On the Calibration of Probabilistic Classifier Sets

Thomas Mortier
KERMIT
Ghent University

Viktor Bengs
Institute of Informatics
LMU Munich

Eyke Hüllermeier
Institute of Informatics
LMU Munich

Stijn Luca
KERMIT
Ghent University

Willem Waegeman
KERMIT
Ghent University

Abstract

Multi-class classification methods that produce sets of probabilistic classifiers, such as ensemble learning methods, are able to model aleatoric and epistemic uncertainty. Aleatoric uncertainty is then typically quantified via the Bayes error, and epistemic uncertainty via the size of the set. In this paper, we extend the notion of calibration, which is commonly used to evaluate the validity of the aleatoric uncertainty representation of a single probabilistic classifier, to assess the validity of an epistemic uncertainty representation obtained by sets of probabilistic classifiers. Broadly speaking, we call a set of probabilistic classifiers calibrated if one can find a calibrated convex combination of these classifiers. To evaluate this notion of calibration, we propose a novel non-parametric calibration test that generalizes an existing test for single probabilistic classifiers to the case of sets of probabilistic classifiers. Making use of this test, we empirically show that ensembles of deep neural networks are often not well calibrated.

1 INTRODUCTION

There is a general consensus that trustworthy machine learning systems should not only return accurate predictions, but also a credible representation of their uncertainty. In this regard, two inherently different sources of uncertainty are often distinguished, referred to as *aleatoric* and *epistemic* (Hora, 1996), and various methods that quantify these types of uncertainty have been proposed (Senge et al., 2014; Kendall and Gal, 2017; Hüllermeier and Waegeman, 2021). In multi-class classification problems, which will be the setting of interest in this paper, such methods typically focus on the uncertainty in the outcome $y \in \mathcal{Y}$ given a query instance $\boldsymbol{x} \in \mathcal{X}$ for which a prediction is sought.

Background. Consider a standard multi-class classification setting with instance space $\mathcal{X}$ and label space $\mathcal{Y} = \{1, \ldots, K\}$. We assume that the data is i.i.d. according to an underlying joint probability measure P on $\mathcal{X} \times \mathcal{Y}$. Correspondingly, each instance $\boldsymbol{x} \in \mathcal{X}$ is associated with a conditional distribution $\boldsymbol{p}(\boldsymbol{x})$, where the component $p_k(\boldsymbol{x}) = p(Y = k \mid \boldsymbol{x})$ is the probability to observe class k as an outcome given $\boldsymbol{x}$. A probabilistic classifier $\hat{\boldsymbol{p}}$ can then be defined as an estimator of $\boldsymbol{p}(\boldsymbol{x})$. Usually, $\hat{\boldsymbol{p}}$ is learned from a hypothesis space $\mathcal{H}$, which is a subset of all possible mappings from $\mathcal{X}$ to Δ_K, where Δ_K denotes the $(K-1)$-simplex

$$\Delta_K \coloneqq \left\{ \boldsymbol{\theta} = (\theta_1, \ldots, \theta_K) \in [0,1]^K \mid \|\boldsymbol{\theta}\|_1 = 1 \right\} \quad (1)$$

of probability vectors $\boldsymbol{\theta}$. Each of these vectors identifies a categorical distribution $\mathrm{Cat}(\boldsymbol{\theta})$. Let us remark that we will utilize bold lowercase letters for vectors and vector functions, e.g., $\boldsymbol{\theta}$ and $\hat{\boldsymbol{p}}(\boldsymbol{x})$, whereas a normal font will depict the components of these vectors, e.g., θ_k and $\hat{p}_k(\boldsymbol{x})$.

In this setting, aleatoric uncertainty is usually defined as uncertainty that cannot be reduced by collecting more training samples, as it originates from wrong class annotations or a lack of informative features. Therefore, the "ground-truth" is the above-mentioned conditional probability distribution $\boldsymbol{p}(\boldsymbol{x})$, and the Bayes error of this distribution tells us whether the aleatoric uncertainty is high. Even with perfect knowledge about the underlying data-generating process, the outcome cannot be predicted with certainty. However, the learner does not know $\boldsymbol{p}(\boldsymbol{x})$ in practice. Instead, it uses an estimate $\hat{\boldsymbol{p}}(\boldsymbol{x})$ based on the training data as a surrogate. In essence, epistemic uncertainty refers to the uncertainty about the true $\boldsymbol{p}$, or the "gap" between $\boldsymbol{p}$ and $\hat{\boldsymbol{p}}$. Methods that represent this (second-order) uncertainty typically do not return a single $\hat{\boldsymbol{p}}(\boldsymbol{x})$, but either a distribution over $\hat{\boldsymbol{p}}(\boldsymbol{x})$, as in Bayesian methods, or sets containing different $\hat{\boldsymbol{p}}(\boldsymbol{x})$, as in ensemble methods.

The validity of an aleatoric uncertainty representation is typically evaluated by statistically testing whether a fitted model $\hat{\boldsymbol{p}}(\boldsymbol{x})$ is calibrated, see e.g. (Hosmer and Lemeshow,

Proceedings of the 26th International Conference on Artificial Intelligence and Statistics (AISTATS) 2023, Valencia, Spain. PMLR: Volume 206. Copyright 2023 by the author(s).

2003; Vaicenavicius et al., 2019; Widmann et al., 2019). However, to be very precise, calibration tests do not test whether a model $\hat{\boldsymbol{p}}(\boldsymbol{x})$ corresponds to the ground-truth $\boldsymbol{p}(\boldsymbol{x})$, because calibration is only a necessary condition. Likewise, since the true $\boldsymbol{p}(\boldsymbol{x})$ is in practice never observed, the evaluation of epistemic uncertainty representations is also far from trivial. For lack of an objective ground-truth, most methods assess epistemic uncertainty representations in an indirect manner, using downstream tasks such as out-of-distribution detection (Ovadia et al., 2019), robustness to adversarial attacks (Kopetzki et al., 2021) or active learning (Nguyen et al., 2022). However, all those tasks are characterized by a scenario where training and test data are not identically distributed. Therefore, several recent studies have raised concerns about the usefulness of such tasks w.r.t. epistemic uncertainty evaluation (Abe et al., 2022; Dewolf et al., 2021; Bengs et al., 2022).

Contribution. In this work, we present a statistical approach to test the validity of epistemic uncertainty representations of methods that use sets of distributions for prediction purposes. This includes various types of ensemble methods, such as random forests (Shaker and Hüllermeier, 2020), deep ensembles (Lakshminarayanan et al., 2017; Rahaman and Thiery, 2021; Schulz and Lerch, 2022), dropout networks (Gal, 2016), stacking methods (Ting and Witten, 1999) and linear opinion pooling (Hora, 2004; Ranjan and Gneiting, 2010; Lichtendahl et al., 2013). With ensemble methods and a predefined hypothesis space $\mathcal{H}$ of probabilistic classifiers, one can assume that in total M different probabilistic models are fitted to the training data, denoted as $\mathcal{P} = \{\hat{\boldsymbol{p}}^{(1)}, \ldots, \hat{\boldsymbol{p}}^{(M)}\} \subseteq \mathcal{H}$, where $\hat{\boldsymbol{p}}^{(m)}$ represents the m-th model.

Inspired by the literature on linear opinion pools and stacking, we are interested in the convex set that contains all convex combinations of these M models:

$$S(\mathcal{P}) = \left\{ \hat{\boldsymbol{p}}_{\boldsymbol{\lambda}} \,\middle|\, \hat{\boldsymbol{p}}_{\boldsymbol{\lambda}}(\boldsymbol{x}) = \sum_{m=1}^{M} \lambda_m \hat{\boldsymbol{p}}^{(m)}(\boldsymbol{x}), \boldsymbol{\lambda} \in \Delta_M \right\} \tag{2}$$

with $\Delta_M \coloneqq \left\{\boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_M) \in [0,1]^M \mid \|\boldsymbol{\lambda}\|_1 = 1\right\}$. Two properties are of crucial importance for these convex sets. On the one hand, the *size* of the convex set quantifies the degree of epistemic uncertainty, so this size should be as small as possible. On the other hand, the convex set should be a *valid* representation, i.e., it should contain the true $\boldsymbol{p}$. To measure the size of the convex set, which is typically done in a pointwise manner, various measures are used in the literature, such as the mutual information (Malinin et al., 2020) or the generalized Hartley measure (Hüllermeier et al., 2022). Conversely, to evaluate the validity, no method exists today.

Therefore, we introduce in this paper the first calibration test to evaluate the validity of set-based epistemic uncertainty representations, starting from existing calibration tests for aleatoric uncertainty representations. Informally, we call a set-based epistemic uncertainty representation calibrated if there exists a calibrated convex combination in the set of Eq. (2). This testing problem gives rise to a challenging scenario of multiple hypothesis testing, since all members in the corresponding convex set need to be tested (in the worst case). We propose a novel test based on resampling and analyze empirically the Type I and Type II error for various calibration measures and conditions. Lastly, we apply our newly-developed test to analyze whether deep ensembles and dropout networks represent epistemic uncertainty in a correct manner.

