Justification-Based Reliability in Machine Learning

Nurali Virani,[*] **Naresh Iyer, Zhaoyuan Yang**
GE Research
1Research Circle, Niskayuna, NY 12309
{nurali.virani, iyerna, zhaoyuan.yang}@ge.com

Abstract

With the advent of Deep Learning, the field of machine learning (ML) has surpassed human-level performance on diverse classification tasks. At the same time, there is a stark need to characterize and quantify reliability of a model's prediction on individual samples. This is especially true in applications of such models in safety-critical domains of industrial control and healthcare. To address this need, we link the question of reliability of a model's individual prediction to the epistemic uncertainty of the model's prediction. More specifically, we extend the theory of Justified True Belief (JTB) in epistemology, created to study the validity and limits of human-acquired knowledge, towards characterizing the validity and limits of knowledge in supervised classifiers. We present an analysis of neural network classifiers linking the reliability of its prediction on a test input to characteristics of the support gathered from the input and hidden layers of the network. We hypothesize that the JTB analysis exposes the epistemic uncertainty (or ignorance) of a model with respect to its inference, thereby allowing for the inference to be only as strong as the justification permits. We explore various forms of support (for e.g., k-nearest neighbors (k-NN) and ℓ_p-norm based) generated for an input, using the training data to construct a justification for the prediction with that input. Through experiments conducted on simulated and real datasets, we demonstrate that our approach can provide reliability for individual predictions and characterize regions where such reliability cannot be ascertained.

Introduction

Predictive and prescriptive models based on ML are at the heart of the modern digital revolution. They are applied across several domains, such as energy, finance, defense, network security, and healthcare. These models often directly influence control actions or indirectly influence decisions through recommendations. In many scenarios, especially safety-critical ones, although the average performance or accuracy of such models is important, the reliability of the model on every individual prediction is equally critical. By reliability, we mean the degree to which the result of

---

[*]All authors contributed equally.

the model can be relied on to be accurate. A model with very good average performance is still capable of causing irreparable damage due to the mistakes it makes on a single or few predictions. In this paper, we present an analysis on the reliability of individual predictions of a ML model.

We draw inspiration from epistemology, the theory of knowledge, which deals with validity of methods used to acquire knowledge. Specifically, we examine the applicability of Plato's classical Justified True Belief (JTB) theory in epistemology (Ichikawa and Steup 2001). The JTB theory of knowledge is a classical theory that attempts to provide necessary and sufficient conditions under which a person can be said to *know* something. We extend this theory to ML models by firstly positing that an individual prediction from a model can be seen as a *belief* equivalent to "I *believe* input x is of class y". Next, we propose that in order for such belief to be considered as *knowledge*, an additional step of constructing justification for the belief is necessary. We explore formulations of JTB analysis by which ML models can construct such justifications for their beliefs.

We contend that existing approaches conflate epistemic uncertainty, which pertains to uncertainty due to information that is absent, with aleatoric uncertainty, which pertains to uncertainty arising from variability in the available information. JTB analysis tackles this fundamental confusion by not relying on confidence intervals for point-class assignments to obtain prediction reliability, and by abstaining from making point-class predictions on inputs, if concrete evidence for such assessment is absent. Some existing approaches make use of thresholding on softmax layer values or distance from the decision hyperplane to abstain from making a class assignment. These approaches fail to link prediction reliability to the location of the sample in the overall input space. It is becoming clearer based on recent work (Meng and Chen 2017) that input regions, either close to the decision boundary or in regions of extrapolation, are particularly susceptible with regard to making unreliable predictions. We present a mechanism for characterizing regions in input space as region of extrapolation (i.e., "I don't know" or IDK), region of confusion (i.e., "I may know" or IMK), and region of trust (i.e., "I know" or IK). This novel mechanism to construct justified belief with IK, IMK, and IDK directly

affects the reliability of the decisions made downstream in the analytical workflows. While IMK informs that support is impure and additional information is needed to improve reliability, empty neighborhood in IDK informs that input is anomalous/novel.

Additionally, it has been shown (De Vries, Memisevic, and Courville 2016) that the softmax classifier can make high confidence predictions that are inaccurate even for points that are at a large distance from the decision hyperplane. JTB analysis overcomes these issues by explicitly factoring in the location of a sample in the input space, while making its class prediction, through constructing support for the prediction based on other training data points in the vicinity of the sample. Suggestions related to abstaining from making point-class predictions have been made recently (Shafahi et al. 2018; Rouani et al. 2019; Miller, Wang, and Kesidis 2019) (especially for adversarial attack detection), but we believe our approach is the first to separately characterize overlap (IMK) and extrapolation (IDK) for individual predictions from neural networks. DeepKNN (Papernot and McDaniel 2018) and trust score (Jiang et al. 2018) approaches collect evidence for prediction reliability by identifying nearest neighbors to the sample from the training points across single/multiple layers of a neural network; the class-membership of the neighbors are then used to assign a nonconformity/trust score to the class prediction. Although our approach leverages this work, by considering the k-NN support operator for the JTB analysis, besides other forms of support like ℓ_p-norm based and novel hybrid forms of support, our work is different: it can identify region of extrapolation (IDK) and construct justification for prediction reliability, whereas the statistical nature of trust scores needs thresholding to do the same.

