Author for correspondence:
Dimitrios Boursinos,
E-mail: dimitrios.boursinos@vanderbilt.edu

CAMBRIDGE
UNIVERSITY PRESS

Assurance monitoring of learning-enabled cyber-physical systems using inductive conformal prediction based on distance learning

Dimitrios Boursinos and Xenofon Koutsoukos

Department of Electrical Engineering and Computer Science, Institute for Software Integrated Systems, Vanderbilt University, Nashville, TN, USA

Abstract

Machine learning components such as deep neural networks are used extensively in cyber-physical systems (CPS). However, such components may introduce new types of hazards that can have disastrous consequences and need to be addressed for engineering trustworthy systems. Although deep neural networks offer advanced capabilities, they must be complemented by engineering methods and practices that allow effective integration in CPS. In this paper, we proposed an approach for assurance monitoring of learning-enabled CPS based on the conformal prediction framework. In order to allow real-time assurance monitoring, the approach employs distance learning to transform high-dimensional inputs into lower size embedding representations. By leveraging conformal prediction, the approach provides well-calibrated confidence and ensures a bounded small error rate while limiting the number of inputs for which an accurate prediction cannot be made. We demonstrate the approach using three datasets of mobile robot following a wall, speaker recognition, and traffic sign recognition. The experimental results demonstrate that the error rates are well-calibrated while the number of alarms is very small. Furthermore, the method is computationally efficient and allows real-time assurance monitoring of CPS.

Introduction

Cyber-physical systems (CPS) can benefit by incorporating machine learning components that can handle the uncertainty and variability of the real world. Typical components such as deep neural networks (DNNs) can be used for performing various tasks such as the perception of the environment. In autonomous vehicles, for example, perception components aim at making sense of the surroundings like recognizing correctly traffic signs. However, such DNNs introduce new types of hazards that can have disastrous consequences and need to be addressed for engineering trustworthy systems. Although DNNs offer advanced capabilities, they must be complemented by engineering methods and practices that allow effective integration in CPS.

A DNN is designed using learning techniques that require specification of the task, a measure for evaluating how well the task is performed, and experience which typically includes training and testing data. Using the DNN during system operation presents challenges that must be addressed using innovative engineering methods. The perception of the environment is a functionality that is difficult to specify, and typically, specifications are based on examples. DNNs exhibit some nonzero error rate, the true error rate is unknown, and only an estimate from a design-time statistical process is known. Furthermore, DNNs encode information in a complex manner and it is hard to reason about the encoding. Nontransparency is an obstacle to monitoring because it is more difficult to have confidence that the model is operating as intended.

Our objective in this paper is to complement the prediction of DNNs with a computation of confidence that can be used for decision making. We consider DNNs used for classification in CPS. In addition to the class prediction, we compute set predictors with a given confidence using the conformal prediction framework (Balasubramanian *et al.*, 2014). We focus on computationally efficient algorithms that can be used for real-time monitoring. An efficient and robust approach must ensure a small and well-calibrated error rate while limiting the number of alarms. This enables the design of monitors which can ensure a bounded small error rate while limiting the number of inputs for which an accurate prediction cannot be made.

The proposed approach is based on conformal prediction (CP; Vovk *et al.*, 2005; Balasubramanian *et al.*, 2014). CP aims at associating reliable measures of confidence with set predictions for problems that include classification and regression. An important feature of the CP framework is the calibration of the obtained confidence values in an online setting which is very promising for real-time monitoring in CPS applications. These methods can be applied for a variety of machine learning algorithms that include DNNs. The main idea is to

CrossMark

test if a new input example conforms to the training dataset by utilizing a *nonconformity measure* (NCM) which assigns a numerical score indicating how different the input example is from the training dataset. The next step is to define a *p*-value as the fraction of observations that have nonconformity (NC) scores greater than or equal to the NC scores of the training examples which is then used for estimating the confidence of the prediction for the test input. In order to use the approach online, inductive conformal prediction (ICP) has been developed for computational efficiency (Papadopoulos *et al.*, 2007; Balasubramanian *et al.*, 2014). In ICP, the training dataset is split into the proper training dataset that is used for learning and a calibration dataset that is used to compute the predictions for given confidence levels. Existing methods rely on NCMs computed using techniques such as *k*-nearest neighbors (*k*-NN) and Kernel Density Estimation and do not scale for high-dimensional inputs in CPS.

DNNs have the ability to compute layers of representations of the input data which can then be used to distinguish between available classes (Hinton, 2007; Bengio, 2009). In our previous work, we developed an approach for mapping high-dimensional inputs into lower-dimensional representations to make the application of ICP possible for assurance monitoring of CPS in real time (Boursinos and Koutsoukos, 2020*a*). The approach utilizes the vector of the neuron activations in the penultimate layer of the DNN for a particular input. This low-dimensional representation can be used to compute NC scores efficiently for high-dimensional inputs. In problems where the input data are high-dimensional, such as the classification of traffic sign images in autonomous vehicles, ICP based on these learned embedding representations produces confident predictions. Moreover, the execution time and the required memory are significantly lower than using the original inputs, and the approach can be used for real-time assurance monitoring of the DNN. The use of low-dimensional learned embedding representations results in improved performance compared with ICP based on the original inputs. However, the underlying DNN is still trained to perform classification and does not learn necessarily optimal representations for computing NC scores.

The main challenge addressed in this paper is the efficient computation of embedding representations that allows assurance monitoring based on conformal prediction in real time. The novelty of the approach lies on using distance metric learning to generate representations of the input data and use Euclidean distance as a measure of similarity. Unlike training a classifier where each training input is assigned a ground truth label and the objective is to minimize a loss function so that the prediction of the classifier will be the same as the label, in distance metric learning, the inputs are considered in pairs. The associated loss function is defined using pairwise constraints such that its minimization will make representations of inputs that belong to the same class be close to each other and representations of inputs belonging to different classes be far from each other. Preliminary results on using appropriate representations for a robotic navigation benchmark with low-dimensional inputs are presented in Boursinos and Koutsoukos (2020*c*).

The main contribution of the paper is the leverage of distance metric learning for assurance monitoring of learning-enabled CPS. The proposed approach based on ICP can be used in real time for high-dimensional data that are typically used in CPS. Different NC functions can be used in ICP to evaluate whether new unknown inputs are similar to the data that have been used for training a learning-enabled component such as DNN. An NC function assigns a score to a labeled input reflecting how well it

conforms to the training dataset. Because the choice of the NC function is very important, the proposed approach utilizes neural network architectures for distance metric learning based on siamese (Koch *et al.*, 2015) and triplet networks (Hoffer and Ailon, 2015) to learn representations and define NC functions based on the Euclidean distance. Specifically, the proposed functions compute the NC scores of a new labeled input using (1) the labels of its closest neighbors, (2) how far the closest neighbor of the same class is compared with any other neighbor, and (3) how far the label's centroid is compared with the centroids of the other labels. The main benefit of the approach is that by utilizing distance metric learning in ICP, we reduce the computational requirements without sacrificing accuracy or efficiency.

An important advantage of the approach is that it allows the computation of the optimal significance level that can be used by the assurance monitor to ensure a bounded error rate while limiting the number of inputs for which an accurate prediction cannot be made. Unlike most common machine learning classifiers that assign a single label to an input, ICP computes a set of candidate labels that contains the correct class given a selected significance level. Small significance level values reduce the classification errors but may result in set predictors with multiple candidate labels. In autonomous systems, it is not only important to have predictions with well-calibrated confidence but also to be able to choose the desired significance level based on the application requirements. Even though reducing the number of possible classes may be helpful when the information is provide to a human, in an autonomous system, it is desirable that the prediction is unique. Therefore, we assume that set predictions that contain multiple classes lead to a rejection of the input and require human intervention. For this reason, it is desirable to minimize the number of test inputs with multiple predictions. If the prediction is unique, then the monitor ensures a confident prediction with well-calibrated error rate defined by the significance level. If the predicted set contains multiple predictions, the monitor rejects the prediction and raises an alarm. Finally, if the predicted set is empty, the monitor indicates that no label is probable. We distinguish between multiple and no predictions, because they may lead to different action in the system. For example, no prediction may be the result of out-of-distribution inputs while multiple possible predictions may be an indication that the significance level is smaller than the accuracy of the underlying DNN.

