



Article

Classification Confidence in Exploratory Learning: A User's Guide

Peter Salamon ^{1,*} , David Salamon ¹, V. Adrian Cantu ², Michelle An ³, Tyler Perry ², Robert A. Edwards ⁴ and Anca M. Segall ⁵

¹ Department of Mathematics, San Diego State University, San Diego, CA 92182, USA; dls@lithp.org

² Computational Science Research Center, San Diego State University, San Diego, CA 92182, USA; garbanyo@gmail.com (V.A.C.); tylerjperry16@gmail.com (T.P.)

³ Bioinformatics and Medical Informatics Program, San Diego State University, San Diego, CA 92182, USA; michellean92@gmail.com

⁴ Flinders Accelerator for Microbiome Exploration, Flinders University, Flinders, Adelaide, SA 5001, Australia; robert.edwards@flinders.edu.au

⁵ Department of Biology, San Diego State University, San Diego, CA 92182, USA; asega@sdsu.edu

* Correspondence: psalamon@sdsu.edu

Abstract: This paper investigates the post-hoc calibration of confidence for “exploratory” machine learning classification problems. The difficulty in these problems stems from the continuing desire to push the boundaries of which categories have enough examples to generalize from when curating datasets, and confusion regarding the validity of those categories. We argue that for such problems the “one-versus-all” approach (top-label calibration) must be used rather than the “calibrate-the-full-response-matrix” approach advocated elsewhere in the literature. We introduce and test four new algorithms designed to handle the idiosyncrasies of category-specific confidence estimation using only the test set and the final model. Chief among these methods is the use of kernel density ratios for confidence calibration including a novel algorithm for choosing the bandwidth. We test our claims and explore the limits of calibration on a bioinformatics application (PhANNs) as well as the classic MNIST benchmark. Finally, our analysis argues that post-hoc calibration should always be performed, may be performed using only the test dataset, and should be sanity-checked visually.

Keywords: confidence calibration; top-label confidence calibration; bioinformatics; machine learning; exploratory machine learning



Citation: Salamon, P.; Salamon, D.; Cantu, V.A.; An, M.; Perry, T.; Edwards, R.A.; Segall, A.M. Classification Confidence in Exploratory Learning: A User's Guide. *Mach. Learn. Knowl. Extr.* **2023**, *5*, 803–829. <https://doi.org/10.3390/make5030043>

Academic Editor: Andreas Holzinger

Received: 17 May 2023

Revised: 16 July 2023

Accepted: 17 July 2023

Published: 21 July 2023



Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

The pressing need for accurate estimators of confidence in classification by machine learning systems has recently been discussed by many authors [1–6]. This was mostly in response to Lakshminarayanan et al.'s 2017 NIPS address calling for more attention to the calibration problem [3,5]. A motivating claim here is that modern training methods have focused on improving accuracy of predictions, often at the cost of calibration [5–9]. Miscalibrated predictions can lead to misallocated resources and catastrophic errors. While the need is particularly acute for medical or military decisions, good calibration is important for all applications.

In the present manuscript we focus on the post-hoc calibration of the confidence associated with a particular choice by a classifier, where by confidence we mean the probability that the chosen class is correct, and by calibration we mean the rule that associates such probability to the output of the classifier. The motivation for our focus comes from bioinformatic classification problems, but applies to any area in which the class-specific information captured in the dataset differs wildly between the classes. Our approach is predicated on the conviction, noted in PhANNs [10], that confidence calibration for classification in such problems should be individualized for each class, i.e., confidence assessment should be done separately for each category.

Poorly defined categories with limited examples are typical of “cutting edge” research that deals with exploratory problems. In such fields, machine learning advances the frontiers of knowledge by helping us identify which categories can be objectively identified in the data, for example classes of proteins with distinct functions. Datasets for different protein families do not grow at equal rates, for example, due to changing research priorities, and some functional protein families are less homogeneous than others. In these domains, classifying the shifting and subdividing categories is a daunting task.

2. Specific Applications to Bioinformatics and Biological Datasets

Multi-class prediction problems in bioinformatics rely on datasets that are parsed and curated from rapidly changing databases using search terms—for example, annotations that may be insufficiently documented or protein functional classes that are often not well delineated. The available datasets for the different classes and the associated reliability of the classifications vary widely. Classifying a protein sequence as belonging to one functional category or another can be very difficult for some classes and relatively easy for others, because some proteins diverge much more than others without losing their function, and because some functional categories have significantly more annotated examples than others. For example, the sequences of the spike protein of influenza or corona viruses frequently diverge to a greater extent from each other compared to the RNA polymerases responsible for replicating the genomes of these viruses. The spike proteins are under greater pressure to diversify in order to escape detection and destruction by the immune system, whereas the RNA polymerases are under greater pressure to conserve their ability to copy the RNA genomes of the viruses. The need to report class-specific performance measures has been clear, at least in bioinformatics applications confronted with highly unbalanced, multi-class problems. Such biological applications depend on databases whose information for some classes far exceeds the information available for other classes [10]. Reporting overall accuracy (or its many relatives, such as F1 scores) is not enough to measure the quality of the prediction in this context. The accuracy prediction must be class-specific. So must confidence prediction.

In the next section, we present our (somewhat unique) view of the calibration problem. To illustrate our arguments, we use PhANNs [11], a bioinformatics classifier for bacteriophage proteins that predicts a score for inclusion of a given protein sequence into one of 11 distinct functional classes [10]. The score identifying the most likely predicted class is accompanied by a confidence score that estimates the probability that the predicted category is correct. The PhANNs example provides a wide spectrum of categorical behaviors and illustrates our point that different classes need separate consideration for the calibration of confidence. The second example is the classic MNIST dataset of handwritten digits [12]. It was chosen and trained to provide a comparable example that had two independent test sets and could be interrogated by a statistically significant sample of ensembles of ten networks, all trained with 30,000 examples. The Appendix A gives further details about the training and results for these networks.

Both examples use an ensemble of ten neural networks. The PhANNs scores come from the sum of the strengths of the separately trained networks’ softmax outputs. This results in PhANNs scores being on a 0–10 scale rather than the more customary 0–1 scale. The temptation to divide by ten was resisted on the grounds that the score would be mistaken for a calibrated confidence, which for some categories it certainly was not. For the MNIST scores we used the more usual 0–1 scale.

3. Confidence in Category Prediction

Confidence is a measure of our trust in the category predicted by the machine learning system (MLS). For virtually every author on the subject, this measure is the probability that the prediction is correct. The post-hoc calibration task is to estimate this probability given the post-training MLS and a test set. What information to use in estimating this probability is a matter of some debate. In particular, what information from the MLS is

assumed to be known in calculating the conditional probability is (surprisingly) an issue. In a recent paper, Gupta and Ramdas [13] argue that the category predicted by the MLS should always be part of the given information when associating a confidence to the score. We wholeheartedly agree with their arguments, and have argued for the same category-specific approach (top-label calibration) based on independent reasoning having to do with poorly and heterogeneously defined categories leading to noisy and unbalanced datasets.

Figure 1 shows the joint distribution of true category and predicted category in the PhANNs test set of 46,801 samples. The Figure also shows two normalizations of the joint distribution into conditional distributions, normalized either by row or by column, i.e., conditioned on the true category versus conditioned on the predicted category. The columns of the table on the bottom right give the confidence at this level of information, i.e., conditioned only on the category predicted. In most cases, the diagonal entry in the Reverse Confusion Matrix, i.e., the confidence, is worse, sometimes much worse, than the corresponding diagonal entry in the Confusion Matrix, i.e., the true positive rate. A particularly egregious example is the confidence in the prediction of Minor Capsid; the conditional probability that a sample identified as a Minor Capsid is indeed a Minor Capsid, is 0.07.

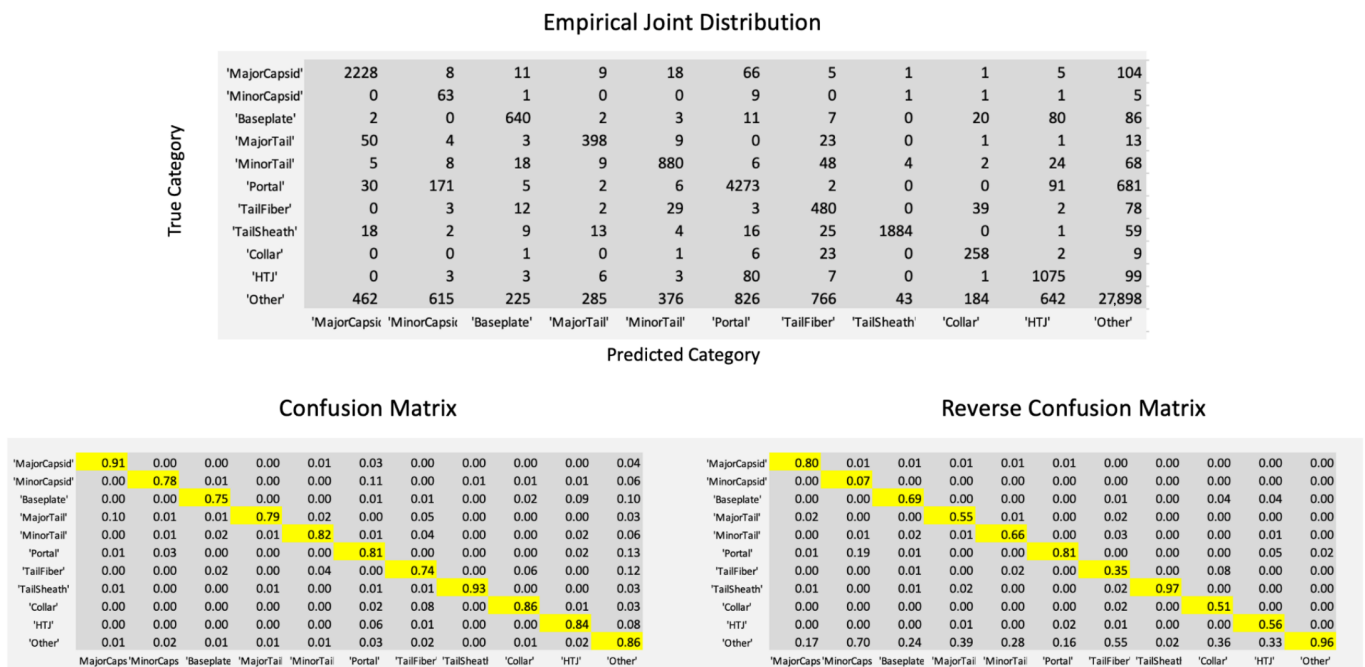


Figure 1. The empirical joint distribution of true and predicted categories in PhANNs. The lower left panel shows the data normalized by row (each entry divided by the sum of the entries in its row). The resulting entries represent empirical estimates of the probabilities of each predicted category given the true category, with diagonals highlighted in yellow. The lower right panel is created similarly, this time normalizing by columns. The resulting entries represent empirical estimates of the probabilities of each true category given the predicted category. The diagonal entries of the Reverse Confusion Matrix, again highlighted in yellow, are the confidence Conf1 in Equation (1).

Figure 2 shows the three levels of output from an MLS trained to perform classification. Given an input vector X , the MLS always predicts the category $k = k(X)$ assigned to X . It may also output a confidence score $S(X)$ or even a probability distribution $Y(X)$ over the categories. This leads to three possible interpretations of **confidence in the predicted category**, depending on the level of information given in the conditions. The first of these is just

$$\text{Conf1}(X) = \text{Prob}(\text{TrueClass}(X) = k(X) \mid k(X)), \quad (1)$$

the empirical estimates of which appear as the diagonal entries of the reverse confusion matrix in Figure 1.

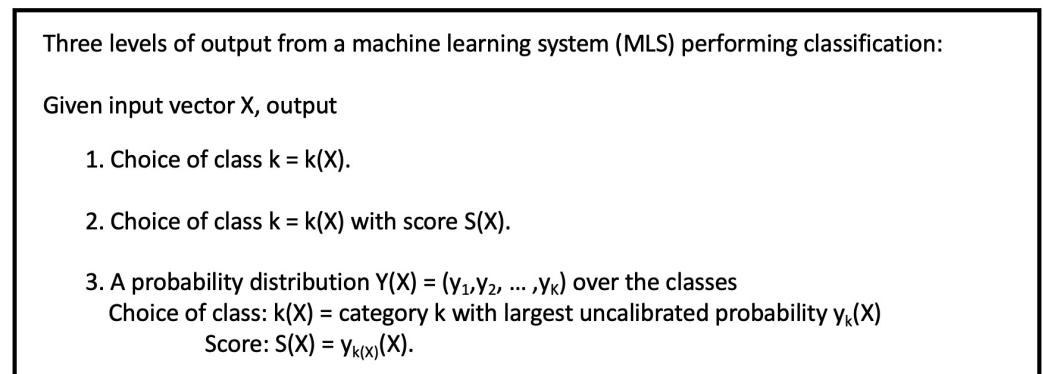


Figure 2. Schematic division of MLS into three groups based on information output from the MLS.

When we have more information in the form of a score function, the corresponding definition of confidence becomes

$$\begin{aligned} \text{Conf}_2(X) &= \text{Prob}(\text{TrueClass}(X) = k(X) \mid k(X), S(X)) \\ &= \text{Conf}_k(S) \end{aligned} \quad (2)$$

This confidence is our primary focus and the one that is the most natural within the top-label paradigm. Calibration at this level is the estimation of the probability density in Equation (2) as a function of score and comes in the form of the K functions $\text{Conf}_k, k = 1, \dots, K$, that convert scores S on samples having $k(x) = k$ to probabilities of being correctly classified. Note that this divides the confidence estimation portion of the multi-classification problem into K separate one-versus-all problems [8,13,14]. We discuss our recommended implementations for finding good estimators of the functions $\text{Conf}_k(S)$ in the following sections.