2 ALEATORIC UNCERTAINTY EVALUATION

In this section we formally review existing calibration tests for multi-class classification problems. With such tests, aleatoric uncertainty representations of classifiers can be evaluated in a natural and direct way. In the following section, these tests will be extended for epistemic uncertainty representations in the form of probabilistic classifier sets. This section can hence be interpreted as a discussion of closely-related work, but it also intends to introduce the mathematical concepts needed further on.

2.1 Different Notions Of Calibration

We start by formally defining confidence calibration (Guo et al., 2017). This notion of calibration is by far the most often used in literature (Filho et al., 2021). Let's assume that we have fitted a probabilistic classifier that estimates $\boldsymbol{p}(\boldsymbol{x})$, leading to the estimate $\hat{\boldsymbol{p}}(\boldsymbol{x})$. Furthermore, let $c(\boldsymbol{x}) = \max_k \hat{p}_k(\boldsymbol{x})$ be the mode of the estimated conditional class distribution for instance $\boldsymbol{x}$ (or confidence score) and let $f(\boldsymbol{x}) = \operatorname{argmax}_k \hat{p}_k(\boldsymbol{x})$ be the corresponding prediction.

Definition 1. *A multi-class classifier is* confidence calibrated *if it holds that*

$$\mathbb{P}_{(\boldsymbol{X},Y)\sim P}\left(Y = f(\boldsymbol{X}) \,|\, c(\boldsymbol{X}) = s\right) = s\,.$$

This is a rather weak notion of calibration, as only the mode of the conditional distribution needs to be calibrated. A stronger form of calibration is classwise calibration (Zadrozny and Elkan, 2001).

Definition 2. *A multi-class classifier is* classwise calibrated *if for all $k \in \{1, ..., K\}$ it holds that*

$$\mathbb{P}_{(\boldsymbol{X},Y)\sim P}\left(Y = k \,|\, \hat{p}_k(\boldsymbol{X}) = s\right) = s\,.$$

A few authors define an even stronger notion of calibration, sometimes referred to as calibration in the strong sense (Widmann et al., 2019; Filho et al., 2021).

Definition 3. *A multi-class classifier is* calibrated in the strong sense *if for all $k \in \{1, ..., K\}$ it holds that*

$$\mathbb{P}_{(\boldsymbol{X},Y)\sim P}\ (Y = k \,|\, \hat{\boldsymbol{p}}(\boldsymbol{X}) = \boldsymbol{s}) = s_k\,,$$

with s_k the k-th component of $\boldsymbol{s}$.

2.2 Calibration Tests

All three definitions of calibration assume that one knows the true underlying distribution P. However, in practice, this distribution is unknown, so one needs to replace the expectation by a finite-sample estimate. In addition, a calibration measure that quantifies the discrepancy between observed and expected frequencies is needed. In a final step, this calibration measure can be used to construct a statistical test that decides whether a multi-class classifier is calibrated or not. In the literature, three different types of tests have been developed. The Hosmer-Lemeshow test is a specific type of chi-squared test that is commonly used as a goodness-of-fit test for logistic and multinomial regression models in statistics (Hosmer and Lemeshow, 2003; Fagerland et al., 2008). The resampling-based test of Vaicenavicius et al. (2019) and the kernel-based test of Widmann et al. (2019) have been proposed more recently in the machine learning literature. The Hosmer-Lemeshow test can only be used to assess classwise calibration, whereas the other tests are more general, because they can be used in combination with a wide range of calibration measures. All tests analyze the null hypothesis $H_0 : \hat{\boldsymbol{p}}$ is calibrated versus the alternative $H_1 : \hat{\boldsymbol{p}}$ is not calibrated, where the notion of calibration (i.e., confidence, classwise or strong) depends on the underlying test.

Hosmer and Lemeshow (2003) originally proposed the Hosmer-Lemeshow (HL) test for binary logistic regression, and later Fagerland et al. (2008) extended it to the more general class of multinomial probabilistic models. Let $\mathcal{D}_{val} = \{(\boldsymbol{x}_1, y_1), ..., (\boldsymbol{x}_N, y_N)\}$ be a validation set of size N, i.i.d. according to P. We will use the shorthand notation $\hat{p}_{ik}$ for the estimated probability $\hat{p}_k(\boldsymbol{x}_i)$. Let $c_i = \max_k \hat{p}_{ik}$ be the highest probability for instance $\boldsymbol{x}_i$ and let $\hat{y}_i = \operatorname{argmax}_k \hat{p}_{ik}$ be the predicted label. For every label y_i, let us consider a K-dimensional vector $\boldsymbol{y}_i$ that defines the one-hot encoding of y_i, i.e., $y_{ik} = 1$ iff $y_i = k$. Similarly as for the true label, we transform $\hat{y}_i$ to a K-dimensional vector $\hat{\boldsymbol{y}}_i$ that defines the one-hot encoding of $\hat{y}_i$, i.e., $\hat{y}_{ik} = 1$ iff $\hat{y}_i = k$.

For class k, the probabilities $\hat{p}_{1k}, ..., \hat{p}_{Nk}$ are sorted, and the corresponding instances are placed into equal-frequency bins $\mathcal{B}_{1k}, ..., \mathcal{B}_{Bk}$ with B the number of bins. Furthermore, the following test statistic is considered:

$$HL_{cwise}(\hat{\boldsymbol{p}}, \mathcal{D}_{val}) = \sum_{k=1}^{K}\sum_{j=1}^{B} \frac{(o(\mathcal{B}_{jk}) - \mathrm{p}(\mathcal{B}_{jk}))^2}{\mathrm{p}(\mathcal{B}_{jk})}\,, \quad (3)$$

$$\text{with} \qquad o(\mathcal{B}_{jk}) = \frac{1}{|\mathcal{B}_{jk}|}\sum_{i:\hat{p}_{ik}\in\mathcal{B}_{jk}} y_{ik}\,,$$

$$\mathrm{p}(\mathcal{B}_{jk}) = \frac{1}{|\mathcal{B}_{jk}|}\sum_{i:\hat{p}_{ik}\in\mathcal{B}_{jk}} \hat{p}_{ik}\,, \quad (4)$$

where $o(\mathcal{B}_{jk})$ and $\mathrm{p}(\mathcal{B}_{jk})$ denote the observed and expected probabilities in bin $\mathcal{B}_{jk}$, respectively. Fagerland et al. (2008) argued that (3) follows approximately a chi-squared distribution with $(K-1)(B-2)$ degrees of freedom, and they derived p-values using this assumption. Let us remark that, besides the HL test, many other alternative goodness-of-fit tests exist in statistics, but these usually do not assess model calibration in a direct manner.

Vaicenavicius et al. (2019) developed a more general (non-parametric) test that can be adopted to assess confidence calibration, classwise calibration and calibration in the strong sense, depending on the calibration measure that is used. This test constructs a bootstrap distribution of any calibration measure empirically, under the null hypothesis that a classifier is calibrated, by resampling new labels multiple times from the probabilistic model that is under assessment. After constructing the distribution of the calibration measure under the null hypothesis, the test verifies, for the observed labels, how likely the calibration measure is under the assumption of a calibrated model.

The test of Vaicenavicius et al. (2019) is often used with the expected calibration error (ECE) as calibration measure. This measure has been originally introduced for binary problems as a way to evaluate reliability diagrams in a quantitative manner. The classwise extension of this measure in fact looks very similar to the Hosmer-Lemeshow test statistic, so it is used to assess classwise calibration (Def. 2):

$$ECE_{cwise}(\hat{\boldsymbol{p}}, \mathcal{D}_{val}) = \frac{1}{K}\sum_{k=1}^{K}\sum_{j=1}^{B} \frac{|\mathcal{B}_{jk}|}{N} |o(\mathcal{B}_{jk}) - \mathrm{p}(\mathcal{B}_{jk})|\,, \quad (5)$$

with $o(\mathcal{B}_{jk})$ and $\mathrm{p}(\mathcal{B}_{jk})$ as in (4). However, for the expected calibration error, the $[0, 1]$-interval is often subdivided in intervals of equal length instead of equal frequency, i.e., $\mathcal{B}_{jk} := \{i : \frac{j-1}{B} \le \hat{p}_{ik} < \frac{j}{B}\}$.

