Reliable Probability Intervals For Classification Using Inductive Venn Predictors Based on Distance Learning

Dimitrios Boursinos and Xenofon Koutsoukos
Institute for Software Integrated Systems
Vanderbilt University
Nashville TN, USA
{dimitrios.boursinos, xenofon.koutsoukos}@vanderbilt.edu

2021 IEEE International Conference on Omni-Layer Intelligent Systems (COINS) | 978-1-6654-3156-9/21/$31.00 ©2021 IEEE | DOI: 10.1109/COINS51742.2021.9524144

***Abstract*—Deep neural networks are frequently used by autonomous systems for their ability to learn complex, non-linear data patterns and make accurate predictions in dynamic environments. However, their use as black boxes introduces risks as the confidence in each prediction is unknown. Different frameworks have been proposed to compute accurate confidence measures along with the predictions but at the same time introduce a number of limitations like execution time overhead or inability to be used with high-dimensional data. In this paper, we use the Inductive Venn Predictors framework for computing probability intervals regarding the correctness of each prediction in real-time. We propose taxonomies based on distance metric learning to compute informative probability intervals in applications involving high-dimensional inputs. Empirical evaluation on image classification and botnet attacks detection in Internet-of-Things (IoT) applications demonstrates improved accuracy and calibration. The proposed method is computationally efficient, and therefore, can be used in real-time.**

***Keywords*—deep neural networks, assurance monitoring, inductive venn predictors, probability intervals**

I. Introduction

Modern Deep Neural Network (DNN) architectures have the capacity to be trained using high-dimensional data and make accurate decisions in dynamic and uncertain environments. This ability makes them a common choice for many autonomous system applications. However, when DNNs are used as black boxes in safety-critical systems, they may result in disastrous consequences if it is not possible to reason about their predictions.

The training of a Learning Enabled Component (LEC) requires specification of the task, performance measure for evaluating how well the task is performed, and experience in the form of training and testing data. An LEC, such as a DNN, during system operation exhibits some nonzero error rate and the true error rate is unknown and can only be approximated during design time using the available data. Confidence values, such as the softmax probabilities which are used by most DNNs for classification, are usually greater than the actual posterior probability that the prediction is correct. Important factors that make modern DNNs overconfident are the depth, width, and techniques like weight decay, and batch normalization [5].

Our objective is to complement the predictions made by DNNs with a computation of confidence. The confidence can be expressed as probability intervals to characterize the correctness of the DNN prediction. An efficient and robust approach must ensure that the actual accuracy of a DNN is contained in the computed intervals and the width of the intervals is small. We focus on computationally efficient algorithms that can be used in real-time. The proposed approach is based on the Inductive Venn Predictors (IVP) framework [28]. IVP computes the probability intervals for an unknown input leveraging knowledge it has acquired from previous predictions on labeled data. Most of the IVP or Venn Predictors (VP) applications in the literature are evaluated on low-dimensional data [13], [14], [19], [20], [28], [29].

The estimation of reliable predictive uncertainty has become an important part of many modern machine learning components used in safety-critical applications. Even though many of the proposed methods produce well-calibrated models, their application in the real world is challenging. In [12], [22], new training algorithms and loss functions are proposed to achieve well-calibrated DNNs. These approaches require training DNN models from scratch and cannot be used with pre-trained ones. Another category of calibration methods like the Platt's scaling [23] and temperature scaling [5] proposes ways of post-processing the outputs of already trained models to produce calibrated confidence measures. In [11], [18], it is shown that these methods are not as well-calibrated as it is reported especially when the validation data are not independent and identically distributed (IID) and in the presence of distribution shifts. The Conformal Prediction (CP) framework is developed to compute prediction sets to satisfy a desired significance level [1], [26], [28]. The confidence value assigned to each possible class is in the form of p-values which is less intuitive than estimating the confidence as probabilities. Another way of obtaining confidence information about predictions is by

978-1-6654-3156-9/21/$31.00 ©2021 IEEE

using algorithms based on the Bayesian framework. The use of this framework, however, require some prior knowledge about the distribution generating the data. In the real world, this distribution is unknown and it has to be chosen arbitrarily. In [21], it is shown that the predictive regions produced by Gaussian Processes, a popular Bayesian machine learning approach, may be incorrect and misleading when the correct prior is not known.