The score function $S(X)$ above can come from several possible sources. Usually it is the (uncalibrated) confidence that came with the classification, and reflects performance on the training and/or validation set. For random forests, it is typically the fraction of trees voting for this classification. For MLS type 3 (see Figure 2) the k -th component of the output vector $Y(X)$ is the uncalibrated probability that the input X belongs to category k . The entry for the chosen category, $y_{k(X)}(X)$, should thus be the confidence we seek. For post-hoc calibration, this entry in Y can serve as the score $S(X) = y_{k(X)}(X)$, and is worth recalibrating [5,6], especially in light of post-training dataset growth [15].

While the most important entry in the MLS output $Y(X) = (y_1, \dots, y_K)$ for deciding confidence calibration is certainly $y_{k(X)}$, the other entries also carry some information. We explore some calibration schemes using the full K -tuple $Y(X)$ in the discussion of parametric models below. Notably, we introduce some top-label oriented extensions to temperature scaling.

4. Histograms

The earliest confidence calibration methods [14,16,17] are called histogram methods. They work by binning the scores and defining the confidence for a bin using the empirical fraction of samples whose predicted category matches the true category. The procedure is illustrated in Figure 3.

Following the dictates of our top-label approach, we pay attention to only those samples X in the test set for which the predicted category is fixed, $k(X) = k_0$. We call these samples positive, and divide them into two subsets, the true positives

$$\text{TP} = \{X; \text{TrueClass}(X) = k_0, \text{ and } k(X) = k_0\} \quad (3)$$

and the false positives

$$FP = \{X; \text{TrueClass}(X) \neq k_0, \text{ and } k(X) = k_0\} \quad (4)$$

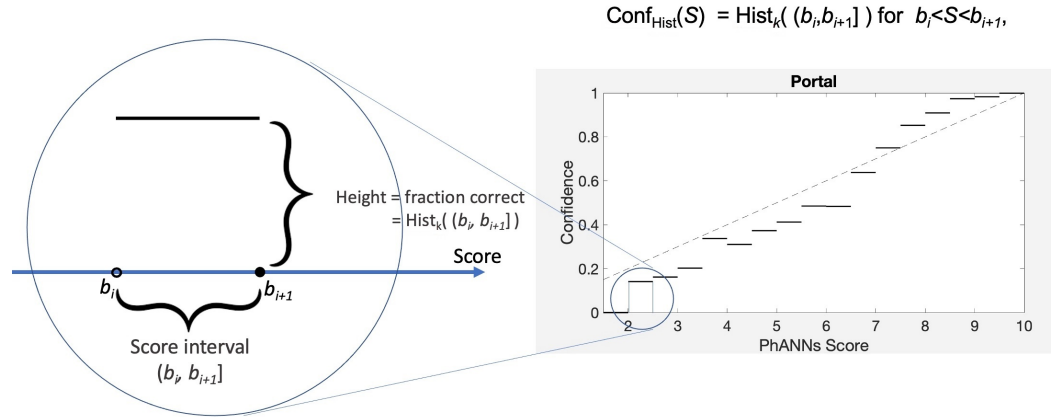


Figure 3. The fraction of true positive samples in a subinterval of the scores range is an estimate of the confidence for that score range. Combining these over a partition of the possible score values creates the graph of $\text{Conf}_{\text{Hist},k}(S)$.

Focusing then on an interval of score values, $a < S \leq b$, we can approximate the true confidence for scores in this interval using the empirical estimate

$$\text{Hist}_{k_0}((a, b]) = \frac{|\text{TP} \cap (a, b]|}{|\text{TP} \cap (a, b]| + |\text{FP} \cap (a, b]|} \quad (5)$$

where $|\{.\}|$ denotes the number of elements in a set. In the formula above, the numerator counts the number of true positive samples with scores in $(a, b]$, while the denominator counts all positive samples with scores in $(a, b]$. The ratio, $\text{Hist}_{k_0}((a, b])$, is an average confidence for scores in the interval $(a, b]$. It is the histogram of “histogram calibration” although it is not a histogram in the usual sense. We can piece together a histogram-based Conf function using the values given by Hist_{k_0} . Formally, consider a partition $B = \{0 = b_1 < b_2 < \dots < b_{n-1} < b_n = 1\}$ of the interval of possible scores. We then define

$$\text{Conf}_{\text{Hist},k}(S) = \text{Hist}_k((b_i, b_{i+1}]) \text{ for } b_i < S < b_{i+1}, \quad (6)$$

by combining histograms on each subinterval as illustrated in Figure 3. We use half-open intervals, so their union covers all values of S exactly once.

Using observed frequencies in the test set to make confidence estimates specific to each bin can spread even a respectably sized test set thin. This is what makes the binning approach rather noisy. But as far as giving a noisy picture of “true confidence”, it is best practice. General bounds on the level of noise from arbitrary distributions have been recently established [18,19]. These bounds are certainly the route to the eventual, high-accuracy confidence values needed for some applications. Gupta and Ramdas’s formalism aimed at much larger datasets than what can be expected to be available for the sort of exploratory problems our PhANNs dataset represents (and which we synthetically emulate with the MNIST dataset). For our target data size range of about 1000 samples from each category, their formalism bounds the histogram error for 10 bins with an expected calibration error under 0.18, ninety percent of the time (see Figure 2 of ref Gupta and Ramdas [19]). For our actual dataset sizes in PhANNs, the noise level is even greater since several of the categories have far fewer than 1000 samples. The MNIST dataset, trained to match PhANNs’ performance and unimpeded by balance issues, can provide an easily

accessible picture of the noise level. Since the MNIST data effectively comes with two independent test sets, we can draw two independent sets of histograms for the same MLS. Shown in Figure 4 are two sets of empirical ConfHist curves for each MNIST category using ten equal size bins. Their differences show the noise level of empirical estimates for the size range of our examples. For MNIST, each test set consisted of 10,000 examples. For PhANNs, there were 46,000 examples in the test set but almost 30,000 of these belonged to the category “Other”.

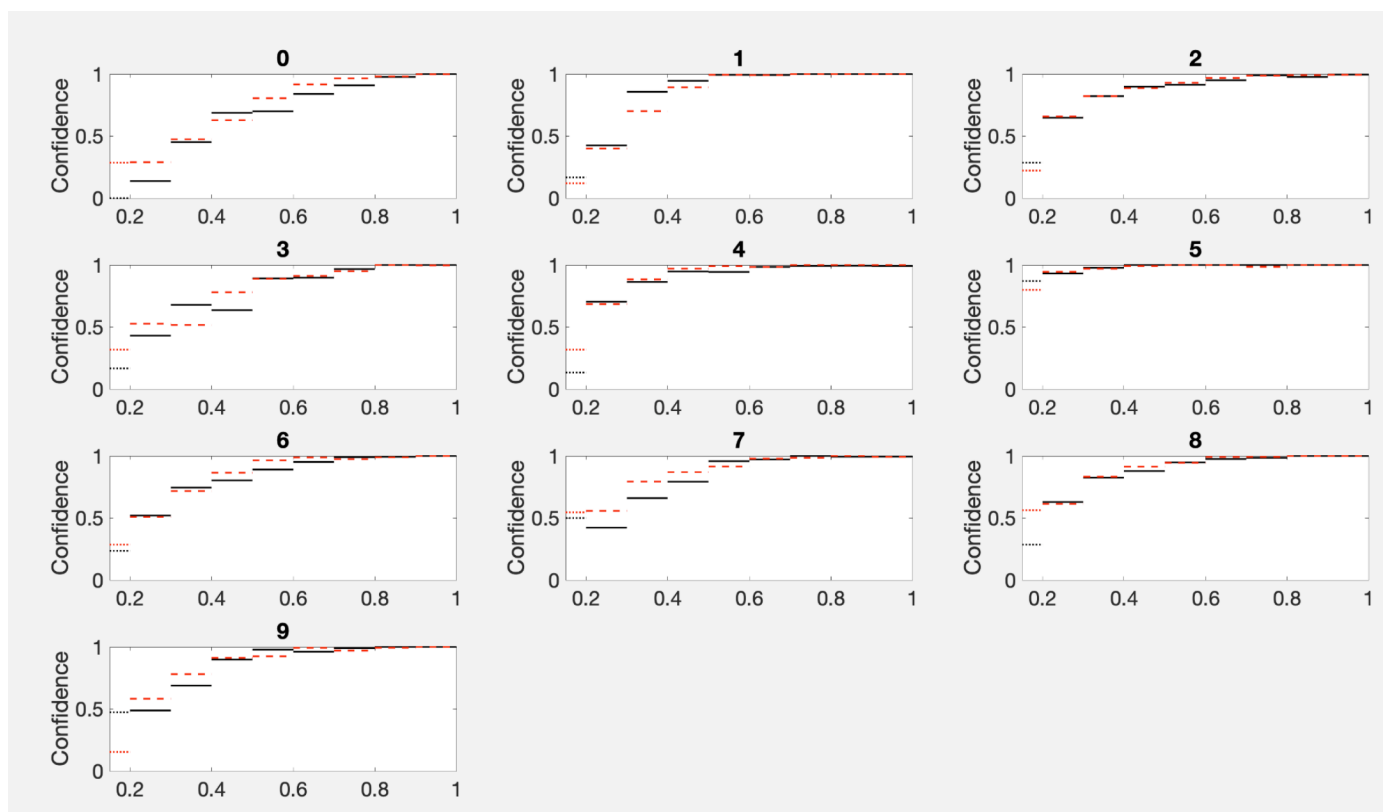


Figure 4. Two sets of histogrammed confidence for the MNIST problem (solid black vs. dashed red) for each category (handwritten digits, 0–9) using the two independent test sets. The differences between red and black histogram heights are entirely due to noise, showing the limitations of the histogram calibration when only 1000 test set examples are available.

Despite the evident noise level shown in Figure 4, histograms usually achieve among the lowest error rates. In fact, the histogram approach to confidence calibration has found new proponents [19] who show that it has impressive performance. As an anchor, we include a plot of $\text{Conf}_{\text{Hist}}(S)$ using 10 equal sized bins for comparison with all our calibration curves.

Histograms are considered a non-parametric method. Our favorite approach, described next, is closely related; it is also non-parametric, but avoids binning.

5. Kernel Densities

Our recommended approach to estimating the correctness probability at a specific score is to use the density of similar scores among true positive samples as a fraction of the density of all positive samples. The two densities are estimated in an approach called kernel density estimation, in which the sample points are fattened to small normal distributions which sum to an approximation of the distribution that gave rise to the sample. One could think of it as viewing our sample “slightly out of focus” so each sample point instead appears as a small blur. The choice of the extent of blurring b (known here as the bandwidth) can have a significant effect; we discuss bandwidth selection in Section 12 below.

Specifically, to find the confidence associated with a prediction $k(X) = k_0$, we again pay attention only to those elements of the test set that had predicted category k_0 . We refer to these samples as positive and recall the definitions of the true positive set, TP, and the false positive set, FP. We then estimate a kernel density for the list of scores in TP

$$\mathcal{K}_{TP}(S) = \frac{1}{|TP|} \sum_{X \in TP} \frac{1}{\sqrt{2\pi}b} \exp\left(-\frac{(S - S(X))^2}{2b^2}\right) \quad (7)$$

and a kernel density for the list of scores in FP

$$\mathcal{K}_{FP}(S) = \frac{1}{|FP|} \sum_{X \in FP} \frac{1}{\sqrt{2\pi}b} \exp\left(-\frac{(S - S(X))^2}{2b^2}\right) \quad (8)$$

We then use these kernels to predict the confidence at any score S as

$$\text{Conf}_{KDE}(S) = \frac{\mathcal{K}_{TP}(S) \cdot |TP|}{\mathcal{K}_{TP}(S) \cdot |TP| + \mathcal{K}_{FP}(S) \cdot |FP|} \quad (9)$$

The procedure is illustrated in Figure 5 for the Baseplate class.

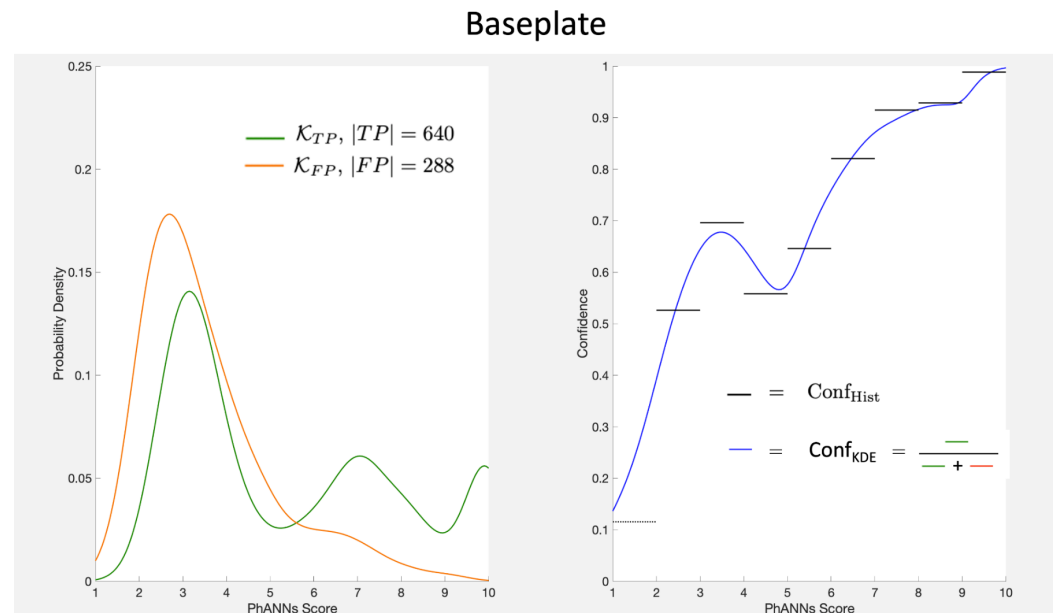


Figure 5. Left Panel: Kernel densities for the probability density of finding a sample with a given score among the TruePositive (green) or the FalsePositive (orange) samples. Right Panel: Confidence from the ratio of kernel density estimates (blue) calculated from Equation (9) compared to the histogram binned confidence (black).