The paper presents a comprehensive empirical evaluation of the approach using three datasets for classification problems in CPS of increasing complexity. The first dataset is the SCITOS-G5 robot navigation dataset (Dua and Graff, 2017) for which we use a fully connected feedforward network architecture. The second is a speech recognition dataset which contains audio files of human speech (Kiplagat, n.d.). For this problem, we learn the embedding representations using a DNN with 1D convolutional layers. The third dataset is the German Traffic Sign Recognition Benchmark (GTSRB; Stallkamp *et al.*, 2012). For this dataset, we use a modified version of the VGG16 architecture (Simonyan and Zisserman, 2014) to learn and generate the embedding representations. We used different combinations of NC functions and distance metric learning architectures and compare them with ICP without distance metric learning. The results demonstrate that the selected or computed significance levels bound the error rate in all cases. Moreover, the representations learned by the siamese or triplet networks result in well-formed clusters for different classes and individual training data typically can be captured by their class centroid. Such representations reduce the memory requirements and

the execution time overhead while still ensure a bounded small error rate with a limited number of prediction sets containing multiple candidate labels.

Related work on confidence estimation and well-calibrated models for different kind of machine learning methods is presented in the "Related work" section. In the section "Problem formulation", we define the problem and present the proposed architecture. Sections "Distance learning", "ICP based on distance learning", and "Assurance monitoring" present the details of ICP based on distance learning and assurance monitoring. Finally, we evaluate the performance of our suggested approach on three different applications in the section "Evaluation".

Related work

Machine learning components tend to be poorly calibrated. Modern, commonly used DNN architectures typically have a softmax layer to produce a probability-like output for each class. The chosen class is the one with the highest probability; however, this generated probability measure is often higher than the actual posterior probability that the prediction is correct. Other factors that affect the calibration in DNNs are the depth, width, weight decay, and Batch Normalization (Guo et al., 2017). The estimation of accurate error-rate bounds is important as it provides assurance guarantees in safety-critical applications but also makes the decision confidence interpretable by humans. Several approaches have been proposed that compute well-calibrated confidence metrics in different ways, like scaling the DNN softmax outputs or other post-processing algorithms.

The calibration methods generally belong to two categories: parametric and nonparametric. The parametric methods assume that the probabilities follow certain well-known distributions whose parameters are to be estimated from the training data. The Platt's scaling method (Platt, 1999) is proposed for the calibration of Support Vector Machine (SVM) outputs. After the training of an SVM, the method computes the parameters of a sigmoid function to map the outputs into probabilities. Piecewise logistic regression is an extension of Platt scaling and assumes that the log-odds of calibrated probabilities follow a piecewise linear function (Zhang and Yang, 2004). Another variant of Platt scaling is temperature scaling (Guo et al., 2017) which can be applied in DNNs with a softmax output layer. After training of a DNN, a temperature scaling factor T is computed on a validation set to scale the softmax outputs. However, while temperature scaling achieves good calibration when the data in the validation dataset are independent and identically distributed (IID), there is no calibration guarantee under distribution shifts (Ovadia et al., 2019). Experiments in Kumar et al. (2019) show that Platt scaling and temperature scaling are not well-calibrated as it is reported and it is difficult to know how miscalibrated they are.

Histogram binning or quantile binning is a commonly used non-parametric approach with either equal-width or equal-frequency bins. It divides the outputs of a classifier into bins and computes the calibrated probability as the ratio of correct classifications in each bin (Zadrozny and Elkan, 2001). Isotonic Regression is a generalization of histogram binning by jointly optimizing the bin boundaries and bin predictions (Zadrozny and Elkan, 2002). An extention of isotonic regression is a method called ensemble of near-isotonic regression (ENIR) that uses selective Bayesian averaging to ensemble the near-isotonic regression models (Naeini and Cooper, 2018). Adaptive calibration of predictions (ACPs) also use the ratio of correct classifications as the posterior probability in each bin, but

it obtains bins from a 95% confidence interval around each individual prediction (Jiang et al., 2012). Estimating calibrated probability is a more significant issue in class imbalance and class overlap problems. Receiver Operating Characteristics (ROC) Binning uses the ROC curves to construct equal-width bins that provide accurate calibrated probabilities that are robust to changes in the prevalence of the positive class (Sun and Cho, 2018). Bayesian binning into quantiles (BBQs) extend the simple histogram-binning calibration method by considering multiple equal-frequency Histogram Binning models and their combination as the calibration result (Naeini et al., 2015).

Another framework developed to produce well-calibrated confidence values is the CP (Vovk et al., 2005; Shafer and Vovk, 2008; Balasubramanian et al., 2014). The conformal prediction framework can be applied to produce calibrated confidence values with a variety of machine learning algorithms with slight modifications. Using CP together with machine learning models such as DNNs is computationally inefficient. In Papadopoulos et al. (2007), the authors suggest a modified version of the CP framework, ICP that has less computational overhead and they evaluate the results using DNNs as undelying model. Deep k-nearest neighbors (DkNN) is an approach based on ICP for classification problems that uses the activations from all the hidden layers of a neural network as features (Papernot and McDaniel, 2018). The method is based on the assumption that when a DNN makes a wrong prediction, there is a specific hidden layer that generated intermediate results that lead to the wrong prediction. Taking into account all the hidden layers can lead to better interpretability of the predictions. In Johansson et al. (2013), the authors present an empirical investigation of decision trees as conformal predictors and analyzed the effects of different split criteria, such as the Gini index and the entropy, on ICP. There are similar evaluations using ICP with random forests (Devetyarov and Nouretdinov, 2010; Bhattacharyya, 2013) as well as SVMs (Makili et al., 2011). The above methods are applied to datasets and show good results when the input data are IID. In Boursinos and Koutsoukos (2020b), we showed that ICP underperforms when the input data are sequential. Individual frames of a sequence might contain partial information regarding the input and more frames might be needed for ICP to reach a confident prediction. The performance of ICP in this case can be improved by designing a feedback-loop configuration that queries the sensors until a single confident decision can be reached.

Confidence bounds can also be generated for regression problems. In this case instead of sets of multiple candidate labels, we have intervals around a point prediction that include the correct prediction with a desired confidence. There are ICP methods for regression problems with different underlying machine learning algorithms. In Papadopoulos et al. (2011), the authors use the k-nearest neighbors regression (k-NNR) as a predictor and evaluate the effects of different nonconformity functions. Random forests can also be used in regression problems. In Johansson et al. (2014), there is a comparison on the generated confidence bounds using k-NNR and DNNs (Papadopoulos and Haralambous, 2011). An alternative framework used to compute confidence bounds on regression problems is the Simultaneous Confidence Bands. The method presented in Sun and Loader (1994) generates linear confidence bounds centered around the point prediction of a regression model. In this approach, the model used for predictions has to be estimated by a sum of linear models. Models that satisfy this condition are the least squares polynomial models, kernel methods, and smoothing splines. Functional principal
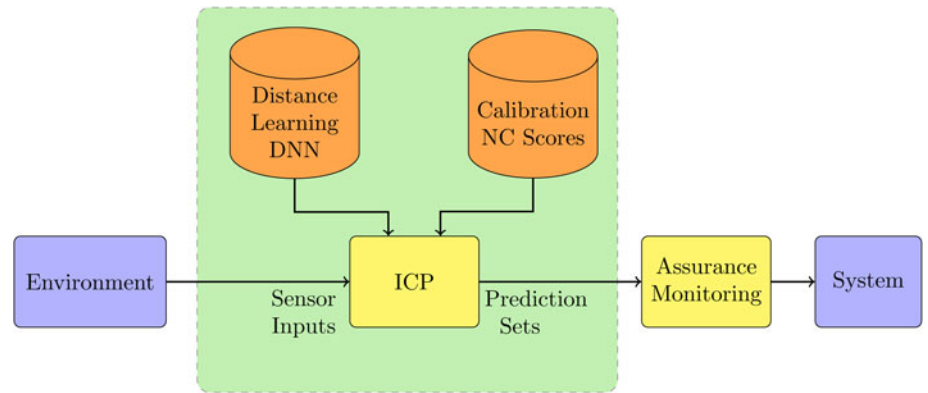
Fig. 1. Assurance monitoring using ICP based on distance learning.

components (FPC) analysis can be used for the decomposition of an arbitrary regression model to a combination of linear models (Goldsmith *et al.*, 2013).

The findings of the state-of-the-art methods described above illustrate the significance of computing well-calibrated and accurate confidence measures. Typically, the main objective is to complement existing machine learning models that are generally unable to produce an accurate estimation of confidence for their predictions with post-processing techniques in order to compute well-calibrated probabilities. An important advantage of such approaches is that they are independent of the underline predictive machine learning models. Therefore, there is no need to redesign and optimize the objective functions used for training which could lead to optimization tasks with high computational complexity.