The main objectives and contributions of this work are given as follows:

- To introduce, formalize, and illustrate the notion of justified belief from epistemology for ML models to gain reliability in individual predictions.

- To contrast performance of Epistemic classifiers with those that rely on distance from hyperplane on real and simulated datasets.

We next introduce Epistemic classifiers, instantiated by the application of JTB analysis to standard classifier models. We use the rest of the paper to define concepts necessary for formalizing the JTB analysis framework, describing algorithmic steps for performing training and inference for JTB, describing experiments with applying Epistemic classifiers and analysis of outcomes from it, and finishing with a discussion of potential future directions and conclusions.

Concept of Epistemic Classifiers

Presently, a classification problem is not evaluated in terms of the true class observability permitted by the training data used to construct classifiers. In reality, the ground truth of a particular instance might not be fully observable from the available training data implying that reliable single-point classification is unachievable. Epistemic classifiers address
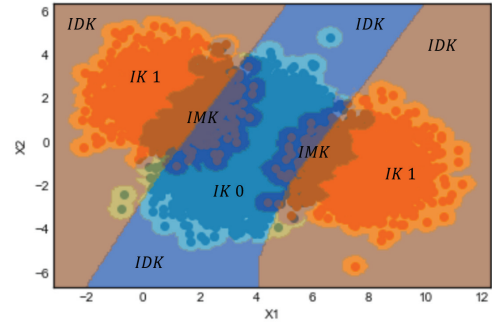

Figure 1: Illustration of region of trust (IK0, IK1), region of confusion (IMK), and region of extrapolation (IDK) for 2-D input binary classification with Epistemic classifier using neural network as base classifier

this issue by characterizing regions of observability and abstain from making class-assignments in regions of the input space where such characterization is found to be difficult to construct. Consider Fig. 1 illustrating the input space of a 2-D binary classification problem. The problem we are interested in is, given a test input x, to be able to characterize the input region for x during inference and provide estimated class-prediction with additional information indicating individual reliability of the prediction. The fundamental mechanism to enable this characterization of the input space is developed by application of the theory of Justified True Belief (JTB) from field of epistemology, which is discussed next.

The JTB theory of knowledge suggests that a subject S *knows* that a proposition P is true if and only if P is true, S believes that P is true, and S is justified in believing that P is true. Most of scientific knowledge is based on justified belief, where belief is a claim or hypothesis and justification is the experimental evidence or the mathematical proof leveraging existing justified beliefs to support the new belief. We extend this theory of knowledge as justified belief to supervised learning-based classifiers.

In current AI systems, *cross-validation accuracy of model is high* is used as a justification to rely on model outputs, but this form of justification, that uses aggregate statistics fails to account for unreliability at the level of individual predictions. To counter this, we introduce justification that gathers evidence using the training set for each individual test input x in the input and the hidden layers of a neural network classifier, where an unambiguous truth state in the neighborhood of x in those embedded spaces provides support to the belief allowing model to declare "I know P". This support and justification process to characterize knowledge, which uses neighborhood to allow generalization beyond labeled instances, is discussed next.

Neighborhood and Support

Consider a training set with predictors X and corresponding labels Y. A neighborhood operator $\mathcal{N}(x)$ for an input x is a function that returns a subset of training set, i.e., $\mathcal{N}(x) \subset X$, such that each data point $z \in \mathcal{N}(x)$ is similar to x using certain distance metric and selection criterion.
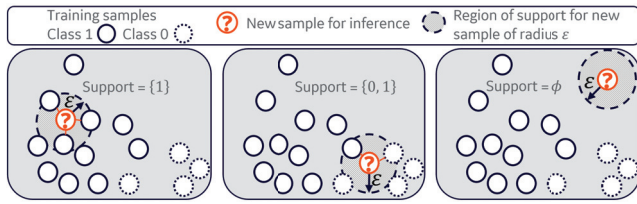
Figure 2: Construction of support in some embedded space using ε-balls as neighborhoods

Specific examples of neighborhood operators include ε-ball neighborhood $B_\varepsilon(x)$, which uses ℓ_p distance metric and criterion that $d(x, z) \leq \varepsilon$ for $\varepsilon \in \mathbb{R}^+$, and k-nearest neighbor neighborhood $N_k(x)$, which uses distance metric d to identify $k \in \mathbb{Z}^+$ nearest neighbors of x from the training set. Let function f map input from training data to set of labels, then given a neighborhood operator $\mathcal{N}_i(\cdot)$, the support of input x in i^{th} layer of neural network is defined as:

$$S_i(x) = \{f(z) : z \in X, h_i(z) \in \mathcal{N}_i(h_i(x))\}, \quad (1)$$

where $h_i(z)$ is the activation values of layer i in neural network with z as input. Thus via the neighborhood operator, the support operator is parametrized by distance bound ε_i or number of neighbors k_i. An illustration of ε-ball neighborhood to construct support is shown in Fig. 2.

