A Geometric Method for Improved Uncertainty Estimation in Real-time

Abstract

Machine learning classifiers are probabilistic in nature, and thus inevitably involve uncertainty. Predicting the probability of a specific input to be correct is called uncertainty (or confidence) estimation and is crucial for risk management. Post-hoc model calibrations can improve models' uncertainty estimations without the need for retraining, and without changing the model. Our work puts forward a geometric-based approach for uncertainty estimation. Roughly speaking, we use the geometric distance of the current input from the existing training inputs as a signal for estimating uncertainty and then calibrate that signal (instead of the model's estimation) using standard post-hoc calibration techniques. We show that our method yields better uncertainty estimations than recently proposed approaches by extensively evaluating multiple datasets and models. In addition, we also demonstrate the possibility of performing our approach in near real-time applications. Our code is available at anonymous-Git.

1 INTRODUCTION

Machine learning models such as neural networks, random forests, and gradient boosted trees are extensively used in domains ranging from computer vision to transportation and are slowly revolutionizing computer science [Niculescu-Mizil and Caruana, 2006, Zhang and Haghani, 2015]. Dealing with uncertainty is a fundamental challenge for all machine learning-based applications. In principle, classifications are always probabilistic, implying that miss-classifications are inevitable.

Uncertainty calibration is the process of adapting machine learning models' confidence estimations to be consistent with the actual success probability of the model [Guo et al., 2017a]. The model's *confidence* evaluation on its classifications, i.e., the model's prediction of the success ratio on a specific input, is an essential aspect of mission-critical machine learning applications as it provides a realistic estimate of the classification's success probability and facilitates informed decisions about the *current* situation. Even a very accurate model may run into an unexpected situation, which could then be communicated to the user by the confidence estimation. For example, consider an autonomous driver that uses a model to identify and classify traffic signs. The model is very accurate, and in most cases, its classifications are correct with high confidence. However, one day, it encountered a traffic sign obscured by, e.g., heavy vegetation. In such a case, the model's classification is more likely to be incorrect. Thus, estimating confidence (i.e., uncertainty) is an essential tool for assessing unavoidable risks, enabling system designers to address the risks better, potentially avoiding unexpected and catastrophic implications. Our autonomous driver, for example, may reduce the speed and activate additional sensors until it reaches higher confidence. Thus, indeed, all popular machine learning models have mechanisms for determining confidence that can be calibrated to maximize the quality of confidence estimations [Niculescu-Mizil and Caruana, 2005, Guo et al., 2017b, Kumar et al., 2019] and there is a concentrated effort to calibrate models better and facilitate more dependable applications [Leistner et al., 2009, Sun et al., 2007].

Existing calibration methods can be divided into two types: *post-hoc* methods that preform a transformation that maps from classifiers' raw outputs to their expected probabilities [Kull et al., 2019, Guo et al., 2017a, Gupta and Ramdas, 2021b], and *ad-hoc* methods that adapt the training process to generate better calibrated design [Thulasidasan et al., 2019, Hendrycks et al., 2019a]. Post-hoc calibration methods are easier to apply as they do not change the model and do not require us to retrain a model. That said, ad-hoc methods may lead us to better model training in the first place and thus better models. With the success of the two approaches, recent approaches suggest using ensemble methods whose

Submitted to the 38th Conference on Uncertainty in Artificial Intelligence (UAI 2022). **To be used for reviewing only**.

estimation is a (weighted) average of multiple calibration methods [Ashukha et al., 2020, Ma et al., 2021].

This paper presents a post-hoc uncertainty estimation method, but of a different kind. While current post-hoc methods use the model itself as their signal for calibration, we use the training dataset (without modifying the model). More precisely, our method is based on geometric notions calculated on the training dataset. In fact, our geometric choice of the signal is orthogonal to the preformed calibration method in the sense that we can employ it using various post-hoc calibration methods.

Roughly speaking, we examine the geometric distance of the current input from the existing training inputs and use it to estimate the model's confidence. Intuitively, the confidence is high when the current input is close to training set inputs in the same classification and is far from training set inputs with other classifications. Dually, the confidence would be low when there are very close training set inputs with different classifications. To maximize this geometric signal the inputs should be normalized. That is, the size, format, etc. of the images should be consistent. Thus, in this paper, we employ such well behaved datasets.

Our work demonstrates that geometry can facilitate better uncertainty estimations for diverse models and datasets. We first provide an algorithm for calculating the maximal geometric *separation* of an input. However, calculating the geometric separation for an input requires evaluating the whole space of training inputs, making it a computationally expensive method that is not always feasible. For example, an autonomous driver needs to reach decisions within a short time frame to be effective. Therefore, we also suggest a lightweight approximation called *fast-separation*, and show that it provides an approximation of geometric separation.

Our next challenge is to move from a separation value to a confidence estimation. For this, we apply numerical analysis tools. Thus, to obtain a confidence estimation in real-time we only need to apply a regression function to the calculated separation value. Interestingly, our extensive simulation across different models and datasets shows that our geometric-based method yields better confidence estimations when compared to popular libraries used in the industry [Pedregosa et al., 2011] , as well as recently proposed calibration methods [Kumar et al., 2019, Gupta and Ramdas, 2021a]. Furthermore, our evaluation shows that using our method with the fast-separation approximation allows for multiple confidence estimations per second, making it real-time applicable.

2 RELATED WORKS

The dependability of machine learning models is a key challenge in the research community [Johnson, 2018]. Various works demonstrate vulnerabilities in popular machine learning models [Biggio et al., 2014b,a], or show explicit methods to generate adversarial inputs to such models [Zhou et al., 2012]. Unfortunately, such vulnerabilities are fundamental to the field and cannot be avoided.