Computing well-calibrated confidence is extremely important for designing autonomous systems because accurate measures of confidence are necessary to estimate the risk associated with each decision. The main limitation of existing methods comes from the fact that it is very difficult to select desired confidence values according to the application requirements and ensure bounded error rate. This is especially important in autonomous CPS applications where decisions can be safety critical. Another important challenge is to investigate how the computed confidence measures can be used for decision making by autonomous systems and how to handle data for which a confident decision cannot be taken.

The proposed work based on ICP produces prediction sets and computes a significance level that will bound the expected error rate. Similar to existing methods, since the approach is based on ICP, it can be used with any machine learning component without the need of retraining. ICP methods provide very promising results especially when the input data are not very high-dimensional and there are not stringent time constraints. However, ICP can be impractical when the inputs are, for example, images because of the excessive memory requirements and high execution times. The proposed approach aims to learn appropriate lower-dimensional representations of high-dimensional inputs that make the task of computing confidence measures based on similarities much easier.

Problem formulation

A perception component in a CPS aims to observe and interpret the environment in order to provide information for decision making. For example, in autonomous vehicles, a DNN can be used to classify traffic signs. The problem is to complement the prediction of the DNN with a computation of confidence. An efficient and robust approach must ensure a small and well-calibrated error rate while limiting the number of alarms to enable real-time monitoring. That is, maximize the autonomous operation time while keeping the error-rate bounded according to the application requirements. Finally, the computation of well-calibrated predictions must be computationally efficient for applications with high-dimensional inputs that require fast decision as, for example, in autonomous vehicles.

During the system operation of a CPS, inputs arrive one by one. After receiving each input, the objective is to compute a valid measure of the confidence of the prediction. The objective is twofold: (1) provide guarantees for the error rate of the prediction and (2) design a monitor which limits the number of input examples for which a confident prediction cannot be made. Such a monitor can be used, for example, by generating warnings that require human intervention.

The conformal prediction framework allows computing set predictors for a given confidence expressed as a significance value (Balasubramanian *et al.*, 2014). The confidence is generated by comparing how similar a test is to the training data using different nonconformity functions. In our previous work (Boursinos and Koutsoukos, 2020c) we used DNNs to produce embedding representations for more efficient application of ICP. The additional problem we are solving is the computation of appropriate embedding representations that will lead to more confident decisions. The proposed approach is illustrated in Figure 1. The main idea is to use distance learning and enable DNNs to learn a lower-dimensional representation for each input on an embedding space where the Euclidean distance between the input representations is a measure of similarity between the original inputs themselves. The ICP approach is applied using the low-dimensional embedding representations and estimates the similarity between a new input and the available data in the training set using an NC function. Using such a representation not only reduces the execution time and the memory requirements but is also more efficient in producing useful predictions. Based on a chosen significance level, ICP generates a set of possible predictions. If the computed set contains a single prediction, the confidence is a well-calibrated and a valid indication of the expected error. If the computed set contains multiple predictions or no predictions, an alarm can be raised to indicate the need for additional information.

In CPS, it is desirable to minimize the number of alarms while performing the required computations in real time. An evaluation of the method must be based on metrics that quantify the error rate, the number of alarms, and the computational efficiency.
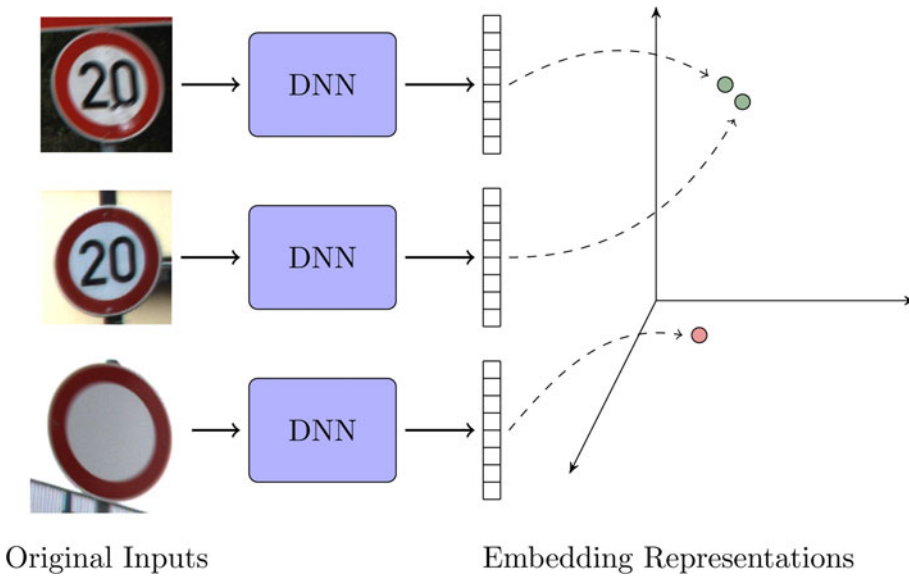
For real-time operation, the time and memory requirements of the monitoring approach must be similar to the computational requirements of the DNNs used in the CPS architecture. Figure 1 illustrates the proposed architecture for assurance monitoring. At design time, a DNN is trained to produce embedding representations using distance metric learning techniques. Then, NC scores are computed for a labeled calibration set that is not used for training of the DNN. During system operation, the assurance monitor employs the trained DNN to map new sensor inputs to lower-dimensional representations. Using the NC scores of the calibration data, the method produces prediction sets including well-calibrated confidence of the predictions. Ideally, a prediction set should include exactly one class to enable decision making. Alarms can be raised if either the prediction set include multiple possible classes or if it does not contain any.

Distance learning

The ICP framework requires computing the similarity between the training data and a test input. This can be done efficiently by learning representations of the inputs for which the Euclidean distance is a metric of similarity, meaning that similar inputs will be close to each other as illustrated in Figure 2. There are different approaches based on DNN architectures that generate embedding representations for distance metric learning.

A *siamese network* is composed using two copies of the same neural network with shared parameters (Koch *et al.*, 2015) as shown in Figure 3a. During training, each identical copy of the siamese network is fed with different training samples x_1 and x_2 belonging to classes y_1 and y_2. The embedding representations produced by each network copy are $r_1 = \text{Net}(x_1)$ and $r_2 = \text{Net}(x_2)$. The learning goal is to minimize the Euclidean distance between the embedding representations of inputs belonging to the same class and maximize it for inputs belonging to different classes as described below:

$$\begin{cases} \min d(r_1, r_2), & \text{if } y_1 = y_2, \\ \max d(r_1, r_2), & \text{otherwise.} \end{cases} \quad (1)$$

This optimization problem can be solved using the *contrastive loss*
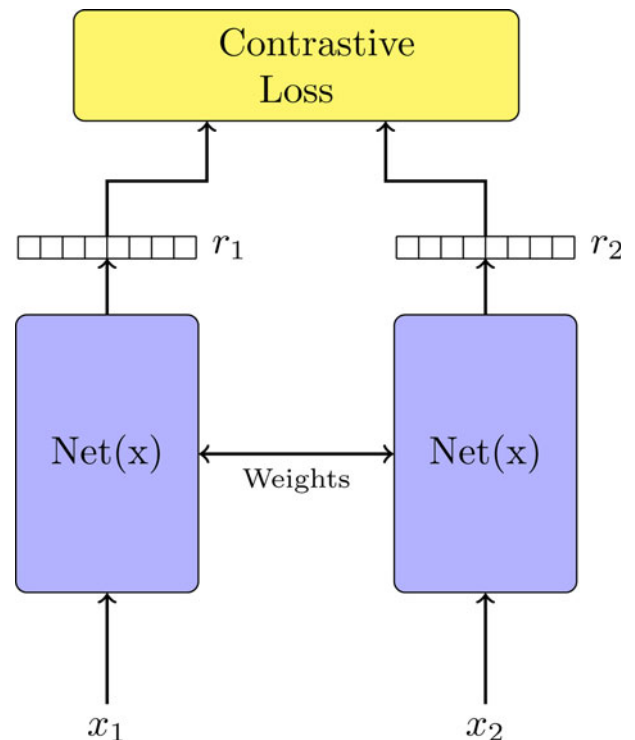
Fig. 3. (a) Siamese network architecture and (b) triplet network architecture.

function (Melekhov *et al.*, 2016):

$$L(r_1, r_2, y) = y \cdot d(r1, r2) + (1 - y) \max [0, m - d(r_1, r_2)],$$

where y is a binary flag equal to 0 if $y_1 = y_2$ and to 1 if $y_1 \neq y_2$ and m is a margin parameter. In particular, when $y_1 \neq y_2$, $L = 0$, when $d(r_1, r_2) \geq m$, otherwise the parameters of the network are updated to produce more distant representations for those two elements. The reason behind the use of the margin is that when the distance between pairs of different classes are large enough and at most m, there is no reason to update the network to put

the representations even further away from each other and instead focus the training on harder examples.