The main contribution of our work is that we compute low-dimensional, appropriate, embedding representations of the original inputs in a space where the Euclidean distance is a measure of similarity between the original inputs, in order to handle high-dimensional inputs in real-time. Then, we implement four different taxonomies that split the low-dimensional data into categories based on their similarity. Last, we present an empirical evaluation of the approach using two datasets for image classification problems with a large number of classes as well as detection of botnet attacks in an IoT device. The underlying models are chosen according to the input size and shape keeping into account the low-latency and low-power properties to meet the resource constraints of the variety of use cases [8].

II. Problem

A perception component in an autonomous system aims to observe and interpret the environment in order to provide information for decision-making. For example, a DNN can be used for classifying traffic signs in autonomous vehicles. 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 to enable real-time operation. The approach must ensure a bounded small error rate while limiting the number of inputs for which an accurate prediction cannot be made.

During system operation, for each new input a prediction is made, usually by a LEC and the objective is to compute a valid measure of the prediction's confidence. The objective is twofold: (1) provide guarantees for the error rate of the prediction and (2) limit the number of input examples for which a confident prediction cannot be made. Well-calibrated confidence in terms of probabilities can be used for decision-making, for example, by generating warnings when human intervention is required.

The Venn Prediction (VP) framework can produce predictions with well-calibrated confidence intervals that guarantee to include the true probabilities for each class output to occur [28]. The confidence intervals for a test input are generated by considering the class distribution of labeled inputs assigned to the same category that are collected offline and are available to the system. In the literature, VP implementations use Support Vector Machines (SVMs) or DNN classifiers to create categories of labeled data [13], [20], [28]. The additional problem we are considering is the computation of appropriate embedding representations that can lead to more efficient VPs. The main idea is to use distance metric 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. Using such representations we define taxonomies to form categories of similar input data. This not only reduces the memory requirements but is also more efficient in producing more informative intervals.

III. Probability Intervals based on Distance Metric Learning

Venn Predictors is a machine learning framework that can be combined with existing classifier architectures for producing well-calibrated multi-probability predictions under the IID assumption [1], [28]. This means that the confidence assigned to a prediction is a probability distribution which in effect defines lower and upper bounds regarding the probability of correctness for all possible classes. VPs are well-calibrated and the probability bounds asymptotically contain the corresponding true conditional probabilities (proof in [28]). However the framework is computationally inefficient as it requires training the underlying algorithm after every new test input. Computational efficiency can be addressed using the Inductive Venn Predictors [13], [14], an extension of the VP framework.

Central to the VP and IVP frameworks is the definition of a Venn taxonomy. This is a way of clustering data points into a number of categories according to their similarity and is based on an underlying algorithm. For example a taxonomy can be defined to put in the same category examples that are classified in the same class by a DNN. The main idea of our approach is that the taxonomy can be defined efficiently by learning embedding representations of the inputs for which the Euclidean distance is a measure of similarity. To compute the embedding representations of the inputs we train a *siamese network* using contrastive loss [6], [9].

We consider the training examples, $z_1, \ldots, z_l$ from $\boldsymbol{Z}$, where each z_i is a pair (x_i, y_i) with x_i the feature vector and y_i the corresponding label. We also consider a test input x_{l+1} which we wish to classify. IVP assumes that all the examples $z_1, \ldots, z_{l+1}$ are independent and identically distributed (IID) generated from the same but usually unknown probability distribution. The available training examples are split into two parts: the *proper training set* with q examples and the calibration set with $l-q$ examples. The examples in the proper training set are used to train the siamese network which is used to define different Venn taxonomies. The roll of the taxonomy is to divide the $l-q$ calibration examples into a number of categories based on their similarity. This process takes place during the design time.

After placing the calibration data into categories using the underlying algorithm for the taxonomy, during execution time we consider a test input x_{l+1} and place it in a category k_{l+1}. The true class y_{l+1} is unknown and IVP computes a lower and an upper probability $[L(Y_j), U(Y_j)]$ for every possible class $j = 1, \ldots, c$ based on the number of samples of each class in k_{l+1}. The predicted class for the classification is computed as:

$$j_{best} = \arg \max_{j=1,\ldots,c} \overline{p(Y_j)} \tag{1}$$

where $\overline{p(Y_j)}$ is the mean of the probability interval assigned to Y_j. Along with the class $Y_{j_{best}}$ the IVP framework outputs the probability interval $[L(Y_{j_{best}}), U(Y_{j_{best}})]$. The steps taking place during execution are illustrated in Fig. 1.

