Scalable Utility-Aware Multiclass Calibration

Anonymous Author(s)
Affiliation
Address
email

Abstract

Ensuring that classifiers are well-calibrated, i.e., their predictions align with observed frequencies, is a minimal and fundamental requirement for classifiers to be viewed as trustworthy. Existing methods for assessing multiclass calibration often focus on specific aspects associated with prediction (e.g., top-class confidence, class-wise calibration) or utilize computationally challenging variational formulations. We instead propose *utility calibration*, a general framework designed to evaluate model calibration directly through the lens of downstream applications. This approach measures the calibration error relative to a specific *utility function* that encapsulates the goals or decision criteria relevant to the end user. As such, utility calibration provides a task-specific perspective on reliability. We demonstrate how this framework can *unify and re-interpret several existing calibration metrics*, particularly allowing for more robust versions of the top-class and class-wise calibration metrics, and to go beyond such binarized approaches, towards assessing calibration for richer classes of downstream utilities.

1 Introduction

Calibration is a fundamental property of probabilistic predictors. A calibrated model produces predictions that, on average, align with observed frequencies. For instance, if a weather forecaster predicts a 30% chance of rain on a given day, rain should occur on approximately 30% of such days. In multiclass classification problems, calibration ensures that the predicted probabilities reflect the true likelihood of each class. Formally, let $\mathcal{X}$ denote the input space, $\mathcal{Y} = \{e_1, \dots, e_C\}$ the output space, where e_i is the i-th canonical basis vector in $\mathbb{R}^C$, and $\Delta^{C-1} := \left\{x \in \mathbb{R}^C_+ | \sum_i x_i \leq 1\right\}$ denote the simplex in $\mathbb{R}^C$. A predictor $f : \mathcal{X} \to \Delta^{C-1}$ is said to be perfectly calibrated with respect to a distribution D over $\mathcal{X} \times \mathcal{Y}$ if $\mathbb{E}[Y \mid f(X)] = f(X)$. The most direct metric for quantifying the deviation from perfect calibration is the Mean Calibration Error (MCE).

Definition 1.1 (Mean Calibration Error)**.** *For a distribution D such that $(X, Y) \sim D$ and a predictor f, the mean calibration error is defined as* $\mathrm{MCE}(f) := \mathbb{E}\left[\|\mathbb{E}[Y \mid f(X)] - f(X)\|_2\right]$.

Without further assumptions, the MCE is fundamentally impossible to estimate, even in the binary setting [1, 2]. While assumptions like Hölder continuity of $\mathbb{E}\left[Y|f(X)\right]$ allow for consistent estimators of $\mathbb{E}\left[Y|f(X)\right]$ or minimax optimal tests for $\mathrm{MCE}(f)$ [1, 3, 4], their sample complexity scales exponentially with the dimension C, making MCE estimation intractable in high dimensions.

Due to the difficulty of measuring MCE, multiple relaxations are proposed, falling into two main categories: *binarized* and *variational*. First, binarized approaches [5–7] simplify the problem by focusing on specific binary events derived from the multiclass predictions, e.g. top-class or class-wise calibration. However, these methods are by nature presumptive of downstream tasks. Moreover, their reliance on binning schemes or kernel estimators for the underlying binary subproblems introduce sensitivity to estimator choices and can suffer from high bias [8]. Second, variational approaches [9–14] assess calibration through optimization problems, such as the distance to the nearest perfectly

Submitted to 39th Conference on Neural Information Processing Systems (NeurIPS 2025). Do not distribute.

calibrated predictor or the worst-case error against a class of witness functions. Unfortunately, these methods can be computationally intensive and can scale poorly as the number of classes C increases.

To address these limitations and provide an application-focused perspective on calibration, we introduce *utility calibration*. This framework evaluates a model f by considering a downstream user who employs its predictions $f(X)$. The core idea is to measure calibration error relative to a specific *utility function*, denoted u, which encapsulates the goals, costs, or decision criteria relevant to this end user. Utility calibration then assesses how well the *expected utility* (as estimated by the user based on $f(X)$ and u) aligns with the *realized utility* (obtained when the true outcome Y is observed). In practice, models often serve diverse users or a single user with multiple objectives. We thus extend utility calibration to handle *classes of utility functions*. The overall utility calibration for a class $\mathcal{U}$ can be defined as the worst-case error over $u \in \mathcal{U}$, denoted $\mathrm{UC}(f, \mathcal{U})$. A notable aspect of this class-based formulation is that it provides a structured way to express and analyze various existing calibration notions. In particular, by defining appropriate utility functions within $\mathcal{U}$, concepts such as top-class and class-wise calibration can be cast within the utility calibration framework. This offers a unified perspective and a superior alternative to binning for examining those notions of calibration.

Contributions and outline: In Section 2, we review related literature on calibration metrics and post-hoc calibration methods. In Section 3, we define utility calibration and relate it to existing measures of calibration. In addition, we demonstrate how this framework can be used to frame several existing calibration concepts within a common utility-centric perspective, offering consistent interpretations and providing examples of relevant utility classes. To characterize the difficulty of achieving utility calibration for classes of utility functions, we introduce the notions of *proactive* and *interactive* measurability. While, for rich utility classes, proactive measurability is not possible, we show that interactive measurability is achievable for many classes of interest. Drawing on these insights, we empirically demonstrate the application of our proposed metrics and evaluation methodology, in Section 4, to that end, we formulate a practical and scalable methodology for evaluating calibration against interactively measurable utility classes in Section 4.

Notation: For any vector $w \in \mathbb{R}^C$, w_i denotes its i-th component and $\gamma(w) \coloneqq \mathrm{argmax}_i w_i$. For a probability vector $p \in \Delta^{C-1}$, we write $Z \sim p$ to denote a categorical random variable Z taking values in $\mathcal{Y} = \{e_1, \ldots, e_C\}$ such that $\mathbb{P}\{Z = e_i\} = p_i$, where e_i is the i-th canonical basis vector. We use $\mathbb{1}\{\cdot\}$ for the indicator function. $\mathbb{E}[\cdot]$ denotes expectation, which is taken typically w.r.t. $(X, Y) \sim D$ and, for $k \in \mathbb{N}_+$, $[k] = \{1, \ldots, k\}$. Finally, for $a, b \in \mathbb{R}$ with $a < b$, we denote $\mathbb{I}[a, b]$ to be the set of closed interval subsets of $[a, b]$.

2 Related Work

In this section, we review three classical and related approaches to measuring or ensuring a form of calibration, namely binarized relaxations, variational approaches, and post-hoc calibration methods.

First, *binarized relaxations* aim to circumvent the difficulty of measuring the calibration error of a high-dimensional predictor f by measuring the MCE of a single or multiple downstream binary versions of f instead. Two commonly used relaxations are the Top-Class calibration Error (TCE) [7] and the Class-Wise calibration Error (CWE) [6], which are respectively defined as

$$\mathrm{TCE}(f) \coloneqq \mathbb{E}\left[\left|\mathbb{E}[\mathbb{1}\left\{Y = e_{\gamma(f(X))}\right\} | f(X)_{\gamma(f(X))}] - f(X)_{\gamma(f(X))}\right|\right],$$
$$\mathrm{CWE}(f) \coloneqq \sum_{i \in [C]} w_i \, \mathbb{E}\left[\left|\mathbb{E}[\mathbb{1}\{Y = e_i\} | f(X)_i] - f(X)_i\right|\right],$$

where w_i is a class-dependent weight, which can be set to $1/C$, $w_i = \mathbb{P}\{Y = e_i\}$, or another choice. Typically, TCE and CWE are estimated using binning schemes. Concretely, for $(B_j)_{j \in [m]}$ m disjoint subsets of $[0, 1]$ such that $\cup_{j \in [m]} B_j = [0, 1]$, we consider the following binned estimators

$$\mathrm{TCE}^{\mathrm{bin}}(f) = \sum_{j \in [m]} \left|\mathbb{E}\left[\left(f(X)_{\gamma(f(X))} - \mathbb{1}\left\{Y = e_{\gamma(f(X))}\right\}\right) \mathbb{1}\left\{f(X)_{\gamma(f(X))} \in B_j\right\}\right]\right|, \quad (2.1)$$

$$\mathrm{CWE}^{\mathrm{bin}}(f) = \sum_{i \in [C]} \sum_{j \in [m]} w_i \left|\mathbb{E}\left[\left(f(X)_i - \mathbb{1}\{Y = e_i\}\right) \mathbb{1}\{f(X)_i \in B_j\}\right]\right|. \quad (2.2)$$