Another architecture trained to produce embedding representations for distance learning is the *triplet network* (Hoffer and Ailon, 2015). A triplet is composed using three copies of the same neural network with shared parameters as shown in Figure 3b. The training examples consist of three samples, the anchor sample x, the positive sample x^+, and the negative sample x^-. The samples x and x^+ belong to the same class, while x^- belongs to a different class. The embedding representations produced by each network copy will be $r = \text{Net}(x)$, $r^+ = \text{Net}(x^+)$, and $r^- = \text{Net}(x^-)$. The optimization problem described by Eq. (1) is solved by training the triplet network copies using the *triplet loss* function:

$$L(r, r^+, r^-) = \max\left[d(r, r^+) - d(r, r^-) + m, 0\right].$$

The margin parameter m separates pairs of different classes by at most m and it is used so that the network parameters will not be updated trying to push a pair even further away when a positive sample is already at least m closer to an anchor than a negative sample. Instead, the training is more efficient when harder triplets are used. The input triplets to the network copies can be sampled randomly from the training data. However, as training progresses it is harder to randomly find triplets that produce $L(r, r^+, r^-) > 0$ that will update the triplet network parameters. This leads to slow training and underfitted models. The training can be improved by carefully mining the training data that produce a large loss (Xuan *et al.*, 2019). For each training iteration, first, the anchor training data are randomly chosen. For each anchor, the hardest positive sample is chosen, meaning a sample from the same class as the anchor that is located the furthest away from the anchor. Then, the triplets are formed by mining hard negative samples that satisfy $d(r, r^-) < d(r, r^+)$ or semi-hard negatives that satisfy $d(r, r^-) < d(r, r^+) + m$. This way the formed triplet batches will produce gradients to update the shared weights between the DNN copies.

ICP based on distance learning

We consider a training set $\{z_1, \ldots, z_l\}$ of examples, where each $z_i \in Z$ is a pair (x_i, y_i) with x_i the feature vector and y_i the label of that example. For a given unlabeled input x_{l+1} and a chosen significance level ϵ, the task is to compute a prediction set Γ^ϵ for which $P(y_{l+1} \notin \Gamma^\epsilon) < \epsilon$, where y_{l+1} is the ground truth label of the input x_{l+1}. ICP computes well-calibrated prediction sets with the underlying assumption that all examples (x_i, y_i), $i = 1, 2, \ldots$ are IID generated from the same but typically unknown probability distribution.

Central to the application of ICP is a *nonconformity function* or NCM which shows how different a labeled input is from the examples in the training set. For a given test example z_i with candidate label $\tilde{y}_i$, an NC function assigns a numerical score indicating how different the example z_i is from the examples in $\{z_1, \ldots, z_{i-1}, z_{i+1}, \ldots, z_n\}$. There are many possible NC functions that can be used (Vovk *et al.*, 2005; Shafer and Vovk, 2008; Balasubramanian *et al.*, 2014; Johansson *et al.*, 2017; Boursinos and Koutsoukos, 2020a). For example, an NC function can be defined as the number of the k-NN to z_{l+1} in the training set with label different from the candidate label $\tilde{y}_{l+1}$ (k-NN NCM). The input space is often high-dimensional which makes storing the whole training set impractical and the computation of the NC scores inefficient. To address this challenge, the proposed approach leverages

distance metric learning methods to learn representations that enable applying ICP in real time.

Nonconformity functions that can be defined in the embedding space learned by siamese and triplet networks are (1) the k-NN (Papernot and McDaniel, 2018), (2) the *one Nearest Neighbor* (1-NN; Vovk *et al.*, 2005), and (3) the *Nearest Centroid* (Balasubramanian *et al.*, 2014). The k-NN NCM finds the k most similar examples of a test input x in the training data and counts how many of those are labeled different from the candidate label y. We denote $f : X \to V$ the mapping from the input space X to the embedding space V defined by either a siamese or a triplet network. Using the trained neural network, the encodings $v_i = f(x_i)$ are computed and stored for all the training data x_i. Given a test input x with encoding $v = f(x)$, we compute the k-NN in V and store their labels in a multi-set Ω. The k-NN NCM of input x with a candidate label y is defined as

$$\alpha(x, y) = |i \in \Omega : i \neq y|.$$

The 1-NN NCM requires to find the most similar example of a test input x in the training set that is labeled the same as the candidate label y as well as the most similar example in the training set that belongs to any class other than y and is defined as

$$\alpha(x, y) = \frac{\min_{i=1,\ldots,n : y_i = y} d(v, v_i)}{\min_{i=1,\ldots,n : y_i \neq y} d(v, v_i)}$$

where $v = f(x)$, $v_i = f(x_i)$, and d is the Euclidean distance metric in the V space.

The nearest centroid NCM simplifies the task of computing individual training examples that are similar to a test input when there is a large amount of training data. We expect examples that belong to a particular class to be close to each other in the embedding space, so for each class y_i, we compute its centroid $\mu_{y_i} = \sum_{j=1}^{n_i} v_j^i / n_i$, where v_j^i is the embedding representation of the jth training example from class y_i and n_i is the number of training examples in class y_i. The NC function is then defined as

$$\alpha(x, y) = \frac{d(\mu_y, v)}{\min_{i=1,\ldots,n : y_i \neq y} d(\mu_{y_i}, v)}$$

where $v = f(x)$. It should be noted that for computing the nearest centroid NCM, we need to store only the centroid for each class.

The NC score is an indication of how uncommon a test input is compared with the training data. Input data that come from the same distribution as the training data will produce low NC scores and are expected to lead to more confident classifications while unusual inputs will have higher NC score. However, this measure does not provide clear confidence information by itself, but it can be used by comparing it with NCM scores computed using a *calibration set* of known labeled data. Consider the training set $\{z_1, \ldots, z_l\}$. This set is split into two parts, the proper training set $\{z_1, \ldots, z_m\}$ of size $m < l$ that will also be used for the training of the siamese or triplet network and the calibration set $\{z_{m+1}, \ldots, z_l\}$ of size $l - m$. The NC scores $a(x_i, y_i)$, $i = m + 1, \ldots, l$, of the examples in the calibration set are computed and stored before applying the online monitoring algorithm. Given a test input x

with an unknown label y, the method generates a set $|\Gamma^\epsilon|$ of possible labels $\tilde{y}$ so that $P(y \notin |\Gamma^\epsilon|) < \epsilon$. For all the candidate labels $\tilde{y}$, ICP computes the empirical p-value defined as

$$p_j(x) = \frac{|\{\alpha \in A : \alpha \geq \alpha(x, j)\}|}{|A|}.$$

which is the fraction of NC scores of the calibration data that are equal or larger than the NC score of a test input. A candidate label is added to Γ^ϵ if $p_j(x) > \epsilon$. It is shown in Balasubramanian *et al.* (2014) that the prediction sets computed by ICP are valid, that is, the probability of error will not exceed ϵ for any $\epsilon \in [0, 1]$ for any choice of NC function. Our approach focuses on computing small prediction sets in an efficient manner that allow assurance monitoring approach in real time.

Assurance monitoring

In CPS, it is not only important to have predictions with well-calibrated confidence but also to be able to choose the desired significance level based on the application requirements. ICP computes a prediction set Γ^ϵ with a chosen significance level ϵ and Γ^ϵ may include any subset of all possible classes. Even though reducing the number of possible classes may be helpful when the information is provided to a human, in an autonomous system it is desirable that the prediction is unique, that is, $|\Gamma^\epsilon| = 1$. Therefore, we assume that set predictions that contain multiple classes, that is, $|\Gamma^\epsilon| > 1$, lead to a rejection of the input and require human intervention. For this reason, it is desirable to minimize the number of test inputs with multiple predictions and we define a monitor with output defined as

$$\text{out} = \begin{cases} 0, & \text{if } |\Gamma^\epsilon| = 0, \\ 1, & \text{if } |\Gamma^\epsilon| = 1, \\ \text{reject}, & \text{if } |\Gamma^\epsilon| > 1. \end{cases}$$

If the set Γ^ϵ contains a single prediction, the monitor outputs out = 1 to indicate a confident prediction with well-calibrated error rate ϵ. If the predicted set contains multiple predictions, the monitor rejects the prediction and raises an alarm. Finally, if the predicted set is empty, the monitor outputs out = 0 to indicate that no label is probable. We distinguish between multiple and no predictions, because they may lead to a different action in the system. For example, no prediction may be the result of out-of-distribution inputs while multiple possible predictions may be an indication that the significance level is smaller than the accuracy of the underlying DNN. Choosing a relatively small significance level that can consistently produce prediction sets with only one class is important. To do this, we apply ICP on the data in the calibration/validation set and compute the smallest significance level ϵ that does not produce any prediction set with $|\Gamma^\epsilon| > 1$. Assuming that the distribution of the test set is the same as the one of the calibration/validation set we expect the same value of ϵ to minimize the prediction sets with multiple classes on the test data.