IV. Distance-based Taxonomies

As proved in [28] the probability intervals assigned to each classification by the VP are well-calibrated regardless of the choice of the Venn taxonomy and this holds in practice for IVP as well [13]. However, the choice of the taxonomy affects the efficiency of the IVP. The probability intervals are desirable to be relatively narrow to minimize the uncertainty in the probability of correctness as well as create better separation between the probabilities of each class. We propose four different Venn taxonomies based on distance metric learning. The first two taxonomies are based on a k-Nearest Neighbors classifier. The naive approach, that we call *k-NN* V_1, trains a k-NN classifier using the embedding representations of the proper training set. Then the calibration data, as well as each new test input, are placed to a category that is defined by the k-NN prediction using the computed embedding representations. That is, for a data point x_{l+1} that needs to be placed into a category, its embedding representation is computed using the siamese network, $r_{l+1} = \text{Net}(x_{l+1})$ and its k nearest training data are found. Depending on the class $\hat{y}_{l+1}$ that most neighbors belong to, the data point is assigned to the category

$$k_{l+1} = \hat{y}_{l+1}. \tag{2}$$

This taxonomy leads to a number of categories that is equal to the number of classes in the dataset. An extension of the previous taxonomy, *k-NN* V_2, is also based on a k-Nearest Neighbors classifier. However, we attempt to more accurately split the data into categories by taking into account how many of the k nearest training data points are labeled different than the predicted class. For a data point x_{l+1} with embedding representation r_{l+1} that needs to be placed into a category we compute the k-nearest neighbors in the training set and store their labels in a multi-set Ω. The category where x_{l+1} is placed is computed as:

$$k_{l+1} = \hat{y}_{l+1} \times (k - \left\lfloor \frac{k}{c} \right\rfloor) + |i \in \Omega : i \neq \hat{y}_{l+1}| \tag{3}$$

where $\hat{y}_{l+1}$ is the k-NN classification of r_{l+1}, k is the number of nearest neighbors and c is the number of different classes. This taxonomy aims at further improving the similarity of the data in each category leveraging the classifier's confidence. It is expected that the more similar labeled neighbor training data points, the higher the chance of the corresponding class being the correct one. That way each category of *k-NN* V_1 is further split into $k - \lfloor \frac{k}{c} \rfloor$ new categories.

The ability of siamese networks to create clusters of similar data can be used to further reduce the Venn taxonomy computational requirements when there is a large amount of training data. Each class cluster i corresponding to class $Y_i, i = 1 \ldots, c$ can then be represented by its centroid $\mu_i = \frac{\sum_{j=1}^{n_i} r_j^i}{n_i}$, where r_j^i is the embedding representation of the j^{th} training example from class Y_i and n_i is the number of training examples labeled as Y_i. We propose another family of taxonomies based on the *Nearest Centroids*. The *NC* V_1 places the calibration data as well as each new test input to a category that is the same as the class assigned to their nearest centroid. The category where an example x_{l+1} is placed is computed as:

$$k_{l+1} = \arg \min_{j=1,\ldots,c} d(r_{l+1}, \mu_j) \tag{4}$$

where d the Euclidean distance. This leads to a number of categories that is equal to the number of classes in the dataset. An extension of this taxonomy, the *NC* V_2, attempts to form more accurate categories by taking into account the classification confidence. We expect data points of the same class to be more similar to each other when their embedding representations are placed at similar distances to their class centroid. That way each category of *NC* V_1 is further split into two categories based on how close an example x_{l+1} is to its nearest centroid:

$$k_{l+1} = 2 \times \arg \min_{j=1,\ldots,c} d(r_{l+1}, \mu_j) + h, \tag{5}$$

$$h = \begin{cases} 0, & \text{if } d(r_{l+1}, \mu_{\text{min}}) \leq \theta \\ 1, & \text{otherwise} \end{cases}$$

where $\mu_{\text{min}} = \arg\min_{j=1,\ldots,c} d(r_{l+1}, \mu_j)$ is the distance to the nearest centroid and θ a chosen distance threshold.

V. Evaluation Metrics

The performance of IVP based on the proposed taxonomies is evaluated regarding the accuracy, calibration and efficiency. The objective is for the computed probability intervals to contain the true probability of correctness for each prediction. The probability interval for a given input x with predicted class $\hat{y}$ is $[L(\hat{y}), U(\hat{y})]$. Equivalently, the probability that $\hat{y}$ is not the correct classification will be in the complimentary interval $[1-U(\hat{y}), 1-L(\hat{y})]$, called *error probability interval*. The true probability of correctness for a single prediction is unknown so the correctness of the computed intervals is evaluated over a number of samples. To do this we use the following metrics:

- cumulative errors

$$E_n = \sum_{i=1}^{n} err_i, \tag{6}$$

$$err_i = \begin{cases} 1, & \text{if classification } \hat{y}_i \text{ is incorrect} \\ 0, & \text{otherwise} \end{cases}$$