For the confidence-based extension of expected calibration error, which is used to assess confidence calibration (Def. 1), binning only has to be done once (so, not for every class k). Let us divide the unit interval into B subintervals of equal length. With the j-th subinterval we associate the bin $\mathcal{B}_j := \{i : \frac{j-1}{B} \le c_i < \frac{j}{B}\}$[1]. Then, the calibration measure becomes:

$$ECE_{conf}(\hat{\boldsymbol{p}}, \mathcal{D}_{val}) = \sum_{j=1}^{B} \frac{|\mathcal{B}_j|}{N} |\mathrm{Acc}(\mathcal{B}_j) - \mathrm{c}(\mathcal{B}_j)|\,, \quad (6)$$

[1] the last bins $\mathcal{B}_{Bk}$ and $\mathcal{B}_B$ are right-closed.

$$\text{with} \quad \mathrm{Acc}(\mathcal{B}_j) = \frac{1}{|\mathcal{B}_j|} \sum_{i:c_i \in \mathcal{B}_j} \sum_{k=1}^{K} y_{ik} \hat{y}_{ik} \,,$$
$$\mathrm{c}(\mathcal{B}_j) = \frac{1}{|\mathcal{B}_j|} \sum_{i:c_i \in \mathcal{B}_j} c_i \,.$$

Widmann et al. (2019) proposed a class of measures derived from matrix-valued kernel functions, the so-called squared kernel calibration error (SKCE), to test calibration in the strong sense, without the need for binning. For example, if one analyses all pairs of instances, while using a general matrix-valued kernel $\Gamma : \Delta_K \times \Delta_K \to \mathbb{R}^{K \times K}$, then the measure becomes:

$$\begin{aligned}\widehat{SKCE}_{uq}(\hat{\boldsymbol{p}}, \mathcal{D}_{val}) = &\binom{N}{2}^{-1} \\ &\times \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} \sum_{s=1}^{K} \sum_{t=1}^{K} (\hat{p}_{is} - y_{is}) \\ &\times (\hat{p}_{jt} - y_{jt}) \Gamma_{st}(\hat{\boldsymbol{p}}(\boldsymbol{x}_i), \hat{\boldsymbol{p}}(\boldsymbol{x}_j)) \,. \end{aligned} \tag{7}$$

However, it is clear that calculating the above measure is computationally expensive when the number of instances N increases. For that reason, a similar estimator was proposed by the same authors:

$$\begin{aligned}\widehat{SKCE}_{ul}(\hat{\boldsymbol{p}}, \mathcal{D}_{val}) = &\frac{1}{\lfloor N/2 \rfloor} \\ &\times \sum_{i=1}^{\lfloor N/2 \rfloor} \sum_{s,t=1}^{K} (\hat{p}_{(2i-1)s} - y_{(2i-1)s})(\hat{p}_{2it} - y_{2it}) \\ &\times \Gamma_{st}(\hat{\boldsymbol{p}}(\boldsymbol{x}_{2i-1}), \hat{\boldsymbol{p}}(\boldsymbol{x}_{2i})) \,, \end{aligned} \tag{8}$$

which is linear in N. Widmann et al. (2019) also consider other calibration measures, and they derive bounds and approximations on the p-value for the null hypothesis H_0 that the model is calibrated. These tests require a lot of space to explain, so we refer to the original paper for a further discussion.

3 EPISTEMIC UNCERTAINTY EVALUATION

In this section we develop calibration tests for epistemic uncertainty representations of type $S(\mathcal{P})$, as defined in Eq. (2), starting from the methodology that was reviewed in the previous section. In the literature, this type of set is sometimes called a credal set, which is typically assumed to be convex and closed (Walley, 1991; Corani and Antonucci, 2014; Yang et al., 2014). Furthermore, from a statistical point of view, there is also a close connection to linear opinion pooling, where the goal is to aggregate probability expert forecasts in an appropriate way, e.g. keeping in mind a proper scoring rule (Hora, 2004; Ranjan and Gneiting, 2010; Lichtendahl et al., 2013). In this work, $S(\mathcal{P})$ represents epistemic uncertainty, i.e., due to a limited training dataset, one cannot estimate the ground-truth probability distribution precisely, but this distribution should be contained in the set. An interpretation of that kind allows us to present natural extensions of confidence calibration, classwise calibration, and calibration in the strong sense for probabilistic classifier sets.

Definition 4. *A probabilistic classifier set $S(\mathcal{P})$ is confidence calibrated (cf. classwise calibrated or calibrated in the strong sense) if there exists a probabilistic model $\hat{\boldsymbol{p}}_{\boldsymbol{\lambda}}(\boldsymbol{x}) \in S(\mathcal{P})$ that is confidence calibrated (cf. classwise calibrated or calibrated in the strong sense).*

In what follows, we develop a statistical test to verify whether a probabilistic classifier set is calibrated according to Def. 4, which translates to the following hypotheses:

$$H_0 : S(\mathcal{P}) \text{ is calibrated}, \; H_1 : \neg H_0 \tag{9}$$

The above set of hypotheses can also be written as:

$$H_0 : \exists \boldsymbol{\lambda} \in \Delta_M \text{ s.t. } \hat{\boldsymbol{p}}_{\boldsymbol{\lambda}} \text{ is calibrated}, \; H_1 : \neg H_0 \tag{10}$$

Furthermore, in what follows, we will use the generic term "calibrated" or "calibration" to denote any notion of calibration. While the test problem in Section 2.2 focuses on the question whether a *single* fixed probabilistic classifier $\hat{\boldsymbol{p}}$ is calibrated, the test problem in (9) or (10) asks for the existence of a calibrated convex combination of the finite set of probabilistic classifiers. Thus, the latter is essentially a multiple comparison problem, as we simultaneously test for all $\boldsymbol{\lambda}$ the hypotheses

$$H_{0,\boldsymbol{\lambda}} : \hat{\boldsymbol{p}}_{\boldsymbol{\lambda}} \text{ is calibrated}, \; H_{1,\boldsymbol{\lambda}} : \neg H_{0,\boldsymbol{\lambda}} \tag{11}$$

Since the number of possible convex combinations is infinite, one cannot resort to standard ways for addressing multiple hypothesis testing problems, such as Bonferroni-correction or the Holm-Bonferroni method. In what follows, we will adopt the extreme value approach for high-dimensional testing (Dickhaus, 2015). All calibration measures considered in Section 2 are in essence decreasing functions of the *degree of calibration* of a probabilistic model, so one can simply search for the minimum of any calibration measure of interest over $\boldsymbol{\lambda} \in \Delta_M$. Then, the multiple hypothesis testing problem in (10) can be addressed by considering the distribution of the minimum under the null hypothesis.

This principle leads to a nonparametric test that is in fact a direct generalization of the test of Vaicenavicius et al. (2019). The pseudocode of this procedure is given in Alg. 1. Starting from a set $\mathcal{P}$ containing M fitted probabilistic models, a validation dataset $\mathcal{D}_{val} = \{(\boldsymbol{x}_1, y_1), ..., (\boldsymbol{x}_N, y_N)\}$, and a general calibration measure g, such as HL_{cwise} or ECE_{conf}, the method first constructs the distribution of the calibration measure under the null hypothesis. To this end, it performs in total D

Algorithm 1 Calibration Test for Probabilistic Classifier Sets – **input:** $\mathcal{P}, \mathcal{D}_{\text{val}}, g, \alpha, D$

1: **for** $d = 1, \ldots, D$ **do**
2: $\quad \mathcal{D}_d \leftarrow$ extract bootstrap sample of size N from $\mathcal{D}_{\text{val}}$ (only features are further used)
3: $\quad$ Sample uniformly $\boldsymbol{\lambda}_{0,d} \in \Delta_M$
4: $\quad$ For all $\boldsymbol{x}_i \in \mathcal{D}_d$, sample y_i from $Cat(\hat{\boldsymbol{p}}_{\boldsymbol{\lambda}_{0,d}}(\boldsymbol{x}_i))$
5: $\quad t_{0d} \leftarrow g(\hat{\boldsymbol{p}}_{\boldsymbol{\lambda}_{0,d}}, \mathcal{D}_d)$ $\quad \triangleright$ g represents an arbitrary calibration measure
6: $q_{1-\alpha} \leftarrow$ compute $(1-\alpha)$-quantile of empirical distribution $\{t_{0,1}, \ldots, t_{0,D}\}$
7: $t \leftarrow \min_{\boldsymbol{\lambda} \in \Delta_M} g(\hat{\boldsymbol{p}}_{\boldsymbol{\lambda}}, \mathcal{D}_{\text{val}})$ $\quad \triangleright$ Use an appropriate optimization algorithm for g
8: reject H_0 if $t > q_{1-\alpha}$ and don't reject H_0 otherwise

runs with D a hyperparameter that trades off runtime versus p-value precision. In each run, a bootstrap sample of size N is sampled from the original validation dataset (line 2). Then, one $\boldsymbol{\lambda} \in \Delta_M$ is selected at random and the corresponding element of the convex set $S(\mathcal{P})$ is chosen as the ground-truth probability distribution, since we are working under the assumption that the null hypothesis is true (line 3). We assume that every element of $S(\mathcal{P})$ is equally likely to correspond to the ground-truth distribution P, thus a uniform sample is drawn. Subsequently, for every instance in the bootstrap replicate, a label is randomly drawn from the selected probability distribution (line 4). In the last step of every run, the calibration measure is computed for the generated artificial labels and the ground-truth probability distribution (line 5). After D runs, we assume that a good estimate of the distribution of the calibration measure is obtained, under the assumption that the null hypothesis is true. Given a controlled Type I error rate α, the $(1-\alpha)$-quantile of this distribution is computed (line 6). This quantile defines the maximum allowed value of the calibration measure to not reject the null hypothesis. Subsequently, the minimum of the calibration measure is computed for all members of the convex set, using now the original validation dataset (line 7). The null hypothesis is rejected if this minimum exceeds the threshold that was found under the empirical null distribution (line 8).

