Robust Calibration with Multi-domain Temperature Scaling

Anonymous Author(s)
Affiliation
Address
email

Abstract

Uncertainty quantification is essential for the reliable deployment of machine learning models to high-stakes application domains. Uncertainty quantification is all the more challenging when training distribution and test distribution are different, even the distribution shifts are mild. Despite the ubiquity of distribution shifts in real-world applications, existing uncertainty quantification approaches mainly study the in-distribution setting where the train and test distributions are the same. In this paper, we develop a systematic calibration model to handle distribution shifts by leveraging data from multiple domains. Our proposed method—multi-domain temperature scaling—uses the heterogeneity in the domains to improve calibration robustness under distribution shift. Through experiments on three benchmark data sets, we find our proposed method outperforms existing methods as measured on both in-distribution and out-of-distribution test sets.

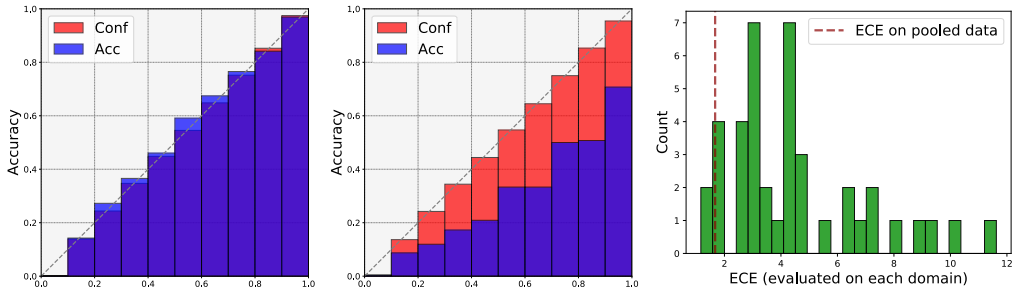
1 Introduction

To make learning systems reliable and fault-tolerant, predictions must be accompanied by uncertainty estimates. A significant challenge to accurately codifying uncertainty is the distribution shift that typically arises over the course of a system's deployment [35]. For example, suppose health providers from 20 different hospitals employ a model to make diagnostic predictions from fMRI data. The distributions across hospitals could be quite different as a result of differing patient populations, machine conditions, and so on. In such a setting, it is critical to provide uncertainty quantification that is valid for *every* hospital—not just on average across all hospitals. Going even further, our uncertainty quantification should be informative when a new 21st hospital goes online, even if the distribution shifts from those already encountered. In this work, we study calibration in the multi-domain setting. We find that by requiring accurate calibration across all observed domains, our method provides more accurate uncertainty quantification on unseen domains.

Calibration is a core topic in learning [33, 28, 9, 24, 12, 3], but most techniques are targeted at settings with no distribution shift. To see this, we consider a simple experiment on the ImageNet-C [14] dataset, which consists of 76 domains. Here, each domain corresponds to one type of data corruption applied with a certain severity. We apply the temperature scaling technique [12] on the pooled data from all domains. In Figure 1(a) and 1(b), we display the reliability diagrams for the pooled data and for one individual domain. We find that even under a relatively mild distribution shift—i.e., subpopulation shift from the mixture of all domains to the single domain—temperature scaling does not produce calibrated confidence estimates on the stand-alone domain. This behavior is pervasive; in Figure 1(c), we see that the calibration on individual domains is much worse than the the reliability diagram from the pooled data would suggest.

To address this issue, we develop a new algorithm, multi-domain temperature scaling, that leverages multi-domain structure in the data. Our algorithm takes a base model and learns a calibration function

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.



(a) Evaluated on pooled data. (b) Evaluated on one domain. (c) ECE evaluated on every domain.