As mentioned above, uncertainty calibration is about estimating the model's success probability of classifying a given example. Post-hoc calibration methods apply some transformation to the model's confidence (without changing the model) such transformations include Temperature Scaling (TS) [Guo et al., 2017a, Kull et al., 2019], Ensemble Temperature Scaling (ETS) [Zhang et al., 2020], and cubic spline [Gupta and Ramdas, 2021a]. In brief, these methods are limited by the best learn able mapping between the model's confidence estimations, and the actual confidence. That is, post-hoc calibration methods are limited in mapping each confidence value to another calibrated value. In comparison, our method uses geometric distance as a signal for calibration and its improvement over post-hoc calibration is because geometric distances better differentiate then the model's predicted probabilities in the models and datasets included in our evaluation. The recently proposed work of [Kumar et al., 2019] uses a fitting function on the confidence values and then divides the inputs into bins of equal size and outputs the function's average in each bin. The work of [Gupta and Ramdas, 2021a] uses a similar idea but divide the inputs into uniform-mass (rather than equal size) bins. It is interesting to note that while most post-hoc calibration methods are model agnostic, recent methods have begun to look on a neural network non-probabilistic output called logits[Guo et al., 2017b, Z.Ding et al., 2020, J.Wenger et al., 2019] (before applying softmax). Thus, some of the new post-hoc calibration methods are applicable only to neural networks.

Ensemble methods are similar to post-hoc calibration methods as they do not change the model, but they consider multiple signals to determine the model's confidence [Ashukha et al., 2020, Ma et al., 2021]. In principle, ensemble methods complement our approach. For example, one can include our estimator in an ensemble, e.g., by averaging its prediction with other methods. Ad-hoc calibration is about training models in new manners aimed to yield better uncertainty estimations. Important techniques in this category include mixup training [Thulasidasan et al., 2019], pre-training [Hendrycks et al., 2019a], label-smoothing [Müller et al., 2019], data augmentation [Ashukha et al., 2020], self-supervised learning [Hendrycks et al., 2019b], Bayesian approximation (MC-dropout) [Gal and Ghahramani, 2016, Gal et al., 2017], Deep Ensemble (DE) [Lakshminarayanan et al., 2017], Snapshot Ensemble [Huang et al., 2017a], Fast Geometric Ensembling (FGE) [Garipov et al., 2018], and SWA-Gaussian (SWAG) [Maddox et al., 2019].

Ad-hoc calibration is perhaps the best approach in public as it tackles the core of model's calibration directly. However, because it offers specific training methods it is of

less use to large and already trained models, and the impact of each work is limited to a specific model type (e.g., DNNs in [Garipov et al., 2018]). In compression, ad-hoc and ensemble methods (and our own method) often work for numerous models.

Our geometric method is largely inspired by the approach of robustness proving in machine learning models. In this field, formal methods are used to prove that specific inputs are robust to small adversarial perturbations. That is, we formally prove that all images in a certain geometric radius around a specific train-set image receive the same classification [Narodytska et al., 2018, Katz et al., 2017, Huang et al., 2017b, Gehr et al., 2018, Ehlers, 2017, Einziger et al., 2019]. These works are not applicable to uncertainty calibration as they can only produce proves in an offline manner, and thus only to training set inputs rather than to the current input. However, the underlying intuition is that inputs that are geometrically similar should be classified the same also appears in our approach. Indeed, our work shows that geometric properties of the inputs can help us quantify the uncertainty in certain inputs, and that in general inputs that are less geometrically separated and are 'on the edge' between multiple classifications are more error prune than points that are highly separated from other classes. Thus our work reinforces the intuition behind applying formal methods to prove robustness and support the intuition that more robust training models would be more dependable.

3 GEOMETRIC CONFIDENCE EVALUATION

This section lays the foundations for a geometric estimation of the model's confidence level on a given instance. Our work assumes that the inputs are normalized. That is, they are fixed-sized images and within the same format. Under such conditions, we can measure the geometric distance between various inputs.

Formally, a model receives a data input, x, and outputs the pair $\langle \mathcal{C}(x), conf(x) \rangle$, where $\mathcal{C}(x)$ is the model's classification of x and $conf(x)$ reflects the probability that the classification is correct. Our current work evaluates $conf(x)$ from a geometric point of view. We estimate the environment around x where points are closer to inputs of certain classifications over the others. In Section 3.1 we define a geometric separation measure, and provide an algorithm to calculate it. We explain that such a computation is too cumbersome for real-time systems, and so we suggest a lightweight approximation in Section 3.2. Finally, Section 3.3 explains how we use the geometric signal to derive $conf(x)$. That is, mapping a real number corresponding to the geometric separation to a number in $[0, 1]$ corresponding to the confidence ratio.

3.1 SEPARATION MEASURE

We look at the displacement of x compared to nearby data inputs within the training set. Intuitively, when x is close to other inputs in $\mathcal{C}(x)$ (i.e., inputs with the same classification as x) and is far from inputs with other classifications, then the model is correct with a high probability, implying that $conf(x)$ should be high. On the other hand, when there are training inputs with a different classification close to x, we estimate that $\mathcal{C}(x)$ is more likely to be incorrect.

Below we provide definitions that allow us to formalize this intuitive account. In what follows, we consider a model $\mathcal{M}$ to consist of a machine learning model (e.g., a gradient boosted tree or a neural network), along with a labeled train set, Tr, used to generate the model. We use an implicit notion of distance, and denote by $d(x, y)$ the distance between inputs x and y, and by $D(x, A)$ the distance between the input x and the set A (i.e., the minimal distance between x and the inputs in A).

Definition 1 (Safe and Dangerous inputs)**.** *Let $\mathcal{M}$ be a model. For an input x in the sample space we define:*

$$F_{\mathcal{M}}(x) := \{x' \in Tr : \mathcal{C}(x') = \mathcal{C}(x)\}.$$

We denote by $\overline{F}_{\mathcal{M}}(x)$ the set $Tr \setminus F_{\mathcal{M}}(x)$.
An input $x \in \mathcal{X}$ is labeled as safe *if it is closer to $F_{\mathcal{M}}(x)$ than to $\overline{F}_{\mathcal{M}}(x)$, and it is labeled as* dangerous *otherwise.*

Definition 2 (Zones)**.** *Let x be a safe (dangerous) point. A* zone *for x, denoted z_x, is such that for any input y, if $d(x, y) < z_x$, then $D(y, F_{\mathcal{M}}(x)) < D(y, \overline{F}_{\mathcal{M}}(x))$ ($D(y, F_{\mathcal{M}}(x)) \geq D(y, \overline{F}_{\mathcal{M}}(x))$). For each x we denote the maximal such zone by $\mathcal{Z}(x)$.*

In other words, a zone of a safe (dangerous) input x, $\mathcal{Z}(x)$, is a radius around x such that all inputs in this ball are closer to an input in $F_{\mathcal{M}}(x)$ ($\overline{F}_{\mathcal{M}}(x)$) than to any input in $\overline{F}_{\mathcal{M}}(x)$ ($F_{\mathcal{M}}(x)$), respectively.