Gupta and Ramdas [5] unified multiple instances of binarized proxies of MCE, such as TCE, CWE and topK confidence calibration, introduced in [15], and proposed additional binarized reductions

which offer stronger notions of calibration. Unfortunately, the binning schemes used in such binarized proxies are known to have a large effect on the estimated error [8, 16]. Apart from the simpler equal-size bins [7] and equal-weight bins [17], multiple binning schemes built on top of different heuristics have been proposed [see, e.g., 8, 18–20]. Gupta and Ramdas [21] showed a simple equal-weight binning scheme with better sample complexity guarantees for estimating bin averages. Kumar et al. [22] developed adaptive binning schemes with guarantees for discrete f and showed that for any binning scheme, there exists a worst-case continuous f such that the bias of $\mathrm{TCE}^{\mathrm{bin}}(f)$ as an estimate of $\mathrm{TCE}(f)$ is lower bounded by 0.49 (noting that by construction TCE is bounded between 0 and 1). On the other hand, there exist binning-free alternatives for binarized reductions [see, e.g., 3, 15]. Nonetheless, in an assumption-free setting, it is generally impossible to consistently estimate the MCE of binary predictors [1, 2, 23]. As such, it is generally difficult to control the calibration error defined by binarized relaxations.

Second, *variational approaches* do not strictly aim to measure the MCE. Instead, they consider alternative formulations that do not require direct estimation of the conditional expectation. For example, Distance to Calibration (DC) quantifies the calibration error of a predictor f as the distance between f and the nearest perfectly calibrated predictors [10]:

$$\mathrm{DC}(f) \coloneqq \inf_{\mathrm{MCE}(g)=0} \mathbb{E}\big[\|f(X) - g(X)\|_1\big].$$

A unified formulation of variational measures of calibration is weighted calibration, which assesses the calibration error against a class of witness functions [9]. Concretely, let $\mathcal{W}$ be a class of functions mapping Δ^{C-1} to $[-1,1]^C$. Then, weighted calibration error with witness class $\mathcal{W}$ is

$$\mathrm{CE}_{\mathcal{W}}(f) \;=\; \sup_{w\in\mathcal{W}} \mathbb{E}_{X,Y}\left[\big\langle w\big(f(X)\big), f(X) - Y\big\rangle\right]. \tag{2.3}$$

A specific instance of weighted calibration is the Kernel Calibration Error (KCE) [24], which sets $\mathcal{W}$ to be the unit ball of the reproducing kernel Hilbert space (RKHS) of a multivariate universal kernel. This allows for efficient computation of the supremum but it remains hard to interpret the impact of low KCE for a user of f. Błasiok et al. [10] showed that in the binary setting, $\mathrm{DC}(f)$ and $\mathrm{CE}_{\mathrm{Lip}(1)}(f)$ are equivalent up to a (low-degree) polynomial scaling, where $\mathrm{Lip}(1)$ is the class of 1-Lipschitz functions from Δ^{C-1} to $[-1,1]$. In addition, the authors proved that, for the binary setting, $\mathrm{CE}_{\mathrm{Lip}(1)}(f)$ can be well approximated by the RKHS of the Laplace kernel allowing for efficient assessment of $\mathrm{DC}(f)$ using a calibration metric originally proposed by Kumar et al. [12].

The result on the equivalence between $\mathrm{CE}_{\mathrm{Lip}(1)}(f)$ and $\mathrm{DC}(f)$ was further extended to the multiclass setting in [2, Theorem 15.5.5] and [11, Lemma 3.3]. In particular, Gopalan et al. [25] showed that measuring either $\mathrm{DC}(f)$ or $\mathrm{CE}_{\mathrm{Lip}(1)}(f)$ requires an exponential number of samples with respect to C [11, Theorem 3.2. and Theorem 3.4.]. Thus, even though $\mathrm{DC}\,(f)$ can be efficiently assessed in the binary setting, it is quickly intractable as the dimension increases.

A particular case is *Decision calibration*, introduced by Zhao et al. [14], that tailors calibration guarantees to downstream decision-making tasks. A predictor f is considered decision calibrated of order K if, for any decision problem involving at most K actions, the expected loss computed using the model's predictions $f(X)$ accurately matches the true expected loss incurred. Formally, for any loss function ℓ mapping an outcome-action pair to a real-valued loss, decision calibration of order K requires:

$$\mathbb{E}\Big[\ell\big(\hat{Y}, \delta(f(X))\big)\Big] = \mathbb{E}\Big[\ell(Y, \delta(f(X)))\Big],$$

where $\hat{Y} \sim f(X)$ and δ is a decision rule that picks the best action among K actions under the model's prediction $f(X)$. This ensures that decision-makers can reliably estimate the consequences of their choices when using the predictor. A key contribution of Zhao et al. [14] is showing that decision calibration of order K can be achieved by having $\sup_{p\in P(K)} \|\,\mathbb{E}[(Y - f(X))\mathbb{1}\{f(X)\in p\}]\| = 0$, where $P(K)$ is the set of polytopes with at most K supporting hyperplanes. Unfortunately, computational complexity is again an issue—Gopalan et al. [11] showed that even for $K = 2$ the computational complexity of measuring decision calibration is exponential with respect to C.

In summary, practitioners are faced with a dilemma in assessing the calibration error. On one hand, for binarized approaches, it is generally impossible to have consistent estimation of the calibration error of the binary subproblems. In addition, by preemptively only assessing specific binary subproblems, they are fundamentally presumptive of the downstream usage of the model. On the other hand,

variational approaches can offer more robust and well-motivated assessment of the calibration error but they are computationally infeasible as the dimension grows.

Independently, *post-hoc calibration* refers to techniques applied to a pre-trained model's outputs to improve the alignment between its predicted probabilities and the true likelihood of outcomes, without altering the original model parameters. Such methods are advantageous as they decouple the calibration process from the training process.

Common post-hoc calibration methods often adjust the model's outputs; popular examples include Temperature Scaling and its multi-parameter extensions, Vector Scaling and Matrix Scaling [7], which may all be regarded as a multiclass extension of Platt's scaling [26]. Dirichlet calibration assumes the model's predicted probability vectors can be modeled by a Dirichlet distribution, whose parameters are learned on a calibration set to transform the original probabilities [27]. Nonparametric methods such as Histogram Binning [17] and Isotonic Regression [28] learn calibration maps by discretizing the probability space or fitting monotonic (order-preserving) functions, respectively. Other methods also include: [18], which applies a specific binning strategy followed by recalibration to minimize class-wise calibration error, [29], which uses order-preserving transformations for recalibration to maintain accuracy. Finally, a related body of literature aims to improve calibration by changing or regularizing the training objective, e.g. [30, 3, 31, 12].