Figure 1: Reliability diagrams and expected calibration error histograms for temperature scaling with a ResNet-50 on ImageNet-C. We use temperature scaling to obtain adjusted confidences for the ResNet-50 model. **(a)** Reliability diagram evaluated on the pooled data of ImageNet-C. **(b)** Reliability diagram evaluated on data from one domain (Gaussian corruption with severity 5) in ImageNet-C. **(c)** Calibration evaluated on every domain in ImageNet-C as well as the pooled ImageNet-C (measured in ECE, lower is better).

that maps each input to a different temperature parameter that is used for adjusting confidence in the base model. Empirically, we find our algorithm significantly outperforms temperature scaling on three real-world multi-domain datasets. In particular, in contrast to temperature scaling, our proposed algorithm is able to provide well-calibrated confidence on each domain. Moreover, our algorithm largely improves robustness of calibration under distribution shifts. This is expected, because if the calibration method performs well on every domain, it is likely to have learned some structure that generalizes to unseen domains. Theoretically, we analyze the multi-domain calibration problem in the regression setting, providing guidance about the conditions under which robust calibration is possible.

Contributions. The main contributions of our work are as follows: Algorithmically, we develop a new calibration method that generalizes the widely used temperature scaling concept from single-domain to multi-domain. The proposed new method exploits multi-domain structure in the data distribution, which enables model calibration on every domain. We conduct detailed experiments on three real-world multi-domain datasets and demonstrate that our method significantly outperforms existing calibration methods on *both in-distribution domains and unseen out-of-distribution domains*. Theoretically, we study multi-domain calibration in the regression setting and develop a theoretical understanding of robust calibration in this setting.

Related Work

Calibration methods. There is a large literature on calibrating the well-trained machine learning models, including histogram binning [45], isotonic regression [46], conformal prediction [39], Platt scaling [33], and temperature scaling [12]. These calibration methods apply a validation set and post-process the model outputs. As shown in Guo et al. [12], temperature scaling, a simple method that uses a single (temperature) parameter for rescaling the logits, performs surprisingly well on calibrating confidences for deep neural networks. We focus on this approach in our work. More broadly, there has been much recent work develop methods to improve calibration for deep learning models, including augmentation-based training [37, 17], calibration for neural machine translation [23], neural stochastic differential equation [22], self-supervised learning [16], ensembling [24], and Bayesian neural networks [9, 10], as well as statistical guarantees for calibration with black-box models [1].

Calibration under distribution shifts. Ovadia et al. [29] conduct an empirical study on model calibration under distribution shifts and find that models are much less calibrated under distribution shifts. Minderer et al. [26] revisit calibration of recent state-of-the-art image classification models under distribution shifts and study the relationship between calibration and accuracy. Wald et al. [40] study model calibration and out-of-distribution generalization. Other works consider providing uncertainty estimates under structured distribution shifts, such as covariate shift [38, 31], label shift [34], and f-divergence balls [6]. Another line of work studies calibration in the domain adaptation setting [41, 30], which require unlabeled samples from the target domain.

2 Problem setup

Notation. We denote the input space and the label set by $\mathcal{X} \subseteq \mathbb{R}^d$ and $\mathcal{Y} = \{1, \ldots, J\}$. We let $[x]_i$ denote the i-th element of vector x. We use $\mathcal{P}(X)$ to denote the marginal feature distribution on input

space $\mathcal{X}$, $\mathcal{P}(Y|X)$ to denote the conditional distribution, and $\mathcal{P}(X, Y)$ to denote the joint distribution. For the multiple domains scenario, we let $\mathcal{P}_k(X)$ and $\mathcal{P}_k(Y|X)$ denote the feature distribution and conditional distribution for the k-th domain. We let $f : \mathcal{X} \to \mathbb{R}^J$ denote the base model, e.g., a deep neural network, where J is the total number of classes. We assume f returns an (unnormalized) vector of logits. Throughout the paper, the base model is trained with training data and will not be modified. The class prediction of model f on input $x \in \mathcal{X}$ is denoted by $\hat{y} = \operatorname{argmax}_{j \in \{1,\ldots,J\}} [f(x; \theta)]_j$. We use $\mathbf{1}\{\cdot\}$ to represent the indicator function. We use $h(\cdot; f, \beta) : \mathcal{X} \to [0, 1]$ to denote a *calibration map* (parameterized by β) that takes an input $x \in \mathcal{X}$ and returns a confidence score—this is a post-processing of the base model f. We let $\hat{\pi} = h(x; f, \beta) \in [0, 1]$ denote the confidence estimate for sample x when using model f. For instance, if we have 100 predictions $\{\hat{y}_1, \ldots, \hat{y}_{100}\}$ with confidence $\hat{\pi}_1 = \cdots = \hat{\pi}_{100} = 0.7$, then the accuracy of f is expected to be 70% on these 100 samples (if the confidence estimate is well calibrated). Data from the domains $\mathcal{P}_1, \ldots, \mathcal{P}_K$ are used for learning the calibration models, and we call the *in-distribution* (InD) domains. We use $\widetilde{\mathcal{P}}$ to denote the unseen *out-of-distribution* (OOD) domain which is not used for calibrating the base model. Our goal is to learn a calibration map h that is well calibrated on the OOD domain $\widetilde{\mathcal{P}}$. To do this, we will learn a calibration map that does well on all InD domains simultaneously.