- cumulative lower and upper error probabilities

$$LEP_n = \sum_{i=1}^{n} [1 - U(\hat{y})], \quad UEP_n = \sum_{i=1}^{n} [1 - L(\hat{y})] \tag{7}$$

To compare the IVP implementations based on our proposed taxonomies with the baseline taxonomies, scalar metrics are used that represent the performance regarding accuracy, calibration, and efficiency. Unlike the NN classifiers that produce a single softmax probability for each class, the IVP framework

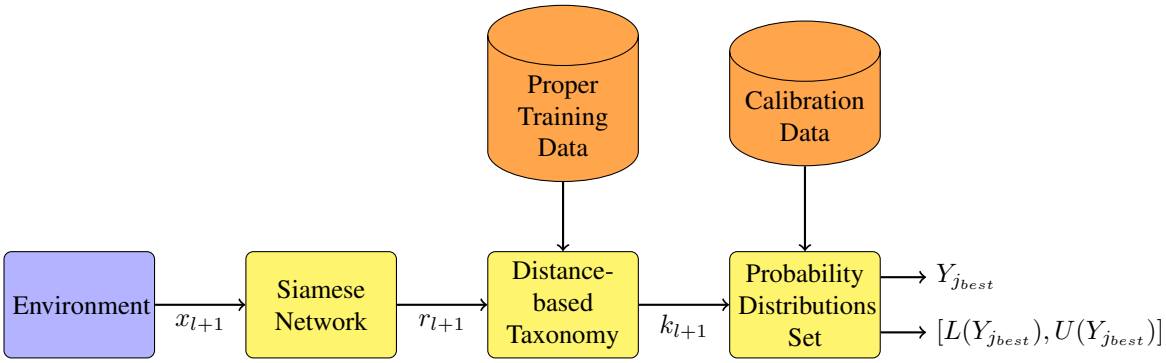

Figure 1. IVP classifier based on distance metric learning

produces probability intervals. For the computation of the evaluation metrics the probability assigned to a class Y_j will be $\overline{p(Y_j)}$ like in (1). The accuracy of an IVP implementation is evaluated as the number of correct classifications over the number of attempted classifications and it is computed as

$$accuracy = 1 - \frac{E_n}{n}. \tag{8}$$

An efficient, or informative, IVP is one that makes predictions with small diameter probability intervals and their median is as close to zero or one. The most popular quality metrics for probability assessments are the negative log-likelihood (NLL) and the Brier score (BS) [4]. NLL is the simplest out of the two and only considers the probability assigned to the predicted class in (1). It is computed as

$$NLL = -\sum_{i=1}^{n}\sum_{j=1}^{c} t_i^j \log(o_i^j), \tag{9}$$

where $o_i^j = \overline{p(Y_j)}$ of example i and t_i^j the one-hot representation of the ground truth classification label y_i of example i, that is

$$t_i^j = \begin{cases} 1, & \text{if classification } y_i = Y_j \\ 0, & \text{otherwise} \end{cases}$$

This metric is minimized by producing intervals that are narrow and have median probability close to one assigned to the correct class. Computational issues may occur as the log score explodes if we observe an event that the classifier considers impossible. BS is computed as

$$BS = \frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{c}(o_i^j - t_i^j)^2 \tag{10}$$

This is, in effect, the mean squared error of the predictions. Unlike NLL, BS considers the probabilities assigned to all possible classes and will penalize probability intervals assigned to incorrect classes that are not close to zero. There are different views in the literature regarding which scoring rule is more appropriate. [2] emphasizes in the importance of the locality property, meaning, the scoring rule should only depend on the probability of events that actually occur and only NLL satisfies this. On the other hand, [25] states that a scoring rule should be symmetric and only BS satisfies this. This means that if the true class probability is p and the predicted probability is $\hat{p}$, then the score should be equal to the case where the true probability is $\hat{p}$ and the predicted probability is p. However, we think that both metrics produce useful insights in probability assessment so both are reported in our experiment results. The interval size has a significant role on how informative and interpretable a prediction is. We evaluate the size of the probability intervals by computing the average interval diameter as

$$D = \frac{\sum_{i=1}^{n} U(\hat{y}) - \sum_{i=1}^{n} L(\hat{y})}{n} \tag{11}$$

A well-calibrated IVP computes probability intervals that are representative of the true correctness likelihood. Formally a model is well-calibrated when

$$\mathbb{P}\left(\hat{y} = Y | \hat{p} = p\right) = p, \qquad \forall p \in [0,1] \tag{12}$$

