algorithmic learning in a random world (2005)
23. Wald, A.: Sequential tests of statistical hypotheses. The Annals of Mathematical Statistics **16**(2), 117–186 (1945)