Two lines in the algorithm deserve some more discussion. In line 5, we compute the minimum of the calibration measure without solving an optimization problem. This is because we know the ground-truth probability distribution in this case. In the limit, when the sample size grows to infinity, the calibration measure g will even be zero, so in fact we are looking for the natural deviation from zero for a sample of size N.

Conversely, in line 7, we have to solve an optimization problem to find the minimum over $\boldsymbol{\lambda} \in \Delta_M$. Specific solvers are needed here, because the objective functions are in most cases not differentiable, e.g., for the measures in (3), (5) and (6) this is not the case. The objective functions constructed from (7) and (8) are differentiable, but not convex. In this work we use constrained optimization by linear approximation (COBYLA), which is a numerical optimization method for constrained problems where the derivative of the objective function is not known (Powell, 1994). In principle, any constrained derivative-free optimization algorithm can be used in line 7, and exploring other solvers will be considered in future work.

4 EXPERIMENTS

Two types of experiments are considered in this work. First, we analyse the empirical Type I and II error of our statistical test for synthetic datasets. Second, we apply our test on several real-world datasets and popular ensemble-based methods to investigate whether these methods return calibrated representations. Detailed information regarding the experimental setup and datasets can be found in the supplementary materials (see Section 1).

4.1 Type I And II Error Analysis

In a first set of experiments, we analyse the empirical Type I and II error rate for Alg. 1, with number of bootstrap samples $D = 100$, where we consider the calibration measures that have been discussed in Section 2: $SKCE_{ul}$, HL_{cwise}, ECE_{conf} and ECE_{cwise}. For computational reasons, we choose not to incorporate $SKCE_{uq}$. For simplicity, we use a bin size of $B = 5$ and $B = 10$ for HL_{cwise} and $ECE_{conf,cwise}$. Similar as in Widmann et al. (2019), for $SKCE_{ul}$, we use the matrix-valued kernel $\Gamma(\boldsymbol{p}, \boldsymbol{p}') = \exp(-\|\boldsymbol{p} - \boldsymbol{p}'\|/2)\boldsymbol{I}_K$, combining the commonly-used total variation distance and the $K \times K$ identity matrix $\boldsymbol{I}_K$. Moreover, we analyse the statistical Type I and II error under three different scenarios, which we denote by S1, S2 and S3, respectively. S1 generates data under the null hypothesis that the probabilistic classifier set is calibrated, whereas S2 and S3 generate data under the alternative hypothesis that the probabilistic classifier set is not calibrated. To this end, we construct $R = 1000$ synthetic datasets that contain $N = 100$ instances. For each instance, we generate predictions for M probabilistic models and a ground-truth label: $\{(\boldsymbol{p}^{(1)}(\boldsymbol{X}_i), \ldots, \boldsymbol{p}^{(M)}(\boldsymbol{X}_i), Y_i)\}_{i=1}^N$, with ensemble size $M = 10$, for $K = 10$ classes. For each dataset, we assume a mean $\boldsymbol{p}_e | \boldsymbol{X}_i \sim Dir(\boldsymbol{a}_e)$, with $a_{ek} = 1/K$ for all $k = 1, \ldots, K$, and sample an ensemble from the prior $\boldsymbol{p}^{(1)}, \ldots, \boldsymbol{p}^{(M)} | \boldsymbol{X}_i \sim Dir(K\boldsymbol{p}_e/u)$. By means of this prior, we are able to control the center $\boldsymbol{p}_e$ and uncertainty (or spread) u of the convex sets, similarly as in (Sensoy et al., 2018). Then, we simulate the corresponding labels Y_i conditionally on the probabilistic classifier set in three ways:

S1: The null hypothesis is true. For each dataset, we sample uniformly at random $\boldsymbol{\lambda} \in \Delta_M$, and select $\boldsymbol{p}_{\boldsymbol{\lambda}}$ as the ground-truth probability distribution, i.e., the labels are generated according to $Cat(\boldsymbol{p}_{\boldsymbol{\lambda}})$.

S2: The null hypothesis is false. For each dataset, we generate labels using a categorical distribution that is randomly chosen outside the convex set on the line segment between the closest corner in Δ^K and $\boldsymbol{p}_e$.

S3: The null hypothesis is false, too. Similarly as in S2, a categorical distribution is randomly chosen outside the convex set, but this time on the line segment between a randomly chosen corner in Δ^K and $\boldsymbol{p}_e$. In this case, it should be somewhat easier to reject the null hypothesis than in scenario S2.

The three different scenarios are illustrated in Fig. 1 (left) for $K = 3$ and $M = 10$. For scenarios S2 and S3, an additional algorithm is needed to compute the line segment outside the convex set. This algorithm is explained in the supplementary materials (see Section 1).

The results are shown in Fig. 1 (right). Our test with $SKCE_{ul}$ is not correct, because it makes more errors than defined by the significance level. However, for the other measures, our test is more reliable, because the Type I error is mostly lower than the significance level. HL_{cwise} does not seem to yield high power for S2, for different bin sizes. Empirically, our test with $ECE_{conf,cwise}$ results in reliable tests when it comes to both the Type I and Type II error. In the supplementary materials (see Section 2), we also show some results obtained for $u = 0.1$ (Fig. 1), $M = 100$ (Fig. 2) and $K = 100$ (Fig. 3). Similar findings are obtained w.r.t. the calibration measure $SKCE_{ul}$. For most cases, larger probabilistic classifier sets result in more conservative tests, as can be observed in Fig. 1. Increasing the ensemble size, however, does not seem to have a significant influence on the results, following Fig. 2. Finally, when looking at Fig. 3, by increasing the number of classes, the empirical test error approaches the significance level, making most tests less conservative.

4.2 Calibration Of Probabilistic Classifier Sets Based On Deep Neural Networks

In a last set of experiments, we apply our test in Alg. 1 on six benchmark datasets, where we consider a single classifier (S) and three ensemble-based models: two dropout networks with rate 0.1 and 0.6 (DN(0.1), DN(0.6)) and a deep ensemble (DE), where we use an ensemble size of ten. For the deep ensembles, a probabilistic classifier set is obtained by training ten different models, i.e., with different initializations of the weights (Lakshminarayanan et al., 2017). For the dropout networks, a probabilistic classifier set is obtained by using dropout in the last layer and sampling ten predictions (Gal, 2016). For the image datasets, we consider a small and large neural network architecture that are commonly used in the literature: MobileNetV2 (MOB) with 3.4×10^6 parameters and VGG16 (VGG) with 138×10^6 parameters, respectively (Sandler et al., 2018; Simonyan and Zisserman, 2014). For the last two biological datasets, a simple neural network with one hidden layer is considered, together with textual feature representations. For more information related to the training of the models and datasets, we refer the reader to the supplementary materials (see Section 1). Furthermore, we analyse the calibration of the probabilistic classifier sets by means of the ECE_{conf} and ECE_{cwise} calibration measures, since those measures gave reliable tests in terms of the Type I and Type II error in the previous simulations.

The results are shown in Table 1. For each classifier, we report results for the ensemble average, i.e., one the average of the ensemble predictions, and the weighted ensemble average, with the weights $\boldsymbol{\lambda}$ obtained in line 7 of Alg. 1. More precisely, two different weighted ensemble averages are considered by running our test with the ECE_{conf} and ECE_{cwise} measure, respectively, on a specific calibration set. For each weighted average, we show the outcome of our test w.r.t. the set of hypotheses in (9), together with the average accuracy and calibration measure obtained on the test set. For the single classifier case, our test in Alg. 1 corresponds to the specific test from Vaicenavicius et al. (2019).

The single classifiers based on deep neural networks are in general not calibrated, and this corresponds to similar findings that have been reported in the literature (Guo et al., 2017). For most cases, the classifiers are not calibrated in terms of ECE_{cwise}, which is not so surprising, because classwise calibration is already a very strong form of calibration. Our findings are in line with what has been reported in the literature on linear opinion pooling and deep ensembles, namely, a linear opinion pool is often not calibrated in the strong sense, even in the ideal case in which the individual models are calibrated (Hora, 2004; Ranjan and Gneiting, 2010; Lichtendahl et al., 2013; Rahaman and Thiery, 2021; Schulz and Lerch, 2022). When it comes to ECE_{conf} and dropout networks, in some cases, our test does not reject the null hypothesis when a higher dropout rate is considered. This indicates that dropout networks tend to be better confidence calibrated when using a higher dropout rate, which is also confirmed by comparing the average ECE_{conf} on the test set between DN(0.1) and DN(0.6). A possible explanation could be given by the fact that the prior, from which models are sampled in the dropout network with higher dropout rate, is more diverse and results in large probabilistic classifier sets that include the ground-truth distribution. Finally, when it comes to ECE_{conf}, deep ensembles appear to be calibrated for most datasets and architectures, since the null hypothesis is almost never rejected. For those classifiers, there is also a significant difference in terms of ECE_{conf} between the average and weighted average, which indicates that using a simple averaging strategy results in less calibrated deep ensembles.