Our approach using the kernel density ratio, Conf_{KDE} in Equation (9), is a smoothed alternative to the histograms. We are not the first to suggest their use. Zhang et al. [5] recommend evaluating the calibration error using kernel density estimators (KDEs) for the unknown distributions to enable stable evaluation of the integrals involved. They even recommend integrated deviation from their KDE estimators as the way to estimate calibration error. What they do not seem to realize is that their KDE estimate is in fact a great calibrator. They also miss a crucially important restriction forced by the category predicted. This is the topic of our next section.

6. Top-Label Confidence

When we say that confidence is the fraction of samples that are correct, we are referring to something like Equation (5). The numerator of that fraction is clear but the denominator could be imagined without the $k(X) = k$ condition. In fact Zhang et al. [5] use kernel denominators that do not impose the predicted category condition.

As we have stressed above, our perspective forces us to focus only on the positive examples, i.e., the ones with $k(X) = k$. There is much information in the predicted category and it should be exploited. Furthermore, it is information that we always have. Thus reporting confidence without using the predicted class is, at best, underutilizing data. The right panel in Figure 6 shows the effect of counting all scores on category k (red curve) compared with counting just the scores of samples that predicted category k (blue curve). As the picture makes clear, it is better to count only with the positive scores. Note that scores above 5 automatically forced the sample to predict category k and so the two ways of counting only diverge for scores below 5.

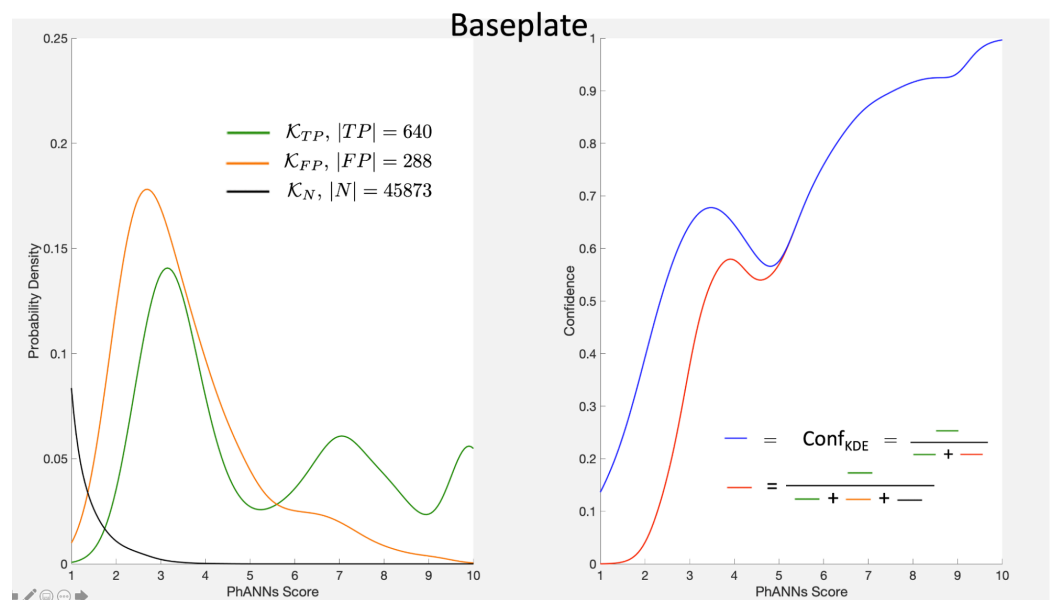


Figure 6. The effect of counting scores that did not predict the category. Left Panel: Kernel densities for the probability density of finding a sample with a given score among the TruePositive (green), the FalsePositive (orange), and the score on this category by negative samples (black). Note the much higher weight attached to the negative samples, since most samples did not predict this category. Right Panel: Confidence from the ratio of kernel density estimates (blue) compared with Confidence that would result if all the negative samples were used in the denominator (red). The lesson is to only use those y_i with $i = k(X)$ rather than using a diluted signal.

This brings us to our last version of the definition of confidence appropriate for the third type of MLS in Figure 2, one whose output $Y(X) = (y_1, y_2, \dots, y_K)$ is in fact a probability distribution over the classes.

$$\text{Conf3}(X) = \text{Prob}\left(\text{TrueClass}(X) = k(X) \mid k(X), Y(X)\right) \quad (10)$$

Note that we have stressed that the predicted category, $k(X)$, be part of the given conditions. The entries in the output probability vector $Y(X)$ are supposed to be the probabilities for each category. Thus $y_{k(X)}$ is the (uncalibrated) estimate of the probability we seek. But is it the value on the red or the blue curve above? **In fact standard training does not take into account the required difference between the red and blue curves in Figure 6 based on slight changes in activations.** This is one important reason a post-hoc calibration is needed even though it might be (and typically is) using the same test

set that was used to measure proficiency. The confidence estimates need to know the predicted categories.

We recommend using the score function $S(X) = y_{k(X)}$ and calibrating with Conf_{KDE} as above. Note, that this does not use any of the entries in Y other than $y_{k(X)}$. We also discuss some parametric alternatives below that use the full Y vector. Either way, recalibration is worthwhile and cannot hurt. While the uncalibrated confidence estimates $y_{k(X)}$ may in fact be good estimates of the confidence, recalibration will show this to be the case. If the data was calibrated to begin with, a final calibration finds a calibration function Conf_k very close to the identity function. Two examples are shown in Figure 7 for the PhANNs categories Minor Tail and Portal. The calibration function $\text{Conf}(S) = S/10$ (the effective identity here since PhANNs scores run 0–10) comes very close in these examples to the results of calibrating using kernel densities or histograms or temperature scaling with awards (see below).

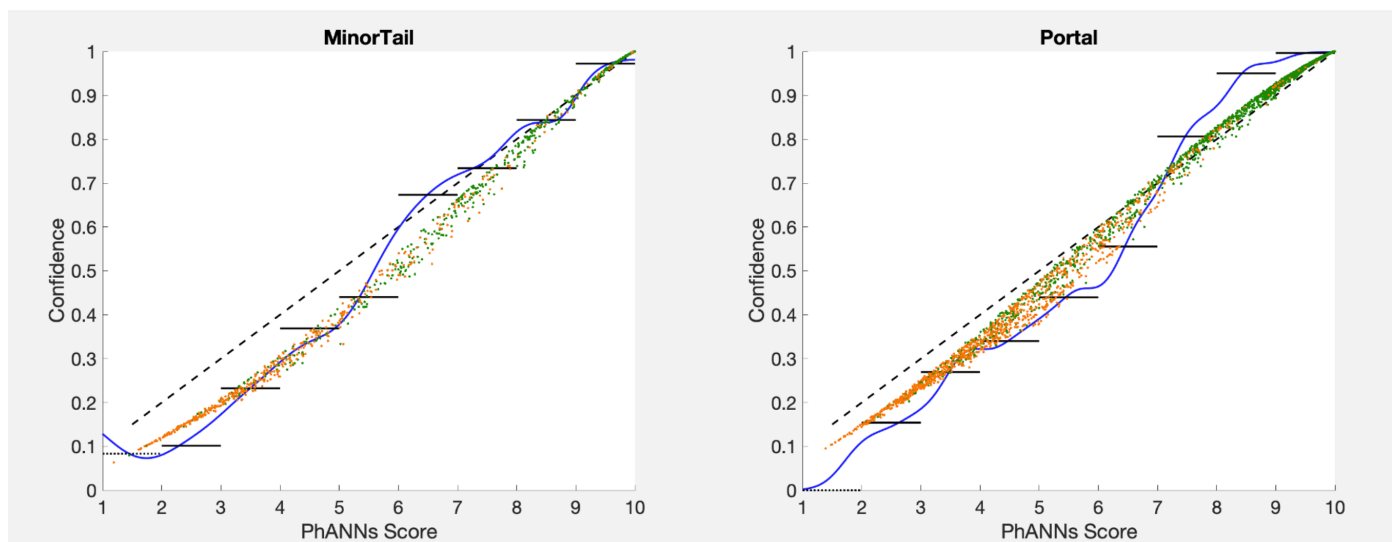


Figure 7. The scoring function PhANNs Score/10 is already very close to calibrated for the categories Minor Tail and Portal. This can be seen from the proximity of the uncalibrated score (dashed line) to our three best confidence estimators: (1) the kernel density estimator Conf_{KDE} (blue), (2) the histogram estimator $\text{Conf}_{\text{Hist}}$ (black), and (3) the cloud of estimates from temperature scaling with awards (green and orange), described in detail in Section 11 (TSwA).

7. The PhANNs Approach Tweaked

The original PhANNs paper assigned confidence values in category k by associating to a sample X with score $S(X)$ the empirical fraction $\text{Hist}_k([S(X), \infty])$. This corresponds to the very liberal strategy of grouping a score S with all scores greater than or equal to S . We shall refer to it as the cumulative confidence Conf_{Cum} .

$$\text{Conf}_{\text{Cum}}(S) = \text{Hist}_k([S, \infty]) \quad (11)$$

Using cumulative distributions to mitigate the noisy empirical estimates of small bins is less sensitive to sampling noise than binning strategies. The approach is also very robust, i.e., it will work on examples where the other methods do not. It does require a careful statement warning the user that the returned value is the probability of a score of S or better, rather than the probability density at S . Here we discuss adapting Conf_{Cum} as an estimator of the probability density.

The cumulative features imply that in general Conf_{Cum} is much slower to come down toward lower estimates for lower scores. This is illustrated in the top left panel of Figure 8 for the category Tail Fiber. Not surprisingly, this confidence estimator badly overestimates

the confidence for low scores. A simple trick however improves this estimator significantly.

$$\text{Conf}_{\text{Cum}+}(S) = \begin{cases} \text{Conf}_{\text{Cum}}(S) & \text{if } S \geq \theta \\ \text{Hist}_k([0, \theta]) & \text{if } S < \theta \end{cases} \quad (12)$$

The confidence function $\text{Conf}_{\text{Cum}+}$ uses Conf_{Cum} from the max score down to score $S = \theta$, below which it uses just one big histogram bin. If we choose θ equal to the median of all scores that go with category choice k , the technique remains non-parametric. The result of doing this is shown in the top right panel of Figure 8. While it makes the method more competitive, it is still far worse than kernel density based estimates or histogram estimates.

With an optimized cutoff θ_{opt} , chosen to minimize negative log likelihood in calibration, the resulting confidence function

$$\text{Conf}_{\text{Cum}*}(S) = \begin{cases} \text{Conf}_{\text{Cum}}(S) & \text{if } S \geq \theta_{\text{opt}} \\ \text{Hist}_k([0, \theta_{\text{opt}}]) & \text{if } S < \theta_{\text{opt}} \end{cases} \quad (13)$$

becomes competitive, but not stellar. Figure 9 shows a noticeably worse fit between the green optimal cutoff curves and the histograms than blue KDE ratio estimator manages. This is further corroborated by the charts comparing the performance of our methods on the PhANNs dataset in Section 14 where we refer to this method using the acronym CwOC, cumulative confidence with optimal cutoff.

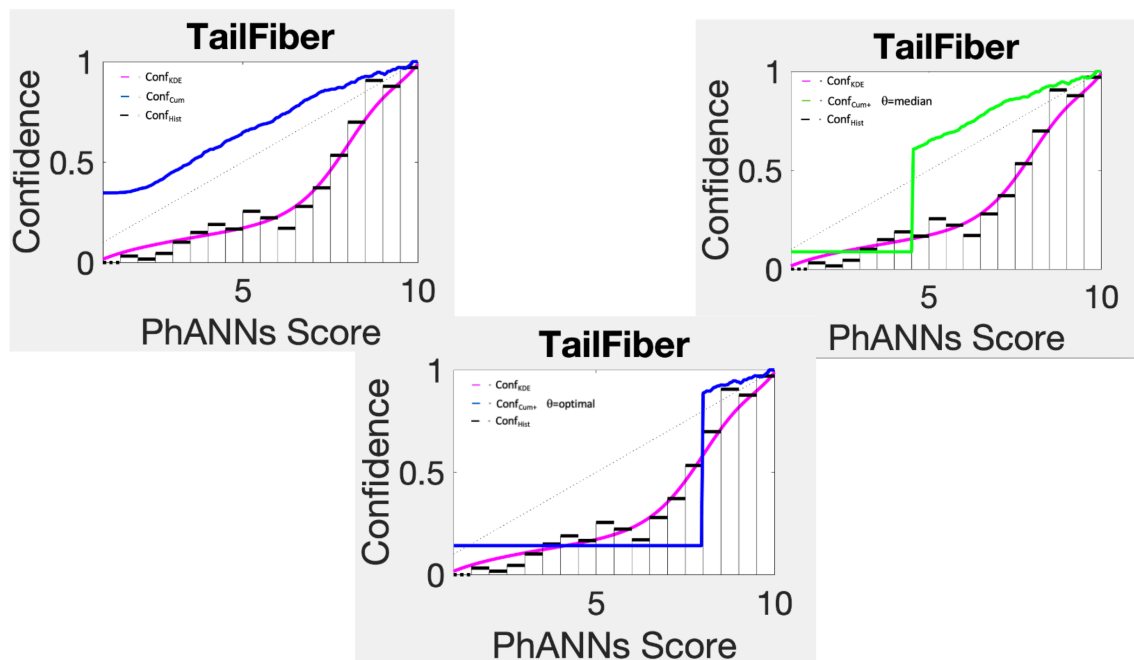


Figure 8. Top left panel shows cumulative confidence Conf_{Cum} compared to KDE and histogram confidence estimates. The other two panels compare $\text{Conf}_{\text{Cum}+}$ using the median as the cutoff (top right panel) and the optimal cutoff (bottom panel).

On the other hand, an optimized cutoff also moves this method into the category of parametric models, where the competition is stiffer. This is the topic of the next several sections.