The assurance monitoring approach is illustrated in Algorithms 1 and 2. Algorithm 1 shows the tasks that need to be performed at design time where, first, a distance metric learning network f is trained using the proper training set (X, Y) so that the computed embedding representations will form clusters for each class. Then, using the calibration data, both the NC

Algorithm 1. Training, calibration, and significance level computation

Require: training data (X, Y), calibration data (X^c, Y^c)
Require: DNN architecture f for distance metric learning
Require: Nonconformity function α
1: Train f using $(x, y) \in (X, Y)$ ▷ Training
2: // Compute the representations
3: $V = f(X)$
4: $V^c = f(X^c)$
5: // Compute the nonconformity scores for the calibration data
6: $A = \{\alpha(v^c, y^c) : (v^c, y^c) \in (V^c, Y^c)\}$ ▷ Calibration
7: **for** each v_i^c in V^c, $i = 1, \ldots, l - m$ **do**
8:      **for** each label $j \in 1, \ldots, n$
9:          Compute the nonconformity score $\alpha(v_i^c, j)$
10:          $p_{ij} = p_j(v_i^c) = \frac{|\{\alpha \in A : \alpha \geq \alpha(v_i^c, j)\}|}{|A|}$ ▷ empirical p-value
11:      **end for**
12:      Store all p_{ij}
13: **end for**
14: Compute ϵ such that for each $i \in [1, \ldots, l - m]$ no more that than 1 of the p-values $p_{ij} \geq \epsilon$

Algorithm 2. Assurance monitoring

Require: Nonconformity function α
Require: training data or centroids (V, Y) depending on the used nonconformity function α
Require: trained siamese or triplet neural network f for distance metric learning
Require: test input $z_t = (x_t, y_t)$
Require: significance level threshold ϵ
1: // Generate prediction sets for each test data x_t
2: Compute embedding representation $v_t = f(x_t)$
3: **for** each label $j \in 1, \ldots, n$ **do**
4:      Compute the nonconformity score $\alpha(v_t, j)$
5:      $p_j(z_t) = \frac{|\{\alpha \in A : \alpha \geq \alpha(z, j)\}|}{|A|}$ ▷ empirical p-value
6:      **if** $p_j(z) \geq \epsilon$
7:          Add j to the prediction set Γ^ϵ ▷ Γ^ϵ formation
8:      **end if**
9: **end for**
10: **if** $|\Gamma^\epsilon| = 0$ **then**
11:      **return** 0
12: **else if** $|\Gamma^\epsilon| = 1$ **then**
13:      **return** 1
14: **else**
15:      **return** Reject
16: **end if**

scores A and the optimal significance level ϵ are computed and stored. Algorithm 2 shows the tasks that are performed at runtime for a sensor input x_t. The input first needs to be mapped to its embedding representation v_t. Then, using the same NC function that is used for the calibration data, we compute the NC scores and the p-values assuming every label j as a candidate label. Then, the p-values p_j and ϵ are used to compute the set of candidate labels Γ^ϵ.

Evaluation

Our assurance monitor design leverages distance metric learning techniques to compress the input data to lower dimensions in order to make the ICP application more efficient and with lower memory requirements. The objective of the evaluation is to compare how the suggested architecture performs against the baseline

ICP approaches as well as investigate the validity/calibration and efficiency (size of set predictions).

Experimental setup

For the evaluation, we experiment with three datasets of variable complexity and input size. First, we use a dataset generated by the SCITOS-G5 mobile robot (Dua and Graff, 2017). This robot is equipped with 24 ultrasound sensors around it that are sampled at a rate of 9 samples per second. Its task is to navigate itself around a room counter-clockwise in close proximity to the walls. The possible actions the robot can take to accomplish this are "Move-Forward", "Sharp-Right-Turn", "Slight-Left-Turn", and "Slight-Right-Turn". The SCITOS-G5 dataset contains 5456 raw values of the ultrasound sensor measurements as well as the decision it took in each sample. Because of the small sensor number, the inputs have one dimension and their size is relatively small. Second, we use a speech recognition dataset which contains 7501 audio samples from speeches of five prominent leaders; Benjamin Netanyahu, Jens Stoltenberg, Julia Gillard, Margaret Thatcher, and Nelson Mandela, made available by the American Rhetoric (Kiplagat, n.d.). Each audio sample has 1 s duration, the sampling rate is 16 kHz and use pulse-code modulation (PCM) encoding. Third, the German Traffic Sign Recognition Benchmark (GTSRB) dataset is a collection of traffic sign images to be classified in 43 classes (each class corresponds to a type of traffic sign) (Stallkamp et al., 2012). The dataset has 26,640 labeled images of various sizes between 15×15 and 250×250 depending on the distance of the traffic sign to the vehicle. For all datasets, we split the available data so that 10% of the samples is used for testing. From the remaining 90% of the data, 80% is used for training and 20% for calibration and/or validation. In the ICP implementations that use the k-NN NC function, the number of neighbors k are chosen to be 20, 15,

and 40, respectively, for the three datasets, values that produce stability to outlier data points. The choice of DNN architectures happened according to the complexity of each application so that they will be simple enough to reduce the computational requirements but at the same time achieve good accuracy and data clustering without overfitting. All the experiments run in a desktop computer equipped with Intel(R) Core(TM) i9-9900K CPU, 32 GB RAM and a Geforce RTX 2080 GPU with 8 GB memory.

Baseline

The proposed approach assigns the original inputs into embedding representations for which the Euclidean distance is a measure of similarity between the inputs themselves. In order to understand the effect of the distance metric learning in ICP, we compare it with the approaches that we used in our previous work (Boursinos and Koutsoukos, 2020a). First, the most basic way of applying ICP is using only the original inputs. Then, we compare it with the approach that we presented in our previous work that uses embedding representations without distance metric learning and this will be the baseline in the following experiments.

The baseline approach computes the embedding representations using the activations of the penultimate layer of a DNN. A DNN is trained as a classifier to predict the class of the input data. The vector of activations of the neurons in the penultimate layer will be considered as the embedding representation of the input. In Figure 4, there is an illustration of how the embedding representations are generated in the baseline using a DNN with four input neurons that classify inputs to two possible classes. The embedding representations are generated in the penultimate layer and are typically reduced in size compared with the inputs. For an accurate comparison between the baseline and the
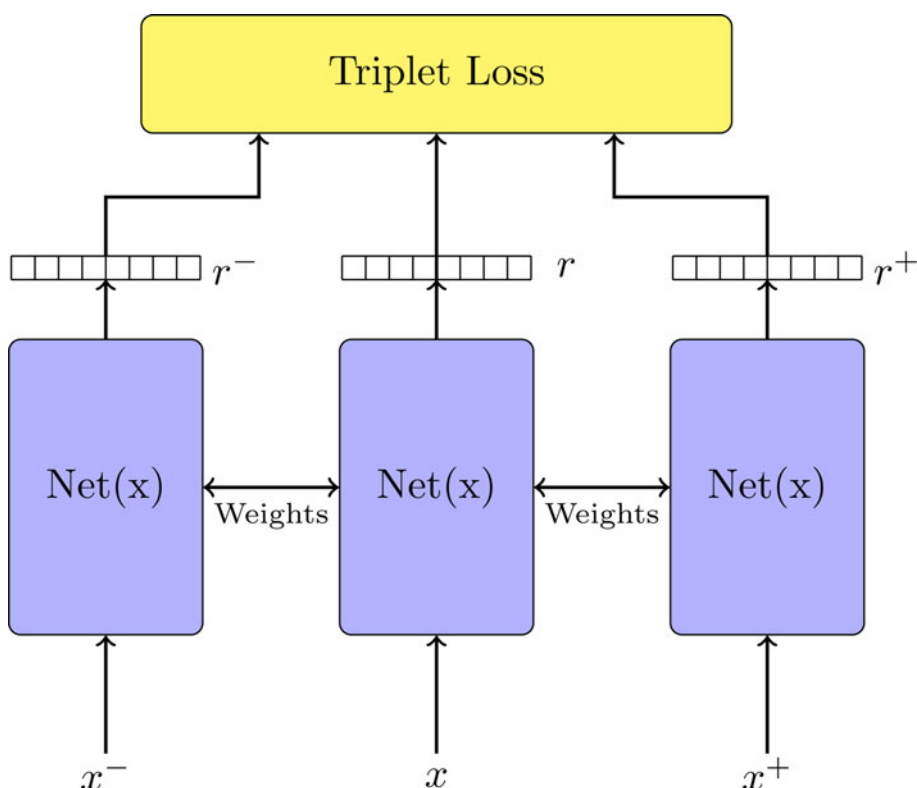

Fig. 4. Baseline DNN architecture.

proposed improvements using either the triplet or the siamese network, all of these approaches use the same DNN architecture, meaning that the embedding representations will also be of the same size.