Definition 3 (Separation)**.** *The separation of a data input x with respect to the model $\mathcal{M}$ is $\mathcal{Z}(x)$ when x is a safe input, and $-1 \cdot \mathcal{Z}(x)$ when x is a dangerous input.*

That is, the separation of x encapsulates the maximal zone for x (provided by the absolute value) together with an indication of whether the point is safe or dangerous (provided by the sign). The separation of x depends only on the classification of x by the model and the train set. This is because our definition partitions the inputs in Tr into two sets: one with $\mathcal{C}(x)$, $F_{\mathcal{M}}(x)$, and one with all other classifications, $\overline{F}_{\mathcal{M}}(x)$. These sets vary between models only when they disagree on the classification of x. Note that x's for which the distance from $F_{\mathcal{M}}(x)$ equals the distance from $\overline{F}_{\mathcal{M}}(x)$ are considered dangerous inputs, and their separation measure will be zero.

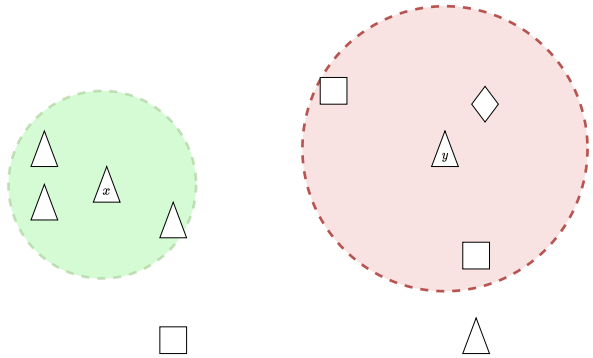


Figure 1: Geometric representation of safe and dangerous inputs, maximal zones, and separation values. The various classifications are illustrated via different shapes.

As mentioned, Definition 2 and Definition 3 use an implicit notion of distance. Such notion can accept any distance metric (e.g., L_1, L_2 or L_∞). However, in this work, we fix the metric to L_2 as it is a standard measure for safety features in adversarial machine learning [Moosavi-Dezfooli et al., 2017], in addition to it being easy to illustrate and intuitive to understand. Furthermore, our methodology relies on calculating the nearest neighbors of a given input, and for L_2, this can be done using standard and well-optimized libraries. Accordingly, all our definitions and calculations assume the L_2 metrics (Euclidean distances).

Figure 1 provides a geometric illustration of safe and danger zones, and separation values. For illustration purposes, the figure uses the L_2 norm with two dimensions, whereas our data usually includes many more dimensions. For example, a 30×30 traffic sign image will have 900 dimensions. In the figure, x is a safe input, and the green highlighted ball represents its maximal zone which reflects how far we can get from x and still be closer to training set inputs classified the same as x than any other inputs. The input y is a dangerous input, and the red highlighted ball represents its maximal zone which dually represents how far we need to distance ourselves from x so that inputs classified as x become closer than other inputs. Thus, x will have a positive separation value, while y will have a negative one.

Next, we provide a formula for calculating the separation of a given input x within the L_2 distance metric.

Definition 4. *Given a model $\mathcal{M}$ and an input x, define:*

$$\overline{\mathcal{S}}^{\mathcal{M}}(x) = \min_{x'' \in \overline{F}_{\mathcal{M}}(x)} \max_{x' \in F_{\mathcal{M}}(x)} \frac{d^2(x, x'') - d^2(x, x')}{2d(x', x'')}$$

Lemma 1. *Let $x, x', x'' \in \mathbb{R}^n$ be inputs such that $d(x, x') < d(x, x'')$. The maximal distance $M(x, x', x'')$ for which if $y \in \mathbb{R}^n$ such that $d(x, y) < M(x, x', x'')$, then $d(y, x') < d(y, x'')$ is*

$$\frac{d^2(x, x'') - d^2(x, x')}{2d(x', x'')}.$$

Proof. Since any three points in space define a plane we focus on the plane defined by these three points.

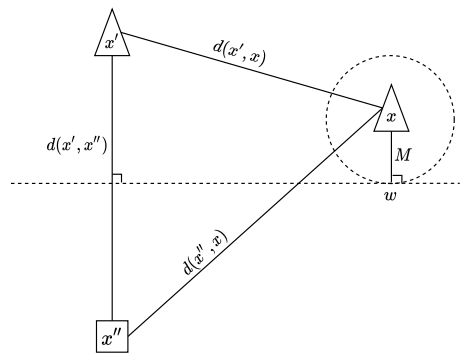


Figure 2: Illustration of the proof of Lemma 1

Figure 2 demonstrates a geometric positioning of the points, and the main constructions in the proof. The perpendicular bisector to the line between x' and x'' divides the plane into two parts: one in which all the points are closer to x'' than to x' (the lower part in the figure) and one in which all the points are closer to x' than to x'' (the upper part in the figure). Our goal is thus to establish the distance between x and the lower part of the plane. Hence, $M(x, x', x'')$ amounts to the distance from x to the perpendicular bisector to the line between x' and x''. Using trigonometric calculations, it is straightforward to verify that indeed

$$M(x, x', x'') = \frac{d^2(x, x'') - d^2(x, x')}{2d(x', x'')}.$$

□

Proposition 1. *$\overline{\mathcal{S}}^{\mathcal{M}}(x)$ is the separation of x with respect to the model $\mathcal{M}$ (in Definition 3).*

Proof. Let x be a safe input, and y be an input such that

$$d(x, y) < \min_{x'' \in \overline{F}_{\mathcal{M}}(x)} \max_{x' \in F_{\mathcal{M}}(x)} \frac{d^2(x, x'') - d^2(x, x')}{2d(x', x'')}.$$

We first show that y is closer to $F_{\mathcal{M}}(x)$ than to $\overline{F}_{\mathcal{M}}(x)$. Let $z'' \in \overline{F}_{\mathcal{M}}(x)$, it suffices to show that there exist $z' \in F_{\mathcal{M}}(x)$ such that $d(y, z') < d(y, z'')$. Notice that

$$d(x, y) < \max_{x' \in F_{\mathcal{M}}(x)} \frac{d^2(x, z'') - d^2(x, x')}{2d(x', z'')}.$$

Therefore, there exist a $z' \in F_{\mathcal{M}}(x)$ for which

$$d(x, y) < \frac{d^2(x, z'') - d^2(x, z')}{2d(z', z'')}$$

Thus, since x is a safe point, using Lemma 1, we conclude that $d(y, z') < d(y, z'')$. The proof follows similar arguments for dangerous points, taking the distances as $-\overline{\mathcal{S}}^{\mathcal{M}}$ and flipping the inequalities.