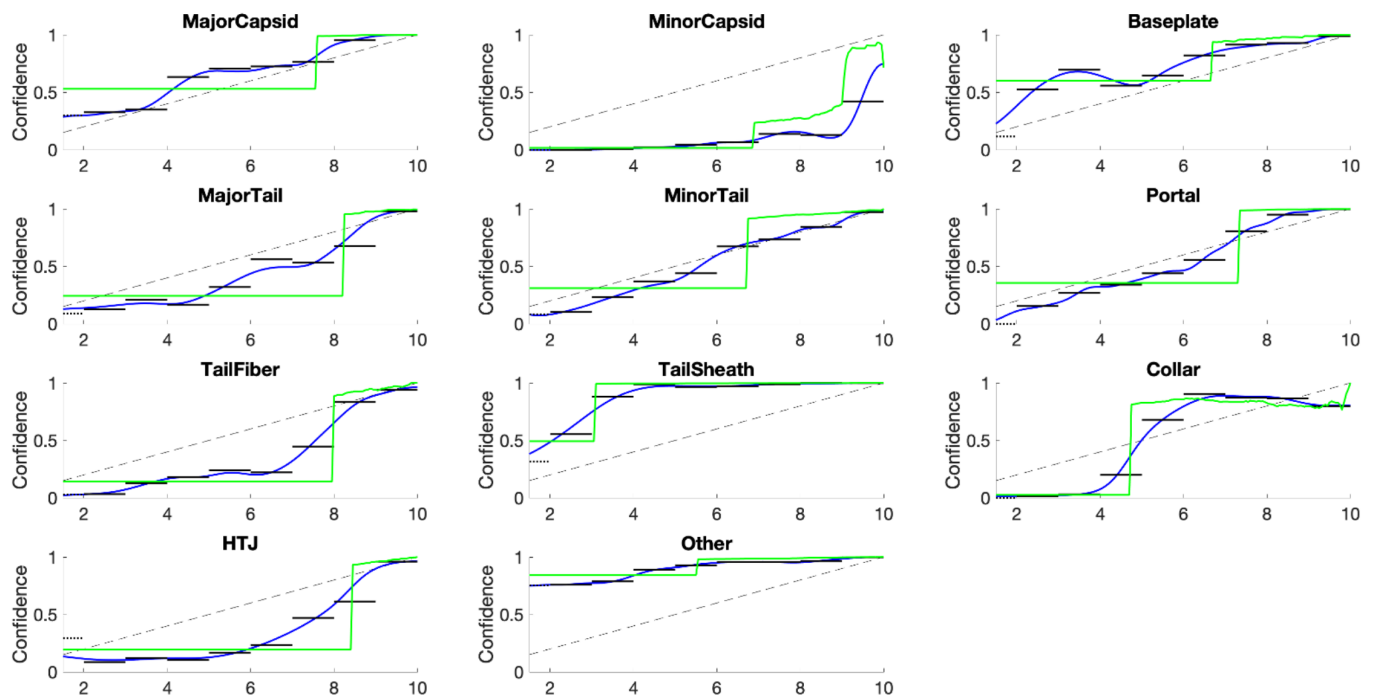


Figure 9. Optimally chopped cumulative confidence, $\text{Conf}_{\text{Cum}*}$ (green) compared to $\text{Conf}_{\text{Hist}}$ (black) and Conf_{KDE} (blue) on PhANNs categories.

8. Parametric Models

Parametric models involve the selection of a parameter based on data. If that data involves the test set, any predictions using the parameter (such as confidence predictions) face a bias-variance tradeoff. There are four possible approaches—all of them used.

The first approach basically views the calibration problem as a second machine learning task that therefore needs a second test set [5,7] and can be attacked by elaborate methods. For the exploratory problems we are considering here, this is unlikely to be a viable approach due to paucity of data.

The second approach is to use the validation set to train these parameters, after the network parameters are fixed. In that case the resulting calibrations are part of the MLS output and should in turn be calibrated post-hoc on the test set by the techniques described above.

A third approach is to use cross-validation on test data. This requires more work but can extract nearly all of the information in the test set with minimal bias. It tends to be quite unstable with sparse data, which is the case for several regions of several categories in the PhANNs dataset.

The fourth approach is to just train on the test set. This is our approach here and is the only truly post-hoc approach in the presence of scarce data. With only a few, sufficiently generic parameters, there is little risk of overtraining. Is it a valid confidence calibrator? Certainly. Can we make an unbiased prediction of how well it will perform on other data? Alas, no. For that, we need a second test set. Using the fitted parameters to estimate the error on the same set that was used for the estimate tends to underestimate such error. Here we used experiments with the MNIST dataset to assess the magnitude of the resulting bias, which could not be distinguished from noise for our models below. This is detailed further in Appendix A.

To train our parameters, we followed the procedure adopted by Ovadia et al. and minimized training error on the test set [15]. Initially, we trained using NLL loss and Brier score. These are proper scoring functions [20] so the results are acceptable, but slightly better results are obtained by using NLL or Brier loss following conversion to the effective

binary problem for the top-label calibration procedure as recommended by [13]. Specifically, we used the following loss

$$\text{Loss}_{\text{NLL}} = - \sum_{X \in \text{TP}} \log(S(X)) - \sum_{X \in \text{FP}} \log(1 - S(X)) \quad (14)$$

which gave results very close to what one obtains using

$$\text{Loss}_{\text{Brier}} = \sum_{X \in \text{TP}} (1 - S(X))^2 + \sum_{X \in \text{FP}} S(X)^2 \quad (15)$$

where TP and FP are the category specific true positive and false positive samples as defined above.

Temperature scaling is a popular parametric technique that estimates one parameter, a temperature T , to (retroactively) adjust the harshness of the softmax used to get Y . Introduced by Guo et al. [6], it has been positively reported on by several research groups [5,7,8,21]. Our interest here is not temperature scaling per se, but rather some tweaks on temperature scaling that work very well for top-label calibration. Nonetheless, we begin with the standard version.

9. Temperature Scaling

Specifically, temperature scaling remaps the MLS output $Y = (y_1, y_2, \dots, y_K)$ to

$$\hat{Y} = \text{softmax}\left(\frac{\log(y_1)}{T}, \frac{\log(y_2)}{T}, \dots, \frac{\log(y_K)}{T}\right) \quad (16)$$

where the temperature T is chosen to optimize the fit to the data using a proper scoring function, exactly like when one is training the MLS. Note that this makes

$$\hat{y}_i = \frac{y_i^{1/T}}{\sum_{k=1}^K y_k^{1/T}} \quad (17)$$

Note that temperature scaling alters all the scores the same way for all categories with one parameter. While this works well for the balanced MNIST dataset where all of the curves are quite similar, it does not work nearly as well for the PhANNs dataset where some of the categories need a temperature greater than one (concave down) and some need a temperature less than one (concave up). This makes temperature scaling a benefit for some categories but a harm to others. In Figure 10 we show the results of temperature scaling for both the PhANNs and the MNIST datasets.

The cloud of points results because temperature scaling sends a certain score y_k to different \hat{y}_k depending on how the remaining $(1 - y_k)$ fraction of the probability is split among the other categories. The Figure shows movement (from the diagonal) generally in the direction of the right answer, shown here by the $\text{Conf}_{\text{Hist}}$ lines.

While we see that temperature scaling improved the calibration of the Other category, it was at the cost of worse calibration on categories such as “HTJ”, which would have benefited from a temperature greater than one moving all the points downward. Unfortunately, one parameter to recalibrate the full K dimensional output is asking too much, especially when the different categories pull in different directions, as is the case for PhANNs. The PhANNs dataset was dominated by the very populous category “Other” and hence the irresistible pull toward $T < 1$. Since the algorithm only gets to choose one temperature for all categories, temperature scaling only goes so far.

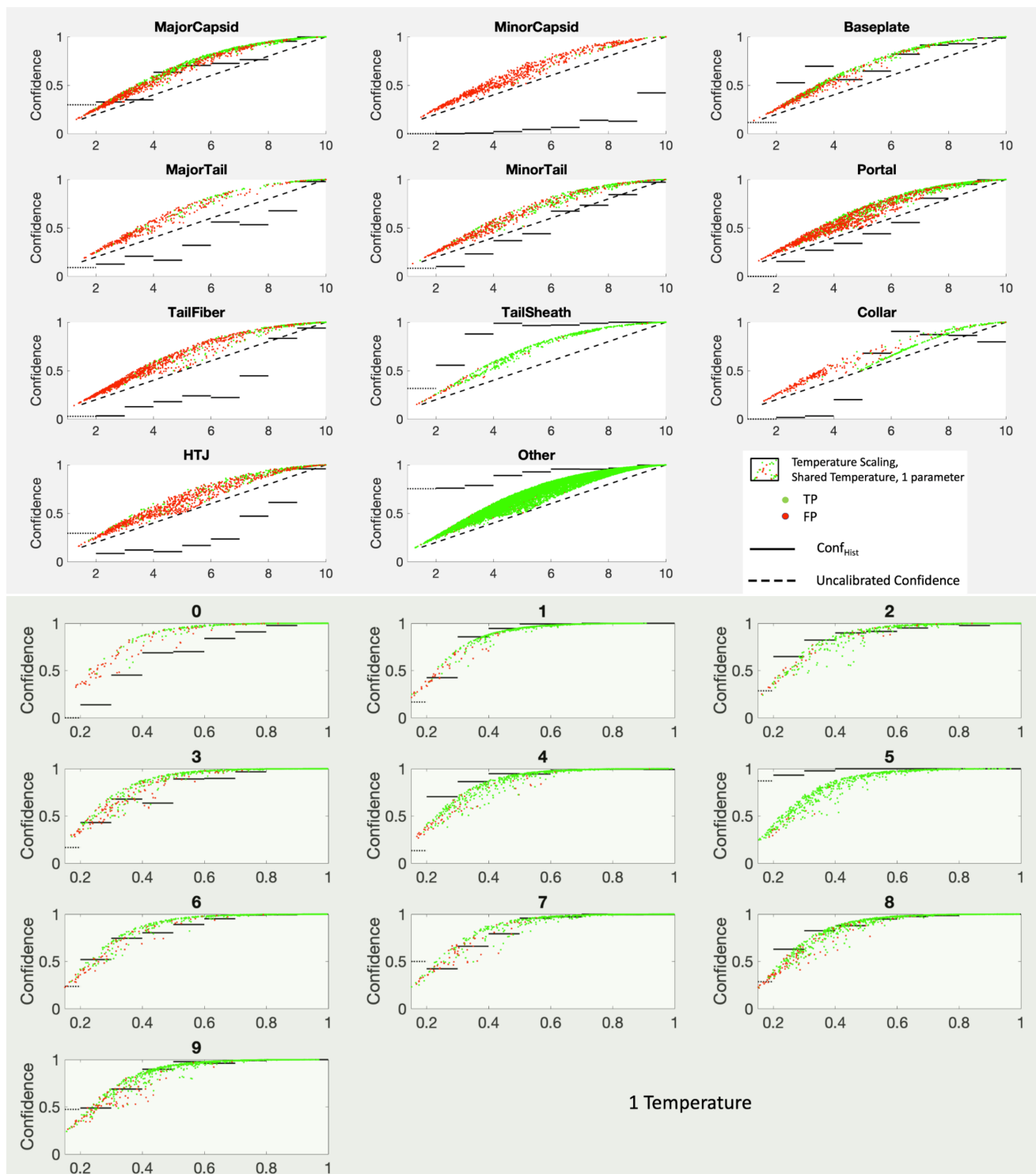


Figure 10. Shared temperature scaling for PhANNs categories in the top panel, and for MNIST handwritten digit categories in the bottom panel. Note that at each score there is a range of corresponding temperature scaled confidence values. Also note that while shared temperature scaling does quite well for MNIST, several categories in PhANNs, e.g., TailFiber and HTJ, would be better fit with the reversed convexity. In each subplot, the horizontal axes show the score while the vertical axes show confidence measured as y_k .

10. Category Specific Temperature Scaling (CSTS)

One obvious modification given our philosophy of top-label calibration is to use K different temperatures. The loss function must be adjusted, however, to count only those

examples with $k(x) = k$ using the correct answer to be 0 or 1. The result is a K -parameter model that is surprisingly competitive with KDEs. The resulting confidence curves are shown in Figure 11.