A k-NN neighborhood identifies k similar instances. Thus, it will find k neighbors even when the sample is extrapolating and is semantically different from all neighbors. On other hand, ε-ball neighborhood operates on a fixed region in which it looks for evidence and that evidence varies based on data density and can also lead to empty support under extrapolating regions. However, this favorable variability is expected to lead to ε that might be conservative in one part of input space and loose in another part. Since, the formulation can handle general notion of neighborhood, we also propose hybrid versions of neighborhood operators: 1) H-1 : when ε-ball neighborhood is empty, then use k-NN neighborhood, and 2) H-2 : when ε-ball neighborhood is non-empty, compute union with k-NN neighborhood. Note that when ε-ball neighborhood is empty, H-1 provides more information about nearest neighbors, but it would potentially reduce robustness as those evidence might be obtained from semantically dissimilar classes. H-2 provides stronger robustness as it increases the set of evidence, but it would potentially increase impurity of evidence, i.e. data points from different classes. In this work, the neighborhood operators that are considered to construct support in layer i, given the choice of distance metric d_i, can be parametrized by size of ball neighborhood ε_i and number of neighbors k_i, i.e. we can represent neighborhood operator as $\mathcal{N}_i = \mathcal{N}_{d_i, \varepsilon_i, k_i}$. This support operator is evaluated over each layer of interest and then justification operator acts over these sets of support. This process of justification is explained next.

Justification and Knowledge

The justification operator takes as input a collection of supports from multiple layers and produces a set of justified class assignments for an input using (2).

$$J(x) = \begin{cases} \phi & \text{if for any } S_i(x) = \phi, \\ \bigcup_{i \in I} S_i(x) & \text{otherwise.} \end{cases} \quad (2)$$

If any support is empty, then $J(x)$ is also empty, else we use the disjunction or union for the justification operator. Thus, the justified class assigned to input x is a union of the class-assignments made by each of the supports from chosen layers. The disjunction makes for a liberal justification operator since a class that is contained even in one of the supports is carried on to the output.

Based on the outcome of applying the justification operator $J(x)$ and prediction or belief $g(x)$ for a given input x, the Epistemic classifier using justified belief may assert IK, IMK, or IDK. These assignments capture the epistemic uncertainty of the classifier with respect to the input's class. The IDK assertion captures the epistemic uncertainty encountered when a sample finds no support for the belief from existing training data (e.g., when the inference involves extrapolation), i.e. $g(x) \not\subset J(x)$. The IMK assertion captures the epistemic uncertainty when a sample finds conflicting support from the training data for two or more classes which also consists of the belief, i.e., $g(x)$ is proper subset of $J(x)$. For example, in Fig. 1, if test input $x = (2, 2)$, then $J(x) = \{0, 1\}$ from support in input layer and $g(x) = \{1\}$ leading to IMK assertion. Additionally, the IK assertion is provided when the belief and justification are exactly the same $g(x) = J(x)$, e.g., when $x = (8, 0)$ in Fig. 1. Clearly, these three degrees of epistemic uncertainty are ordered wherein IDK captures a relatively higher degree of ignorance about the sample class than the IMK and IMK has higher uncertainty than IK. Now that concept of the Epistemic classifier and intuition have been explained, we will formalize this classifier further.

Definition (Epistemic classifier). *Epistemic classifier (E) is a function mapping from input space to output label and epistemic certainty. It is defined by a tuple $E = (g, I, \mathcal{N})$, where g is the base neural network classifier, I is the set of chosen layers, and $\mathcal{N} = \{\mathcal{N}_1, \ldots, \mathcal{N}_{|I|}\}$ is a set consisting of neighborhood operator for each layer. For a given input x, this classifier obtains belief as $g(x)$ with base classifier and constructs justification $J(x)$ using (2) with neighborhood and support operators using (1) over layers in I. Then output $E(x) = (g(x), j \in \{IK, IMK, IDK\})$ consists of belief and assertion of IK (if $g(x) = J(x)$), IMK (if $g(x) \subset J(x)$), or IDK (if $g(x) \not\subset J(x)$) as the degree of epistemic certainty in the belief.*

Next we will show the training and inference algorithms as well as some analysis on choice of model parameters.

Algorithm and Analysis

In order to build an Epistemic classifier, we need a trained neural network model g, which is the base classifier. The evidence for justification process will be based on the training set (X, Y). The parameter optimization is conducted by evaluating metrics over the validation set (X^v, Y^v). We demonstrate our training approach in Algorithm 1.

Algorithm 1 – Training Epistemic Classifiers.

Require: training set (X, Y), validation set (X^v, Y^v)
Require: trained neural network g
Require: set of layers I for support construction
Require: metrics for each layer $\bar{d} = \{d_1, ..., d_{|I|}\}$
 1: $\Omega = \{\}$ ▷ Set of one search tree per layer in I
 2: **for** each layer $i \in I$ **do**
 3:     $\Gamma_i^X \leftarrow$ Extract activation $h_i(X)$ for training set X
 4:     $\Omega_i \leftarrow$ NeighborSearchTree(Γ_i^X, Y, d_i)
 5:     $\Omega \leftarrow \Omega \bigcup \Omega_i$
 6: **end for**
 7: $\bar{\varepsilon}, \bar{k} =$ ParameterSelection$(X^v, Y^v, \Omega, g, I, \bar{d})$
 8: **return** $g, I, \bar{d}, \bar{\varepsilon}, \bar{k}, \Omega$

Algorithm 2 – Inference with Epistemic Classifiers.