To measure calibration, we first review the definition of approximate expected calibration error.

Definition 2.1 (ECE)**.** *For a set of samples $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n$ with $(x_i, y_i) \overset{\text{i.i.d.}}{\sim} \mathcal{P}(X, Y)$, the (empirical)* expected calibration error *(ECE) with M bins evaluated on $\mathcal{D}$ is defined as*

$$\mathsf{ECE}(\mathcal{D}, M) = \sum_{m=1}^{M} \frac{|B_m|}{n} \left|\mathsf{Acc}(B_m) - \mathsf{Conf}(B_m)\right|, \tag{1}$$

and B_m, $\operatorname{acc}(B_m)$, $\operatorname{conf}(B_m)$ are defined as

$$B_m = \{i \in [n] : \hat{\pi}_i \in ((m-1)/M, m/M]\},$$
$$\mathsf{Acc}(B_m) = (1/|B_m|) \sum_{i \in B_m} \mathbf{1}\{\hat{y}_i = y_i\}, \quad \mathsf{Conf}(B_m) = (1/|B_m|) \sum_{i \in B_m} \hat{\pi}_i,$$

where $\hat{\pi}_i$ and $\hat{y}_i$ are the confidence and predicted label of sample x_i.

The empirical ECE defined in Eq. (1) approximates the expected calibration error (ECE) $\mathbb{E}[|p - \mathbb{P}(\hat{y} = y|\hat{\pi} = p)|]$ with bin size equal to M [28, 12]; see [25] for statistical results about about the empirical ECE as an estimator. The perfect calibrated map corresponds to the case when $\mathbb{P}(\hat{y} = y|\hat{\pi} = p) = p$ holds for all $p \in [0, 1]$.

Multi-domain calibration. Although the standard ECE measurement in Eq. (1) provides informative evaluations for various calibration methods in the single-domain scenario, it does not provide fine-grained evaluations when the dataset consists of multiple domains, $\mathcal{P}_1, \ldots, \mathcal{P}_K$. It is possible that the ECE evaluated on the pooled data $\mathcal{D}_K^{\text{pool}} = \mathcal{D}_1 \cup \cdots \cup \mathcal{D}_K$ is small while the ECE evaluated on one of the domains is large. For example, as shown in Figure 1(c), there may exist a domain, $k \in [K]$, such that the ECE evaluated on domain k is much higher than the ECE evaluated on the pooled dataset, i.e., $\operatorname{ECE}(\mathcal{D}_k) \gg \operatorname{ECE}(\mathcal{D}_K^{\text{pool}})$. In the fMRI application mentioned in Section 1, producing well-calibrated confidence on data from every hospital is a more desirable property compared to only being calibrated on the pooled data from all hospitals. Therefore, it is natural to consider the ECE evaluated on every domain, which we refer to as "per-domain ECE." Next, we introduce the notion of Multi-domain ECE to formalize per-domain calibration.