However, $\hat{p}$ is a continuous random variable so the probability in (12) cannot be approximated using finitely many samples. According to (12) a measure of miscalibration can be expressed as $\mathop{\mathbb{E}}_{\hat{p}}\left[\left|\mathbb{P}\left(\hat{y} = y | \hat{p} = p\right) - p\right|\right]$. The *Expected Calibration Error* (ECE) [17] computes an approximation of this expected value across bins:

$$\text{ECE} = \sum_{m=1}^{M} \frac{|B_m|}{n} \left|\text{acc}(B_m) - \text{conf}(B_m)\right| \tag{13}$$

where $|B_m|$ is the number of samples in bin B_m, n is the total number of samples and $\text{acc}(B_m)$ and $\text{conf}(B_m)$ are the accuracy and confidence of bin B_m respectively as defined in [17]. Many times in safety critical applications it is more useful to compute the maximum miscalibration of a model than the mean value. This metric is called Maximum Calibration Error (MCE) [17] and is computed as:

$$\text{MCE} = \max_{m \in \{1,\ldots,M\}} \left|\text{acc}(B_m) - \text{conf}(B_m)\right| \tag{14}$$

VI. Evaluation

In this section, we evaluate the IVPs that use distance-based taxonomies with regard to accuracy, calibration, and efficiency. Additionally, for the evaluation of our proposed

taxonomies, we use metrics regarding the performance of the siamese network in clustering similar input data, the execution time of the framework, and the required memory.

A. Experimental Setup

The embedding representation computations, part of our proposed taxonomies, are not application-specific and can improve the performance of IVP in cases where inputs are high-dimensional. We evaluate the performance of IVP with distance-learning in two different classification problems. First, we have two case studies in image classification. The German Traffic Sign Recognition Benchmark (GTSRB) dataset is a collection of traffic sign images to be classified in 43 classes [27]. The labeled sign images are of various sizes between 15x15 to 250x250 pixels depending on the observed distance. We convert all the images to a fixed shape of 96x96 pixels. The second dataset is the Fruits360 [16]. This dataset contains images of 131 different kinds of fruits and vegetables. The input data are used in their original size, 100x100 pixels.

The second classification problem we consider is the detection of botnet attacks in IoT devices. As part of the evaluation in [15], authors made available data regarding network traffic while infecting different common IoT devices two families of botnets. Mirai and BASHLITE are two common IoT-based botnets and their harmful capabilities are presented in [10]. In the dataset there are data for the following ten attacks:

- BASHLITE Attacks
  1) Scan: Scanning the network for vulnerable devices
  2) Junk: Sending spam data
  3) UDP: UDP flooding
  4) TCP: TCP flooding
  5) COMBO: Sending spam data and opening a connection to a specified IP address and port
- Mirai Attacks
  1) Scan: Scanning the network for vulnerable devices
  2) Ack: Ack flooding
  3) Syn: Syn flooding
  4) UDP: UDP flooding
  5) UDPplain: UDP flooding with fewer options, optimized for higher PPS

Including the benign network traffic we approach this as a classification problem with eleven classes. The available data are in the form of 115 statistical features extracted from the raw network traffic. The same 23 features, presented in [15], are extracted from five time windows of the most recent 100ms, 500ms, 1.5sec, 10sec, 1min. The features summarize the traffic in each of these time windows that has (1) the same source IP address, (2) the same source IP and MAC address, (3) been sent between the source and destination IP address, (4) been sent between the source and destination TCP/UDP sockets. These features are computed incrementally and in real-time.

The available data are used throughout the evaluation process the same way the same way in every dataset. 10% of the data are taken out to be used for testing and the rest is the training set. The training set is then split into the proper training set and the calibration set. The proper training set is randomly chosen as 80% of the training set and is used to train the underlying models and for the computation of the categories. The calibration set is the remaining 20% of the available training data is used only to form the categories during the design time. The reported evaluation results are computed on the separate test set. All the experiments run in a desktop computer equipped with and Intel(R) Core(TM) i9-9900K CPU and 32 GB RAM and a Geforce RTX 2080 GPU with 8 GB memory.