3 Utility Calibration

We consider the following utility-centric formulation of calibration. In particular, we are interested in the setting, where for some input X, a downstream user leverages $f(X)$ as an estimation of $\mathbb{E}[Y|X]$. Based on this estimation of the conditional expectation, the user may then take arbitrary actions or decisions. Finally, the user observes the true realization of the label Y and based on this realization, may then suffer some loss or achieve some gain. To model such a pipeline of observation, action, then consequences, we consider a utility function $u : \Delta^{C-1} \times \mathcal{Y} \to [-1, 1]$ such that $u(f(X), Y)$ models the reward obtained or the loss suffered by the decision-makers after using $f(X)$ to take arbitrary actions/decisions. In such a setting, predictability is highly desirable, in the sense that when using the predictor f, the utility obtained is similar to the utility expected. More concretely, for $\hat{Y} \sim f(X)$ and a given input X, the user can use $f(X)$ to construct the following estimate of utility:

$$v_u(X) \coloneqq \mathbb{E}\left[u(f(X), \hat{Y})|X\right] = \langle f(X), \vec{u}(X)\rangle\,, \tag{3.1}$$

where $\vec{u} : \mathcal{X} \to [-1, 1]^C$ is defined as $\vec{u}(X) \coloneqq (u(f(X), e_i))_{i\in C}$. Ideally, we want the function $v_u(X)$ to be an unbiased estimator of the true utility. As such, we define the utility calibration with respect to a utility function u as

$$\mathrm{UC}(f, u) \coloneqq \sup_{I\in\mathbb{I}[-1,1]} |\mathbb{E}\left[(u(f(X), Y) - v_u(X))\,\mathbb{1}\left\{v_u(X) \in I\right\}\right]| \tag{3.2}$$

and say that f is ε-calibrated with respect to a utility function u if $\mathrm{UC}(f, u) \le \varepsilon$. Note that for any $I = [a, b]$, the inner optimization problem in (3.2) can be rewritten as

$$\left|\mathbb{E}\left[u(f(X), Y) - u(f(X), \hat{Y})\middle|v_u(X) \in [a, b]\right]\right| \mathbb{P}\left\{v_u(X) \in [a, b]\right\}.$$

In words, looking at the instances where $v_u(X) \in [a, b]$, the bias between the utility the decision-maker expects to get (while using $f(X)$ to take decisions and to estimate the utility) and the actual utility the decision-maker achieves (when using $f(X)$ to take decisions), is at most ε after being weighted by the probability of the event $\{v_u(X) \in [a, b]\}$.

Combining (3.1) and (3.2) above, one obtains that $\mathrm{UC}(f, u)$ is equivalent to

$$\mathrm{UC}(f, u) = \sup_{I\in\mathbb{I}[-1,1]} |\mathbb{E}\left[\langle Y - f(X), \vec{u}(X)\mathbb{1}\left\{v_u(X) \in I\right\}\rangle\right]|\,. \tag{3.3}$$

Thus, utility calibration is equivalent to weighted calibration (2.3), with the witness class $\mathcal{W}$ set to $\mathcal{W}(u) \coloneqq \{x \mapsto \xi\vec{u}(x)\mathbb{1}\left\{v_u(x) \in I\right\} | I \in \mathbb{I}[-1, 1]\}$. In addition, our notion of utility calibration requires that the predicted label $\hat{Y} \sim f(X)$ can be used for an unbiased estimation of the utility. This is related to Outcome Indistinguishability (OI) [32], where a predictor f is considered reliable if its simulated outcomes $\hat{Y} \sim f(X)$ are computationally indistinguishable from Nature's true outcomes Y. We also note that this perspective also connects to recent work that leverages OI variants to establish links between loss minimization guarantees, omnipredictors, and multicalibration [33–35].

3.1 Decision-Theoretic Implications of Utility Calibration

In a very recent work, for the binary classification setting, Rossellini et al. [23] introduced the CutOff calibration metric, which assesses the calibration error by measuring against the worst-case bin, and demonstrated that it provide robust decision-theoretic guarantees. We defer a more detailed discussion of CutOff calibration to Appendix B.1. By assessing the $\mathrm{UC}(f, u)$ on the worst-case interval of $v_u(\cdot)$, our construction of utility calibration can be seen as a generalization of CutOff calibration to multiple dimensions and arbitrary utility functions, and that in fact inherits analogous decision-theoretic guarantees to the one shown in Rossellini et al. [23, Prop 2.1 and 3.2].

In particular, consider a decision rule based on thresholding the predicted utility $v_u(X)$ at some level $t_0 \in [-1, 1]$, i.e., taking the action $\hat{U}_{t_0} := \mathbb{1}\{v_u(X) \geq t_0\}$. This models the situation in which a user needs to commit a binary decision after estimating the utility using $f(X)$. Then, the quality of this decision can be assessed by the loss $\ell_{\mathrm{util}}(\tilde{u}, \hat{U}; t) = |\tilde{u} - t|\, \mathbb{1}\{\hat{U} \neq \mathbb{1}\{u \geq t\}\}$, which penalizes the *deviation* between the true utility u_Y and the decision threshold t_0 when a mismatch between $\hat{U}_{t_0}$ and the ideal decision occurs. Consequently, let $R_{\mathrm{util}}(g; t_0) = \mathbb{E}[\ell_{\mathrm{util}}(u(f(X), Y), \hat{U}_{t_0}; t_0)]$ be the associated risk. Then, we show that the decision process $\hat{U}_{t_0}$ cannot significantly be improved by any simple post-processing of $v_u(\cdot)$ through a composition with a monotone function.

Proposition 3.1 (Utility Risk Gap)**.** *Let $u : \Delta^{C-1} \times \mathcal{Y} \to [-1, 1]$ be a utility function and $v_u(X)$ be the predicted expected utility. For any threshold $t_0 \in [-1, 1]$ and the loss function ℓ_{util} as described above,*

$$R_{\mathrm{util}}(v_u(X); t_0) - \inf_{\substack{h:[-1,1]\to[-1,1] \\ \text{monotone}}} R_{\mathrm{util}}(h(v_u(X)); t_0) \leq 2\mathrm{UC}(f, u).$$

In words, Proposition 3.1 indicates that, if f is utility calibrated, in such a binary decision-making scenario, the user can barely benefit from any monotonic post-processing to v_u. Another interpretation of $v_u(X)$ is as a regressor for the realized utility $u_Y := u(f(X), Y) \in [-1, 1]$. Similar to Rossellini et al. [23, Prop 2.1], we can show that the regressor v_u satisfies a notion of calibration itself. First, note that distance from calibration naturally extends to such a single-dimension regression problem by considering a function $g_u(X)$ to be a perfectly calibrated predictor of u_Y if $\mathbb{E}[u_Y \mid g_u(X)] = g_u(X)$ almost surely. We denote this extended notion of distance from calibration as $\mathrm{DCU}(f, u)$, the Distance to Calibrated Utility Predictor for $v_u(X)$ with respect to the realized utility $u(f(X), Y)$:

$$\mathrm{DCU}(f) := \inf_{\substack{g_u:\mathcal{X}\to[-1,1] \\ \mathbb{E}[u_Y \mid g_u(X)] = g_u(X)}} \mathbb{E}\left|g_u(X) - v_u(X)\right| .$$

We show that $\mathrm{DCU}(f, u)$ can be effectively controlled through $\mathrm{UC}(f, u)$.

Proposition 3.2 (Utility Calibration upper Bounds DCU)**.** *Let $u : \Delta^{C-1} \times \mathcal{Y} \to [-1, 1]$ be a utility function. Then,*

$$\mathrm{DCU}(f) \leq \sqrt{8\mathrm{UC}(f, u)} + \mathrm{UC}(f, u).$$

Proposition 3.2 implies that if $\mathrm{UC}(f, u)$ is small, then $v_u(X)$, seen as a regressor for the true utility $u(f(X), Y)$, is a calibrated predictor itself. This further strengthens the interpretation of $\mathrm{UC}(f, u)$: not only does it *ensure actionable decisions based on* $v_u(X)$, but it also *guarantees that* $v_u(X)$ *itself is not far from calibration*. We thus turn to the question of how to estimate $\mathrm{UC}(f, u)$.