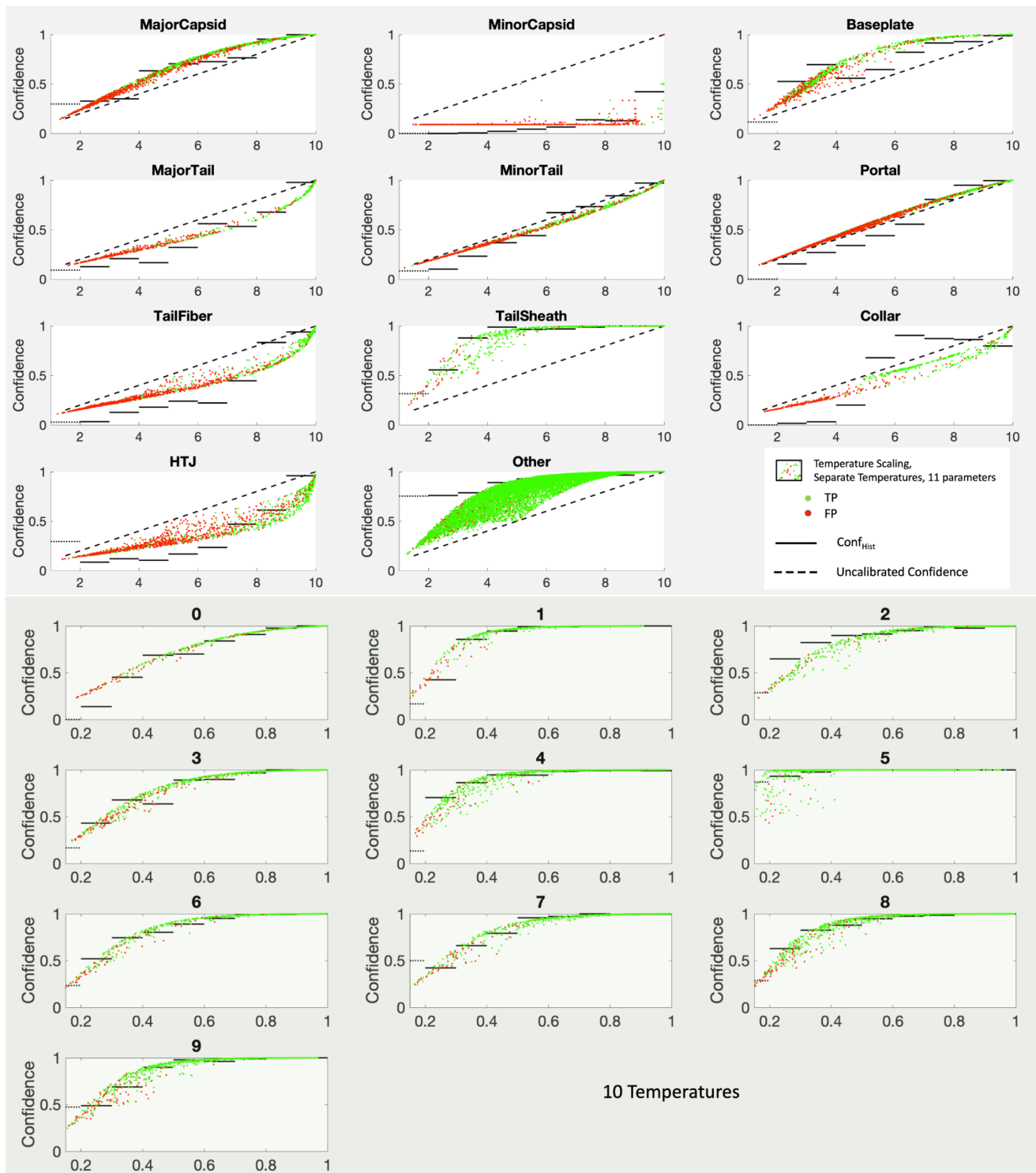


Figure 11. Results for category-specific temperature scaling (CSTS) for PhANNs phenotypes (top panel) and for MNIST handwritten digits (0–9) (bottom panel). Note the convexity changes for TailFiber and HTJ, allowing much better fits than they got with one shared temperature. In each subplot, the horizontal axes show the score while the vertical axes show confidence measured as y_k .

11. Temperature Scaling with Awards (TSwA)

A second modification is our attempt to exploit the additional information inherent in the choice of category by adding an “award” term to the score of the chosen category to account for the difference between the two curves in Figure 6. This results in

$$\hat{Y} = \text{softmax}\left(\frac{\log(y_1) + \delta_{1,k(X)}A_1}{T}, \frac{\log(y_2) + \delta_{2,k(X)}A_2}{T}, \dots, \frac{\log(y_K) + \delta_{K,k(X)}A_K}{T}\right) \quad (18)$$

where δ_{ij} is the Kronecker delta, whose value is 1 if $i = j$ and otherwise 0. The $K + 1$ vector $(A_1, A_2, \dots, A_K, T)$ is chosen to optimize the fit to the data for all categories at once. Thus it is the only technique we discuss that does NOT decompose the problem into K separate fits.

We should mention that temperature scaling with awards can violate the first desiderata listed by Zhang et al. [5]. If any of our A_i is less than zero, the winning category from Y may no longer be the winning category from \hat{Y} . This means we have to separately store the value of the chosen category. Since we calibrate one category at a time, this does not seem an undue burden, especially in light of our incentive to award the winner. The purpose of \hat{Y} is to have the correct confidence sitting in the $\hat{y}_{k(X)}$ entry, not to enable us to recalculate $k(X)$.

Temperature scaling with awards is our favorite parametric technique, and outperforms CSTS on the PhANNs data (compare Figures 11 and 12). Notably either CSTS or TSwA performs much better than simple temperature scaling on PhANNs as well as MNIST (See Appendix A). While they admittedly need K or $K + 1$ parameters rather than 1, they are compatible with our top-label calibration philosophy, whereas simple temperature scaling is not.

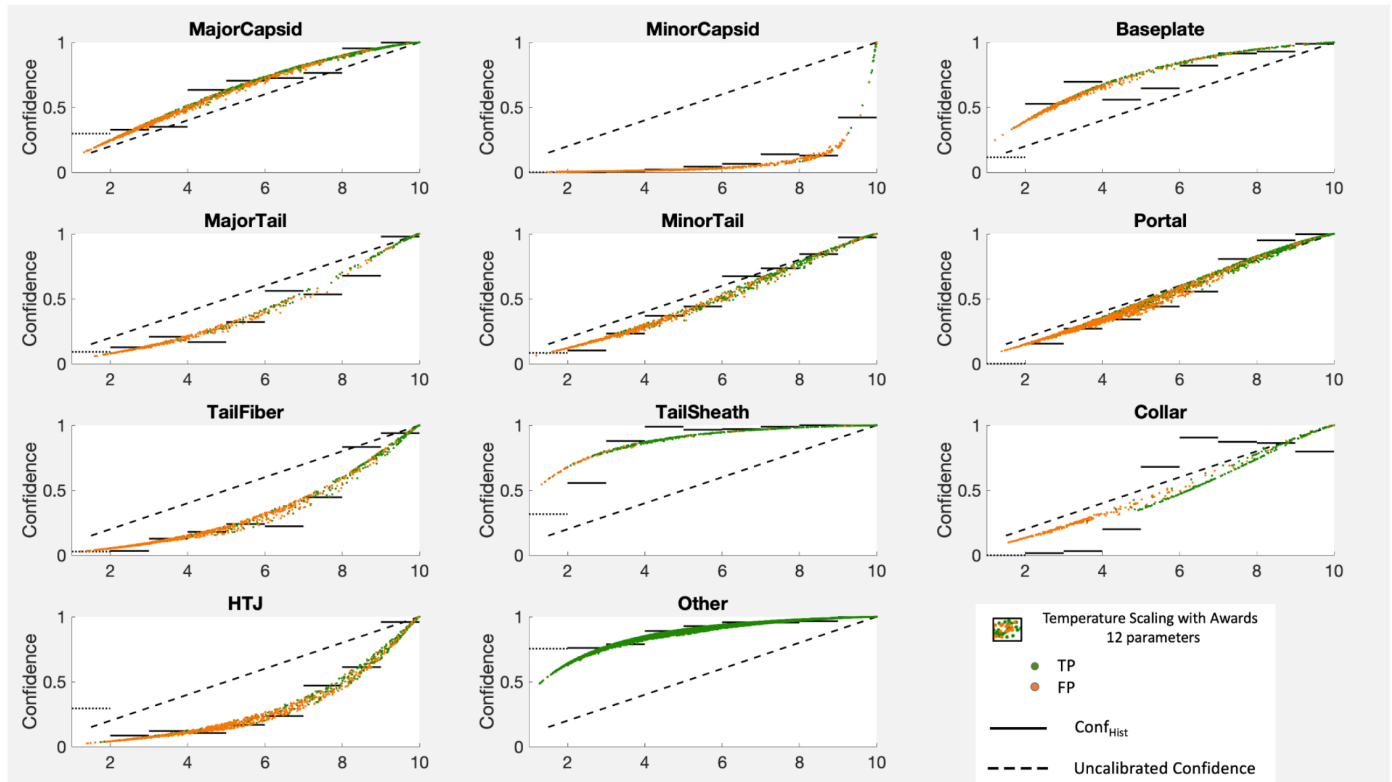


Figure 12. Cont.

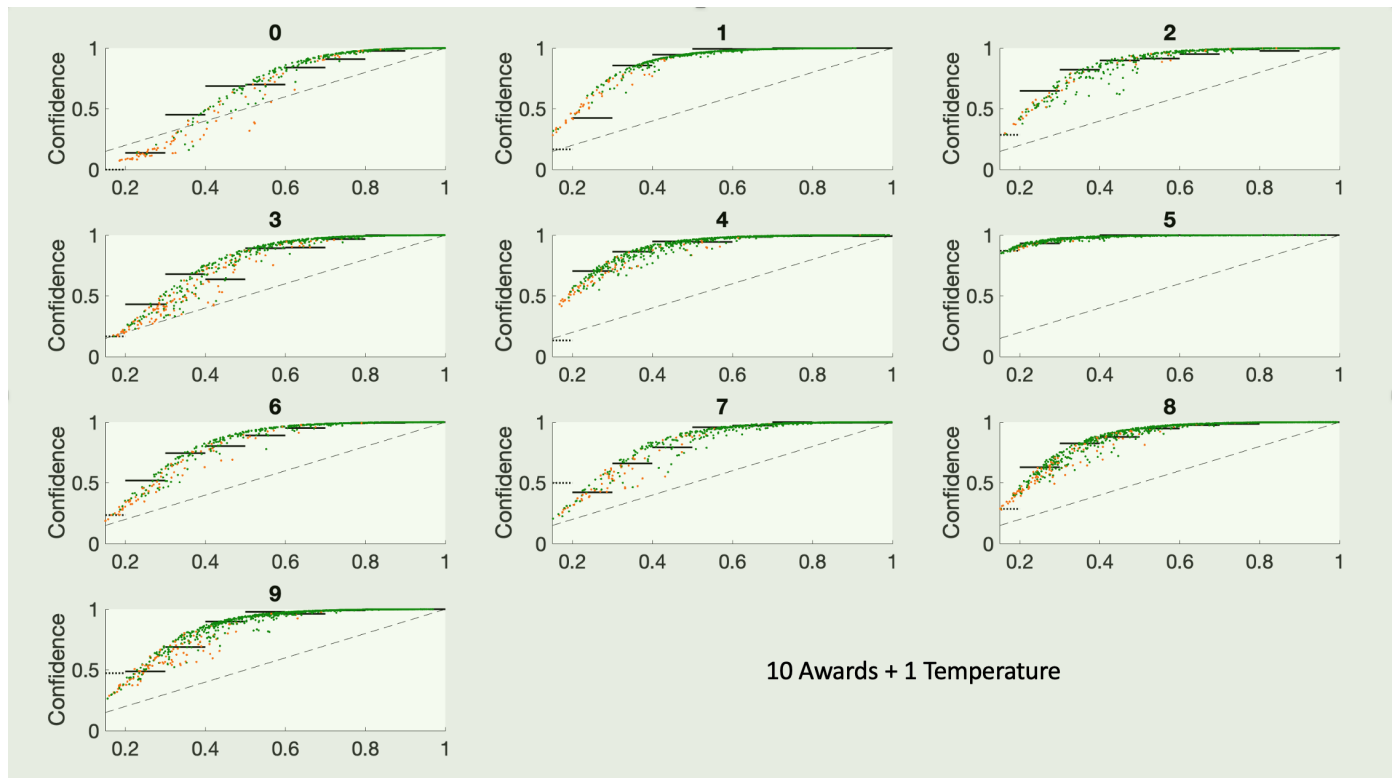


Figure 12. Results for temperature scaling with awards (TSwA), again showing PhANNs phenotypes in the top panel and MNIST handwritten digits (0–9) in the bottom panel. Note the much tighter range of confidence values at each score, strikingly illustrated by the “Other” category. Note also the ability of K awards with a single temperature to show the expressivity to capture not only the different convexities, but even resulting in a surprisingly good fit for “Minor Capsid”. In each subplot, the horizontal axes show the score while the vertical axes show confidence measured as y_k .

It is possible to individualize both a temperature and an award to each category, resulting in a $2K$ parameter model. In the Appendix A, we report on the resulting fits for MNIST. The performance is comparable to CSTS and TSwA. Given its doubling of the number of parameters without significant reduction in error, any criterion that penalizes a model for the number of parameters, e.g., AIC [22] or Occam’s Razor, would turn the $2K$ parameter model down in favor of the two models we presented above.

12. Adjusting the Bandwidth

As a general rule, choosing the bandwidth for KDEs in a data-limited environment is more difficult than fitting temperatures or awards. KDEs suffer from the “trivial maximum at 0” problem [23]. (The same is true for histograms. The smaller the bandwidth or the bin size, the better the fit to the data the kernels or the bins describe). This is not a problem if a second test set is available, but that is a rare luxury for post-hoc calibrations on exploratory problems.

The clue to a simpler, and much more stable way to choose the bandwidth comes from the early histogram binning literature in the form of an algorithm known as isotonic regression [24]. This classic paper added constraints to the histogram method described above to force the resulting function $\text{Conf}_{\text{Hist}}$ to be monotonic. The goal of monotonicity (or near monotonicity) was the key.

The main failure mode for possible methods of choosing bandwidth is ending up with “wiggly” confidence curves, i.e., Conf_{KDE} , with too many local minima. KDEs are sufficiently expressive to fit noise, leading to spurious “wiggles” in the resulting confidence curves (see Figure 13). Choosing the bandwidth by limiting the number of sign changes in the slope is a reliable, fast algorithm that can be easily implemented with no additional

data. To facilitate our comparison of KDE based confidence estimators, we modify our notation to reflect the explicit b dependence in the functions $\text{Conf}_{\text{KDE}(b)}$.