B. Baseline

To understand the effect of the distance metric learning in IVP we compare it with approaches that use DNN classifiers as underlying algorithms. A variety of Venn taxonomy definitions based on DNNs is proposed in [20]. V_1 assigns two examples to the same category if their maximum softmax outputs correspond to the same class. V_2, divides the examples in the categories defined by V_1 into two smaller categories based on the value of their maximum softmax output. Their chosen threshold for the maximum output to create the two smaller categories is 0.75. V_3 divides the examples in the categories defined by V_1 into two smaller categories but this time based on the second highest softmax output. Their chosen threshold for the second-highest output is 0.25. V_4 divides each category of taxonomy V_1 in two, based on the difference between the highest and second-highest softmax outputs. The threshold for this difference is 0.5. In the same paper, they proposed a fifth taxonomy that creates the categories based on which classes have softmax outputs above a certain threshold. This taxonomy creates 2^C number of categories making its use infeasible in our evaluation datasets.

C. Evaluation Results

The difficulty to assign an input to a category and the memory demands increase as the size and complexity of the inputs increases. Our goal is to evaluate our method using general-purpose and lightweight DNNs. For the image classification problems, we use the MobileNet architecture for both the embedding representation computation as well as the classifier used for the baseline taxonomies for its low latency and low memory requirements. The trade-off between accuracy and latency is configured by the hyperparameter α. We set $\alpha = 0.5$ in the case of GTSRB and $\alpha = 1$ for the Fruit360. In both cases the embedding representation vectors are of size 128. In the case of the botnet attacks detection, the input data are arranged in vectors of 115 values so we use a fully connected DNN with two hidden layers, the first has 10 units, and the second which produces the embedding representations has 32 units.

After training the siamese network and before it is used as part of the taxonomies we need to evaluate how well it performs in clustering similar inputs. For comparison, we use the embedding space produced by the penultimate layer of the DNN classifier [7]. A commonly used metric of the separation

between class clusters is the *silhouette coefficient* [24]. This metric evaluates how close together samples from the same class are, and far from samples of different classes and takes values in [-1,1]. The results on the silhouette analysis for the test inputs from both datasets are shown in Table I. The siamese network produces representations that are well clustered based on their similarity and better than the representations produced by the classifier DNN. This is important for constructing efficient categories using our proposed distance-based taxonomies.

Table I
SILHOUETTE COEFFICIENT COMPARISON

	Classifier Embeddings	Siamese Embeddings
GTSRB	0.56	0.98
Fruits360	0.52	0.85
Ecobee Thermostat	0.27	0.46

For illustration, the cumulative upper and lower error probabilities as well as the cumulative error are plotted on the same axis in Fig. 2 using the NC V_2 taxonomy and test data from the GTSRB dataset. The computed probability intervals successfully bound the true error-rate.

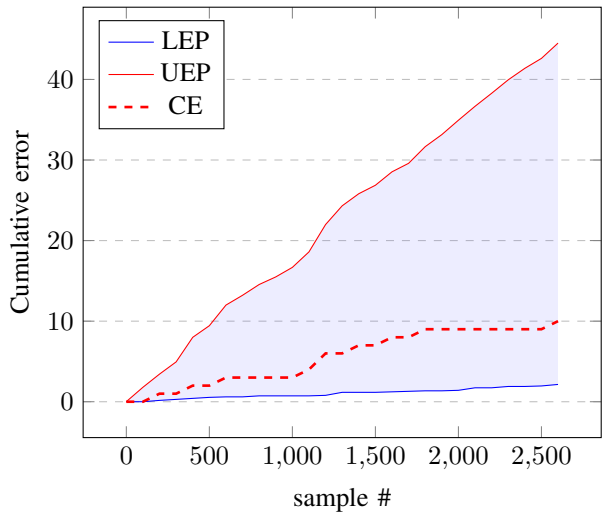

Figure 2. Illustration of cumulative metrics in the GTSRB dataset using the NC V_2 taxonomy

The evaluation results are shown in Table II. For both datasets, we observe that using the proposed distance-based taxonomies, IVP produces more accurate classifications. Even though the baseline V_1 taxonomy produces probability intervals that are as narrow as the intervals produced by some of the proposed taxonomies, the proposed taxonomies produce better quality intervals by keeping the intervals assigned to the correct class close to 1 and the intervals of the incorrect classes close to 0, as shown by the NLL and BS metrics. The differences in ECE are not significant but most of the proposed taxonomies produce probabilities that are better calibrated in the whole probability space $[0, 1]$ with no areas of miscalibration as indicated by MCE.