3.2 Measuring $\mathrm{UC}(f, u)$

A naturally arising question is on the difficulty of measuring and achieving a small utility calibration error. We show in Lemma 3.3 that both the computational and sample complexity of estimating $\mathrm{UC}(f, u)$ are generally feasible and of limited dependence on the dimension, allowing its scalability to predictors with thousands of classes.

Lemma 3.3 (Estimating Utility Calibration Against a Single Function)**.** *Let $u : \Delta^{C-1} \times \mathcal{Y} \to [-1, 1]$ be a fixed utility function and $f : \mathcal{X} \to \Delta^{C-1}$ be a given predictor. Define the empirical estimator $\widehat{\mathrm{UC}}(f, u; S)$ based on n i.i.d. samples $S = \{(X_i, Y_i)\}_{i=1}^n \sim D^n$ as*

$$\widehat{\mathrm{UC}}(f, u; S) := \sup_{I \in \mathbb{I}[-1,1]} \left| \frac{1}{n} \sum_{i=1}^{n} \left[(u(f(X_i), Y_i) - V(X_i))\mathbb{1}\{V(X_i) \in I\}\right] \right| .$$

Then, for any $\delta > 0$, with probability at least $1 - \delta$ over the draws of the sample S,

$$|\widehat{\mathrm{UC}}(f, u; S) - \mathrm{UC}(f, u)| \leq \tilde{O}\left(\sqrt{\frac{\log(1/\delta)}{n}}\right). \tag{3.4}$$

Furthermore, $\widehat{\mathrm{UC}}(f, u; S)$ can be computed from S in $O(n^2 + nT_{eval})$ time, where T_{eval} is the time to evaluate $f(X_i)$ and $u(\cdot, \cdot)$.

First, we note that the constants hidden in the $\tilde{O}(\cdot)$ in (3.4) are dimension-independent. Similarly, the only dimension-dependent term in the computational complexity is T_{eval}. As such, $\mathrm{UC}(f, u)$ is a completely scalable notion of calibration, allowing it to be implemented for classifier with a thousand classes – as exemplified in Section 4. In addition, given that $\mathrm{UC}(f, u)$ can be formulated as weighted calibration (see eq. (3.3)) and that $\widehat{\mathrm{UC}}(f, u; S)$ is both a computationally and sample efficient, we can leverage the common patching-style post-hoc calibration algorithm, eg: [9, 36, 2] to recalibrate f in order to minimize $\mathrm{UC}(f, u)$ while decreasing its Brier score. We summarize this fact informally in Lemma 3.4 and defer to a more detailed discussion and experimental evaluation of the recalibration patching algorithm in Appendix A.

Lemma 3.4 (Informal)**.** *For $\varepsilon > 0$, there exists an algorithm, which given a classifier $f : \mathcal{X} \to \mathcal{Y}$, outputs a recalibrated classifier $\tilde{f} : \mathcal{X} \to \mathcal{Y}$ such that $\mathrm{UC}(\tilde{f}, u) \leq \varepsilon$ and its Brier score decreases:*

$$\mathbb{E}\left[\|\tilde{f}(X) - Y\|_2^2\right] \leq \mathbb{E}\left[\|f(X) - Y\|_2^2\right].$$

Those encouraging facts on the utility calibration w.r.t. a single u being established, we next turn out attention to Utility Calibration against a function classe $\mathcal{U}$.

3.3 Utility Calibration against a Function Class

In many real-world scenarios, a single probabilistic predictor f might serve multiple downstream users, or a single user might employ it under varying conditions or objectives. The exact utility function relevant at the time of decision-making may not be known beforehand by the model provider, or it might even change over time (e.g., due to changing costs, available actions, or strategic goals), or might be fundamentally user-dependent.

Therefore, ensuring reliability often requires guarantees that hold not just for a single, pre-specified utility function, but for an entire class of plausible or relevant utility functions, denoted by $\mathcal{U}$. This provides a more robust assurance that the model's predictions are trustworthy across a range of potential downstream applications. To capture this requirement, overloading the notion, we define utility calibration against a function class as the worst-case performace over the class, i.e.

$$\mathrm{UC}(f, \mathcal{U}) = \sup_{u \in \mathcal{U}} \mathrm{UC}(f, u). \tag{3.5}$$

To illustrate the practical relevance of this concept, we exhibit hereafter several examples of utility classes, each motivated by different downstream tasks. We first demonstrate how to recover similar notions to top-class (2.1) and class-wise (2.2) using the framework of utility calibration (3.5).

Example 3.5 (Top-Class and Class-Wise Utilities ($\mathcal{U}_{\mathrm{TCE}}, \mathcal{U}_{\mathrm{CWE}}$))**.** *Define the top-class utility function $u_{\mathrm{top}}(p, y) = \mathbb{1}\{y = e_{\gamma(p)}\}$, where we recall that $\gamma(p) = \arg\max_k p_k$, and the class-wise utility function for class $c \in [C]$ as $u^c(p, y) = \mathbb{1}\{y = e_c\}$. The corresponding utility classes are respectively $\mathcal{U}_{\mathrm{TCE}} = \{u_{\mathrm{top}}\}$ and $\mathcal{U}_{\mathrm{CWE}} = \{u^c,\ c \in [C]\}$. It results in defining:*

$$\mathrm{UC}(f, \mathcal{U}_{\mathrm{TCE}}) = \sup_{I \in \mathbb{I}[0,1]} \left|\mathbb{E}\left[\left(\mathbb{1}\left\{Y = e_{\gamma(f(X))}\right\} - f(X)_{\gamma(f(X))}\right) \mathbb{1}\left\{f(X)_{\gamma(f(X))} \in I\right\}\right]\right|,$$

$$\mathrm{UC}(f, \mathcal{U}_{\mathrm{CWE}}) = \sup_{c \in [C]} \sup_{I \in \mathbb{I}[0,1]} \left|\mathbb{E}\left[\left(\mathbb{1}\{Y = e_c\} - f(X)_c\right) \mathbb{1}\{f(X)_c \in I\}\right]\right|.$$

In contrast to the binned estimators $\mathrm{TCE}^{\mathrm{bin}}$ (2.1) and $\mathrm{CWE}^{\mathrm{bin}}$ (2.2), utility calibration with the classes $\mathcal{U}_{\mathrm{TCE}}$ and $\mathcal{U}_{\mathrm{CWE}}$ offers a more robust, binning-free, computable assessment. Specifically, $\mathrm{UC}(f, \mathcal{U}_{\mathrm{TCE}})$ and $\mathrm{UC}(f, \mathcal{U}_{\mathrm{CWE}})$ are determined by maximizing the calibration deviation over *any* possible interval $I \subseteq [0, 1]$ (and additionally over classes for $\mathcal{U}_{\mathrm{CWE}}$), effectively identifying the

worst-case interval-based error. This approach inherently avoids fixed binning schemes, thereby circumventing pathologies where bin choices drastically alter estimated errors [8, 22]. Consequently, for any binning scheme using m bins, $m \cdot \mathrm{UC}(f, \mathcal{U}_{\mathrm{TCE}})$ and $m \cdot \mathrm{UC}(f, \mathcal{U}_{\mathrm{CWE}})$ upper bound $\mathrm{TCE}^{\mathrm{bin}}(f)$ and $\mathrm{CWE}^{\mathrm{bin}}(f)$ respectively, while the converse is not true. We refer to Appendix B.2 for the formal statement. Furthermore, by Proposition 3.1, a small $\mathrm{UC}(f, \mathcal{U}_{\mathrm{TCE}})$ guarantees that decisions based on thresholding top-class confidence are robust to monotonic recalibration, and by Proposition 3.2 that this confidence is a calibrated predictor of actual top-class accuracy. Analogous guarantees hold for $\mathrm{UC}(f, \mathcal{U}_{\mathrm{CWE}})$ for individual class confidences, offering assurances for downstream applications.