Preprocessing and distance learning

The difficulty to compute the NC functions and the memory demands increase as the input size increases. Here, we see how the original high-dimensional inputs are mapped to lower-dimensional representations so that the application of ICP will be more efficient as well as the Euclidean distance between two embedding representations is a metric of similarity, property useful in the computation of the NC scores. We evaluate how the use of the embedding representations affect the application of ICP when it is applied on datasets of increasing complexity. First, the input data to the SCITOS-G5 mobile robot is a vector of 24 values. We use a fully connected feedforward DNN to generate embedding representations with size 8. The DNN is trained in either a siamese or triplet network for distance metric learning. The triplet network is trained without mining since this is a small dataset. Second, the speech recognition dataset contain audio samples with duration 1 s. For each audio sample, we add different kind of noises like dishwasher, running tap, and exercise bike on half the volume of the speech sample. Then, we use FFT to convert the audio samples to their frequency domain. The sampling rate of the speech files is 16 kHz, so in the frequency domain, it has 8000 components according to the Nyquist–Shannon sampling theorem (Shannon, 1949). A convolutional DNN is used to generate embedding representations of each audio wave in the frequency domain with size 32. In the case when the triplet architecture is used for the DNN's training, the semi-hard negatives mining produce the best results. Finally, the GTSRB dataset contains traffic sign images of variable sizes. In order to be able to use a single DNN to produce embedding representations for the image data, every image is either up-sampled by interpolation or down-sampled to $96 \times 96 \times 3$. A convolutional DNN is used to generate embedding representations with size 128. In the triplet case, the training produced better results when mining for hard negatives is used.

We first look at how well the distance metric learning methods cluster data of each class. A commonly used metric of the separation between classes is the *Silhouette* (Rousseeuw, 1987). For each sample, we first compute the mean distance between i and all other data points in the same cluster in the embedding space

$$a(i) = \frac{1}{|C_i| - 1} \sum_{j \in C_i, i \neq j} d(i, j).$$

Then, we compute the smallest mean distance from i to all the data points in any other cluster

$$b(i) = \min_{k \notin i} \frac{1}{|C_k|} \sum_{j \in C_k} d(i, j).$$

The silhouette value is defined as

$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}}.$$

Table 1. Clustering comparison using the silhouette coefficient

		Training silhouette	Validation silhouette
SCITOS-G5	Triplet embeddings	0.72	0.64
	Siamese embeddings	0.94	0.8
	Baseline embeddings	0.23	0.23
	Original inputs	−0.03	−0.03
Speaker recognition	Triplet embeddings	0.64	0.58
	Siamese embeddings	0.8	0.66
	Baseline embeddings	0.19	0.19
	Original inputs	−0.03	−0.03
GTSRB	Triplet embeddings	0.43	0.43
	Siamese embeddings	0.75	0.72
	Baseline embeddings	0.23	0.24
	Original inputs	−0.22	−0.23

Each sample i in the embedding space is assigned a silhouette value $-1 \leq s(i) \leq 1$ depending on how close and how far it is to samples belonging to the same and different classes, respectively. The closer $s(i)$ is to 1, the closer the sample is to samples of the same class and further from samples belonging to other classes. To compare the representations learned using the different methods as well as compute how much the clustering improves over the original inputs, we compute the mean silhouette over the training data and the validation data separately. In Table 1, we see that the representations learned by either the siamese or the triplet network form well-defined clusters and are improved over the baseline clusters. On the other hand, the original inputs are not arranged in clusters.

Selecting the significance level

First, we illustrate the assurance monitoring algorithm with a test example from the GSTRB dataset. The left side of Figure 5 shows the image of a 60 km/h speed limit sign. Using nearest centroid as the NC function and the siamese network, Algorithm 2 can be used to generate sets of possible predicted labels. In the following, we vary the significance level ϵ and we report the set predictions. When $\epsilon \in [0.001, 0.004)$, the possible labels are "Speed limit 50 km/h", "Speed limit 60 km/h", "Speed limit 80 km/h"; when $\epsilon \in [0.004, 0.006)$, the possible labels are "Speed limit 50 km/h", "Speed limit 60 km/h", and finally, when $\epsilon \in [0.006, 0.0124]$, the algorithm produces a single prediction "Speed limit 60 km/h" which is obviously correct.

For monitoring of CPS, one can either choose ϵ to be small enough given the system requirements or compute ϵ to minimize the number of multiple predictions. Since the number of multiple predictions decreases when ϵ increases, we can select ϵ as the smallest value that eliminates multiple predictions for a calibration/validation set. This can be seen in Figure 6 where for each dataset, the optimal ϵ is selected as the significance level value where the performance curve goes to 0. The nearest centroid NC function is used for the plots in this figure.
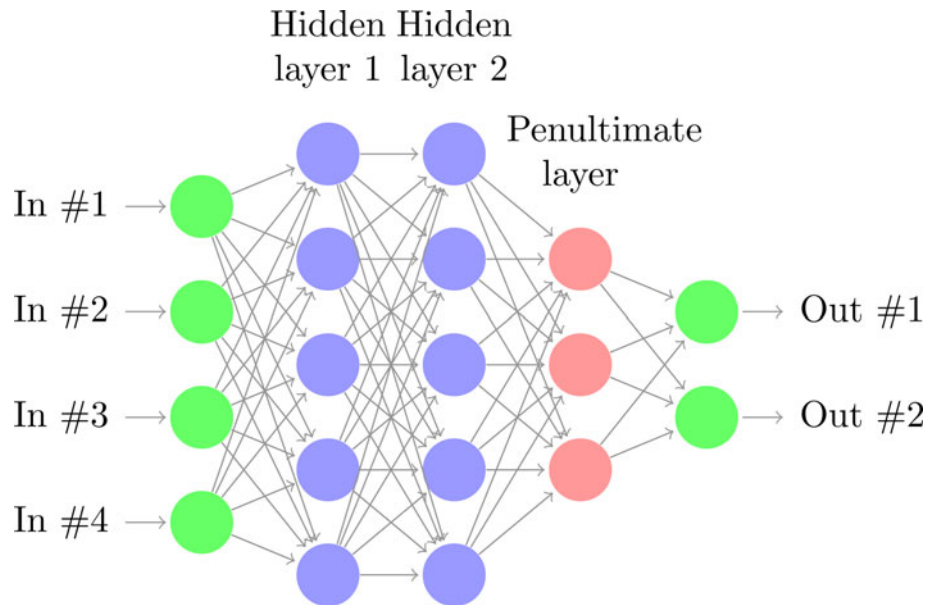
Fig. 5. Illustrative example.

Fig. 6. Performance and calibration curves formed using the validation data from the different datasets using the nearest centroid NC function.

Table 2 shows the results for the different datasets and the various NC functions. First using the calibration/validation dataset, we select ϵ to eliminate sets of multiple predictions and we report the errors in the predictions for the testing dataset. The algorithm successfully did not generate any set with multiple predictions for the testing datasets for any of the NC functions other than the 1-NN when it was used in the SCITOS-G5 dataset with representations computed with the triplet network. In this particular case, there was no ϵ that could eliminate the prediction sets with multiple classes and even when $\epsilon = 1$, 38.6% of the test inputs produced prediction sets with multiple classes. The error rates are well-calibrated and bounded by the computed or the chosen significance level. One way to compare the different NCMs is by looking at the significance level that is required for ICP to make single predictions. The use of embedding representations could always produce single predictions using significance levels much lower than when the original inputs are used. The significance of the distance metric learning techniques is apparent in the case of the nearest centroid NCM on all the datasets. This is an appealing NCM for its simplicity and the reduced memory requirements. When used as part of the baseline the performance was not as good as the more expensive NCMs. However, leveraging the better clustering that distance metric learning methods achieve, the nearest centroid NCM performs as well or better than the rest of the NCMs on making predictions with low significance level while retaining the computational efficiency. We also evaluate how well the different approaches bound the error rate for two different values of the significance level. The errors are bounded in most cases no matter if embedding representations are used or not. The percentage of set predictions on the test data that have multiple candidate classes tend to increase the lower the chosen ϵ is compared with the estimated optimal ϵ.