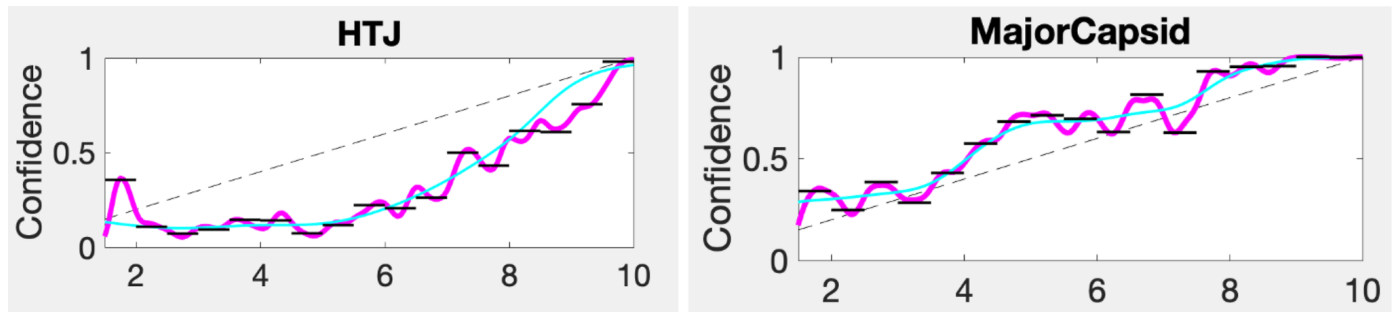


Figure 13. The maroon curves shown are $\text{Conf}_{\text{KDE}(b_{\text{opt}})}$ curves using the optimal bandwidth from leave-one-out cross-validation fits to the data for the categories HTJ and Major Capsid. Light blue is $\text{Conf}_{\text{KDE}(b_{\text{mon2}})}$ and the black lines are histograms. Too small a bandwidth always leads to wiggles that reflect the accidental (or real) clustering of the scores in the test set.

If we choose the number of sign changes to be zero, the corresponding bandwidth is

$$b_{\text{mon}} = \min\{b \mid \text{Conf}_{\text{KDE}(b)} \text{ is monotonic}\} \quad (19)$$

The $\text{Conf}_{\text{KDE}(b_{\text{mon}})}$ curves using category-specific b_{mon} for the PhANNs dataset are shown in the top panel of Figure 14. The fits are not bad, but can be improved significantly by relaxing the strict monotonicity requirements. The bottom panel shows the fits allowing one local minimum, i.e., using

$$b_{\text{mon2}} = \min\left\{b \mid \frac{d\text{Conf}_{\text{KDE}(b)}}{d\text{Score}} \text{ changes sign at most twice}\right\} \quad (20)$$

For most categories, allowing one local minimum changes the curve only slightly while improving the expressivity of the method immensely. Our recommendation is to use b_{mon2} for the bandwidth for both the TP and the FP estimates. The algorithm for finding b_{mon2} proceeds simply by starting at a low b , calculating the number of sign changes in ΔConf on a grid of S values and incrementing b until the number of sign changes reaches two or less.

Using b_{mon2} improves generalization error over b_{mon} . On the other hand, this is even more true for b_{mon4} and b_{opt} (optimized with cross validation), but at the cost of many more wiggles and is achieved, in our opinion, by fitting the noise. For the more adventurous, exploring smaller bandwidths holds some rewards but is fraught with pitfalls. *We would not recommend using any calibration without a visual check.*

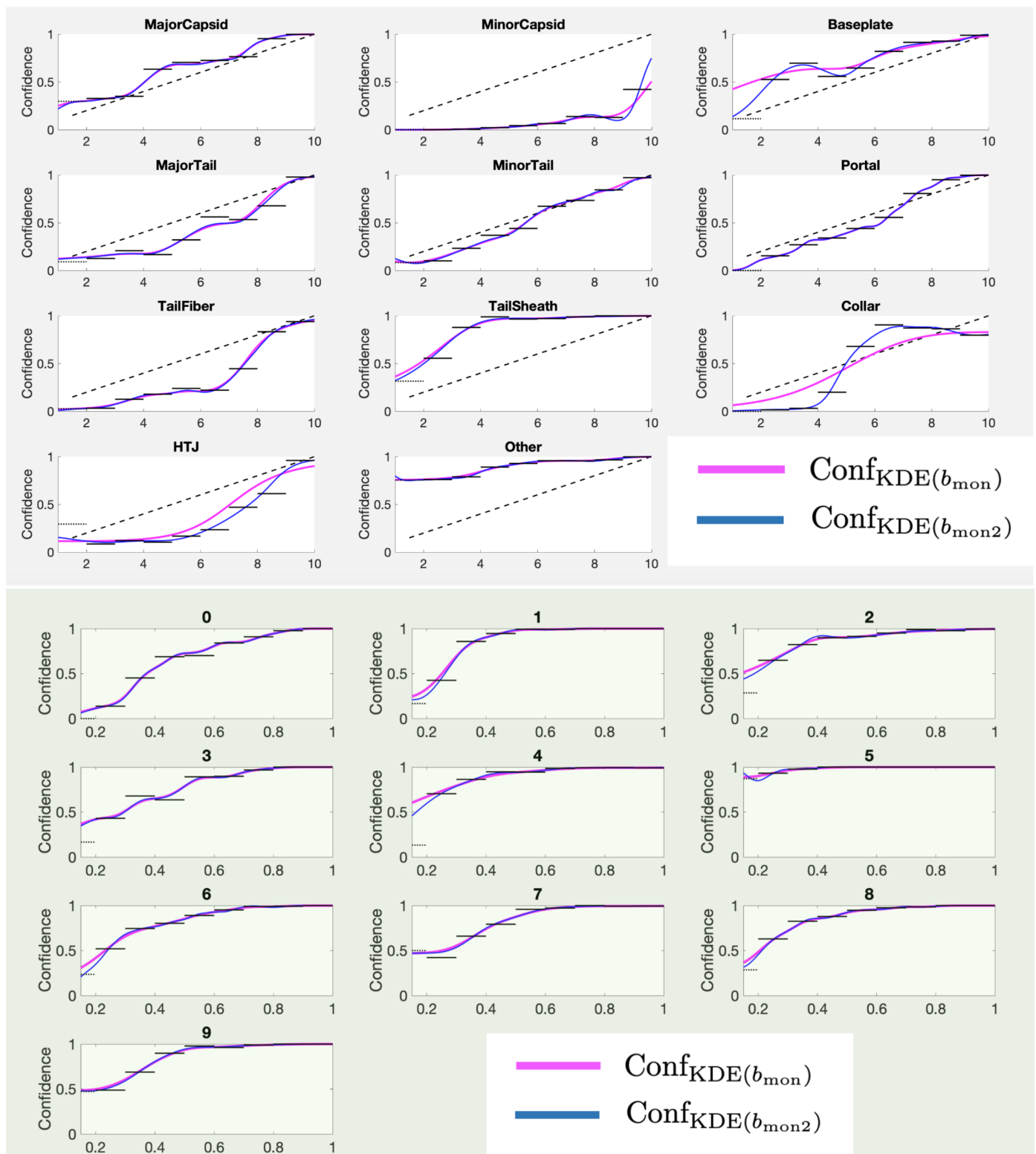


Figure 14. Comparing monotonic performance of $\text{Conf}_{\text{KDE}}(b_{\text{mon}})$ (blue) and $\text{Conf}_{\text{KDE}}(b_{\text{mon2}})$ (magenta) for the PhANNs phenotypes (top panel) and the MNIST handwritten digits (0–9) (bottom panel).

13. Discussion

13.1. Fundamental Differences between PhANNs and MNIST Datasets

The application of machine learning algorithms to a problem is closely related to the application of compression methods to the same datasets. The difficulty of bioinformatic classification problems has been previously discussed in connection with the remarkable difference in lossless compressibility between text/images and biostrings [25,26]. While standard compression tools (zip, rar, etc.) are commonly used to compress genome data, the level of compression these tools achieve when run on UNICODE encoded biostrings is 1:2 or 1:4, whereas bioinformatic specific compression can achieve lossless ratios of 1:1200 or more for higher organisms [27]. Furthermore, the compression algorithms employed are not similar, indicating that machine learning system performance may not generalize from image or text processing to bioinformatics processing, inasmuch as machine learning systems generalize across data using the same domain-specific structure that compression systems use to succinctly describe such data. The connection between compression and machine learning performance is seemingly well grounded, given that neural network architectures designed for natural language processing rather than compression are currently the best known method for compressing large text corpuses when runtime speed is not considered [28].

13.2. Dataset Composition Frequencies

During the course of this project, an additional trove of about 3000 new HTJ sequences became available for the PhANNs dataset. We added these additional samples to the 1227 previous HTJs in the test set and recalibrated. Figure 15 shows the results. The left panel shows the old and new estimated kernels. The first thing to note is that FP, and hence \mathcal{K}_{FP} , is the same with or without the new samples. Recall that these are the false positives, those samples classified as HTJ whose true classes were not HTJ. Since every sample in the new trove has true class HTJ, the new trove of data had no effect on the false positive (red) distribution shown. This implies that confidence for HTJ (as defined in Equation (6) or in Equation (9)) can only go up by adding additional true HTJ samples. The lesson, though not new, is that the frequency of the representation of the different categories in the test set has a huge effect on the confidence scores. The bigger issue is: how well do the frequencies of category representation in the use-set mirror the frequencies in the test-set?

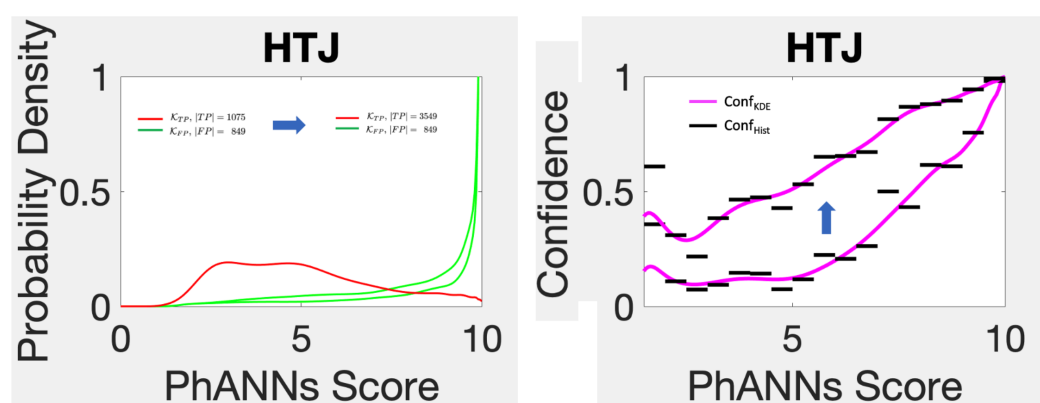


Figure 15. The effect of adding an additional trove of “HTJ” samples to the test set used to predict confidence. Since this does not change FP, Conf_{KDE} can only increase.

The HTJ datatrove example serves as a broader reminder that the category representation in the test set has a huge effect on the confidence. Gupta and Ramdas have recently managed to push the quantitative assessment of error for histogram approaches to new levels. On the other hand, their analysis does not address the points made in the previous paragraphs: error from changes in the underlying distributions. In fact we believe the largest source of (unassessed) uncertainty is the mismatch between the frequency of the

representations of the different categories in the test-set and its frequency in the use-set. This problem is called the label-shift problem [29]. With the rise of post-hoc calibration, an obvious suggestion is to keep use-statistics for the net and periodically recalibrate [30] with a test set that is curated to match the use-set statistics. For biological datasets in particular, this will be difficult to achieve.

This situation for PhANNs was exacerbated by (a reasonable) request from the journal (PLOS Computational Biology) that steps be taken to reduce the amount of interpolation rather than generalization. To this end, the ORFs being classified were de-replicated at 40% and the remaining “seeds” were split up into the 12 groups, 11 for the cross-validation and one for the test set. When these seeds were expanded to include the de-replicated protein sequences, the size of the resulting groups varied widely and had a large effect on the representation of the different categories in the test set. This conflict between controlling representation frequency of the different categories and trying to keep easy interpolation answers out of the test set is likely to continue.

13.3. Which Densities?

In the above presentation we introduced the two density estimators \mathcal{K}_{TP} and \mathcal{K}_{FP} . It is of course equivalent to the more usual approach that estimates \mathcal{K}_{TP} and \mathcal{K}_P , where

$$\mathcal{K}_P = \frac{\mathcal{K}_{\text{TP}}|\text{TP}| + \mathcal{K}_{\text{FP}}|\text{FP}|}{|\text{TP}| + |\text{FP}|} \quad (21)$$

Our choice was predicated on the fact that separating out the false positives focuses on just how dependent our calculations are on the other categories and their representation in the test set. Note that \mathcal{K}_P gives an alternate expression for the denominator in Equation (9).

13.4. The Number of Datapoints

It came as somewhat of a surprise that the number of datapoints available for the calibration problem is not what one would naively expect. We got a lesson about this from the MNIST dataset. When examining calibration results for fully-trained nets (see Appendix A), the results were surprising. Having most of the approximately 1000 points sitting at the score $S = 1$ made the calibration effort nearly impossible. For example, for category “0”, there were only four TP scores less than 0.99 and eleven FP scores total. This is far too sparse to assess score dependence. On the other hand, it is not a likely scenario for the exploratory problem-type we focus on in this work. We specifically included the comparison to MNIST in order to underline such features.

In Figure 16 we show the fully-trained MNIST results in the form of the joint distribution and the resulting calibration curves using kernel densities with $b_{\text{mon}2}$. A reasonable confidence estimation approach in this case is to just stick to confidence given only the category, as in Equation (1). If a top-label approach is required, $\text{Conf}_{\text{KDE}(b_{\text{mon}})}$ produces acceptable results as does $\text{Conf}_{\text{CUM}*}$, though most of the resulting optimal cutoffs θ_{opt} fall between 0.99 and 1.00.

Sparse data is an issue for confidence calibration more generally. Typically, this happens in two regions: high scores for false positives and low scores for true positives. Local sparsity was also the most common reason for our Conf_{KDE} curves to develop spurious wiggles. Our better bandwidth selection scheme gets around most of these problems but some persist. At scores well below the first datapoint, the tails of the different normal distributions added up can make the kernel density’s imagination go wild. This is the reason our plots begin at PhANNs score 1.5 (MNIST 0.15) and the lowest histogram bar is shown dashed in all of our figures.