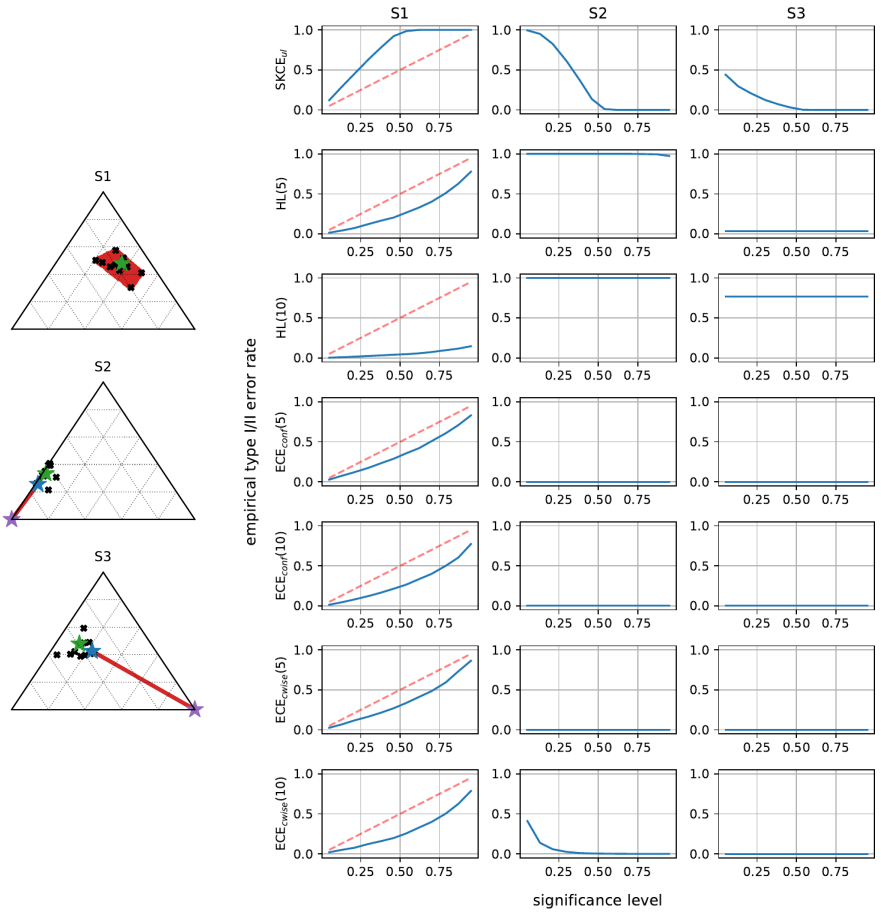

Figure 1: Left: A visualization of the setup in scenarios S1, S2 and S3. In all three scenarios, black stars correspond to an ensemble that has been sampled as outlined in the main text. In S1, the red dots correspond to a convex set from which the ground-truth distribution is uniformly sampled. In S2 and S3, the ground-truth distribution is uniformly sampled from the red line segment, outside the convex set. For S2, the line segment connects the calculated boundary (blue star) and closest corner in the simplex (purple star) of the convex set. For S3, the line segments connects a boundary of the convex set and a random corner in the simplex. Right: empirical Type I (S1) and Type II (S2, S3) error rate for different calibration measures in function of the significance level for $R = 1000$ randomly sampled datasets from S1, S2 and S3 and with $N = 100, M = 10, K = 10$ and $u = 0.01$. For all tests we use $D = 100$ bootstrap samples.

Table 1: Results obtained for four different classifiers: single classifier (S), dropout network with rate 0.1 (DN(0.1)) and 0.6 (DN(0.6)) and a deep ensemble (DE). Classifiers are tested on six benchmark datasets. For the image datasets, we use two different architectures: VGG16 (VGG) and MobileNetV2 (MOB). For each classifier we consider the average and two different weighted averages, defined by the convex combination found in line 7 of Alg. 1, for ECE_{conf} and ECE_{cwse}, respectively. We report the average accuracy (Acc.) and calibration measure ($ECE_{conf,cwise}$) obtained on the test sets. For the two weighted averages, which correspond to confidence and classwise calibration, respectively, we also present the outcome (ie. reject or not/$\neg$ reject H_0) of our test on a separate calibration dataset.

			Avg.			Weighted avg.					
						$\boldsymbol{\lambda}_{conf}$			$\boldsymbol{\lambda}_{cwise}$		
Dataset	Arch.	Classifier	Acc.	ECE_{conf}	ECE_{cwise}	H_0	Acc.	ECE_{conf}	H_0	Acc.	ECE_{cwise}
CIFAR-10	MOB	S	0.7804	0.0194	0.0071	Rej.	0.7804	0.0194	Rej.	0.7804	0.0071
		DN(0.1)	0.7740	0.0612	0.0134	Rej.	0.7726	0.0629	Rej.	0.7730	0.0137
		DN(0.6)	0.3039	0.0196	0.0108	¬Rej.	0.3025	0.0142	Rej.	0.3007	0.0111
		DE	0.8329	0.1680	0.0327	¬Rej.	0.7642	0.0150	Rej.	0.7668	0.0062
	VGG	S	0.8438	0.0828	0.0183	Rej.	0.8438	0.0828	Rej.	0.8438	0.0183
		DN(0.1)	0.8476	0.0821	0.0183	Rej.	0.8474	0.0832	Rej.	0.8462	0.0181
		DN(0.6)	0.8355	0.0950	0.0202	Rej.	0.8353	0.0954	Rej.	0.8351	0.0203
		DE	0.8754	0.0044	0.0042	¬Rej.	0.8756	0.0052	¬Rej.	0.8766	0.0044
Caltech-101	MOB	S	0.9477	0.0135	0.0010	Rej.	0.9477	0.0135	Rej.	0.9477	0.0010
		DN(0.1)	0.9384	0.0081	0.0010	¬Rej.	0.9389	0.0087	Rej.	0.9361	0.0010
		DN(0.6)	0.9338	0.0129	0.0012	¬Rej.	0.9338	0.0107	Rej.	0.9324	0.0012
		DE	0.9625	0.0175	0.0009	¬Rej.	0.9509	0.0076	¬Rej.	0.9579	0.0009
	VGG	S	0.9287	0.0395	0.0013	Rej.	0.9287	0.0395	Rej.	0.9287	0.0013
		DN(0.1)	0.9218	0.0192	0.0013	¬Rej.	0.9204	0.0212	Rej.	0.9218	0.0014
		DN(0.6)	0.9259	0.0155	0.0015	¬Rej.	0.9162	0.0062	Rej.	0.9157	0.0015
		DE	0.9500	0.0231	0.0011	¬Rej.	0.9329	0.0079	¬Rej.	0.9306	0.0011
Caltech-256	MOB	S	0.7829	0.0638	0.0009	Rej.	0.7829	0.0638	Rej.	0.7829	0.0009
		DN(0.1)	0.7820	0.0473	0.0008	Rej.	0.7816	0.0480	Rej.	0.7823	0.0008
		DN(0.6)	0.7395	0.0201	0.0009	¬Rej.	0.7313	0.0085	Rej.	0.7336	0.0009
		DE	0.8383	0.0358	0.0007	¬Rej.	0.8082	0.0130	Rej.	0.8312	0.0006
	VGG	S	0.7552	0.0948	0.0011	Rej.	0.7552	0.0948	Rej.	0.7552	0.0011
		DN(0.1)	0.7458	0.0645	0.0011	Rej.	0.7452	0.0663	Rej.	0.7421	0.0011
		DN(0.6)	0.7427	0.0106	0.0010	¬Rej.	0.7414	0.0103	Rej.	0.7421	0.0010
		DE	0.8128	0.0326	0.0007	¬Rej.	0.7864	0.0104	Rej.	0.8108	0.0007
PlantCLEF2015	MOB	S	0.4842	0.1300	0.0005	Rej.	0.4842	0.1300	Rej.	0.4842	0.0005
		DN(0.1)	0.4357	0.0866	0.0005	¬Rej.	0.4320	0.0920	Rej.	0.4319	0.0005
		DN(0.6)	0.4382	0.0199	0.0005	¬Rej.	0.4343	0.0245	Rej.	0.4348	0.0005
		DE	0.5973	0.0993	0.0004	¬Rej.	0.5364	0.0171	Rej.	0.5757	0.0004
	VGG	S	0.4085	0.1362	0.0006	Rej.	0.4085	0.1362	Rej.	0.4085	0.0006
		DN(0.1)	0.4156	0.1009	0.0006	¬Rej.	0.4151	0.1020	Rej.	0.4146	0.0006
		DN(0.6)	0.3927	0.0566	0.0005	¬Rej.	0.3928	0.0566	Rej.	0.3894	0.0005
		DE	0.5488	0.1045	0.0004	¬Rej.	0.4705	0.0197	Rej.	0.5236	0.0004
Bacteria	–	S	0.8785	0.0507	0.0002	¬Rej.	0.8785	0.0507	¬Rej.	0.8785	0.0002
		DN(0.1)	0.8398	0.1504	0.0002	Rej.	0.8345	0.1508	Rej.	0.8371	0.0002
		DN(0.6)	0.8407	0.2085	0.0002	¬Rej.	0.7491	0.0829	¬Rej.	0.7879	0.0002
		DE	0.8926	0.1437	0.0002	¬Rej.	0.8565	0.0642	¬Rej.	0.9032	0.0001
Proteins	–	S	0.8001	0.0454	0.0001	Rej.	0.8001	0.0454	Rej.	0.8001	0.0001
		DN(0.1)	0.7909	0.0788	0.0001	Rej.	0.7895	0.0779	Rej.	0.7915	0.0001
		DN(0.6)	0.8117	0.0393	0.0001	Rej.	0.8033	0.0353	Rej.	0.8119	0.0001
		DE	0.8076	0.0764	0.0001	Rej.	0.7968	0.0528	Rej.	0.8050	0.0001