To show maximality, observe that the intersection point marked by w in Figure 2, which is at distance $\overline{\mathcal{S}}^{\mathcal{M}}(x)$ from x, can be easily shown to be of equal distances from $F_{\mathcal{M}}(x)$ and $\overline{F}_{\mathcal{M}}(x)$. □

While separation provides the maximal zone, it is expensive to calculate. As can be seen in Definition 4, to estimate the separation of one specific input, we go over many triplets of inputs. The exact amount is unbounded and depends on the dataset. Thus, separation is infeasible to compute in near real-time. Therefore, when time or computation resources are limited, we require a different and computationally simpler notion. Accordingly, the following section provides an efficient approximation of the separation measure.

3.2 FAST-SEPARATION APPROXIMATION

We approximate the separation of a given input using only its distance from $F_{\mathcal{M}}(x)$ and its distance from $\overline{F}_{\mathcal{M}}(x)$. This simplification allows us to calculate a zone for any given point, which is not necessarily the maximal one. The reliance on these two distances enables a faster calculation since we do not perform an exhaustive search over many triplets of inputs. In particular, we do not consider the geometric positioning of the inputs that determine the distance from these sets.

Definition 5 (Fast-Separation)**.** *Given a model $\mathcal{M}$, the fast-separation of an input x, denoted $\underline{\mathcal{S}}^{\mathcal{M}}(x)$, is defined as:*

$$\underline{\mathcal{S}}^{\mathcal{M}}(x) = \frac{D(x, \overline{F}_{\mathcal{M}}(x)) - D(x, F_{\mathcal{M}}(x))}{2}$$

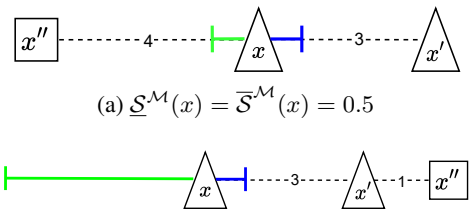
Notice that just as is the case for separation, if x is a safe input, its fast-separation value will be strictly positive and non-positive otherwise.

Figure 3 illustrates the notion of fast-separation. In particular, it exemplifies why it only provides an approximation of the more accurate separation measure. It encapsulates a zone that is less than or equal to that of separation. Sub-figure (a) demonstrates a case in which $\underline{\mathcal{S}}^{\mathcal{M}}(x) = \overline{\mathcal{S}}^{\mathcal{M}}(x)$, while sub-figure (b) presents a case where $\overline{\mathcal{S}}^{\mathcal{M}}(x)$ is considerably larger than $\underline{\mathcal{S}}^{\mathcal{M}}(x)$.

Proposition 2. *$\underline{\mathcal{S}}^{\mathcal{M}}$ is a lower bound of the separation $\overline{\mathcal{S}}^{\mathcal{M}}$ in the sense that for every safe (dangerous) input $0 \leq \underline{\mathcal{S}}^{\mathcal{M}}(x) \leq \overline{\mathcal{S}}^{\mathcal{M}}(x)$ ($\overline{\mathcal{S}}^{\mathcal{M}}(x) \leq \underline{\mathcal{S}}^{\mathcal{M}}(x) \leq 0$).*

Proof. Let x be a safe input. Since Proposition 1 shows that $\overline{\mathcal{S}}^{\mathcal{M}}(x)$ is the *maximal* zone, it suffices to show that $\underline{\mathcal{S}}^{\mathcal{M}}(x)$ is a zone of x. Let y be a point such that

$$d(x, y) < \underline{\mathcal{S}}^{\mathcal{M}} = \frac{D(x, \overline{F}_{\mathcal{M}}(x)) - D(x, F_{\mathcal{M}}(x))}{2}.$$



(a) $\underline{\mathcal{S}}^{\mathcal{M}}(x) = \overline{\mathcal{S}}^{\mathcal{M}}(x) = 0.5$

(b) $0.5 = \underline{\mathcal{S}}^{\mathcal{M}}(x) \neq \overline{\mathcal{S}}^{\mathcal{M}}(x) = 3.5$

Figure 3: Geometric representation of the induced zones of $\underline{\mathcal{S}}^{\mathcal{M}}$ and $\overline{\mathcal{S}}^{\mathcal{M}}$ for different input alignments. $\underline{\mathcal{S}}^{\mathcal{M}}$ is represented by blue arrows and $\overline{\mathcal{S}}^{\mathcal{M}}$ by green arrows.

We show that $D(y, F_{\mathcal{M}}(x)) < D(y, \overline{F}_{\mathcal{M}}(x))$. Take $z' \in F_{\mathcal{M}}(x)$ and $z'', w \in \overline{F}_{\mathcal{M}}(x)$ such that $d(x, z') = D(x, F_{\mathcal{M}}(x))$, $d(x, z'') = D(x, \overline{F}_{\mathcal{M}}(x))$, and $d(y, w) = D(y, \overline{F}_{\mathcal{M}}(x))$. Using the triangle inequality we get:

$$\begin{aligned} D(y, F_{\mathcal{M}}(x)) &\leq d(y, z') \leq d(x, z') + d(x, y) \\ &< d(x, z') + \frac{d(x, z'') - d(x, z')}{2} = \frac{d(x, z'') + d(x, z')}{2} \\ &= d(x, z'') - \frac{d(x, z'') - d(x, z')}{2} < d(x, z'') - d(x, y) \\ &\leq d(x, w) - d(x, y) \leq d(y, w) = D(y, \overline{F}_{\mathcal{M}}(x)) \end{aligned}$$

For dangerous points, the proof follows similar arguments, switching $F_{\mathcal{M}}(x))$ and $\overline{F}_{\mathcal{M}}(x))$. □

Proposition 2 shows that the absolute value of $\underline{\mathcal{S}}^{\mathcal{M}}(x)$ is always smaller than or equal to that of $\overline{\mathcal{S}}^{\mathcal{M}}(x)$ and that they have the same sign. Thus, fast-separation is an approximation of separation in the sense that it uses smaller zones.