Require: Test input x
Require: Epistemic classifier $G = (g, I, \mathcal{N})$
 1: Compute belief as prediction $g(x)$
 2: **for** each layer $i \in I$ **do**
 3:     $\Gamma_i^x \leftarrow$ Extract activation $h_i(x)$ for test input x
 4:     $S_i(x) \leftarrow \Psi(\Gamma_i^x, \mathcal{N}_i)$ ▷ support of x in layer i
 5: **end for**
 6: Get justification $J(x)$ from supports $S_i(x)$ using (2).
 7: **if** $g(x) = J(x)$ **then**
 8:     output $\leftarrow$ (IK, $g(x)$)
 9: **else if** $g(x) \subset J(x)$ **then**   ▷ proper subset
10:     output $\leftarrow$ (IMK, $g(x)$)
11: **else**   ▷ implies $g(x) \not\subset J(x)$
12:     output $\leftarrow$ (IDK, $g(x)$)
13: **end if**
14: **return** output

Once we obtain the base classifier g, we feed the training set X into neural network model g and extract Γ_i^X, which is the activation from layer $i \in I$ from all training instances using $h_i(X)$. Recall that $h_i(\cdot)$ gives activation values of layer i for a given input. NeighborSearchTree represents a function which builds a nearest neighbor search tree based on the information from training data and distance metric. ParameterSelection represents a function, which selects a set of parameters $\bar{\varepsilon} = \{\varepsilon_i : i \in I\}$ and $\bar{k} = \{k_i : i \in I\}$ for support construction. We will discuss these functions next.

In NeighborSearchTree, we construct a set of ball-trees (Omohundro 1989) denoted by Ω to store activation values and their corresponding labels for each layer i. Unlike k-d trees (Bentley 1975), ball-trees lead to $O(\log |X|)$ average case complexity for exact search in moderately high dimensions, where complexity for k-d tree is linear. These ball-trees enable to efficiently conduct both nearest neighbor search and range search, i.e., find all neighbors within distance ε_i from test input z. In our implementation, we use ball-tree method from scikit-learn (Pedregosa et al. 2011) to construct the neighbor search tree.

Let us now consider a case where support is constructed only from one layer, then we study the effect of choice of ε over {IK, IMK, IDK} regions. Let F_{IK} be fraction of sam-
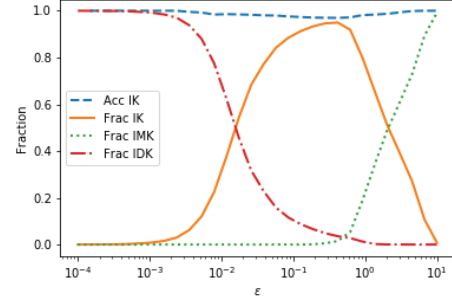

Figure 3: Illustration of behavior of fraction of IK, IMK, and IDK as well as accuracy on IK from Epistemic classifier with parameter of neighborhood operator

ples that are asserted to be "I know", similarly, we have F_{IMK} and F_{IDK} as fraction of samples for "I may know" and "I don't know" respectively. Then, we can make the following remarks about their behavior with ε:

Remark (Behavior of {IK, IMK, IDK} with ε). *Note that $F_{IK} + F_{IMK} + F_{IDK} = 1$, since {IK, IMK, IDK} are exhaustive and mutually-exclusive options for justified belief. If $\varepsilon \to 0$, then $F_{IK} = 0$, $F_{IMK} = 0$, $F_{IDK} = 1$, since very small neighborhoods will not be able to find any evidence. If $\varepsilon \to \infty$, then $F_{IK} = 0$, $F_{IMK} = 1$, $F_{IDK} = 0$, since very large neighborhoods will include training points from all classes as evidence. F_{IMK} is a monotonically-increasing function of ε, since impure support cannot become pure or empty, but pure or empty support of a data point can become impure as size of neighborhood increases. Similarly, F_{IDK} is a monotonically-decreasing function of ε, since empty neighborhood might get new evidence, but nonempty neighborhoods will not become empty as ε increases.*

Figure 3 illustrates these intuitive behaviors using a toy dataset (Gaussian distribution with some overlap) for a binary classification task. The metric F_{IK} or fraction of IK samples is also called coverage. Note that accuracy for IK samples stays high across different coverage levels as desired. Thus, in this work, the objective function used to choose the optimal ε is coverage and not accuracy. The relation of coverage with ε is piece-wise constant as data points move between regions of IDK or IMK and IK. Thus, this optimization for ParameterSelection function can be accomplished with grid search or by using Bayesian optimization (BO) (Snoek, Larochelle, and Adams 2012), where coverage metric is evaluated over validation set.

Given ε_i for one layer-case, we derive an upper bound of the ε_{i+1} such that if data point z is an ε_i-neighbor of the testing input x in layer i, then it is guaranteed that z is an ε_{i+1}-neighbor of the testing input x in layer $i + 1$.

Remark (ε bound). *If $W_i \in \mathbb{R}^{m \times n}$ is weight matrix of the layer i with input dimension m and output dimension n, λ_i^* is the largest eigenvalue of the matrix $W_i W_i^T$, L_i is the Lipschitz constant of the activation function of layer i, then given*

ε_i for layer i, the value of ε_{i+1} is bounded as follows:

$$\varepsilon_{i+1} \leq L_i \sqrt{\lambda_i^*} \varepsilon_i.$$

Proof. See Supplementary material (S1) in (Virani, Iyer, and Yang 2019). □

This result is used to choose conservative values of ε across multiple layers by setting value of ε_{i+1} at the upper bound. Note that the weight matrix of layer i leads to an affine projection of ε_i-ball, thus considering only scaling will lead to a loose bound in most of the directions except of the largest principal component. Thus, we also introduce a weighted distance metric based on the weight matrices of the neural network that is constructed for each layer. We define the following metric for each layer i, $d_i(x_a, x_b) = \sqrt{(x_a - x_b)^T D_i (x_a - x_b)}$ and D_i is a symmetric positive semi-definite matrix. The following remark informs how to choose the values for D_i.