5 DISCUSSION

In this paper we addressed the following question: What does it mean that a probabilistic classisfier set represents epistemic uncertainty in a faithful manner? To answer this question, we referred to the notion of calibration of probabilistic classifiers and extended it to probabilistic classifier sets. We called a probabilistic classifier set $S(\mathcal{P})$ calibrated if the set contained at least one calibrated classifier. To verify this property for the important case of ensemble-based models, we proposed a novel nonparametric calibration test that generalizes existing tests for probabilistic classifiers to the case of probabilistic classifier sets. In our experiments, we analyzed the Type I and II error of the newly-proposed test for different scenarios. The best results were obtained when using expected calibration error as underlying calibration measure, but for most measures the Type I and II error were both sufficiently low. Making use of this test, we empirically show that probabilistic classifier sets based on deep neural networks are often not calibrated.

Acknowledgements

For this work W.W. received funding from the Flemish government under the "Onderzoeksprogramma Artificiële Intelligentie (AI) Vlaanderen" Programme (Number 174L00121).

References

Abe, T., Buchanan, E. K., Pleiss, G., Zemel, R., and Cunningham, J. P. (2022). Deep ensembles work, but are they necessary?

Bengs, V., Hüllermeier, E., and Waegeman, W. (2022). On the difficulty of epistemic uncertainty quantification in machine learning: The case of direct uncertainty estimation through loss minimisation.

Corani, G. and Antonucci, A. (2014). Credal ensembles of classifiers. *Computational Statistics & Data Analysis*, 71:818–831.

Deng, J., Dong, W., Socher, R., Li, L., Li, K., and Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *IEEE CVPR*, pages 248–255.

Dewolf, N., Baets, B. D., and Waegeman, W. (2021). Valid prediction intervals for regression problems.

Dickhaus, T. (2015). Simultaneous test procedures in high dimensions: the extreme value approach. In *International Conference on Computational Mathematics, Computational Geometry and Statistics*.

Fagerland, M., Hosmer, D., and Bofin, A. (2008). Multinomial goodness-of-fit tests for logistic regression models. *Statistics in Medicine*, 27(12):4238–4253.

Fiannaca, A., Paglia, L. L., Rosa, M. L., Bosco, G. L., Renda, G., Rizzo, R., Gaglio, S., and Urso, A. (2018). Deep learning models for bacteria taxonomic classification of metagenomic data. *BMC Bioinformatics*, 19-S(7):61–76.

Filho, T. S., Song, H., Perelló-Nieto, M., Santos-Rodríguez, R., Kull, M., and Flach, P. A. (2021). Classifier calibration: How to assess and improve predicted class probabilities: a survey. *CoRR*, abs/2112.10327.

Gal, Y. (2016). *Uncertainty in Deep Learning*. PhD thesis, University of Cambridge.

Goëau, H., Bonnet, P., and Joly, A. (2015). Lifeclef plant identification task 2015. In *Working Notes of CLEF 2015*, volume 1391.

Griffin, G., Holub, A., and Perona, P. (2007). Caltech-256 object category dataset. Technical Report 7694, California Institute of Technology.

Guo, C., Pleiss, G., Sun, Y., and Weinberger, K. Q. (2017). On calibration of modern neural networks. In *ICML*, volume 70 of *Proceedings of Machine Learning Research*, pages 1321–1330.

Hora, S. (1996). Aleatory and epistemic uncertainty in probability elicitation with an example from hazardous waste management. *Reliability Engineering and System Safety*, 54(2–3):217–223.

Hora, S. C. (2004). Probability judgments for continuous quantities: Linear combinations and calibration. *Management Science*, 50(5):597–604.

Hosmer, D. and Lemeshow, S. (2003). *Applied Logistic Regression*. Wiley.

Hüllermeier, E., Destercke, S., and Shaker, M. H. (2022). Quantification of credal uncertainty in machine learning: A critical analysis and empirical comparison. In *Proceedings of the Thirty-Eighth Conference on Uncertainty in Artificial Intelligence*, volume 180 of *Proceedings of Machine Learning Research*, pages 548–557. PMLR.

Hüllermeier, E. and Waegeman, W. (2021). Aleatoric and epistemic uncertainty in machine learning: An introduction to concepts and methods. *Machine Learning*, 110(3):457–506.

Kendall, A. and Gal, Y. (2017). What uncertainties do we need in Bayesian deep learning for computer vision? In *Proc. NIPS, Advances in Neural Information Processing Systems*, pages 5574–5584.

Kopetzki, A. K., Charpentier, B., Zügner, D., Giri, S., and Günnemann, S. (2021). Evaluating robustness of predictive uncertainty estimation: Are dirichlet-based models reliable? In *Proceedings of the 38th International Conference on Machine Learning, ICML*, pages 5707–5718.

Krizhevsky, A., Nair, V., and Hinton, G. (2010). Cifar-10 (canadian institute for advanced research). Technical report, Canadian Institute for Advanced Research.

Lakshminarayanan, B., Pritzel, A., and Blundell, C. (2017). Simple and scalable predictive uncertainty estimation using deep ensembles. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems*, pages 6402–6413.

Li, F., Andreetto, M., and Ranzato, M. A. (2003). Caltech101 image dataset. Technical report, California Institute of Technology.

Li, Y., Wang, S., Umarov, R., and et al. (2018). Deepre: sequence-based enzyme EC number prediction by deep learning. *BMC Bioinformatics*, 34(5):760–769.

Lichtendahl, K., Grushka-Cockayne, Y., and Winkler, R. L. (2013). Is it better to average probabilities or quantiles? *Management Science*, 59(7):1594–1611.

Malinin, A., Prokhorenkova, L., and Ustimenko, A. (2020). Uncertainty in gradient boosting via ensembles.

Nguyen, V., Shaker, M. H., and Hüllermeier, E. (2022). How to measure uncertainty in uncertainty sampling for active learning. *Mach. Learn.*, 111(1):89–122.

Ovadia, Y., Fertig, E., Ren, J., Nado, Z., Sculley, D., Nowozin, S., Dillon, J. V., Lakshminarayanan, B., and Snoek, J. (2019). Can you trust your model's uncertainty? evaluating predictive uncertainty under dataset shift. *Neurips*.

Paszke, A., Gross, S., Chintala, S., and et al. (2017). Automatic differentiation in pytorch. In *NIPS-W*.

Powell, M. J. D. (1994). *A Direct Search Optimization Method That Models the Objective and Constraint Functions by Linear Interpolation*, pages 51–67. Kluwer Academic.

Rahaman, R. and Thiery, A. (2021). Uncertainty quantification and deep ensembles. In *Advances in Neural Information Processing Systems*, volume 34, pages 20063–20075. Curran Associates, Inc.

Ranjan, R. and Gneiting, T. (2010). Combining probability forecasts. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 72(1):71–91.

RIKEN (2013). Genomic-based 16s ribosomal rna database.

Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., and Chen, L. (2018). Mobilenetv2: Inverted residuals and linear bottlenecks.

Schulz, B. and Lerch, S. (2022). Aggregating distribution forecasts from deep ensembles.

Senge, R., Bösner, S., Dembczynski, K., Haasenritter, J., Hirsch, O., Donner-Banzhoff, N., and Hüllermeier, E. (2014). Reliable classification: Learning classifiers that distinguish aleatoric and epistemic uncertainty. *Information Sciences*, 255:16–29.

Sensoy, M., Kaplan, L., and Kandemir, M. (2018). Evidential deep learning to quantify classification uncertainty. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc.

Shaker, M. and Hüllermeier, E. (2020). Aleatoric and epistemic uncertainty with random forests. In *Proc. IDA, 18th Int. Symposium on Intelligent Data Analysis*, volume 12080 of *LNCS*, pages 444–456, Konstanz, Germany. Springer.