3.3 PREDICTING CONFIDENCE FROM SEPARATION

At this point, we showed how to calculate geometric separation and fast-separation for points (the latter can be done efficiently). Next, we take the calculated (fast-)separation value and derive a confidence estimation $conf$. We use a validation set that is disjoint from the train and test sets to calculate for each input $\overline{\mathcal{S}}^{\mathcal{M}}$ or $\underline{\mathcal{S}}^{\mathcal{M}}$ values. We perform a fitting to map $\overline{\mathcal{S}}^{\mathcal{M}}$ or $\underline{\mathcal{S}}^{\mathcal{M}}$ values to confidence probabilities. The fitting is done between $\overline{\mathcal{S}}^{\mathcal{M}}$ (or $\underline{\mathcal{S}}^{\mathcal{M}}$) values and the ratios of correct classifications (on the validation set) for each unique value. E.g., if for $\underline{\mathcal{S}}^{\mathcal{M}}$ value of 10 we see that 90% of the points are classified correctly then we'll add the pair $\langle 10, 0.9 \rangle$ to the fitting function. We expect very low confidence values for highly negative (fast-)separation values, and we expect to approach 100% confidence when the values become positive enough. The regression function we finally get accepts a (fast-)separation value in $\mathbb{R}$ and outputs a scalar in $[0, 1]$ indicating the confidence estimation,

i.e., the predicted success probability for inputs with that (fast-)separation score.

In principle, our method can accept most post-hoc calibration methods to perform the fitting. In this paper, we use isotonic regression as our fitting fuction. Such a function was shown to work best for the tested workloads both for our geometric signal and for the model's original signal, as done in [Pedregosa et al., 2011].

4 EXPERIMENTAL RESULTS

This section provides experimental results following the method described in the previous section. We first introduce the datasets, models, and evaluation criteria and then continue to experimental results.

4.1 METHODOLOGY

4.1.1 Datasets

Our evaluation uses the following standard datasets.

- *Modified National Institute of Standards and Technology database (MNIST)* [LeCun and Cortes, 2010], which consists of hand-written images designed for training various image processing systems. It includes 70,000 28×28 grayscale images belonging to one of ten labels.
- *Fashion MNIST (Fashion)* [Xiao et al., 2017], which is a dataset comprising of 28×28 grayscale images of 70,000 fashion products from 10 categories.
- *German Traffic Signs Recognition Benchmark (GTSRB)* [Houben et al., 2013], which is a large image set of traffic signs devised for the single-image, multi-class classification problem. It consists of 50,000 RGB 30×30 images of traffic signs, belonging to 43 classes.
- *American Sign Language (SignLang)* [Techperson, 2017], which is a database of hand gestures representing a multi-class problem with 24 classes of letters. It consist of 30,000 28×28 grayscale images.
- *Canadian Institute for Advanced Research (CIFAR10)* [A.Krizhevsky et al., 2009], which is a dataset containing 32x32 RGB images of 60,000 objects from 10 classes.

For each dataset, we randomly partitioned the data into three subsets: train set Tr (60%), validation set Vs (20%) and test set Ts (20%). The train set is used to calculate fast separation and train the model. The validation set is used to evaluate the confidence estimation associated with each fast-separation value. These values, in turn, are used to fit a Sigmoid function. Finally, the test set is used to evaluate the confidence on new inputs that were *not* present in the train and validation sets.

4.1.2 Models

In our evaluation, we use the following popular machine learning models: Random Forest (RF) [Breiman, 2001], Gradient Boosting Decision Trees (GBDT) [Mason et al., 1999], and Convolutional Neural Network (CNN) [Gu et al., 2018]. We chose these models because they are different: RF and GBDT are tree-based, while CNN is a neural network. For RF and GBDT, we configured the meta parameters (e.g., the maximal depth of trees) by cross-validation on the train set. For CNN, we used the configuration suggested by practitioners. Our specific configurations are detailed in `https://tinyurl.com/ybx87rtw`.

4.1.3 Evaluation Criteria

To evaluate our method, we compare our fast-separation-based confidence estimation to: (a) the built-in confidence in the Sklearn library, (b) the scaling-binning calibrator [Kumar et al., 2019] which we call SBC, and (c) the histogram binning calibrator [Gupta and Ramdas, 2021a] which we call HB. Each method received the same baseline model as an input yielding a slightly different calibrated model. Note that our method is evaluated against the uncalibrated model as our method does not affect the model. Moreover, it allows us to compare our method against different calibration methods, as shown in Table 1.

To evaluate the confidence predictions, we use the *Expected Calibration Error (ECE)*, which is a standard method to evaluate confidence calibration of a model [Xing et al., 2020, Krishnan and Tickoo, 2020]. Concretely, the predictions sample of size n are partitioned into M equally spaced bins $(B_m)_{m \leq M}$, and ECE measures the difference between the sample accuracy in the mth bin and the the average confidence in it [Naeini et al., 2015]. Formally, ECE is calculated by the following formula:

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

where:

- $\text{acc}\left(B_m\right) = \frac{1}{|B_m|} \cdot |\{x \in B_m : \mathcal{C}(x) \text{ is correct}\}|$, and
- $\text{conf}\left(B_m\right) = \frac{1}{|B_m|} \sum_{x \in B_m} conf(x)$.

4.2 FITTING FUNCTION

Post-hoc calibration methods based on fitting functions typically use either a logistic (Sigmoid) or an isotonic regression [Zadrozny and Elkan, 2002]. Isotonic regression fits a non-decreasing free-form line to a sequence of observations. In comparison, Sigmoid is a continuous step function. We used both fitting functions on our (fast-)separation values and obtained obtained similar accuracy. We opt here to

Table 1: ECE measures with 95% confidence intervals when varying the calibration method, model and dataset.