Remark (Bound with weighted distance). *Let* $\widetilde{W}_i = W_1 W_2 ... W_i$, $\Lambda = diag([\lambda_1, \lambda_2, ...\lambda_n])$ *and* $U = [v_1, v_2, ...v_n]$ *as the first n non-zero eigenvalues and corresponding eigenvectors of matrix* $\widetilde{W}_i \widetilde{W}_i^T$, *and d_0 is ℓ_2 distance metric in input space. If $d_0(x, z) \leq \varepsilon_0$ then the distance function matrix D_i for neighbor search in any layer i:*

$$D_i = U^T \widetilde{W}_i \Lambda^{-1} \widetilde{W}_i^T U \qquad (3)$$

satisfies the following relation:

$$\sqrt{(h_i(x) - h_i(z))^T D_i (h_i(x) - h_i(z))} \leq \varepsilon_0$$

Proof. See Supplementary material (S1) in (Virani, Iyer, and Yang 2019). □

Thus, with this layer-specific metric for each layer, we can use the same value of ε_0 for each layer of interest. Due to activation functions, which are applied after the affine projection, the bound obtained by using weighted distance will still be conservative in several directions. Moreover, computation of eigenvalue decomposition for large weight matrices can also be intractable. We provide an alternate mechanism to compute the values of $\bar{\varepsilon}$ parameters. BO is proposed for selecting the value of $\bar{\varepsilon} = \{\varepsilon_i : i \in I\}$ for each layer. BO is computationally tractable for most situations as parameter tuning for Epistemic classifiers does not require base model retraining and repetition of ball-tree construction. Moreover, layers of interest are chosen to be very few $(1 - 3)$, so the number of parameters to jointly tune is very limited. The choice of layers where support is constructed is not considered as a hyperparameter for optimization. If we consider only output layer, then classifier will lack robustness against adversarial perturbations as the neighborhood might already be consistent with targeted class. If we consider only input layer, then it might be computationally expensive due to data dimensionality and in some cases we might lose some robustness as semantic similarity in input space might not be well-represented by ℓ_p-norms. Moreover, if neighborhood is pure in or close to input layer and is also pure in or close to final layer, and these neighbors are consistent, then the
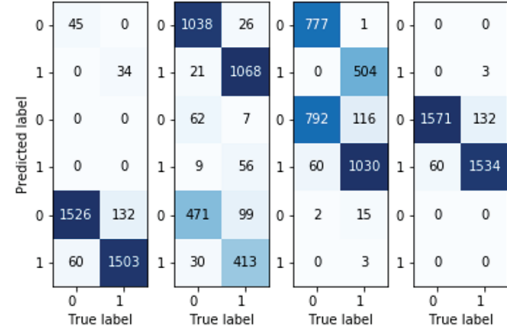

Figure 4: Effect of ε $(0.001, 0.017, 0.237, 3.162)$ on augmented confusion metric with noise std. of 1.0.

benign/malign perturbation was not successful in leading to misclassification or incorrect belief. Thus, a combination of layer closer to input, where search is tractable, and layer closer to output are considered. We will further justify this choice with experimental results.

During the inference stage, a testing sample x is used with the Epistemic classifier according to Algorithm 2. The activations due to the sample x from layers of interest I are extracted. The Epistemic classifier finds the neighborhood and support for the testing input x across the selected layers based on the extracted activation values. This operation to find support in layer i is denoted by operator Ψ, which uses activation from test point Γ_i^x in layer i and neighborhood operator $\mathcal{N}_i$. Justification is then created using (2) with support from layers of interest. This justification and belief is then used to obtain justified belief.

Experiments and Discussion

In this section, we will first identify metrics for comparison and then describe experiments with our inferences and insights from the results. We introduce the concept of augmented confusion matrix (ACM) that consists of three sub-matrices, where each sub-matrix is a confusion matrix for predicted label versus true label under assertion of IK (top), IMK (middle), and IDK (bottom). See Fig. 4 for examples of ACM. Several robust generalization metrics can be devised using this matrix. In this work, we will use coverage or fraction of IK (F_{IK}), accuracy over IK samples (A_{IK}), and accuracy over non-IK samples $(A_{\neg IK})$.