Simonyan, K. and Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556.

Ting, K. M. and Witten, I. H. (1999). Issues in stacked generalization. *J. Artif. Int. Res.*, 10(1):271–289.

Vaicenavicius, J., Widmann, D., Andersson, C. R., Lindsten, F., Roll, J., and Schön, T. B. (2019). Evaluating model calibration in classification. In *The 22nd International Conference on Artificial Intelligence and Statistics, AISTATS*, volume 89 of *Proceedings of Machine Learning Research*, pages 3459–3467. PMLR.

Walley, P. (1991). *Statistical Reasoning with Imprecise Probabilities*. Chapman and Hall.

Widmann, D., Lindsten, F., and Zachariah, D. (2019). Calibration tests in multi-class classification: A unifying framework. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems*, pages 12236–12246.

Yang, G., Destercke, S., and Masson, M. (2014). Nested dichotomies with probability sets for multi-class classification. In *ECAI*, pages 363–368.

Zadrozny, B. and Elkan, C. (2001). Obtaining calibrated probability estimates from decision trees and naive bayesian classifiers. In *ICML*, pages 609–616.

A EXPERIMENTAL SETUP

A.1 Type I And II Error Analysis

Here we explain in detail how the ground-truth distribution is generated in scenarios S2 and S3. In both cases, the null hypothesis is false, so one needs to sample a ground-truth distribution outside the convex set. To find a distribution outside the convex set, for every $\boldsymbol{x}$, we need to find the largest $\lambda_b \in [0,1]$, such that $\boldsymbol{p}_{\lambda_b} \in S(\mathcal{P})$, with $\boldsymbol{p}_{\lambda_b} = (1-\lambda_b)\boldsymbol{p}_e + \lambda_b \boldsymbol{p}_c$, $\boldsymbol{p}_e$ the mean and $\boldsymbol{p}_c$ the randomly chosen corner. This can be calculated by means of an exhaustive line search for λ over the interval $[0,1]$ and a linear program. Pseudocode for this procedure is given by Alg. 2. After finding the boundary, one can simply sample a random distribution on the line segment between $\boldsymbol{p}_{\lambda_b}$ and $\boldsymbol{p}_c$.

Algorithm 2 findBoundary – **input:** $S(\mathcal{P})$, $\boldsymbol{p}_0$, $\boldsymbol{p}_c$, LP

$\boldsymbol{P} \leftarrow [\boldsymbol{p}^{(1)}; \ldots; \boldsymbol{p}^{(M)}]$, i.e., $\boldsymbol{P} \in [0,1]^{M \times K}$ ▷ $\boldsymbol{P}$ represents $\mathcal{P}$ in matrix-notation
$\boldsymbol{A} = [\boldsymbol{P}^T; \boldsymbol{1}_M]$ with row vector $\boldsymbol{1}_M = [1, \ldots, 1]$
$\boldsymbol{p}_{\lambda_b} \leftarrow \boldsymbol{p}_0$
for $\lambda' = 0$ to 1 **do** ▷ Begin exhaustive line search
    $\boldsymbol{p}_{\lambda'} \leftarrow (1-\lambda')\boldsymbol{p}_0 + \lambda' \boldsymbol{p}_c$
    $\boldsymbol{z} = [\boldsymbol{p}_{\lambda'}; 1]$
    if LP$(\boldsymbol{A}, \boldsymbol{z})$ finds a solution **then** ▷ Check if $\boldsymbol{p}_{\lambda'}$ falls inside the convex
        $\boldsymbol{p}_{\lambda_b} \leftarrow \boldsymbol{p}_{\lambda'}$
    else
        break ▷ We are outside the convex set, hence, break and return previous solution
return $\boldsymbol{p}_{\lambda_b}$

A.2 Calibration of probabilistic classifier sets based on deep neural networks

Table 2: Overview of of image (top) and text (bottom) datasets used in the experiments. Notation: K – number of classes, D – number of features, N – number of samples for training, calibration and test set.

Dataset	K	D	$\mathbf{N_{train}}$	$\mathbf{N_{cal.}}$	$\mathbf{N_{test}}$
CIFAR-10 (Krizhevsky et al., 2010)	10	1000	50000	4992	4992
Caltech-101 (Li et al., 2003)	97	1000	4338	2160	2160
Caltech-256 (Griffin et al., 2007)	256	1000	14890	7440	7440
PlantCLEF2015 (Goëau et al., 2015)	1000	1000	91758	10720	10720
Bacteria (RIKEN, 2013)	2659	1000	10587	1136	1136
Proteins (Li et al., 2018)	3485	1000	11830	5088	5088

We use a MobileNetV2 or VGG16 convolutional neural network (Sandler et al., 2018; Simonyan and Zisserman, 2014), pretrained on ImageNet (Deng et al., 2009), in order to obtain hidden representations for all image datasets. For the bacteria dataset, tf-idf representations are obtained by means of extracting 3-, 4- and 5-grams from the 16S rRNA sequences that were provided in the dataset (Fiannaca et al., 2018). For the proteins dataset, tf-idf representations are obtained by considering 3-grams only. Furthermore, to comply with literature, the tf-idf representations are concatenated with functional domain encodings, which contain distinct functional and evolutional information about the protein sequence (Li et al., 2018). Next, obtained feature representations for the biological datasets are passed through a single-layer neural net with 1000 output neurons and a ReLU activation function. We use the categorical cross-entropy loss by means of stochastic gradient descent with momentum, where the learning rate and momentum are set to $1e-5$ and 0.99, respectively. We set the number of epochs to 2 and 20, for the Caltech and other datasets, respectively. We train all models end-to-end on a GPU, by using the PyTorch library (Paszke et al., 2017) and infrastructure with the following specifications:

- **CPU:** i7-6800K 3.4 GHz (3.8 GHz Turbo Boost) – 6 cores / 12 threads,
- **GPU:** 2x Nvidia GTX 1080 Ti 11GB + 1x Nvidia Tesla K40c 11GB,
- **RAM:** 64GB DDR4-2666.

B ADDITIONAL EXPERIMENTS

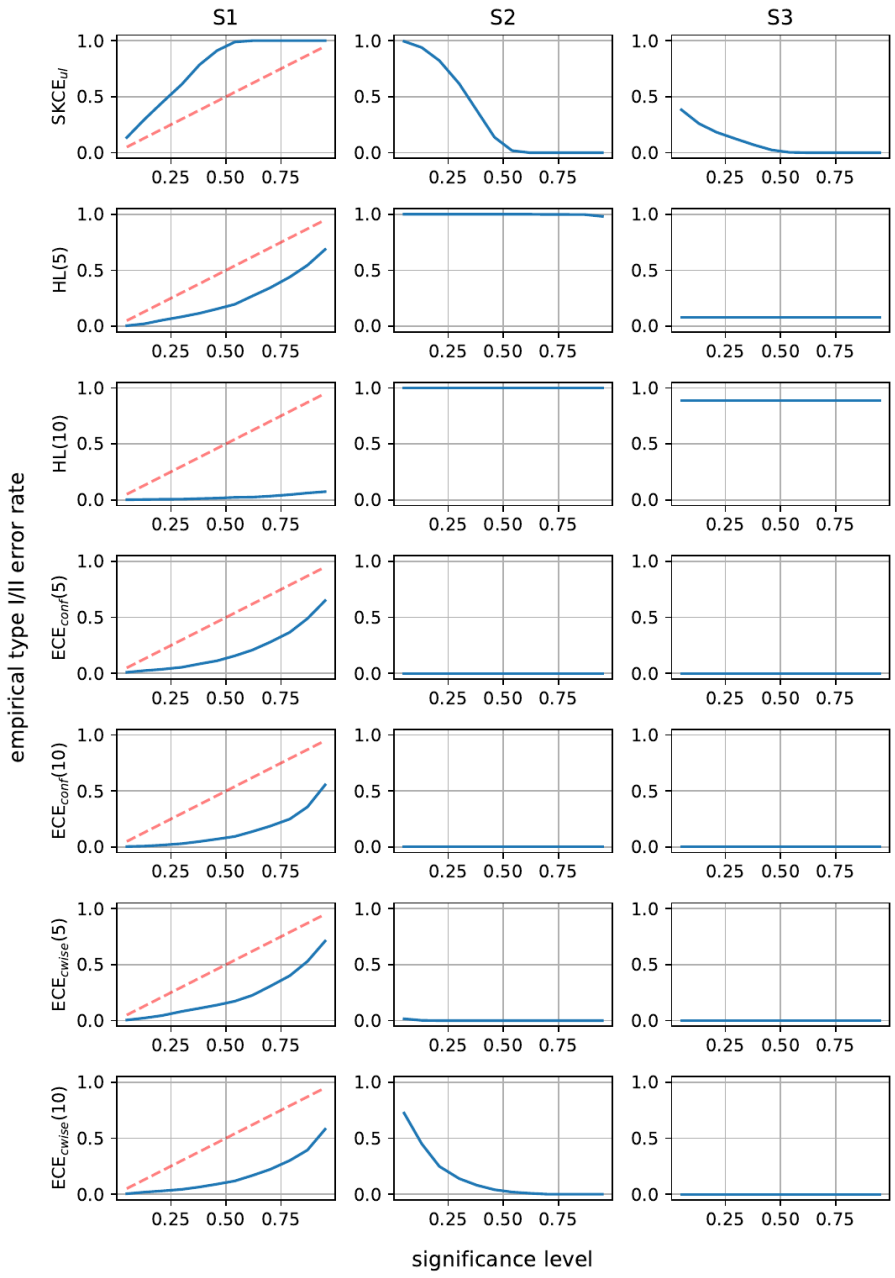

Figure 2: Empirical Type I (S1) and Type II (S2, S3) error for different calibration measures in function of the significance level for $R = 1000$ randomly sampled datasets from S1, S2 and S3 and with $N = 100, M = 10, K = 10$ and $u = 0.1$. For all tests we use $D = 100$ bootstrap samples.