Beyond the binarized perspectives offered by $\mathcal{U}_{\mathrm{TCE}}$ and $\mathcal{U}_{\mathrm{CWE}}$, the utility calibration framework readily accommodates richer and more complex classes of utility functions. This allows us to move beyond presumptive binary events and consider more nuanced downstream applications. In particular, consider settings where the utility derived from an outcome Y is intrinsic to the outcome itself, independent of the model's prediction $f(X)$. For example, in medical diagnosis, the cost or severity tied to a specific disease $Y = e_j$ might be a fixed value a_j, irrespective of the diagnostic prediction. Formally, such situations can be modeled using a utility function $u_a : \Delta^{C-1} \times \mathcal{Y} \to [-1, 1]$ defined by a payoff vector $a \in [-1, 1]^C$, where utility function and the expected utility are respectively $u_a(\cdot, e_j) = a_j$ and $v_{u_a}(X) = \langle f(X), a \rangle$, with a_j represents the utility if the true outcome is e_j.

Example 3.6 (Linear Utilities ($\mathcal{U}_{\mathrm{lin}}$))**.** *Define the class of linear utilities as* $\mathcal{U}_{\mathrm{lin}} \coloneqq \{u_a \mid a \in [-1, 1]^C\}$*, noting that the predicted utility* $v_{u_a}(X)$ *is linear in the prediction* $f(X)$*.*

A small $\mathrm{UC}(f, \mathcal{U}_{\mathrm{lin}})$ ensures that for any payoff vector a, the predicted expected utility $v_{u_a}(X)$, as a regressor of the realized utility, is close to calibration.

Alternatively, in applications like information retrieval or recommender systems, the realized utility depends on the rank assigned to the true outcome $Y = e_j$. Given a model's prediction $p = f(X)$, assuming $p_1, \ldots, p_C$ are distinct (or that ties are broken arbitrarily/randomly among equal coordinates), the rank of class j, denoted $\mathrm{rank}(p, j)$, is its position across p, i.e. $\mathrm{rank}(p, j) \coloneqq \sum_{i \in [C]} \mathbb{1}\{p_j \leq p_i\}$. Using a valuation vector $\theta \in [-1, 1]^C$, a rank-based utility function can then be constructed as $u_\theta(p, e_j) = \theta_{\mathrm{rank}(p,j)}$ with the associated expected utility function $v_{u_\theta}(X) = \sum_{i=1}^{C} f(X)_i \theta_{\mathrm{rank}(f(X),i)}$. Calibrating for such utilities ensures the model's expected rank-based performance aligns with reality. A prominent special case is topK utility, where the valuation vector $\theta^{(K)}$ for a given $K \in [C]$ is defined such that $\theta_r^{(K)} = 1$ if $r \leq K$ and $\theta_r^{(K)} = 0$ if $r > K$.

Example 3.7 (Rank-Based and Top-K Utilities ($\mathcal{U}_{\mathrm{rank}}, \mathcal{U}_{\mathrm{topK}}$))**.** *The class of general rank-based utilities is* $\mathcal{U}_{\mathrm{rank}} \coloneqq \{u_\theta \mid \theta \in [-1, 1]^C\}$*. The class of top-K utilities is then* $\mathcal{U}_{\mathrm{topK}} \coloneqq \{u_{\theta^{(K)}} \mid K \in [C]\}$*, where* $\theta_r^{(K)} = \mathbb{1}\{r \leq K\}$*. Equivalently,* $u_K(p, e_j) = \mathbb{1}\{\mathrm{rank}(p, j) \leq K\}$*. A small* $\mathrm{UC}(f, \mathcal{U}_{\mathrm{rank}})$ *(or* $\mathrm{UC}(f, \mathcal{U}_{\mathrm{topK}})$*) ensures reliable prediction for general rank (or specifically top-K accuracy) valuations, validating the model's ranking capabilities.*

As discussed in Section 2, decision calibration [14] ensures that for problems with up to K actions, the model's predicted utility for its recommended action matches the actual realized utility. We can frame a similar guarantee within utility calibration. For any bounded loss function $l : \mathcal{Y} \times [K] \to [-1, 1]$ and a prediction $p = f(X)$, the optimal action is $\delta_l(p) = \arg\min_{a \in [K]} \mathbb{E}_{\hat{Y} \sim p}[l(\hat{Y}, a)]$. The utility function is then $u_l(p, y) = -l(y, \delta_l(p))$, representing the negative loss from outcome y under action $\delta_l(p)$. The predicted expected utility is $v_{u_l}(X) = -\mathbb{E}_{\hat{Y} \sim f(X)}[l(\hat{Y}, \delta_l(f(X)))|X]$.

Example 3.8 (Decision Calibration Utilities ($\mathcal{U}_{\mathrm{dec},K}$))**.** *Let* $\mathcal{L}_K = \{l : \mathcal{Y} \times [K] \to [-1, 1]\}$ *be the class of all bounded* K*-action loss functions, and the utility class is* $\mathcal{U}_{\mathrm{dec},K} \coloneqq \{u_l, l \in \mathcal{L}_K\}$*. A small* $\mathrm{UC}(f, \mathcal{U}_{\mathrm{dec},K})$ *implies that for any* K*-action decision problem* $l \in \mathcal{L}_K$*, the model's prediction of expected utility for its chosen action* $\delta_l(f(X))$ *reliably reflects the achieved utility* $-l(Y, \delta_l(f(X)))$*.*

These aforementioned examples illustrate that calibrating against classes $\mathcal{U}$ provides guarantees tailored to diverse user needs, moving beyond simplistic binarized assessments. A critical question then arises: how can $\mathrm{UC}(f, \mathcal{U})$ be measured for a given class $\mathcal{U}$, which we address in the next section.

3.4 Measurability of utility calibration

Estimating $\sup_{u \in \mathcal{U}} \mathrm{UC}(f, u)$ in (3.5) presents two key challenges: the *computational complexity* of the optimization, and the *sample complexity* required for the empirical supremum to converge to its

true value. We introduce the two notions of proactive and interactive measurability to decouple these two aspects.

Definition 3.9 (Proactive Measurability)**.** *The utility calibration error w.r.t. class $\mathcal{U}$ is proactively measurable if there exists an algorithm A and polynomial functions N_{poly}, T_{poly} duch that for any $\varepsilon, \delta > 0$ and $n \geq N_{poly}(C, 1/\varepsilon, 1/\delta)$ samples $S \sim D^n$, algorithm $A(S)$ outputs $\hat{u}$ satisfying $|\mathrm{UC}(f, \hat{u}) - \mathrm{UC}(f, \mathcal{U})| \leq \varepsilon$ with probability at least $1 - \delta$ and the runtime of $A(S)$ is bounded by $T_{poly}(C, n)$.*

Generally, for a finite class $\mathcal{U}$, if $|\mathcal{U}|$ grows polynomially in C then by Lemma 3.3 we can guarantee proactive measurability. Nonetheless, even for simple infinite classes such as $\mathcal{U}_{\mathrm{lin}}$, proactive measurability reduces to a non-convex optimization problem that cannot be generally solved in polynomial time. In fact, even aiming for a weaker notion, namely *improper auditing*, Gopalan et al. [11] showed that assessing both weaker and stronger notions than $\mathrm{UC}(f, \mathcal{U}_{\mathrm{lin}})$ cannot be done in polynomial time in both the error ε^{-1} and the dimension C [11, Theorem 1.3, Theorem 5.2, and Theorem 8.6]. A more detailed description of Gopalan et al. [11] hardness results is in Appendix B.3. The primary bottleneck is the *computation time*. Next, we thus propose an alternative criteria of measurability that decouples the statistical guarantee from the computational complexity of verifying the supremum.