To generate Fig. 4, the Epistemic classifier with a fully-connected neural network is trained with data generated using *make_blobs* (bivariate Gaussian) from scikit-learn. We use ε-ball neighborhood with ℓ_2 distance metric and we use ball-tree to conduct search. Given 4 distinct values of ε on log-scale, the results on test data is shown with ACM in Fig. 4. It is noted that as value of ε is small (resp. large), we have almost all assertions as IDK (resp. IMK). Moreover, in general, the confusion matrix under small ε for IDK and that under large ε for IMK is also similar. As ε increases the points in IDK region get assigned to IK and some to IMK region. At higher values, points in IK region move to IMK and finally at large ε all points are assigned to IMK region.
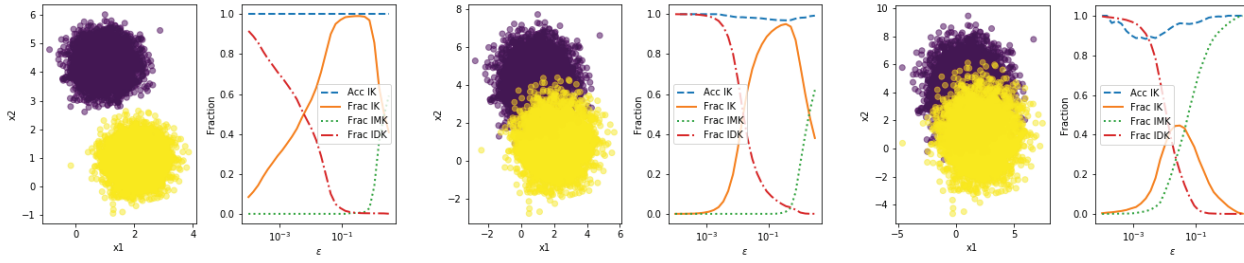
Figure 5: Effect of noise ($\sigma = 0.5, 1.0, 1.5$) and overlap on metrics as a function of ε

		Nominal			Gaussian			Uniform			Large-Noise		
		F_{IK}	A_{IK}	$A_{\neg IK}$	F_{IK}	A_{IK}	$A_{\neg IK}$	F_{IK}	A_{IK}	$A_{\neg IK}$	F_{IK}	A_{IK}	$A_{\neg IK}$
Grid	Base	0.661	0.995	0.826	0.720	0.863	0.573	0.770	0.721	0.557	0.933	0.433	0.494
model:	k-NN	0.692	0.990	0.818	0.720	0.869	0.560	0.754	0.748	0.486	0.776	0.457	0.368
FC	ε-NN	0.615	0.992	0.850	0.532	0.858	0.696	0.331	0.738	0.656	**0.001**	0.750	0.437
(93.74)	H-2	0.581	0.995	0.858	0.494	0.878	0.688	0.301	0.757	0.651	**0.001**	0.750	0.437
Iris	Base	0.889	1.000	1.000	0.867	1.000	1.000	0.822	0.973	1.000	0.800	0.528	0.222
model:	k-NN	0.911	1.000	1.000	0.889	1.000	1.000	0.889	0.975	1.000	0.822	0.459	0.500
FC	ε-NN	0.889	1.000	1.000	0.800	1.000	1.000	0.800	1.000	0.889	**0.178**	0.500	0.459
(100)	H-2	0.889	1.000	1.000	0.800	1.000	1.000	0.800	1.000	0.889	**0.178**	0.500	0.459
Italy	Base	0.897	0.985	0.689	0.862	0.983	0.662	0.814	0.956	0.665	0.796	0.712	0.548
model:	k-NN	0.909	0.981	0.691	0.864	0.983	0.657	0.826	0.954	0.654	0.718	0.717	0.579
CNN	ε-NN	0.894	0.977	0.761	0.821	0.970	0.793	0.588	0.942	0.844	**0.001**	1.000	0.678
(95.40)	H-2	0.856	0.982	0.791	0.765	0.986	0.785	0.543	0.957	0.836	**0.001**	1.000	0.678
		Nominal			Gaussian			Uniform			Adv-Noise		
		F_{IK}	A_{IK}	$A_{\neg IK}$	F_{IK}	A_{IK}	$A_{\neg IK}$	F_{IK}	A_{IK}	$A_{\neg IK}$	F_{IK}	A_{IK}	$A_{\neg IK}$
SynCon	Base	0.907	0.996	0.929	0.730	0.995	0.728	0.727	0.977	0.756	0.243	0.562	0.577
model:	k-NN	0.907	1.000	0.893	0.733	0.991	0.738	0.730	0.977	0.753	**0.440**	0.955	0.274
CNN	ε-NN	0.897	1.000	0.903	0.493	0.980	0.868	0.433	0.985	0.865	0.343	0.942	0.381
(99.00)	H-2	0.863	1.000	0.927	0.437	0.992	0.870	0.393	0.983	0.874	0.307	0.957	0.404
MNIST	Base	0.622	0.998	0.899	0.424	0.988	0.738	0.306	0.888	0.491	0.851	0.026	0.011
model:	k-NN	0.671	0.999	0.881	0.449	0.996	0.720	0.234	0.976	0.501	0.222	0.028	0.023
FC	ε-NN	0.590	0.997	0.907	0.371	0.982	0.763	0.108	0.901	0.577	**0.000**	0.000	0.024
(96.03)	H-2	0.495	0.999	0.922	0.279	0.997	0.785	0.071	0.982	0.584	**0.000**	0.000	0.024
GTSRB	Base	0.466	1.000	0.952	0.475	1.000	0.913	0.485	1.000	0.833	0.631	0.064	0.112
model:	k-NN	0.595	1.000	0.936	0.574	1.000	0.892	0.535	0.999	0.816	0.414	0.099	0.069
CNN	ε-NN	0.488	0.999	0.950	0.488	0.998	0.912	0.485	0.996	0.837	**0.140**	0.015	0.092
(97.41)	H-2	0.377	1.000	0.959	0.373	1.000	0.927	0.359	1.000	0.865	**0.102**	0.018	0.089

Table 1: Results on feature-based (Grid Stability and Iris), time-series (Italy Power and SynCon), as well as image (MNIST and GTSRB) datasets. Base model test accuracy on nominal data is provided in the first column.