The times required for the computation of a classification and the probability intervals when a new input arrives are similar in both the baseline and our proposed taxonomies and indicate they can be used for real-time operation. The speed bottleneck is the computations by the DNNs for either the classifications or the representation mapping. The k-NN computation step in the low-dimensional embedding representation space adds minimal overhead in the execution time. The memory requirements have two main parts: the memory required to store the DNN weights and the memory required to store the categories after calibration. The proposed taxonomies have the additional requirement to store either the embedding representations of the training data to be used by the k-NN or the centroid of each class. The representations of the training data are stored in a $k-d$ tree [3] for fast k-NN computation. With the use of low-dimensional representations, the additional memory required for the nearest centroid based taxonomies is small compared to the underlying DNN size.

VII. CONCLUSION

Although DNNs offer advanced capabilities, they must be complemented by engineering methods and practices for them to provide accurate measures of prediction confidence. For classification tasks, the IVP framework computes probability intervals that contain the probability of the prediction's correctness by examining the underlying model's accuracy on similar data. We presented computationally efficient algorithms based on appropriate embedding representations learned by siamese networks that make it possible for IVP to be used with high-dimensional data for real-time applications. The evaluation results demonstrate that the IVP framework using distance-based taxonomies produces high accuracy and probability intervals that are efficient and well-calibrated. Our choice of lightweight DNNs and small embedding representation size make the approach computationally efficient and can be used in real-time. A direction for future extension of this work is to improve the probability intervals, regarding their efficiency, during execution time.

ACKNOWLEDGMENT

The material presented in this paper is based upon work supported by the Defense Advanced Research Projects Agency (DARPA) through contract number FA8750-18-C-0089. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of DARPA.

REFERENCES

[1] V. Balasubramanian, S.-S. Ho, and V. Vovk. *Conformal Prediction for Reliable Machine Learning: Theory, Adaptations and Applications*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1st edition, 2014.

[2] R. Benedetti. Scoring rules for forecast verification. *Monthly Weather Review*, 138(1):203–211, 2010.

[3] J. L. Bentley. Multidimensional binary search trees used for associative searching. *Commun. ACM*, 18(9):509–517, Sept. 1975.

[4] G. W. Brier. Verification of forecasts expressed in terms of probability. *Monthly Weather Review*, 78(1):1 – 3, 01 Jan. 1950.

Table II
EVALUATION METRICS RESULTS

Dataset	Taxonomy	Accuracy	NLL	BS	D	ECE	MCE	Time	Memory
GTSRB	V_1	0.994	111.835	0.013	3.29e-06	0.005	0.005	3.6ms	11.2MB
	V_2	0.992	58.104	0.055	5.30e-06	0.011	0.583	6.6ms	11.2MB
	V_3	0.993	75.394	0.038	4.48e-06	0.008	0.750	3.7ms	11.2MB
	V_4	0.991	70.279	0.053	5.06e-06	0.009	0.750	2.9ms	11.2MB
	k-nn V_1	0.998	41.575	0.005	3.30e-06	0.004	0.004	3.2ms	19MB
	k-nn V_2	0.998	41.126	0.005	3.30e-06	0.004	0.004	3.6ms	19.8MB
	NC V_1	0.998	41.575	0.005	3.30e-06	0.004	0.004	2.9ms	3.9MB
	NC V_2	0.996	38.444	0.046	6.21e-06	0.007	0.500	3ms	3.9MB
Fruits360	V_1	0.983	1089.938	0.043	1.19e-06	0.008	0.113	4.4ms	41MB
	V_2	0.986	816.893	0.144	1.57e-06	0.013	0.407	4ms	41.2MB
	V_3	0.985	870.470	0.154	1.55e-06	0.012	0.392	4.6ms	41.2MB
	V_4	0.985	836.295	0.159	1.58e-06	0.013	0.384	2.7ms	41.2MB
	k-nn V_1	0.993	532.314	0.025	1.19e-06	0.010	0.073	3.3ms	127.5MB
	k-nn V_2	0.993	466.311	0.088	1.42e-06	0.011	0.243	3.7ms	128.1MB
	NC V_1	0.991	605.087	0.027	1.19e-06	0.010	0.045	3.6ms	14MB
	NC V_2	0.988	725.556	0.208	2.22e-06	0.018	0.500	3.5ms	14.2MB
Ecobee Thermostat	V_1	0.823	4732.483	0.218	2.67e-08	0.003	0.009	0.7ms	52.2kB
	V_2	0.830	4310.008	0.200	4.20e-08	0.003	0.014	0.7ms	53.2kB
	V_3	0.830	4311.460	0.200	4.33e-08	0.002	0.015	0.7ms	53.2kB
	V_4	0.830	4306.791	0.200	4.26e-08	0.003	0.040	0.6ms	53.2kB
	k-nn V_1	0.935	2872.725	0.113	2.94e-08	0.001	0.003	1.9ms	43.8MB
	k-nn V_2	0.935	2299.023	0.096	1.33e-07	0.006	0.375	2.4ms	43.8MB
	NC V_1	0.794	5550.013	0.255	2.78e-08	0.006	0.017	1ms	24kB
	NC V_2	0.794	5541.171	0.255	4.02e-08	0.006	0.023	0.9ms	25kB

[5] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger. On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, pages 1321–1330. JMLR.org, 2017.