Definition 3.10 (Interactive Measurability)**.** *The utility calibration error w.r.t. class $\mathcal{U}$ is interactively measurable if there exists an estimator $\widehat{\mathrm{UC}}(f, u; S)$ and a polynomial function N_{poly} such that for $n \geq N_{poly}(C, 1/\varepsilon, 1/\delta)$ samples $S \sim D^n$, it holds with probability at least $1 - \delta$ that*

$$\sup_{u \in \mathcal{U}} |\widehat{\mathrm{UC}}(f, u; S) - \mathrm{UC}(f, u)| \leq \varepsilon.$$

Interactive measurability represents a much more achievable goal. For example, while decision calibration is computational hard to measure, Zhao et al. [14] showed that it admits polynomial sample complexity. In Appendix B.4, we further demonstrate the interactive measurability of different utility classes of interest with controlled Rademacher complexity.

In summary, while proactively measuring the worst-case utility calibration error $\mathrm{UC}(f, \mathcal{U}) = \sup_{u \in \mathcal{U}} \mathrm{UC}(f, u)$ is often computationally prohibitive for expressive utility classes $\mathcal{U}$, interactive measurability allows for efficient estimation of $\mathrm{UC}(f, u)$ uniformly for any *specific* $u \in \mathcal{U}$. Next, we leverage this distinction to propose a scalable evaluation methodology that, instead of pursuing the intractable worst-case error, characterizes the *distribution* of utility calibration errors across $\mathcal{U}$. This provides a more nuanced understanding of a model f's calibration reliability over a spectrum of potential downstream applications, that we then evaluate in experiments.

4 Scalable Evaluation of Utility Calibration and Experiment

Scalable Evaluation of Utility Calibration. Our approach considers a probability distribution $\mathcal{D}_{\mathcal{U}}$ over the utility class $\mathcal{U}$. Many utility classes of interest admit a finite-dimensional parameterization, making sampling from $\mathcal{D}_{\mathcal{U}}$ practical. We sample M utility functions $\{u_m\}_{m=1}^M$ from $\mathcal{D}_{\mathcal{U}}$ and, for each u_m, compute its estimated error $\widehat{E}_{m,n} := \widehat{\mathrm{UC}}(f, u_m; S)$ using n data points from a sample S. These M error estimates then form an *empirical Cumulative Distribution Function (eCDF)*,

$$\widehat{F}_{E,M,n}(e) := \tfrac{1}{M} \textstyle\sum_{m=1}^{M} \mathbb{1}\{\widehat{E}_{m,n} \leq e\},$$

which serves as an empirical proxy for the true CDF, $F_E(e) := \mathbb{P}_{u \sim \mathcal{D}_{\mathcal{U}}}(\mathrm{UC}(f, u) \leq e)$. We provide guarantees on the difference between $F_E(e)$ and $\hat{F}_{E,M,n}(e)$ in Appendix B.5.

In particular, $\mathcal{U}_{\mathrm{lin}}$ (Example 3.6) and $\mathcal{U}_{\mathrm{rank}}$ (Example 3.7) both admit finite-dimension parameterization. For $\mathcal{U}_{\mathrm{lin}}$, we construct $\mathcal{D}_{\mathcal{U}_{\mathrm{lin}}}$ by sampling the payoff vectors a uniformly in $[-1, 1]^C$. Meanwhile, for $\mathcal{U}_{\mathrm{rank}}$, we also sample from $\mathcal{D}_{\mathcal{U}_{\mathrm{rank}}}$ by uniformly sampling valuation vectors $\theta \in [-1, 1]^C$, which satisfy $\theta_1 \geq \theta_2 \geq \cdots \geq \theta_C$. This is to reflect a rational preference for better ranks, i.e. the higher the rank of the true realization within the predictions of $f(X)$, the higher the utility.

Numerical experiments. We now demonstrate how our approach can be used to empirically validate model calibration. For all of our experiments, we used pretrained models for ImageNet and CIFAR10/100 [37, 38]. In Appendix D, we further detail our experimental setup, provide additional results, and list the licenses of all the assets used. Here, we present the results of two

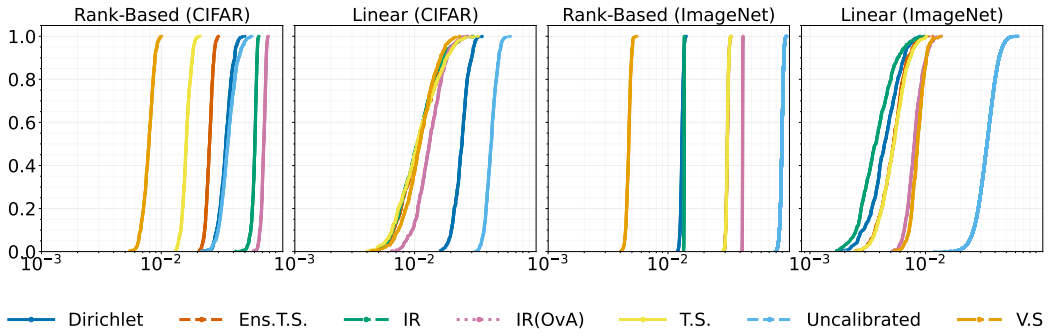

Figure 1: eCDF of utility calibration errors for `ResNet20` on CIFAR100 (left two panels) and `ViT` on ImageNet-1K (right two panels).

settings: (1) `ResNet20` [39] on CIFAR100 and a Vision Transformer `ViT` [40] on ImageNet-1K. For post-hoc calibration, we applied Temperature Scaling (T.S.) [26], Vector Scaling (V.S.) [41], Ensemble Temperature Scaling (Ens. T.S.) [42], and Dirichlet recalibration [27]. In addition, we fitted a shared Isotonic Regression (I.R.) [28] across different classes and an Isotonic Regression for each class using one-vs-all approach (IR OvA).

In Table 1, we present a detailed comparison for the `ResNet20` model on CIFAR100. This table compares standard metrics (accuracy, Brier score), binned binarized metrics ($\mathrm{TCE}_{\mathrm{binned}}$, $\mathrm{CWE}_{\mathrm{binned}}$ with 15 equal-weight bins), and our utility calibration metrics for specific utility classes: top-class ($\mathcal{U}_{\mathrm{TCE}}$), class-wise ($\mathcal{U}_{\mathrm{CWE}}$), and top-$K$ ($\mathcal{U}_{\mathrm{TopK}}$). As expected, most post-hoc methods improve Brier scores and reduce binned error over the uncalibrated model, often with minimal accuracy impact. Our binning-free utility calibration metrics, $\mathcal{U}_{\mathrm{TCE}}$, $\mathcal{U}_{\mathrm{CWE}}$, and $\mathcal{U}_{\mathrm{TopK}}$, show similar improvements. Notably, while $\mathcal{U}_{\mathrm{TCE}}$ and $\mathcal{U}_{\mathrm{TopK}}$ are equal for the uncalibrated model, they can diverge for calibrated models. Since $\mathcal{U}_{\mathrm{TopK}}$ considers all $K \in [C]$, it upper-bounds $\mathcal{U}_{\mathrm{TCE}}$ (the $K = 1$ case). Although calibration methods reduce $\mathcal{U}_{\mathrm{TCE}}$ effectively, the typically higher $\mathcal{U}_{\mathrm{TopK}}$ values can reveal miscalibration for ranks beyond top-1. This suggests $\mathcal{U}_{\mathrm{TopK}}$ as a more comprehensive benchmark.