In Fig. 5, we conduct an experiment with similar setup but now study variation due to extent of overlap. Irrespective of choice of parameters, we should not be able to provide reliability for inputs coming from region that has overlap, but we should be able to characterize that region of confusion. Here we show the scatter plot of complete dataset and variation in accuracy of IK (A_{IK}) and fraction of IK (F_{IK}) metrics with choice of ε. Note that maximum value of F_{IK} reduces as overlap increases implying that region where reliable predictions can be made has reduced, e.g. $F_{IK} \approx 1.00$, when

noise std. $\sigma = 0.5$ and $F_{IK} \approx 0.44$, when noise std. is 1.5.

We then conducted experiments using Grid Stability (Arzamasov, Böhm, and Jochem 2018) and Iris dataset from UCI Repository (Dua and Graff 2017), Italy Power Demand classification (Keogh et al. 2006) and Synthetic Control (SynCon) dataset (Alcock, Manolopoulos, and others 1999) from UCR Time-series Repository (Chen et al. 2015), MNIST image dataset (LeCun 1998), and German Traffic Sign Recognition Benchmark (GTSRB) dataset (Stallkamp et al. 2012) to test prediction reliability under various pertur-

bations for different baseline models. We used convolutional neural networks (CNN) for Italy Power, SynCon, and GT-SRB datasets and for other datasets we used fully-connected (FC) neural networks as base classifiers. After generating these base classifers, we build Epistemic classifiers using Algorithm 1. We show performance of the Epistemic classifier on normal holdout testing data as well as on test datasets with perturbations. In top part of Table 1, we consider Gaussian noise, uniform noise, and large uniform noise to create rubbish and highly overlapping examples. In bottom part of Table 1, we consider Gaussian noise, uniform noise, and adversarial perturbation using Basic Iterative Method (Kurakin, Goodfellow, and Bengio 2016) from Adversarial Robustness Toolbox (Nicolae et al. 2018). Details of individual datasets, base model architectures, and perturbation magnitudes are reported in Supplementary material (S2) in (Virani, Iyer, and Yang 2019). The baseline approach for comparison uses a calibrated threshold over softmax layer output that allows to abstain when the maximum of softmax layer values is below a certain threshold. To make the comparison fair, we first determine the optimal ε for ε-NN neighborhood operator, where F_{IK} is maximized. We tune the parameters for our baseline and k-NN support such that they have the similar coverage as ε-ball support, then we make comparison in the table to compare accuracy and performance under nominal data and various perturbations.

Few insights from Table 1 are discussed next. Softmax thresholding baseline performs well on zero-mean small perturbations and is comparable to performance with k-NN neighborhood operator. However, specifically consider Grid, MNIST, and GTSRB dataset, under large uniform noise and adversarial perturbation cases, where distance to hyperplane and location of other data points would start to differ, we see that baseline has high fraction of IK but very poor accuracy on IK. Under these cases, ideal solution is to have high F_{IK} and high A_{IK} for adversarial robustness or at least be able to assert IDK and get very low F_{IK} for detection of malicious perturbations. This is true for ε-NN and H-2 neighborhood operator as highlighted in bold. Note that accuracy metric for large perturbation does not matter as the test samples are more likely to be rubbish examples in vicinity of the wrong class. The results also highlight that ε-NN and H-2 cases have lower coverage than k-NN in all cases, however it has superior reliability under large perturbations. The results for H-1 support are similar to k-NN operator (see Supplementary material in (Virani, Iyer, and Yang 2019)). In Supplementary material (S3), we have provided additional results where we consider neighborhood only in logit layer as well as in logit and intermediate layers together. Like baseline, support in only logit layer has good coverage, but lacks robustness under perturbations, thus gathering support from at least two layers is preferable. Although, the metrics in Table 1 denote ability of developed approach to provide individual reliability in scenarios of extrapolation (i.e. anomaly detection) and overlap, additional outcomes showing ability to characterize overlap between subset of classes are omitted due to space restrictions.

This paper deals with the reliability due to observability and noise. While BIM-based adversarial attacks have

been considered, we have explored other inference-time and backdoor attacks as well. A detailed treatment of adversarial robustness is being planned for a follow-on paper.

Challenges and Limitations

We identify 5 primary challenges related to the proposed approach. Firstly, large dimensionality of input and embedded space in early layers results in the ball-tree search algorithm to be computationally expensive for large early convolutional layers of state-of-the-art models. Locality-sensitive hashing (Slaney and Casey 2008) and other approximate neighbor search approaches (Andoni and Indyk 2006) can be used to alleviate these issues. Secondly, number of training data points, i.e., $|X|$ leads to $O(\log |X|)$ average case complexity for search and $O(|X|)$ for space, which becomes of concern for large datasets. Thirdly, a more crucial challenge is the closeness of semantic distance between points and the mathematically-convenient ℓ_2-norm distance metric in input and early layers. The closeness of these metrics increases as we go deeper in the network, since neural network training is expected to bring data points from same classes closer to enable linear separation, however in input and initial layers this closeness is usually not available. Fourthly, sparsity of data in neighborhood either due to high dimensionality or low sample size will also adversely affect the performance of this classifier. Finally, the current approach is suited only for neural networks, a model-agnostic approach using suitable input representation and output with Platt scaling (Platt and others 1999) will be explored for other ML models. These challenges will be further explored in near future.