[6] R. Hadsell, S. Chopra, and Y. LeCun. Dimensionality reduction by learning an invariant mapping. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 1735–1742. IEEE, 2006.

[7] G. E. Hinton. Learning multiple layers of representation. *Trends in cognitive sciences*, 11(10):428–434, 2007.

[8] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

[9] G. Koch, R. Zemel, and R. Salakhutdinov. Siamese neural networks for one-shot image recognition. In *ICML deep learning workshop*, volume 2. Lille, 2015.

[10] C. Kolias, G. Kambourakis, A. Stavrou, and J. Voas. Ddos in the iot: Mirai and other botnets. *Computer*, 50(7):80–84, 2017.

[11] A. Kumar, P. S. Liang, and T. Ma. Verified uncertainty calibration. In *Advances in Neural Information Processing Systems*, pages 3792–3803, 2019.

[12] A. Kumar, S. Sarawagi, and U. Jain. Trainable calibration measures for neural networks from kernel mean embeddings. In J. Dy and A. Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 2805–2814, Stockholmsmässan, Stockholm Sweden, 10–15 Jul 2018. PMLR.

[13] A. Lambrou, I. Nouretdinov, and H. Papadopoulos. Inductive venn prediction. *Annals of Mathematics and Artificial Intelligence*, 74(1-2):181–201, 2015.

[14] A. Lambrou, H. Papadopoulos, I. Nouretdinov, and A. Gammerman. Reliable probability estimates based on support vector machines for large multiclass datasets. In *IFIP International Conference on Artificial Intelligence Applications and Innovations*, pages 182–191. Springer, 2012.

[15] Y. Meidan, M. Bohadana, Y. Mathov, Y. Mirsky, A. Shabtai, D. Breitenbacher, and Y. Elovici. N-baiot—network-based detection of iot botnet attacks using deep autoencoders. *IEEE Pervasive Computing*, 17(3):12–22, 2018.

[16] H. Mureşan and M. Oltean. Fruit recognition from images using deep learning. *arXiv preprint arXiv:1712.00580*, 2017.

[17] M. P. Naeini, G. Cooper, and M. Hauskrecht. Obtaining well calibrated probabilities using bayesian binning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 29, 2015.

[18] Y. Ovadia, E. Fertig, J. Ren, Z. Nado, D. Sculley, S. Nowozin, J. Dillon, B. Lakshminarayanan, and J. Snoek. Can you trust your model's uncertainty? evaluating predictive uncertainty under dataset shift. In *Advances in Neural Information Processing Systems*, pages 13991–14002, 2019.

[19] H. Papadopoulos. Reliable probabilistic prediction for medical decision support. In *Artificial Intelligence Applications and Innovations*, pages 265–274. Springer, 2011.

[20] H. Papadopoulos. Reliable probabilistic classification with neural networks. *Neurocomputing*, 107:59–68, 2013.

[21] H. Papadopoulos, V. Vovk, and A. Gammerman. Regression conformal prediction with nearest neighbours. *J. Artif. Int. Res.*, 40(1):815–840, Jan. 2011.

[22] G. Pereyra, G. Tucker, J. Chorowski, Ł. Kaiser, and G. Hinton. Regularizing neural networks by penalizing confident output distributions. *arXiv preprint arXiv:1701.06548*, 2017.

[23] J. C. Platt. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In *ADVANCES IN LARGE MARGIN CLASSIFIERS*, pages 61–74. MIT Press, 1999.

[24] P. J. Rousseeuw. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, 20:53 – 65, 1987.

[25] R. Selten. Axiomatic characterization of the quadratic scoring rule. *Experimental Economics*, 1(1):43–61, 1998.

[26] G. Shafer and V. Vovk. A tutorial on conformal prediction. *J. Mach. Learn. Res.*, 9:371–421, June 2008.

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

[28] V. Vovk, A. Gammerman, and G. Shafer. *Algorithmic Learning in a Random World*. Springer-Verlag, Berlin, Heidelberg, 2005.

[29] V. Vovk, G. Shafer, and I. Nouretdinov. Self-calibrating probability forecasting. In *NIPS*, pages 1133–1140, 2003.