Beyond specific utility functions, Figure 1 displays the eCDFs of utility calibration errors for broader utility classes: rank-based ($\mathcal{U}_{\mathrm{rank}}$) and linear ($\mathcal{U}_{\mathrm{lin}}$). Each eCDF, generated from $M = 1000$ sampled utility functions, shows the proportion of utilities for which the calibration error is below a certain threshold; thus, curves shifted to the left indicate superior calibration across the wider class of utility functions. For the `ResNet20` on CIFAR100 (left panels), the eCDFs reveal interesting dynamics. While most post-hoc methods improved upon the uncalibrated model for $\mathcal{U}_{\mathrm{lin}}$, some methods, specifically I.R. and I.R.(OvA), surprisingly worsened performance for $\mathcal{U}_{\mathrm{rank}}$ compared to the uncalibrated model. This degradation was not apparent from the specific metrics in Table 1, underscoring the necessity of the broader perspective offered by these eCDF plots across a class of utilities. For the Vision Transformer (`ViT`) on ImageNet-1K (right panels), the uncalibrated model exhibits the poorest performance across both $\mathcal{U}_{\mathrm{rank}}$ and $\mathcal{U}_{\mathrm{lin}}$. Nevertheless, the eCDF plots still provide a nuanced way to compare and evaluate different post-hoc methods against each other.

In conclusion, utility calibration provides a robust, unified, and application-centric framework for evaluating classifier reliability. Its specific instantiations, $\mathcal{U}_{\mathrm{CWE}}$ and $\mathcal{U}_{\mathrm{TCE}}$, offer superior, binning-free alternatives to traditional metrics with actionable guarantees, while $\mathcal{U}_{\mathrm{TopK}}$ presents an even more comprehensive ranking assessment. Furthermore, the eCDF plots across broader utility classes deliver crucial nuanced insights into model behavior that single-metric evaluations obscure.

Table 1: `ResNet20`-CIFAR100 calibration results. Comparison of post-hoc methods using Accuracy, binned ECEs ($\mathrm{TCE}_{\mathrm{eqBin}}$, $\mathrm{CWE}_{\mathrm{eqBin}}$), and utility calibration errors: $\mathcal{U}_{\mathrm{TCE}}$ (Top-Class), $\mathcal{U}_{\mathrm{CWE}}$ (Class-Wise), $\mathcal{U}_{\mathrm{topK}}$ (Top-K). Mean $\pm$ maximum devaition over 5 splits.

Method	Accuracy	Brier Score	$\mathrm{CWE}_{\mathrm{binned}}$	$\mathrm{TCE}_{\mathrm{binned}}$	$\mathcal{UC}_{\mathrm{CWE}}$	$\mathcal{UC}_{\mathrm{TCE}}$	$\mathcal{UC}_{\mathrm{TopK}}$
Uncalibrated	0.677 ± 0.010	0.480 ± 0.015	0.00214 ± 0.00016	0.1600 ± 0.008	0.0124 ± 0.0011	0.1590 ± 0.015	0.1590 ± 0.015
Dirichlet	0.666 ± 0.010	0.457 ± 0.008	0.00194 ± 0.00014	0.0727 ± 0.0160	0.0111 ± 0.0004	0.0709 ± 0.0165	0.0818 ± 0.0154
IR	0.677 ± 0.010	0.444 ± 0.011	0.00186 ± 0.00006	0.0264 ± 0.0033	0.0113 ± 0.0005	0.0310 ± 0.0071	0.0756 ± 0.0086
IR (OvA)	0.674 ± 0.010	0.454 ± 0.011	0.00156 ± 0.00016	0.0454 ± 0.0103	0.0108 ± 0.0011	0.0467 ± 0.0190	0.0927 ± 0.0091
T.S.	0.677 ± 0.010	0.440 ± 0.014	0.00188 ± 0.00008	0.0250 ± 0.0066	0.0114 ± 0.0005	0.0322 ± 0.0090	0.0367 ± 0.0046
Ens.T.S	0.677 ± 0.010	0.440 ± 0.010	0.00196 ± 0.00006	0.0212 ± 0.0045	0.0114 ± 0.0005	0.0304 ± 0.0063	0.0393 ± 0.0056
V.S.	0.680 ± 0.010	0.435 ± 0.010	0.00150 ± 0.00010	0.0334 ± 0.0117	0.0107 ± 0.0010	0.0375 ± 0.0148	0.0403 ± 0.0121

References

[1] Donghwan Lee, Xinmeng Huang, Hamed Hassani, and Edgar Dobriban. T-cal: An optimal test for the calibration of predictive models. *Journal of Machine Learning Research*, 24(335):1–72, 2023.

[2] John C. Duchi. Information theory and statistics. https://web.stanford.edu/class/stats311/lecture-notes.pdf, 2024. Lecture Notes for STATS 311 / EE 377, Stanford University. Version from March 12, 2024. Accessed: April 30, 2025.

[3] Teodora Popordanoska, Raphael Sayer, and Matthew Blaschko. A consistent and differentiable lp canonical calibration error estimator. *Advances in Neural Information Processing Systems*, 35:7933–7946, 2022.

[4] Alexandre B Tsybakov. Nonparametric estimators. *Introduction to Nonparametric Estimation*, pages 1–76, 2009.

[5] Chirag Gupta and Aaditya Ramdas. Top-label calibration and multiclass-to-binary reductions. In *International Conference on Learning Representations*, 2022.

[6] Michael Panchenko, Anes Benmerzoug, and Miguel de Benito Delgado. Class-wise and reduced calibration methods. In *2022 21st IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 1093–1100. IEEE, 2022.

[7] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. In *International conference on machine learning*, pages 1321–1330. PMLR, 2017.

[8] Rebecca Roelofs, Nicholas Cain, Jonathon Shlens, and Michael C Mozer. Mitigating bias in calibration error estimation. In *International Conference on Artificial Intelligence and Statistics*, pages 4036–4054. PMLR, 2022.

[9] Christopher Jung, Changhwa Lee, Mallesh Pai, Aaron Roth, and Rakesh Vohra. Moment multicalibration for uncertainty estimation. In *Conference on Learning Theory*, pages 2634–2678. PMLR, 2021.

[10] Jarosław Błasiok, Parikshit Gopalan, Lunjia Hu, and Preetum Nakkiran. A unifying theory of distance from calibration. In *Proceedings of the 55th Annual ACM Symposium on Theory of Computing*, pages 1727–1740, 2023.

[11] Parikshit Gopalan, Lunjia Hu, and Guy N Rothblum. On computationally efficient multi-class calibration. *arXiv preprint arXiv:2402.07821*, 2024.

[12] Aviral Kumar, Sunita Sarawagi, and Ujjwal Jain. Trainable calibration measures for neural networks from kernel mean embeddings. In *International Conference on Machine Learning*, pages 2805–2814. PMLR, 2018.

[13] David Widmann, Fredrik Lindsten, and Dave Zachariah. Calibration tests in multi-class classification: A unifying framework. *Advances in neural information processing systems*, 32, 2019.

[14] Shengjia Zhao, Michael Kim, Roshni Sahoo, Tengyu Ma, and Stefano Ermon. Calibrating predictions to decisions: A novel approach to multi-class calibration. *Advances in Neural Information Processing Systems*, 34:22313–22324, 2021.

[15] Kartik Gupta, Amir Rahimi, Thalaiyasingam Ajanthan, Thomas Mensink, Cristian Sminchisescu, and Richard Hartley. Calibration of neural networks using splines. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=eQe8DEWNN2W.