Conclusion

In this work, we have introduced a novel approach to provide individual reliability of predictions from ML models. We leveraged the notion of Knowledge as Justified Belief from the field of epistemology to create Epistemic classifiers in ML. Epistemic classifier adds more contextual information based on location of training data points in input and hidden layers to add reliability on individual predictions, wherever plausible. We also introduced various domain-agnostic neighborhood operators which are used to gather evidence to construct justification that has utility for example-based explanations (Molnar 2019) for model interpretability. We analyzed the effect of parameter choices, noise in training dataset, and benign/malign perturbations on performance, where performance was reported using new metrics relevant to the epistemic analysis. We contrasted the performance of Epistemic classifiers with a baseline of thresholding softmax layer values and demonstrated superiority in robustness. The results were demonstrated on simulated and real datasets with few features, time-series data, and image data to show flexibility of the solution. Finally, key limitations were highlighted which will pave the way for future research.

Acknowledgments

References

Alcock, R. J.; Manolopoulos, Y.; et al. 1999. Time-series similarity queries employing a feature-based approach. In *7th Hellenic conference on informatics*, 27–29.

Andoni, A., and Indyk, P. 2006. Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. In *2006 47th annual IEEE symposium on foundations of computer science (FOCS'06)*, 459–468. IEEE.

Arzamasov, V.; Böhm, K.; and Jochem, P. 2018. Towards concise models of grid stability. In *2018 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm)*, 1–6. IEEE.

Bentley, J. L. 1975. Multidimensional binary search trees used for associative searching. *Communications of the ACM* 18(9):509–517.

Chen, Y.; Keogh, E.; Hu, B.; Begum, N.; Bagnall, A.; Mueen, A.; and Batista, G. 2015. The UCR time series classification archive. www.cs.ucr.edu/~eamonn/time_series_data/.

De Vries, H.; Memisevic, R.; and Courville, A. C. 2016. Deep learning vector quantization. In *ESANN*.

Dua, D., and Graff, C. 2017. UCI machine learning repository.

Ichikawa, J. J., and Steup, M. 2001. The analysis of knowledge. *The Stanford encyclopedia of philosophy*.

Jiang, H.; Kim, B.; Guan, M.; and Gupta, M. 2018. To trust or not to trust a classifier. In *Advances in neural information processing systems*, 5541–5552.

Keogh, E.; Wei, L.; Xi, X.; Lonardi, S.; Shieh, J.; and Sirowy, S. 2006. Intelligent icons: Integrating lite-weight data mining and visualization into GUI operating systems. In *Sixth International Conference on Data Mining (ICDM'06)*, 912–916. IEEE.

Kurakin, A.; Goodfellow, I.; and Bengio, S. 2016. Adversarial machine learning at scale. *arXiv preprint arXiv:1611.01236*.

LeCun, Y. 1998. The MNIST database of handwritten digits. *http://yann. lecun. com/exdb/mnist/*.

Meng, D., and Chen, H. 2017. Magnet: a two-pronged defense against adversarial examples. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, 135–147. ACM.

Miller, D.; Wang, Y.; and Kesidis, G. 2019. When not to classify: Anomaly detection of attacks (ADA) on DNN classifiers at test time. *Neural computation* 31(8):1624–1670.

Molnar, C. 2019. *Interpretable Machine Learning*. https://christophm.github.io/interpretable-ml-book/.

Nicolae, M.-I.; Sinn, M.; Tran, M. N.; Rawat, A.; Wistuba, M.; Zantedeschi, V.; Baracaldo, N.; Chen, B.; Ludwig, H.; Molloy, I. M.; et al. 2018. Adversarial Robustness Toolbox v0.10.0. *arXiv preprint arXiv:1807.01069*.

Omohundro, S. M. 1989. *Five ball-tree construction algorithms*. International Computer Science Institute Berkeley.

Papernot, N., and McDaniel, P. 2018. Deep k-nearest neighbors: Towards confident, interpretable and robust deep learning. *arXiv preprint arXiv:1803.04765*.

Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V.; et al. 2011. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research* 12(Oct):2825–2830.

Platt, J., et al. 1999. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advances in large margin classifiers* 10(3):61–74.

Rouani, B. D.; Samragh, M.; Javidi, T.; and Koushanfar, F. 2019. Safe machine learning and defeating adversarial attacks. *IEEE Security & Privacy* 17(2):31–38.

Shafahi, A.; Huang, W. R.; Studer, C.; Feizi, S.; and Goldstein, T. 2018. Are adversarial examples inevitable? *arXiv preprint arXiv:1809.02104*.

Slaney, M., and Casey, M. 2008. Locality-sensitive hashing for finding nearest neighbors [lecture notes]. *IEEE Signal processing magazine* 25(2):128–131.

Snoek, J.; Larochelle, H.; and Adams, R. P. 2012. Practical Bayesian optimization of machine learning algorithms. In *Advances in neural information processing systems*, 2951–2959.

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.

Virani, N.; Iyer, N.; and Yang, Z. 2019. Justification-based reliability in machine learning. *arXiv preprint arxiv:1911.07391*.