Dataset	Model	$\underline{\mathcal{S}}^{\mathcal{M}}$	$\overline{\mathcal{S}}^{\mathcal{M}}$	Sklearn (%Imp)	SBC (%Imp)	HB (%Imp)
MNIST	CNN	0.0029±0.0008	0.0031±0.0007	0.0032±0.0004 (9.38%)	0.0190±0.0018 (84.74%)	0.0046±0.0005 (36.96%)
	RF	0.0044±0.0005	0.0045±0.0006	0.0098±0.0012 (55.1%)	0.0331±0.0017 (86.71%)	0.0080±0.0009 (45%)
	GB	0.0040±0.0007	0.0040±0.0008	0.0176±0.0014 (77.27%)	0.0384±0.0025 (89.58%)	0.0074±0.0012 (45.95%)
GTSRB	CNN	0.0014±0.0005	0.0015±0.0003	0.0018±0.0004 (22.22%)	0.1955±0.0072 (99.28%)	0.0021±0.0005 (33.33%)
	RF	0.0042±0.0005	0.0042±0.0007	0.0262±0.0014 (83.97%)	0.0788±0.0047 (94.67%)	0.0065±0.0013 (35.38%)
	GB	0.0071±0.0010	0.0075±0.0013	0.1002±0.0029 (92.91%)	0.1733±0.0045 (95.9%)	0.0175±0.0013 (59.43%)
SignLang	CNN	0.0001±0.0001	0.0001±0.0001	0.0004±0.0002 (75%)	0.1287±0.0151 (99.92%)	0.0008±0.0004 (87.5%)
	RF	0.0008±0.0002	0.0009±0.0003	0.0049±0.0006 (83.67%)	0.0728±0.0079 (98.9%)	0.0026±0.0008 (69.23%)
	GB	0.0008±0.0003	0.0008±0.0002	0.0406±0.0018 (98.03%)	0.2564±0.0207 (99.69%)	0.0065±0.0009 (87.69%)
Fashion	CNN	0.0090±0.0007	0.0087±0.0010	0.0091±0.0013 (1.1%)	0.0266±0.0021 (66.17%)	0.0141±0.0018 (36.17%)
	RF	0.0091±0.0014	0.0101±0.0016	0.0111±0.0011 (18.02%)	0.0506±0.0022 (82.02%)	0.0170±0.0019 (46.47%)
	GB	0.0092±0.0014	0.0093±0.0006	0.0384±0.0019 (76.04%)	0.0667±0.0027 (86.21%)	0.0160±0.0021 (42.5%)
CIFAR10	CNN	0.0110±0.0026	0.0102±0.0020	0.0155±0.0028 (29.03%)	0.0559±0.0017 (80.32%)	0.0188±0.0021 (41.49%)
	RF	0.0142±0.0020	0.0153±0.0017	0.0342±0.0025 (58.48%)	0.1437±0.0020 (90.12%)	0.0370±0.0029 (61.62%)
	GB	0.0155±0.0023	0.0153±0.0016	0.0769±0.0020 (79.84%)	0.2036±0.0028 (92.39%)	0.0404±0.0015 (61.63%)

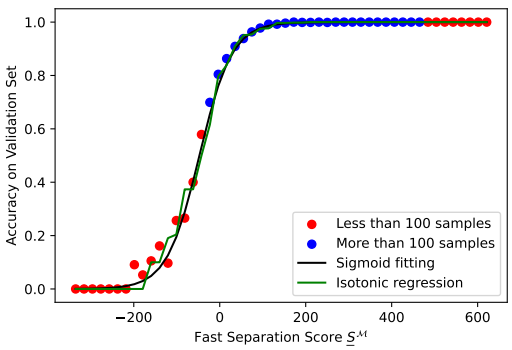


Figure 4: An illustration of the inputs to the fitting function (blue and red dots), and the functions fitted by Sigmoid (black line) and isotonic regression (green line). The inputs are for the MNIST dataset, and the Random Forest model.

present the isotonic regression as it provides the best results, as motivated by Figure 4.

Figure 4 illustrates an example of the success ratio of the Random Forest model for MNIST inputs with varying values of $\underline{\mathcal{S}}^{\mathcal{M}}$ scores (similar behavior were observed for the various models and datasets). We clustered inputs with a similar score together (into 50 bins overall) as each classification is correct or not, and we are looking for the average. The black line represents the Sigmoid function and the green line represents the isotonic regression. As can be observed, both regressions are nearly identical on all the points with positive $\underline{\mathcal{S}}^{\mathcal{M}}$ values. We eventually chose isotonic regression because it better fitted the few points with negative $\underline{\mathcal{S}}^{\mathcal{M}}$ values. Interestingly, these points were consistently a poor fit for the Sigmoid regression rendering slightly less accurate on average. Also, observe that the transition is around the value 0, indicating that the distinction of safe and dangerous points is meaningful in confidence evaluation.

4.3 CONFIDENCE EVALUATION

Table 1 presents the main experimental results of our work. The table summarizes ECEs for our method (with bin size 30).

Each entry in the table describes the ECE, the 95% confidence interval, and (in parenthesis) the improvement of our fast-separation-based method over each competitor method. The improvement is calculated using the ratio between the difference between our ECE and the competitor's ECE. In this experiment, we perform ten random splits of the data into train, validation, and test sets for each model and dataset. We then measure the ECE of the confidence estimation for all test set items, average the result and take the 95% confidence intervals.

First, observe that $\underline{\mathcal{S}}^{\mathcal{M}}$ and $\overline{\mathcal{S}}^{\mathcal{M}}$ yield very similar ECEs, and that the differences between them are usually statistically insignificant. Thus, we conclude that $\underline{\mathcal{S}}^{\mathcal{M}}$ is a very good approximation of $\overline{\mathcal{S}}^{\mathcal{M}}$ despite being considerably simpler to compute. The next interesting comparison is between $\underline{\mathcal{S}}^{\mathcal{M}}$ and SKlearn. We use the same fitting function (Isotonic regression) in both cases, but SKlearn performs the calibration on the model's natural uncertainty estimation, and $\underline{\mathcal{S}}^{\mathcal{M}}$ performs the calibration on geometric distances. Thus, the benefit of our approach stems from the geometric signal and not from the chosen fitting function.

Table 2: Number of confidence estimations per second for the $\underline{\mathcal{S}}^{\mathcal{M}}$-based method with 95% confidence intervals.