[16] Sebastian Gruber and Florian Buettner. Better uncertainty calibration via proper scores for classification and beyond. *Advances in Neural Information Processing Systems*, 35:8618–8632, 2022.

[17] Bianca Zadrozny and Charles Elkan. Obtaining calibrated probability estimates from decision trees and naive bayesian classifiers. In *Icml*, volume 1, pages 609–616, 2001.

[18] Kanil Patel, William Beluch, Bin Yang, Michael Pfeiffer, and Dan Zhang. Multi-class uncertainty calibration via mutual information maximization-based binning. *arXiv preprint arXiv:2006.13092*, 2020.

[19] Mahdi Pakdaman Naeini, Gregory Cooper, and Milos Hauskrecht. Obtaining well calibrated probabilities using bayesian binning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 29, 2015.

[20] Jeremy Nixon, Michael W Dusenberry, Linchuan Zhang, Ghassen Jerfel, and Dustin Tran. Measuring calibration in deep learning.

[21] Chirag Gupta and Aaditya Ramdas. Distribution-free calibration guarantees for histogram binning without sample splitting. In *International conference on machine learning*, pages 3942–3952. PMLR, 2021.

[22] Ananya Kumar, Percy S Liang, and Tengyu Ma. Verified uncertainty calibration. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/f8c0c968632845cd133308b1a494967f-Paper.pdf.

[23] Raphael Rossellini, Jake A Soloff, Rina Foygel Barber, Zhimei Ren, and Rebecca Willett. Can a calibration metric be both testable and actionable? *arXiv preprint arXiv:2502.19851*, 2025.

[24] Zhen Lin, Shubhendu Trivedi, and Jimeng Sun. Taking a step back with KCal: Multi-class kernel-based calibration for deep neural networks. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=p_jIy5QFB7.

[25] Parikshit Gopalan, Michael P Kim, Mihir A Singhal, and Shengjia Zhao. Low-degree multicalibration. In *Conference on Learning Theory*, pages 3193–3234. PMLR, 2022.

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

[27] Meelis Kull, Miquel Perello Nieto, Markus Kängsepp, Telmo Silva Filho, Hao Song, and Peter Flach. Beyond temperature scaling: Obtaining well-calibrated multi-class probabilities with dirichlet calibration. *Advances in neural information processing systems*, 32, 2019.

[28] Bianca Zadrozny and Charles Elkan. Transforming classifier scores into accurate multiclass probability estimates. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 694–699, 2002.

[29] Amir Rahimi, Amirreza Shaban, Ching-An Cheng, Richard Hartley, and Byron Boots. Intra order-preserving functions for calibration of multi-class neural networks. *Advances in Neural Information Processing Systems*, 33:13456–13467, 2020.

[30] Jishnu Mukhoti, Viveka Kulharia, Amartya Sanyal, Stuart Golodetz, Philip Torr, and Puneet Dokania. Calibrating deep neural networks using focal loss. *Advances in neural information processing systems*, 33:15288–15299, 2020.

[31] Charlie Marx, Sofian Zalouk, and Stefano Ermon. Calibration by distribution matching: Trainable kernel calibration metrics. *Advances in Neural Information Processing Systems*, 36, 2024.

[32] Cynthia Dwork, Michael P. Kim, Omer Reingold, Guy N. Rothblum, and Gal Yona. Outcome indistinguishability. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2021, page 1095–1108, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450380539. doi: 10.1145/3406325.3451064. URL https://doi.org/10.1145/3406325.3451064.

[33] Parikshit Gopalan, Adam Tauman Kalai, Omer Reingold, Vatsal Sharan, and Udi Wieder. Omnipredictors. In *13th Innovations in Theoretical Computer Science Conference (ITCS 2022)*, pages 79–1. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2022.

[34] Parikshit Gopalan, Lunjia Hu, Michael P Kim, Omer Reingold, and Udi Wieder. Loss minimization through the lens of outcome indistinguishability. In *14th Innovations in Theoretical Computer Science Conference (ITCS 2023)*, pages 60–1. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2023.

[35] Parikshit Gopalan, Michael Kim, and Omer Reingold. Swap agnostic learning, or characterizing omniprediction via multicalibration. *Advances in Neural Information Processing Systems*, 36: 39936–39956, 2023.

[36] Ursula Hébert-Johnson, Michael Kim, Omer Reingold, and Guy Rothblum. Multicalibration: Calibration for the (computationally-identifiable) masses. In *International Conference on Machine Learning*, pages 1939–1948. PMLR, 2018.

[37] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[38] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images, 2009.

[39] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[40] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=YicbFdNTTy.

[41] Meelis Kull, Telmo M Silva Filho, and Peter Flach. Beyond sigmoids: How to obtain well-calibrated probabilities from binary classifiers with beta calibration. *Electronic Journal of Statistics*, 11:5052–5080, 2017.

[42] Jize Zhang, Bhavya Kailkhura, and T Yong-Jin Han. Mix-n-match: Ensemble and compositional methods for uncertainty calibration in deep learning. In *International conference on machine learning*, pages 11117–11128. PMLR, 2020.

[43] Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. Foundations of machine learning. adaptive computation and machine learning, 2018.

[44] Francis Bach. *Learning theory from first principles*. MIT press, 2024.

[45] Andreas Maurer. A vector-contraction inequality for rademacher complexities. In *Algorithmic Learning Theory: 27th International Conference, ALT 2016, Bari, Italy, October 19-21, 2016, Proceedings 27*, pages 3–17. Springer, 2016.

NeurIPS Paper Checklist

IMPORTANT, please:

- **Delete this instruction block, but keep the section heading "NeurIPS Paper Checklist"**,
- **Keep the checklist subsection headings, questions/answers and guidelines below.**
- **Do not modify the questions and only use the provided macros for your answers**.

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The main aim of the paper is to present a unified framework for assessing calibration that allows for recovering similar notions to existing metrics while circumventing some the difficulties/limitations in assessing them. In addition, it allows going beyond binarized reductions and developing scalable assessment against infinite class through CDF curves. We present the framework, cite the literature to highlight the limititations of existing approaches, and provide proofs in the Appendix.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [NA]

   Justification: The paper aims to introduce a new perspective on evaluating calibration. There are many limitations related to calibration: it is an easy notion to satisfy by trivial predictors, it is hard to measure, it is not the strongest guarantee for trustworthy deployment of machine learning models. Nonetheless, we believe that those limitations are inherit to the underlying problem rather than to the paper itself. Other aspects of the paper can be seen as limitations. For example, proactive measurability is hard, so we propose assessing infinite classes through a distributional approach. We find it hard to judge whether the hardness of proactive measurability is in itself a limitation or not, making this question hard to answer.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.

- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: Proofs included in the Appendix.

   Guidelines:

   - The answer NA means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We used pretrained models for reproducibility. More detailed description of the experimental setup and additional results are available in the appendix.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.

- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: Code and detailed instructions are available in the supplemental material. We intend to open source the code upon acceptance.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: We used pretrained accessible models and standard datasets. Additional hyperparameter are further specified in the appendix.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.

- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We report results over multiple splits using the mean and the maximum deviation from it. We also include standard deviation in the Appendix.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
   - If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Computational resources used are detailed in Appendix D.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes]

   Justification: We satisfy NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.

- If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

    Justification: This work applies to general classifiers and is not specifically tied to particular applications.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: The paper poses no such risk.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: We cited the original assets. The licenses of the assets used are detailed in Appendix D.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [Yes]

Justification: We use standard datasets and accessible open-source pretrained models. Other aspects of the experiments are implemented in code and included in the supplemental material.

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [NA]

Justification: No crowdsourcing or research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [NA]

Justification: No crowdsourcing or research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [NA]

Justification: The paper does not tackle LLMs.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (https://neurips.cc/Conferences/2025/LLM) for what should or should not be described.