Table 2. ICP performance for the different configurations

Dataset	Architecture	NC function	Estimate ϵ		$\epsilon = 0.01$		$\epsilon = 0.02$	
			ϵ	Errors (%)	Errors (%)	Multiples (%)	Errors (%)	Multiples (%)
SCITOS-G5	Triplet	k-NN	0.087	9.2	0	100	0	100
		1-NN	1.0	61.4	0.4	88.3	1	71.1
		Nearest centroid	0.095	8.4	0	96.2	0.5	84.6
	Siamese	k-NN	0.066	6.6	0	100	0	100
		1-NN	0.078	8.2	0.4	71.8	2.2	21.4
		Nearest centroid	0.062	7.1	0.2	45.8	1.5	13.4
	Baseline	k-NN	0.093	7.9	0	100	0.7	36.3
		1-NN	0.074	7.5	0.9	35.7	1.3	27.8
		Nearest centroid	0.133	16.1	0.7	67.9	1.6	55.9
	Original inputs	k-NN	0.198	22.3	0.7	72.1	1.6	58.4
		1-NN	0.122	12.6	1	57.9	3.7	37.5
		Nearest centroid	0.428	43.5	0.5	96.9	0.7	95.8
Speaker recognition	Triplet	k-NN	0.058	5.2	1.5	16.6	2.5	10
		1-NN	0.063	5.9	2.1	22.6	2.8	14.1
		Nearest centroid	0.058	6.3	1.9	20.1	2.5	14
	Siamese	k-NN	0.041	3.6	0	100	2.3	7.3
		1-NN	0.045	4.5	0.9	16.1	2	7.6
		Nearest centroid	0.043	4	1.5	14.8	1.9	7.1
	Baseline	k-NN	0.03	3.1	0.9	6.7	1.6	2.5
		1-NN	0.033	3.6	0.9	6.9	2.3	2.9
		Nearest centroid	0.141	14.6	0.5	78.2	2.1	61.9
	Original inputs	k-NN	0.295	29.6	0	100	0	100
		1-NN	0.231	22.9	0.7	84.82	1.3	76.7
		Nearest centroid	0.336	33.3	0.7	100	1.6	98.3
GTSRB	Triplet	k-NN	0.04	4.8	1.5	9.3	2.6	4.6
		1-NN	0.031	3.8	1.4	8.2	2.7	2.9
		Nearest centroid	0.067	6.3	1.5	22.6	2.5	13.9
	Siamese	k-NN	0.031	3.1	0.9	7.4	1.8	3.4
		1-NN	0.035	3.3	0.8	9.5	1.5	4.8
		Nearest centroid	0.038	3.8	1.2	8.7	2.5	4.1
	Baseline	k-NN	0.011	1.1	1	0.1	2	0
		1-NN	0.003	0.3	1	0	2.1	0
		Nearest centroid	0.182	19.4	0.9	71.8	1.9	62.8
	Original inputs	k-NN	0.731	71.7	1.2	98.5	1.2	98.5
		1-NN	–	–	–	–	–	–
		Nearest centroid	0.824	82.7	0.9	100	2	100

Computational efficiency

In order to evaluate if the approach can be used for real-time monitoring of CPS, we measure the execution times and the memory requirements. Different NC functions lead to different execution times and memory requirements. We compare the average execution time over the testing datasets required for generating a prediction set after the model receives a new test input in Table 3. The 1-NN NC function on the input space of the GTSRB dataset has excessive memory requirements. Below we present the computational requirements for each NC function and explain the higher requirements of the 1-NN function in more detail.

Table 3. Execution times and memory requirements

Dataset	Architecture	NC function	Execution time	Memory
SCITOS-G5	Triplet	k-NN	0.2 ms	700.6 kB
		1-NN	1.8 ms	2 MB
		Nearest centroid	56 μs	324.8 kB
	Siamese	k-NN	0.2 ms	700.6 kB
		1-NN	1.6 ms	2 MB
		Nearest centroid	56 μs	324.8 kB
	Baseline	k-NN	0.2 ms	700.6 kB
		1-NN	1.6 ms	2 MB
		Nearest centroid	58 μs	324.8 kB
	Original inputs	k-NN	0.3 ms	2.4 MB
		1-NN	1.9 ms	4.1 MB
		Nearest centroid	59 μs	763.1 kB
Speaker recognition	Triplet	k-NN	0.2 ms	15.3 MB
		1-NN	2.1 ms	24.6 MB
		Nearest centroid	74 μs	13.1 MB
	Siamese	k-NN	0.2 ms	15.3 MB
		1-NN	2 ms	24.6 MB
		Nearest centroid	0.1 ms	13.1 MB
	Baseline	k-NN	0.3 ms	15.3 MB
		1-NN	2.3 ms	24.6 MB
		Nearest centroid	71 μs	13.1 MB
	Original inputs	k-NN	53 ms	723.9 MB
		1-NN	274 ms	2.3 GB
		Nearest centroid	0.1 ms	378.7 MB
GTSRB	Triplet	k-NN	0.6 ms	70.1 MB
		1-NN	24.6 ms	1.4 GB
		Nearest centroid	0.7 ms	38.4 MB
	Siamese	k-NN	0.5 ms	70.1 MB
		1-NN	21.8 ms	1.4 GB
		Nearest centroid	0.6 ms	38.4 MB
	Baseline	k-NN	0.6 ms	70.1 MB
		1-NN	24.4 ms	1.4 GB
		Nearest centroid	0.7 ms	38.4 MB
	Original inputs	k-NN	654 ms	8.9 GB
		1-NN	–	–
		Nearest centroid	4.8 ms	2.1 GB

Table 3 reports the average execution time for each test input and the required memory space using different NC functions. Datasets with high-dimensional inputs are challenging for applying ICP in real time and the results demonstrate the impact of the embedding representations use on the execution times. All the NC functions require storing the calibration NC scores which are used for computing the test p-values online. The DNN weights need to be stored when embedded representations need to be calculated for every new test input. Furthermore, each NCM has a different memory overhead. In the k-NN case, the encodings of the training data are stored in a $k-d$ tree (Bentley, 1975) that is used to compute efficiently the k-NN. This data structure is used both for the k-NN and 1-NN NC functions. In the 1-NN case, it is required to find the nearest neighbor in the training data for each possible class which is computationally expensive resulting in larger execution time. The nearest centroid NC

function requires storing only the centroids for each class and the additional memory required is minimal.

In conclusion, the evaluation results demonstrate that monitoring based on ICP has well-calibrated error rates in all configurations. Furthermore, the use of embedding representations reduces the computational requirements and can lead to decisions with an improved significance level. Using distance metric learning methods, the training data form well-defined clusters that is essential in the case of the nearest centroid NCM. This improvement makes it a good NCM option for all of the used datasets as it performs as well as the other NCMs but with significantly less computational requirements.

Concluding remarks

CPS incorporate machine learning components such as DNNs for performing various tasks such as the perception of the environment. When used for safety-critical applications, they need to satisfy specific requirements that are defined taking into account the acceptable risk and its cost for incorrect decisions. Although DNNs offer advanced capabilities on the decision making process, they cannot provide guarantees on the estimated error rate. To achieve this, they must be complemented by engineering methods and practices that allow effective integration in CPS where an accurate estimate of confidence is needed.

The paper considers the problem of complementing the prediction of DNNs with a well-calibrated confidence. For classification tasks, the inductive conformal prediction framework allows selecting the significance level according to the requirements of each application. This is a parameter that defines the acceptable error rate and is a trade-off between errors and alarms. We presented computationally efficient algorithms based on representations learned by underlying DNN models that make possible for ICP to be used for real-time monitoring. The proposed approach was evaluated on three different benchmarks of increasing complexity from a mobile robot with ultrasound sensors, to speaker recognition and traffic sign recognition. The evaluation results demonstrate that monitoring based on the inductive conformal prediction framework using embedding representations instead of the original inputs has well-calibrated error rates and can minimize the number of alarms when a confident decision cannot be made. When appropriate embedding representations are computed using distance metric learning methods input data that belong to the same class form well-defined clusters. This property is very important when the similarity of a test input to the test data is estimated. That way the training set can be efficiently represented by the centroids of each class which reduces the computational requirements without any loss in performance when compared with the more computationally expensive approaches.