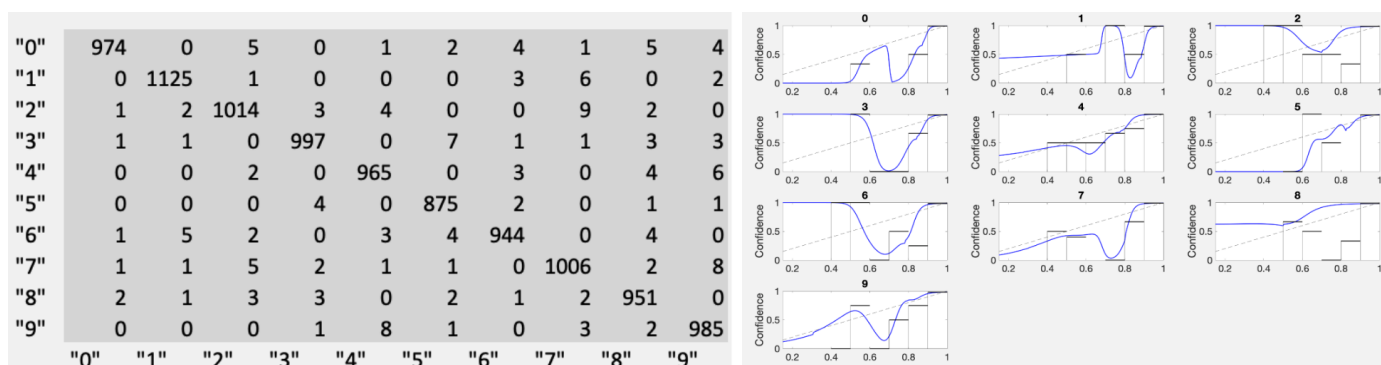


Figure 16. The joint distribution from the “Fully trained” MNIST handwritten digit (0–9) results. Note the very few erroneously classified samples shown here as the counts in off-diagonal entries. The right panel shows the blind application of our recommended $\text{Conf}_{\text{KDE}(b_{\text{mon}2})}$ and provides a warning against using any calibration without a visual check.

13.5. Boundary Correction

Kernel densities on intervals with a boundary do not count quite the way we would like since some of the probability represented in the distribution falls outside the interval. While at overly-small bandwidths this effect can be quite pronounced, at any reasonable bandwidth the effect is minimal and can be ignored.

14. Comparing Methods

Figure 17 compares the summary performance of our new methods to histograms, shared temperature scaling (STS) and uncalibrated networks. Our first observation is that the uncalibrated nets, shown by the last columns in each category, do well for some—notably MinorTail and Portal—but lag significantly in other categories, corroborating our assertion above that post-hoc calibration is always a good idea. Our second observation is that sharing a temperature (STS, in yellow) is not a good idea for the disparate categories typical of exploratory problems, although it performs quite well for MNIST (see Figure A2 in the Appendix A). While histograms of course work very well in all cases [13,18,19], our new techniques work comparably well or better without the noise sensitivity that have made most users shun this method. The chart also shows why we have been featuring KDE methods, especially $\text{KDE}(b_{\text{mon}2})$, which generally work as well or better. The difference between $\text{KDE}(b_{\text{mon}})$ and $\text{KDE}(b_{\text{mon}2})$ is small and in our opinion worthwhile, but as we argue in Section 12 and the Appendix A, which wiggles are real is not a simple call. $\text{KDE}(b_{\text{mon}})$ does not lose out to $\text{KDE}(b_{\text{mon}2})$ by enough to look significant in such aggregate comparison. The detailed visual comparisons in Section 11 are our strongest argument, although that comparison relies on the human eye providing some pattern recognition. Category-specific temperature scaling (CSTS, light blue) performs surprisingly well. Again, our best argument in favor of TSWA over CSTS is through a human eye comparing the fits in Figures 11 and 12. Finally, cumulative confidence with an optimal cutoff (CwOC, dark blue) is competitive but not stellar, just as we noted in Section 7.

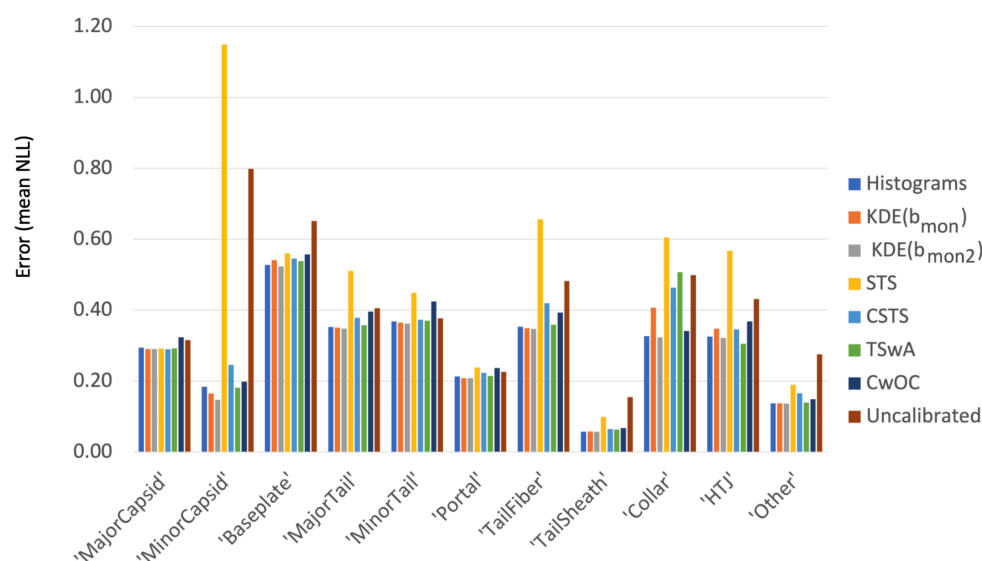


Figure 17. The errors for confidence calibration methods compared on PhANNs categories. The errors shown are the average negative log likelihood per sample. The techniques compared are discussed in the sections above: histograms (Section 4), kernel density based estimators (Sections 5 and 12), shared temperature scaling (STS, Section 9), category specific temperature scaling (CSTS, Section 10), temperature scaling with awards (TSwA, Section 11) and cumulative confidence with optimal cutoff (CwOC, Section 7). The uncalibrated scores use the $y_{k(X)}$ entry from the MLS output Y .

15. Conclusions

In the foregoing manuscript, we describe several new techniques for post-hoc calibration of machine learning systems. Specifically, we aim at a problem size and imbalance typical of exploratory problems—machine learning problems exploring new categories whose very definition depends on a dialog between the MLS and the researchers. We further argue that for such problems, top-label calibration makes better sense than other approaches. Top label calibration is also indicated on any problem where the category specific confidence varies significantly by category.

Our favorite technique for such problems is kernel density estimator-based confidence, $\text{Conf}_{\text{KDE}(b_{\text{mon}2})}$, which comes with a new robust algorithm for bandwidth selection. We suspect that it was exactly the lack of such a robust bandwidth selection algorithm that has limited the prior use of kernel density estimates for confidence calibration. We also introduced three new top-label oriented parametric techniques. The first of these models (CSTS) is a straightforward application of temperature scaling to the K one-versus-all problems that result from breaking up the multi-classification into the top-label versions. The second (TSwA) directly attacks the difference between determining confidence with or without the information provided by the predicted category. K of the $K + 1$ parameters in TSwA are “awards” which are added only to the predicted category’s score before the otherwise standard temperature scaling is applied to the shifted scores. The idea is to modify the initial score to account for the information contained in the MLS selecting a specific category. TSwA performs visibly better than CSTS on PhANNs (compare Figures 11 and 12). Again, we recommend a visual check for any calibration method.

Calibration of confidence, like calibration of accuracy, must be post-hoc to enable objective measurement and to prevent bias. Parametric approaches, with parameters trained from the test set used for the calibration, appear to compromise this benefit. Using only a few generic parameters keeps this bias to a minimum. We use the MNIST problem as a way to estimate how large a bias our generic parameters introduce and find that, at the problem size typical for exploratory problems, the sample noise swamps the bias. Our belief is that the largest source of confidence error comes from yet a different source: the mismatch between the test set used in the calibration and the use-set, i.e., the field

use composition of categories. Compared to such “label-shift” error [29], the differences between the techniques presented are small. The size of the sample and uncertainty in the use-set frequencies of the different categories leave us with a limiting knowledge gap: better calibration is not available without better data.

Our experience with the fully-trained MNIST example highlights the uniqueness of datasets for exploratory problems in contrast to relatively large and balanced datasets that represent “already solved” problems. “Already solved” problems have been the focus of much of the confidence calibration literature and explains why that literature ignores the obvious difference between top-label confidence and approaches that ignore the category predicted. The distinction between the two approaches is moot if all scores are nearly zero or nearly one.

Exploratory problems represent a type of problem that will be around for the foreseeable future. One area of future work that is particularly needed is a good theoretical foundation for selecting the best of multiple valid scores, or perhaps a method of combining multiple valid scores into a still better score. In fact, such problems are more common than we initially envisioned; we thank an anonymous referee for pointing to some potential exploratory classification problems that arise far outside our usual domain as pieces of larger packages including object detection and segmentation [31,32]. Provided these larger packages can supply a score function for the predicted classification, the methods above can be used to predict a corresponding confidence. We hope all explorers find some benefit from this user’s guide.

Author Contributions: Conceptualization, P.S., D.S. and V.A.C.; methodology: P.S. and D.S.; software: D.S. and P.S.; validation: A.M.S.; computational resources: D.S. and P.S.; data curation: V.A.C., A.M.S. and M.A.; original draft preparation: P.S., D.S. and A.M.S.; review and editing: P.S., D.S., A.M.S., M.A., T.P., V.A.C. and R.A.E.; visualization: P.S. and D.S.; supervision: A.M.S. and P.S.; project administration: A.M.S. and R.A.E.; funding acquisition: R.A.E. and A.M.S. All authors have read and agreed to the published version of the manuscript.

Funding: V.A.C., M.A., T.P., R.E. and A.S. and some of the computing resources used were supported by a subcontract to R.E. and A.S. from a NIDDK grant, RC2DK116713 Computational and Experimental Resources for Virome Analysis in Inflammatory Bowel Disease (CERVAID), to D.W.

Data Availability Statement: Our MNIST results can be replicated using [12]. To reproduce our PhANNs results, please use this paper’s supplementary files.

Acknowledgments: We are grateful for discussions with Barbara Bailey, James Nulton, Katelyn McNair and the Biomath group at SDSU. We especially benefited from Barbara Bailey’s repeated references to the eye² method over many years of discussions. We thank A. Ramdas for helpful comments based on an early draft of the manuscript.

Conflicts of Interest: The authors have declared that no competing interest exist.

Appendix A. MNIST Training Details

During the early stages of algorithm research, it is not uncommon to see members of different research groups sharing a preference for a given algorithm. Some of this comes down to subjective preference and local fashion, but a second source of preference differences can come from testing algorithms in different domains. We would not be surprised to learn that different algorithms can more or less closely match the calibration curves for some dataset-algorithm combinations.

To combat the risk of “domain bias”, and thereby temper our group’s biased background largely dealing with biological datasets, we created ensembles of MNIST classifiers. Additionally, to minimize the chance of sampling bias, we created a system which allowed us to compare algorithm performance differences across thousands of runs.

Appendix A.1. MNIST Dataset Preparation

The MNIST dataset comes as 60,000 training examples and 10,000 testing examples. Every example is a pair $(X, \text{TrueClass}(X))$, where X is a 28×28 grayscale image, $X \in \mathbb{R}^{784}$, and TrueClass is a categorical label, $\text{TrueClass} \in \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$.

To prepare our data for ensemble member training, we split the test data into a 50,000 element “training set” group, and reserved 10,000 elements as “test set 1”. The MNIST-provided test set was dubbed “test set 2”.

Appendix A.2. Network Training

Minibatch selection: Mini batches of 1000 elements were sampled, with replacement, from the “training set” group.

Architecture: All neural networks were run in stages:

1. the input data was flattened into a 784 element vector,
2. the output of 1 was run through a fully connected ReLU layer to 300 outputs,
3. the output of 2 was run through a fully connected ReLU layer to 100 outputs,
4. the output of 3 was run through a 10 element softmax layer.

We denote the resulting output as $Y(X)$.

Loss function: We minimized categorical cross entropy, namely

$$-\sum_X (\text{onehot}(\text{TrueClass}(X)) * \log(Y(X)))$$

using the ADAM optimizer [33], with a lambda of 1×10^{-3} .

Training steps: For under-trained networks, we trained for 400 eval-gradient-update steps. Normal accuracy for an under-trained network, measured using “test set 2”, was approximately 88%. Under-trained networks were used to more closely mimic the available data for our exploratory bioinformatics problem. For the fully-trained (over-trained) networks, we trained for 10,000 eval-gradient-update steps. Normal accuracy for an over-trained network, measured using “test set 2”, was approximately 99.7%.

Appendix A.3. Ensemble Sampling

We trained 1000 networks as described above. When a new ensemble output was desired, we sampled from the $\binom{1000}{10}$ possible combinations of 10 members from those 1000 ensembles, ran each network on both datasets, then took the softmax of their summed log outputs. This gave us a fresh pair of test sets, termed “ensemble test set 1” and “ensemble test set 2”, on which to compare calibration algorithms.

Appendix A.4. Calibration Method Tuning

As all of the calibration methods tested had one or more variables that needed tuning to data, and generalization performance was of principal concern. We tuned each calibration method to minimize NLL on the “ensemble test set 1”, then measured the calibration errors ECE1, ECE2, and NLL on both the “ensemble test set 1” and “ensemble test set 2” datasets for reporting.

The temperature scaling methods examined with the above double-test-set design were: single temperature scaling (STS), categorical temperature scaling (CSTS), single temperature scaling with categorical awards (TSwA), and categorical temperature scaling with categorical awards (CSTSwA, 20 parameter model).

The bandwidth selection methods examined with the above double-test-set design were: categorical monotonic bandwidths (b_{mon}), categorical bandwidths allowing one local minimum (b_{mon2}), categorical bandwidths allowing two local minima (b_{mon4}) and optimized categorical bandwidth (b_{opt}), selected using leave-one-out cross validation.