Dataset	Model	Predictions per sec.
MNIST	CNN	22.73 ±1.90
	RF	22.42 ±0.66
	GBDT	22.91 ±0.46
GTSRB	CNN	25.29 ±0.69
	RF	23.23 ±1.19
	GBDT	21.78 ±2.18
SignLang	CNN	46.61 ±0.18
	RF	45.16 ±2.71
	GBDT	46.84 ±0.40
Fashion	CNN	22.99 ±0.16
	RF	22.85 ±0.03
	GBDT	23.20 ±0.31
CIFAR10	CNN	7.08 ±0.24
	RF	6.76 ±0.27
	GBDT	6.84 ±0.41

Observe that our $\underline{\mathcal{S}}^{\mathcal{M}}$ improves the confidence estimations consistently and across the board when compared to SKlearn, SBC, and HB. Specifically, we derive improvements up to 99% in all tested models, and for all tested datasets. Such results demonstrate the potential of geometric signals to improve the effectiveness of uncertainty estimation.

4.4 REAL-TIME COMPUTATION

Table 2 provides the computational advantage of our method. We used a Macbook Pro with an Intel Core i5 with four processor cores@2.3 GHz and 8GB RAM in this experiment. We measure the throughput of confidence evaluations in predictions per second and the 95% confidence intervals using five trials for each measurement.

Observe that the dominant factor in operation speed is the dataset. These differences are due to variations in training set sizes, where larger training sets result in slower operation. Importantly, our method runs in 23–46 predictions per second in all but the CIFAR-10 datasets. Such performance is within the ballpark for camera-based applications. For reference, a TV is broadcast in 60 frames per second, and most animation films use up to 24 frames per second. Thus our performance is within an applicable scale. CIFAR-10 is considerably larger, and thus our performance on that dataset is a bit slow. While we can also use parallelism to obtain a faster runtime and mitigate this issue, we plan to address larger training sets in future work.

5 CONCLUSION

Our work uses post-hoc calibration techniques but on a geometry-based signal rather than on the model's confidence estimation. We demonstrated the feasibility of our approach in estimating uncertainty for multiple models, and for multiple datasets. Our evaluation shows that our fast-separation method ($\underline{\mathcal{S}}^{\mathcal{M}}$) outperforms post-hoc calibration methods based on the model's confidence consistently and across the board. Our approach reduces the error in confidence estimations by up to 99% compared to alternative methods (depending on the specific dataset and model).

In addition, we showed that for moderately-sized standard datasets our method achieves near-real time operation. As suggested by our analysis and indicated by the experimental results, the complexity of calculating $\underline{\mathcal{S}}^{\mathcal{M}}$ depends on the training-set size which implies that very large datasets would be slower, and not run in near real-time. In the future, we will develop ways to control the dataset's effect on the runtime by calculating confidence estimations only on a subset of the dataset. E.g., by pre-processing the data and removing data inputs that are geometrically close to each other. Doing so would increase the applicability of our method on larger datasets.

In the near future, once anonymity restrictions are lifted, we will make all the code used in this work as an open-source library and pursue the adaptation of our technique into standard machine learning libraries. In the meanwhile, our code is available at anonymous-Git.

REFERENCES

A.Krizhevsky, V.Nair, and G.Hinton. Cifar-10 (canadian institute for advanced research). 2009. URL http://www.cs.toronto.edu/~kriz/cifar.html.

A. Ashukha, A. Lyzhov, D. Molchanov, and D. Vetrov. Pitfalls of In-Domain Uncertainty Estimation and Ensembling in Deep Learning. In *International Conference on Learning Representations*, 2020.

B. Biggio, I. Corona, B. Nelson, B.I.P. Rubinstein, D. Maiorca, G. Fumera, G. Giacinto, and F. Roli. *Security Evaluation of Support Vector Machines in Adversarial Environments*, pages 105–153. 2014a.

B. Biggio, G. Fumera, and F. Roli. Security Evaluation of Pattern Classifiers under Attack. *IEEE Transactions on Knowledge and Data Engineering*, 26(4):984–996, 2014b.

L. Breiman. Random Forests. *Machine Learning*, 45(1): 5–32, 2001.

R. Ehlers. Formal Verification of Piece-Wise Linear Feed-Forward Neural Networks. In D. D'Souza and K. N.

Kumar, editors, *Automated Technology for Verification and Analysis*, volume 10482 of *Lecture Notes in Computer Science*, pages 269–286. Springer, 2017.

G. Einziger, M. Goldstein, Y. Sa'ar, and I. Segall. Verifying Robustness of Gradient Boosted Models. In *The Thirty-Third Conference on Artificial Intelligence,*, pages 2446–2453. AAAI, 2019.

Y. Gal and Z. Ghahramani. Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning. In *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 1050–1059, 2016.

Y. Gal, J. Hron, and A. Kendall. Concrete Dropout. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

T. Garipov, P. Izmailov, D. Podoprikhin, D. P. Vetrov, and A. G. Wilson. Loss Surfaces, Mode Connectivity, and Fast Ensembling of DNNs. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018*, pages 8803–8812, 2018.

T. Gehr, M. Mirman, D. Drachsler-Cohen, P. Tsankov, S. Chaudhuri, and M.T. Vechev. Ai2: Safety and robustness certification of neural networks with abstract interpretation. pages 3–18, 05 2018. doi: 10.1109/SP.2018.00058.

J. Gu, Z. Wang, J. Kuen, L. Ma, A. Shahroudy, B. Shuai, T. Liu, X. Wang, G. Wang, J. Cai, and T. Chen. Recent Advances in Convolutional Neural Networks. *Pattern Recognition*, 77:354–377, 2018.

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 of *Proceedings of Machine Learning Research*, pages 1321–1330, 2017a.

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*, ICML, page 1321–1330, 2017b.

C. Gupta and A. Ramdas. Distribution-free Calibration Guarantees for Histogram Binning without Sample Splitting. In *International Conference on Machine Learning*, pages 3942–3952. PMLR, 2021a.

Chirag Gupta and Aaditya Ramdas. Distribution-free calibration guarantees for histogram binning without sample splitting. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 3942–3952. PMLR, 2021b. URL http://proceedings.mlr.press/v139/gupta21b.html.

D. Hendrycks, K. Lee, and M. Mazeika. Using Pre-Training Can Improve Model Robustness and Uncertainty. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 2712–2721, 2019a.