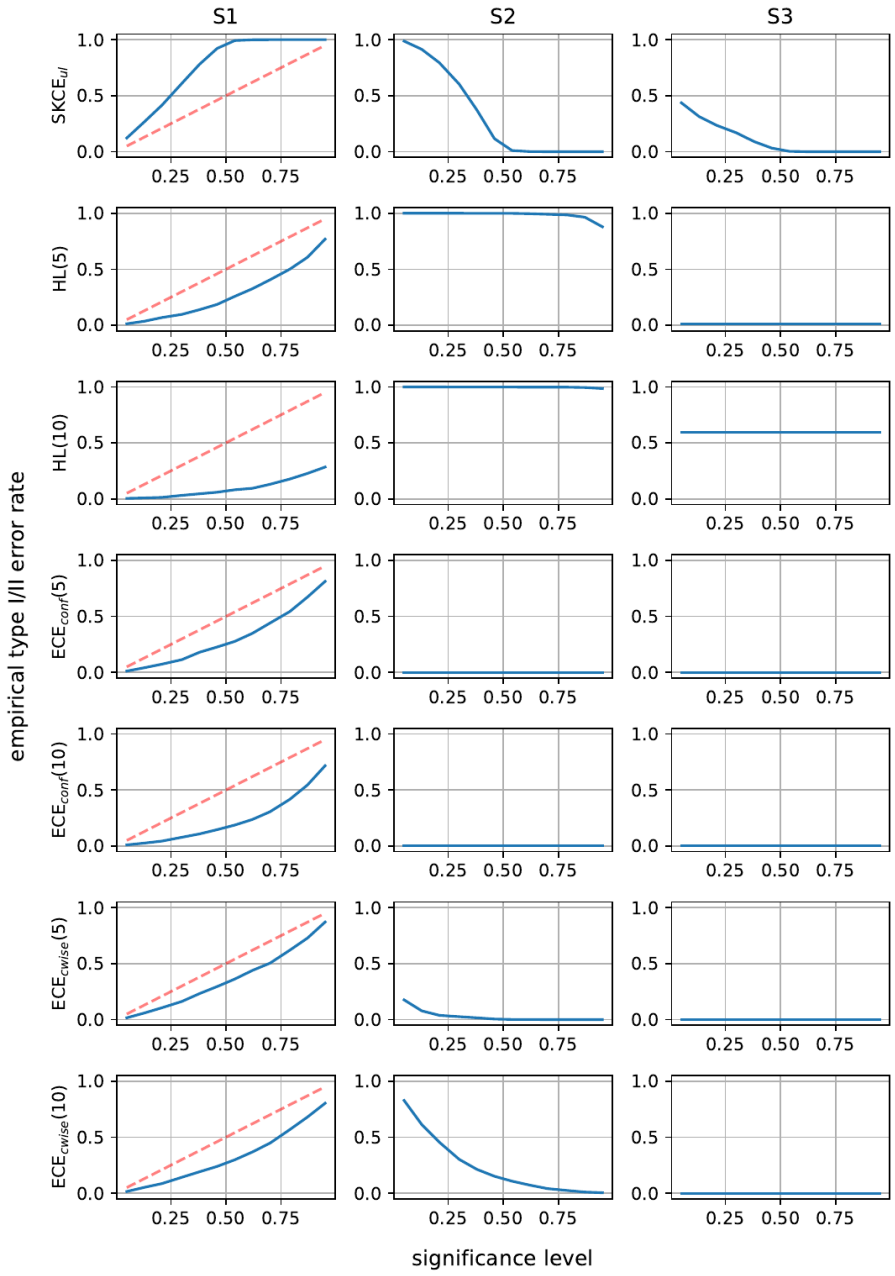

Figure 3: Empirical Type I (S1) and Type II (S2, S3) error for different calibration measures in function of the significance level for $R = 1000$ randomly sampled datasets from S1, S2 and S3 and with $N = 100, M = 100, K = 10$ and $u = 0.01$. For all tests we use $D = 100$ bootstrap samples.

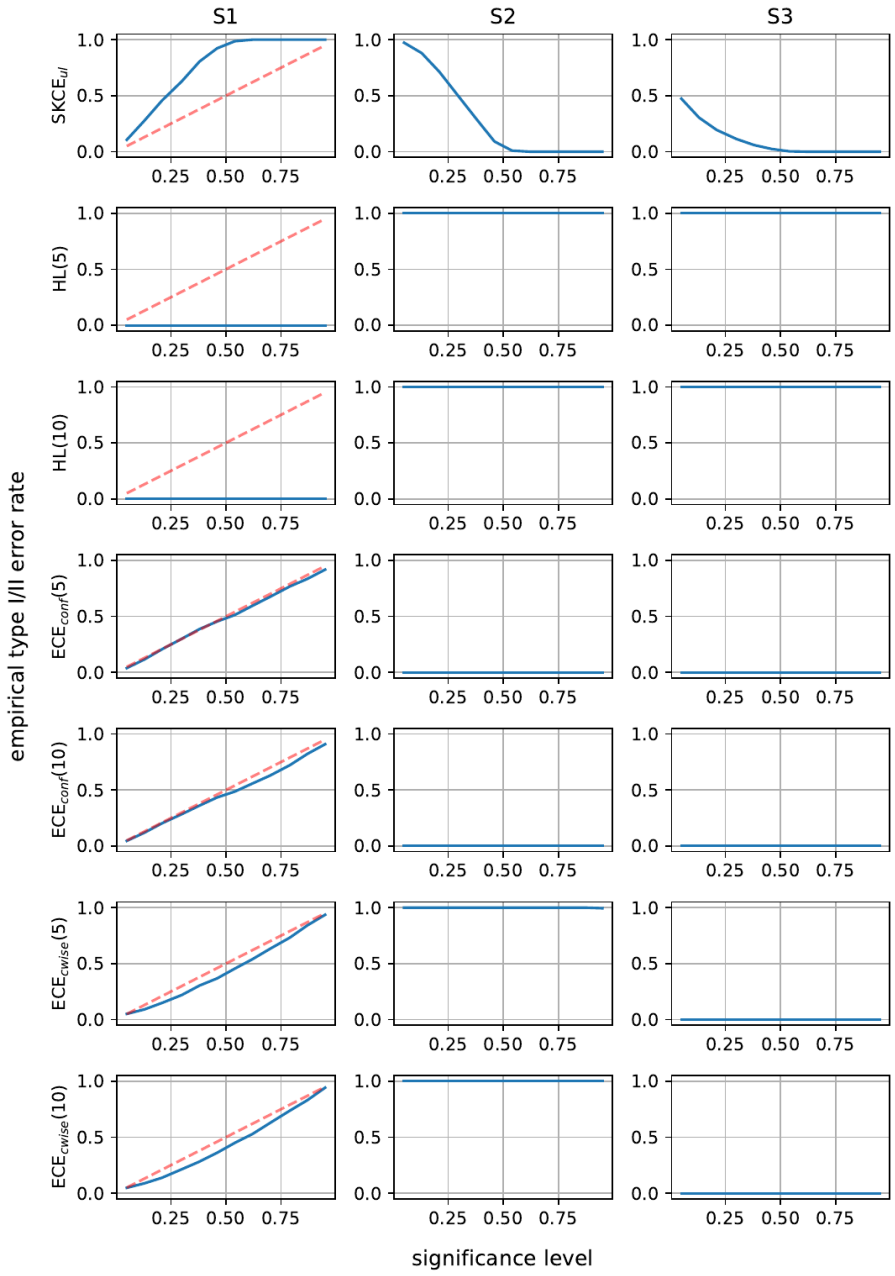

Figure 4: Empirical Type I (S1) and Type II (S2, S3) error for different calibration measures in function of the significance level for $R = 1000$ randomly sampled datasets from S1, S2 and S3 and with $N = 100, M = 10, K = 100$ and $u = 0.01$. For all tests we use $D = 100$ bootstrap samples.