During the experiments, we identified a number of challenges that can lead to poor performance of the proposed method. First, when the datasets are imbalanced both the siamese and the triplet architectures may not learn embedding representations that cluster the under-represented classes well. This affects the efficiency of the NC functions. Second, the training of the triplet networks require mining of training data that will form triplets that lead to large gradients for minimizing the triplet loss function. There is ongoing research for mining algorithms for faster training. One open question for future research is how to utilize all the candidate decisions in the prediction set to deal with the cases when a confident decision cannot be made that will satisfy the significance-level requirements.

References

Balasubramanian V, Ho SS and Vovk V (2014) *Conformal Prediction for Reliable Machine Learning: Theory, Adaptations and Applications*, 1st Edn. San Francisco, CA: Morgan Kaufmann Publishers Inc.

Bengio Y (2009) *Learning Deep Architectures for AI*. Hanover, MA: Now Publishers Inc.

Bentley JL (1975) Multidimensional binary search trees used for associative searching. *Communications of the ACM* **189**, 509–517. doi:10.1145/361002.361007.

Bhattacharyya S (2013) Confidence in predictions from random tree ensembles. *Knowledge and Information Systems* **35**, 391–410. doi:10.1007/s10115-012-0600-z.

Boursinos D and Koutsoukos X (2020a) Assurance monitoring of cyber-physical systems with machine learning components. In *Digital Proceedings of TMCE 2020*, pp. 27–38.

Boursinos D and Koutsoukos X (2020b) Improving prediction confidence in learning-enabled autonomous systems. In *InfoSymbiotics/DDDAS2020*, pp. 217–224.

Boursinos D and Koutsoukos X (2020c) Trusted confidence bounds for learning enabled cyber-physical systems. In *Workshop on Assured Autonomous Systems at SP2020*.

Devetyarov D and Nouretdinov I (2010) Prediction with confidence based on a random forest classifier. In Papadopoulos H, Andreou AS and Bramer M (eds.), *Artificial Intelligence Applications and Innovations*. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 37–44.

Dua D and Graff C (2017) UCI machine learning repository. Available at: http://archive.ics.uci.edu/ml.

Goldsmith J, Greven S and Crainiceanu C (2013) Corrected confidence bands for functional data using principal components. *Biometrics* **69**, 141–151.

Guo C, Pleiss G, Sun Y and Weinberger KQ (2017) On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning*, Vol. 70. JMLR.org, pp. 1321–1330.

Hinton GE (2007) Learning multiple layers of representation. *Trends in Cognitive Sciences* **11**, 428–434.

Hoffer E and Ailon N (2015) Deep metric learning using triplet network. In *International Workshop on Similarity-Based Pattern Recognition*, pp. 84–92.

Jiang X, Osl M, Kim J and Ohno-Machado L (2012) Calibrating predictive model estimates to support personalized medicine. *Journal of the American Medical Informatics Association* **19**, 263–274.

Johansson U, Boström H and Löfström T (2013) Conformal prediction using decision trees. In *2013 IEEE 13th International Conference on Data Mining*, pp. 330–339.

Johansson U, Boström H, Löfström T and Linusson H (2014) Regression conformal prediction with random forests. *Machine Learning* **97**, 155–176.

Johansson U, Linusson H, Löfström T and Boström H (2017). Model-agnostic nonconformity functions for conformal classification. In *2017 International Joint Conference on Neural Networks (IJCNN)*, pp. 2072–2079. doi:10.1109/IJCNN.2017.7966105

Kiplagat E (n.d.). American rhetoric (online speech bank). Available at: https://americanrhetoric.com/speechbank.htm.

Koch G, Zemel R and Salakhutdinov R (2015) Siamese neural networks for one-shot image recognition. In *ICML Deep Learning Workshop*, Vol. 2.

Kumar A, Liang PS and Ma T (2019) Verified uncertainty calibration. In Wallach H, Larochelle H, Beygelzimer A, d'Alché-Buc F, Fox E and Garnett R (eds), *Advances in Neural Information Processing Systems*. Red Hook, NY: Curran Associates, Inc., pp. 3792–3803.

Makili L, Vega J, Dormido-Canto S, Pastor I and Murari A (2011) Computationally efficient SVM multi-class image recognition with confidence measures. *Fusion Engineering and Design* **86**, 1213–1216. (Proceedings of the 26th Symposium of Fusion Technology (SOFT-26)).

Melekhov I, Kannala J and Rahtu E (2016) Siamese network features for image matching. In *2016 23rd International Conference on Pattern Recognition (ICPR)*, pp. 378–383.

Naeini MP and Cooper GF (2018) Binary classifier calibration using an ensemble of piecewise linear regression models. *Knowledge and Information Systems* **54**, 151–170.

Naeini MP, Cooper G and Hauskrecht M (2015) Obtaining well calibrated probabilities using Bayesian Binning. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*.

Ovadia Y, Fertig E, Ren J, Nado Z, Sculley D, Nowozin S and Snoek J (2019) Can you trust your model's uncertainty? Evaluating predictive uncertainty under dataset shift. In Wallach H, Larochelle H, Beygelzimer A, d'Alché-Buc F, Fox E and Garnett R (eds), *Advances in Neural Information Processing Systems*, Red Hook, NY: Curran Associates, Inc., pp. 13991–14002.

Papadopoulos H and Haralambous H (2011) Reliable prediction intervals with regression neural networks. *Neural Networks* **24**, 842–851.

Papadopoulos H, Vovk V and Gammermam A (2007) Conformal prediction with neural networks. In *19th IEEE International Conference on Tools with Artificial Intelligence (ICTAI 2007)*, Vol. 2, pp. 388–395.

Papadopoulos H, Vovk V and Gammerman A (2011) Regression conformal prediction with nearest neighbours. *Journal of Artificial Intelligence Research* **40**, 815–840.

Papernot N and McDaniel P (2018) Deep k-nearest neighbors: towards confident, interpretable and robust deep learning. arXiv:1803.04765.

Platt JC (1999) Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In Smola AJ, Smola AJ, Bartlett P and Schuurmans D (eds), *Advances in Large Margin Classifiers*. One Rogers Street Cambridge, MA: MIT Press, pp. 61–74.

Rousseeuw PJ (1987) Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics* **20**, 53–65.

Shafer G and Vovk V (2008) A tutorial on conformal prediction. *Journal of Machine Learning Research* **9**, 371–421.

Shannon C (1949) Communication in the presence of noise. *Proceedings of the IRE* **37**, 10–21. doi:10.1109/jrproc.1949.232969.

Simonyan K and Zisserman A (2014) Very deep convolutional networks for large-scale image recognition. arXiv:1409.1556.

Stallkamp J, Schlipsing M, Salmen J and Igel C (2012) Man vs. computer: benchmarking machine learning algorithms for traffic sign recognition. *Neural Networks* **32**, 323–332. (Selected Papers from IJCNN 2011).

Sun M and Cho S (2018) Obtaining calibrated probability using ROC Binning. *Pattern Analysis and Applications* **21**, 307–322.

Sun J and Loader CR (1994) Simultaneous confidence bands for linear regression and smoothing. *The Annals of Statistics* **22**, 1328–1345.

Vovk V, Gammerman A and Shafer G (2005) *Algorithmic Learning in a Random World*. Berlin, Heidelberg: Springer-Verlag.

Xuan H, Stylianou A and Pless R (2019) Improved embeddings with easy positive triplet mining. arXiv:1904.04370.

Zadrozny B and Elkan C (2001) Obtaining calibrated probability estimates from decision trees and naive Bayesian classifiers. In Brodley CE and Danyluk AP (eds), *ICML*, Vol. 1, San Francisco, CA: Morgan Kaufmann Publishers Inc., pp. 609–616.

Zadrozny B and Elkan C (2002) Transforming classifier scores into accurate multiclass probability estimates. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. New York, NY: ACM, pp. 694–699. doi:10.1145/775047.775151.

Zhang J and Yang Y (2004) Probabilistic score estimation with piecewise logistic regression. In *Proceedings of the Twenty-First International Conference on Machine Learning*, p. 115.

Dimitrios Boursinos is currently pursuing his PhD degree in Electrical Engineering at Vanderbilt University. He is also a research assistant in the Institute for Software Integrated Systems (ISIS). His research interests focus on machine learning and cyber-physical systems with emphasis on learning-enabled systems.

Xenofon Koutsoukos is a Professor of Computer Science, Computer Engineering, and Electrical Engineering and Chair of the Department of Electrical Engineering and Computer Science at Vanderbilt. He is also a Senior Research Scientist in the Institute for Software Integrated Systems (ISIS). His research work is in the area of cyber-physical systems with emphasis on learning-enabled systems, security and resilience, control and decision making, diagnosis and fault tolerance, formal methods, and adaptive resource management. He is a Fellow of the IEEE for his contributions to the design of resilient cyber-physical systems.