D. Hendrycks, M. Mazeika, S. Kadavath, and D. Song. Using Self-Supervised Learning Can Improve Model Robustness and Uncertainty. In *Advances in Neural Information Processing Systems*, volume 32, 2019b.

S. Houben, J. Stallkamp, J. Salmen, M. Schlipsing, and C. Igel. Detection of Traffic Signs in Real-World Images: The German Traffic Sign Detection Benchmark. *The 2013 International Joint Conference on Neural Networks (IJCNN)*, (1288):1–8, 2013.

G. Huang, Y. Li, G. Pleiss, Z. Liu, J. E. Hopcroft, and K. Q. Weinberger. Snapshot ensembles: Train 1, get M for free. In *5th International Conference on Learning Representations, ICLR*, 2017a.

X. Huang, M. Kwiatkowska, S. Wang, and M. Wu. Safety Verification of Deep Neural Networks. In *Computer Aided Verification*, pages 3–29, 2017b.

P. Johnson. Houston, we have a bug: 9 famous software glitches in space. https://www.arnnet.com.au/slideshow/581015/pictures-houston-we-bug-9-famous-software-glitches-space/, 2018.

J.Wenger, H.Kjellstrm, and R.Triebel. Non-parametric calibration for classification. *CoRR*, abs/1906.04933, 2019.

G. Katz, C.W. Barrett, D.L. Dill, K. Julian, and M.J. Kochenderfer. Reluplex: An Efficient SMT Solver for Verifying Deep Neural Networks. In *Computer Aided Verification - 29th International Conference, CAV*, pages 97–117, 2017.

R. Krishnan and O. Tickoo. Improving Model Calibration with Accuracy versus Uncertainty Optimization. 33: 18237–18248, 2020.

M. Kull, M. Perello Nieto, M. Kängsepp, T. Silva Filho, H. Song, and P. Flach. Beyond Temperature Scaling: Obtaining Well-calibrated Multi-class Probabilities with Dirichlet Calibration. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

A. Kumar, P. Liang, and Te. Ma. Verified Uncertainty Calibration. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 32, 2019.

B. Lakshminarayanan, A. Pritzel, and C. Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 6405–6416, 2017.

Y. LeCun and C. Cortes. MNIST handwritten digit database. 2010. http://yann.lecun.com/exdb/mnist/.

C. Leistner, A. Saffari, P. M. Roth, and H. Bischof. On Robustness of On-line Boosting - a Competitive Study. In *12th International Conference on Computer Vision Workshops, ICCV*, pages 1362–1369, 2009.

C. Ma, Z. Huang, J. Xian, M. Gao, and J. Xu. Improving Uncertainty Calibration of Deep Neural Networks via Truth Discovery and Geometric Optimization. In *Proceedings of the Thirty-Seventh Conference on Uncertainty in Artificial Intelligence*, volume 161 of *Proceedings of Machine Learning Research*, pages 75–85, 2021.

W. J. Maddox, P. Izmailov, T. Garipov, D. P. Vetrov, and A. G. Wilson. A Simple Baseline for Bayesian Uncertainty in Deep Learning. In *Advances in Neural Information Processing Systems*, volume 32, page 12, 2019.

L. Mason, J. Baxter, P. Bartlett, and M. Frean. Boosting Algorithms As Gradient Descent. In *Proceedings of the 12th International Conference on Neural Information Processing Systems*, NIPS, pages 512–518, 1999.

S. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, and P. Frossard. Universal Adversarial Perturbations. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 86–94, 2017.

R. Müller, S. Kornblith, and G. E Hinton. When does label smoothing help? In *Advances in Neural Information Processing Systems*, volume 32, pages 4696–4705, 2019.

M.P. Naeini, G.F. Cooper, and M. Hauskrecht. Obtaining Well Calibrated Probabilities Using Bayesian Binning. In *AAAI*, page 2901–2907, 2015.

N. Narodytska, S.P. Kasiviswanathan, L. Ryzhyk, M. Sagiv, and T. Walsh. Verifying Properties of Binarized Deep Neural Networks. In *32nd AAAI Conference on Artificial Intelligence, AAAI 2018*, pages 6615–6624, 2018.

A. Niculescu-Mizil and R. Caruana. Predicting Good Probabilities with Supervised Learning. In *Proceedings of the 22nd International Conference on Machine Learning*, ICML, page 625–632, 2005.

A. Niculescu-Mizil and R. Caruana. An Empirical Comparison of Supervised Learning Algorithms. In *Proceedings of the 23rd International Conference on Machine Learning*, ICML, pages 161–168, 2006.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

Y. Sun, S. Todorovic, and J. Li. Increasing the Robustness of Boosting Algorithms within the Linear-programming Framework. *The Journal of VLSI Signal Processing Systems for Signal, Image, and Video Technology*, 48(1): 5–20, 2007.

Techperson. Sign Language MNIST. 2017. https://www.kaggle.com/datamunge/sign-language-mnist/.

S. Thulasidasan, G. Chennupati, J. A. Bilmes, T. Bhattacharya, and S. Michalak. On Mixup Training: Improved Calibration and Predictive Uncertainty for Deep Neural Networks. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems, NeurIPS*, pages 13888–13899, 2019.

H. Xiao, K. Rasul, and R. Vollgraf. Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms. *CoRR*, abs/1708.07747, 2017.

C. Xing, S. Arik, Z. Zhang, and T. Pfister. Distance-Based Learning from Errors for Confidence Calibration. In *International Conference on Learning Representations (ICLR)*, 2020.

B. Zadrozny and C. Elkan. Transforming Classifier Scores into Accurate Multiclass Probability Estimates. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, page 694–699, 08 2002.

Z.Ding, X.Han, P.Liu, and M.Niethammer. Local temperature scaling for probability calibration. *CoRR*, abs/2008.05105, 2020.

J. Zhang, B. Kailkhura, and T. YJ Han. Mix-n-Match : Ensemble and Compositional Methods for Uncertainty Calibration in Deep Learning. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 11117–11128, 2020.

Y. Zhang and A. Haghani. A Gradient Boosting Method to Improve Travel Time Prediction. *Transportation Research Part C: Emerging Technologies*, 58:308–324, 2015.

Y. Zhou, M. Kantarcioglu, B. Thuraisingham, and B. Xi. Adversarial Support Vector Machine Learning. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD, pages 1059–1067, 2012.