Appendix A.5. MNIST Results

The figures below show the median of 1000 MNIST ensembles with parameters that were trained on testset 1 and applied to both testset 1 and testset 2. In the figures, each method is associated with a color. The left column of each color always represents the performance on testset 1, the right its performance on testset 2. If significant bias were present from the parameter selection, we would expect that performance on testset 2 would be uniformly worse. Instead, the differences appear random (about as many higher as lower). This is evidence of a lack of bias large enough to dominate the error due to sampling, whose magnitude is revealed in the figures through these left-right differences.

Figure A1 shows the surprisingly similar performance of our four temperature scaling related methods with the notable exception of STS (red), which performs significantly worse for category “5”. As discussed in Section 9, simple temperature scaling works very well on examples like MNIST where all the categories need almost the same temperature. The other three models perform almost identically. We see this as arguments in favor of our 10 and 11 parameter models (CSTS and TSwA respectively) but not the 20 parameter model (shown in blue), whose spurious generalization risk due to the extra parameters is not justified without significant improvement in performance [22], as discussed at the end of Section 11 in the main text.

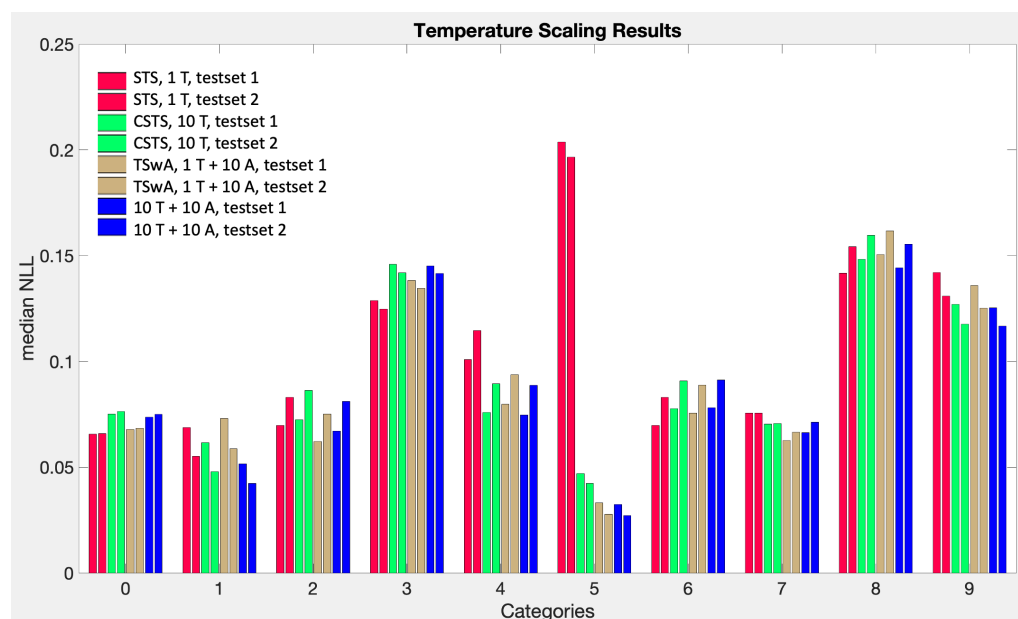


Figure A1. The performance of temperature scaling related methods applied to MNIST. All but the left method (red) represent an adaptation of temperature scaling to top-label calibration.

The MNIST bandwidth selection results shown in Figure A2 are pretty much as expected, with larger bandwidths forcing larger NLL. By definition, the bandwidths $b_{\text{mon}} > b_{\text{mon}2} > b_{\text{mon}4}$ represent progressively weaker constraints on the number of sign changes in the slope of the confidence curve, with the less constrained cases (bigger N in $b_{\text{mon}N}$) producing smaller NLL values.

Recall however, that choosing the bandwidth by optimizing the fit with leave-one-out cross-validation on the PhANNs dataset almost always gave unacceptably wiggly $\text{Conf}_{\text{KDE}(b_{\text{opt}})}$ estimates (See Figure 13). Local pockets of data sparsity or excess in either TP or FP can combine with the “trivial maximum at zero” problem [23], to choose wiggly confidence curves. We believe this instability has kept KDEs from wider use for confidence estimation. This instability is a serious issue for the “exploratory problems” PhANNs represents though a minor issue for the well balanced problems MNIST represents. The b_{mon} and $b_{\text{mon}2}$ fits in Figure 14 are almost identical for the MNIST data. In fact for the MNIST examples we see b_{opt} values even smaller than $b_{\text{mon}4}$ on the average. As discussed

in Section 12, our recommendation is to use $b_{\text{mon}2}$ (green in Figure A2) to avoid overly wiggly confidence curves that fit mostly noise. Note that we again see evidence that the sample noise is significantly bigger than any bias from the training; else we would see uniformly worse performance on testset 2.

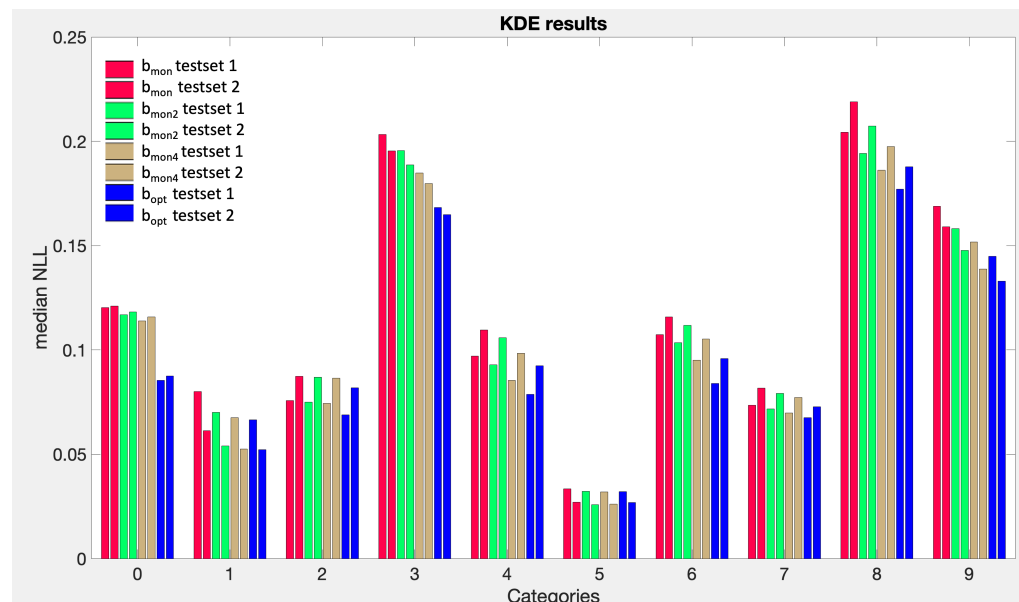


Figure A2. The effect of bandwidth selection on median negative NLL for the MNIST data. The testset1 results are always shown to the left of the testset2 results.

References

- Gawlikowski, J.; Tassi, C.R.N.; Ali, M.; Lee, J.; Humt, M.; Feng, J.; Kruspe, A.; Triebel, R.; Jung, P.; Roscher, R.; et al. A Survey of Uncertainty in Deep Neural Networks. *arXiv* **2021**, arXiv:2107.03342.
- Kuppers, F.; Kronenberger, J.; Shantia, A.; Haselhoff, A. Multivariate Confidence Calibration for Object Detection. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Seattle, WA, USA, 14–19 June 2020; pp. 1322–1330. [\[CrossRef\]](#)
- Lakshminarayanan, B.; Pritzel, A.; Blundell, C. Simple and Scalable Predictive Uncertainty Estimation using Deep Ensembles. *arXiv* **2017**, arXiv:1612.01474.
- Jiang, H.; Kim, B.; Guan, M.Y.; Gupta, M. To Trust or Not to Trust A Classifier. *arXiv* **2018**, arXiv:1805.11783.
- Zhang, J.; Kailkhura, B.; Han, T.Y.J. Mix-n-match: Ensemble and compositional methods for uncertainty calibration in deep learning. In Proceedings of the International Conference on Machine Learning, PMLR, Virtual Event, 13–18 July 2020; pp. 11117–11128.
- Guo, C.; Pleiss, G.; Sun, Y.; Weinberger, K.Q. On calibration of modern neural networks. *arXiv* **2017**, arXiv:1706.04599.
- Kumar, A.; Sarawagi, S.; Jain, U. Trainable calibration measures for neural networks from kernel mean embeddings. In Proceedings of the International Conference on Machine Learning, PMLR, Stockholm, Sweden, 10–15 July 2018; pp. 2805–2814.
- Kull, M.; Perello-Nieto, M.; Kängsepp, M.; Filho, T.S.; Song, H.; Flach, P. Beyond temperature scaling: Obtaining well-calibrated multiclass probabilities with Dirichlet calibration. *arXiv* **2019**, arXiv:1910.12656.
- Wen, Y.; Jerfel, G.; Muller, R.; Dusenberry, M.W.; Snoek, J.; Lakshminarayanan, B.; Tran, D. Combining Ensembles and Data Augmentation can Harm your Calibration. *arXiv* **2020**, arXiv:2010.09875.
- Cantu, V.A.; Salamon, P.; Seguritan, V.; Redfield, J.; Salamon, D.; Edwards, R.A.; Segall, A.M. PhANNs, a fast and accurate tool and web server to classify phage structural proteins. *PLoS Comput. Biol.* **2020**, *16*, e1007845. [\[CrossRef\]](#)
- Cantu, V.A. PhANNs Web Tool. Available online: <http://phanns.com> (accessed on 19 July 2023).
- LeCun, Y. The MNIST Database of Handwritten Digits. 1998. Available online: <http://yann.lecun.com/exdb/mnist/> (accessed on 19 July 2023).
- Gupta, C.; Ramdas, A.K. Top-label calibration and multiclass-to-binary reductions. In Proceedings of the International Conference on Learning Representations, PMLR, Virtual Event, 25–29 April 2022.
- Zadrozny, B.; Elkan, C. Transforming classifier scores into accurate multiclass probability estimates. In Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Edmonton, AB, Canada, 23–36 July 2002; pp. 694–699.
- Ovadia, Y.; Fertig, E.; Ren, J.; Nado, Z.; Sculley, D.; Nowozin, S.; Dillon, J.V.; Lakshminarayanan, B.; Snoek, J. Can You Trust Your Model’s Uncertainty? Evaluating Predictive Uncertainty under Dataset Shift. *arXiv* **2019**, arXiv:1906.02530.

16. Drish, J. *Obtaining Calibrated Probability Estimates from Support Vector Machines*; Technique Report; Department of Computer Science and Engineering, University of California: San Diego, CA, USA, 2001.
17. Platt, J. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Adv. Large Margin Classif.* **1999**, *10*, 61–74.
18. Gupta, C.; Podkopaev, A.; Ramdas, A. Distribution-free binary classification: Prediction sets, confidence intervals and calibration. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 3711–3723.
19. Gupta, C.; Ramdas, A. Distribution-free calibration guarantees for histogram binning without sample splitting. In Proceedings of the International Conference on Machine Learning, PMLR, Virtual Event, 18–24 July 2021; pp. 3942–3952.
20. Gneiting, T.; Raftery, A.E. Strictly proper scoring rules, prediction, and estimation. *J. Am. Stat. Assoc.* **2007**, *102*, 359–378. [[CrossRef](#)]
21. Tomani, C.; Cremers, D.; Buettner, F. Parameterized Temperature Scaling for Boosting the Expressive Power in Post-Hoc Uncertainty Calibration. *arXiv* **2021**, arXiv:2102.12182.
22. Akaike, H. A new look at the statistical model identification. *IEEE Trans. Autom. Control* **1974**, *19*, 716–723. [[CrossRef](#)]
23. Guidoum, A.C. Kernel Estimator and Bandwidth Selection for Density and its Derivatives: The kedd Package. *arXiv* **2020**, arXiv:2012.06102.
24. Zadrozny, B.; Elkan, C. Obtaining calibrated probability estimates from decision trees and naive bayesian classifiers. In Proceedings of the Eighteenth International Conference on Machine Learning, ICML, Williamstown, MA, USA, 28 June–1 July 2001; Volume 1, pp. 609–616.
25. Wandelt, S.; Bux, M.; Leser, U. Trends in genome compression. *Curr. Bioinform.* **2014**, *9*, 315–326. [[CrossRef](#)]
26. Nalbantoglu, O.U.; Russell, D.J.; Sayood, K. Data compression concepts and algorithms and their applications to bioinformatics. *Entropy* **2009**, *12*, 34–52. [[CrossRef](#)] [[PubMed](#)]
27. Pavlichin, D.S.; Weissman, T.; Yona, G. The human genome contracts again. *Bioinformatics* **2013**, *29*, 2199–2202. [[CrossRef](#)] [[PubMed](#)]
28. Melis, G.; Dyer, C.; Blunsom, P. On the state of the art of evaluation in neural language models. *arXiv* **2017**, arXiv:1707.05589.
29. Podkopaev, A.; Ramdas, A. Distribution-free uncertainty quantification for classification under label shift. *arXiv* **2021**, arXiv:2103.03323.
30. Li, M.; Sethi, I.K. Confidence-based active learning. *IEEE Trans. Pattern Anal. Mach. Intell.* **2006**, *28*, 1251–1261.
31. Li, M.; Huang, P.Y.; Chang, X.; Hu, J.; Yang, Y.; Hauptmann, A. Video pivoting unsupervised multi-modal machine translation. *IEEE Trans. Pattern Anal. Mach. Intell.* **2022**, *45*, 3918–3932. [[CrossRef](#)] [[PubMed](#)]
32. Chang, X.; Ren, P.; Xu, P.; Li, Z.; Chen, X.; Hauptmann, A. A comprehensive survey of scene graphs: Generation and application. *IEEE Trans. Pattern Anal. Mach. Intell.* **2021**, *45*, 1–26. [[CrossRef](#)] [[PubMed](#)]
33. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.