Definition 2.2 (Multi-domain ECE)**.** *For a dataset $\mathcal{D}_K^{\text{pool}} = \mathcal{D}_1 \cup \cdots \cup \mathcal{D}_K$ consisting of samples from K domains, where $\mathcal{D}_k = \{(x_{i,k}, y_{i,k})\}_{i=1}^{n_k}$ and $(x_{i,k}, y_{i,k}) \overset{\text{i.i.d.}}{\sim} \mathcal{P}_k(X, Y)$, the (empirical)* multi-domain expected calibration error *(Multi-domain ECE) with M bins evaluated on $\mathcal{D}_K^{\text{pool}}$ is defined as* $\mathsf{MDECE}(\mathcal{D}_K^{\text{pool}}) = \frac{1}{K} \sum_{k=1}^{K} \mathsf{ECE}(\mathcal{D}_k)$.

Compared with the standard ECE evaluated on the pooled dataset, multi-domain ECE provides information about per-domain model calibration. In the multi-domain setting, we aim to learn a calibration map $\hat{h}$ that can produce calibrated confidence estimates on every InD domain. Intuitively, if the unseen OOD domain $\widetilde{\mathcal{D}}$ is similar to one or multiple InD domains, $\hat{h}$ can still provide reliable confidence estimates on the new domain. We formally study the connection between "well-calibrated on each InD domain" and "robust calibration on the OOD domain" in Section 5.

Temperature scaling. Next, we review a simple and effective calibration method, named temperature scaling (TS) [33, 12], that is widely used in single-domain model calibration. Temperature scaling applies a single parameter $T > 0$ and produces the confidence prediction for the base model f as

$$h^{\text{ts}}(x; f, T) = \max_{j \in \{1,\ldots,J\}} [\text{Softmax}(f(x)/T)]_j,$$

where $[\text{Softmax}(z)]_j = \exp([z]_j)/\sum_{i=1}^{J} \exp([z]_i)$. The parameter T is the so-called *temperature*, with larger temperature yielding more diffuse probability estimates. To learn the temperature parameter T from dataset $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{n}$, Guo et al. [12] propose to find T by solving the following convex optimization problem,

$$\min_{T} \mathcal{L}_{\texttt{TS}}(T) := -\sum_{i=1}^{n}\sum_{j=1}^{J} \mathbf{1}\{y_i = j\} \cdot \log([\text{Softmax}(f(x_i)/T)]_j), \tag{2}$$

which optimizes the temperature parameter such that the negative log likelihood is minimized. We use `TS-Alg` to denote the temperature scaling learning algorithm; given inputs dataset $\mathcal{D}$ and base model f, `TS-Alg` outputs the learned temeperature parameter by solving Eq. (2), e.g., $\hat{T} = \texttt{TS-Alg}(\mathcal{D}, f)$.

3 Multi-domain temperature scaling

We propose our algorithm—multi-domain temperature scaling—that aims to improve the calibration on each domain. One key observation is that if we apply temperature scaling to each domain separately, then TS is able to produce calibrated confidence on every domain. Therefore, the question becomes how to "aggregate" these temperature scaling models and learn one calibration model, denoted by $\hat{h}$, that has similar performance to the k-th calibration model $\hat{h}_k$ evaluated on domain k for every $k \in [K]$.

At a high level, we propose to learn a calibration model that maps samples from the input space $\mathcal{X}$ to the temperature space $\mathbb{R}_+$. To start with, we learn the temperature parameter $\hat{T}_k$ for the base model on every domain k by applying temperature scaling on $\mathcal{D}_k$. Next, we apply the base deep model to compute feature embeddings of samples from different domains,[1] and label feature embeddings from the k-th domain with $\hat{T}_k$. In particular, we construct K new datasets, $\hat{\mathcal{D}}_1, \ldots, \hat{\mathcal{D}}_K$, where each dataset contains feature embeddings and temperature labels from one domain, i.e., $\hat{\mathcal{D}}_k = \{(\Psi(x_{i,k}), \hat{T}_k)\}_{i=1}^{n_k}$. Finally, we apply linear regression on these labeled datasets. In detail, our algorithm is as follows:

1. **Learn temperature scaling model for each domain.** For every domain k, we learn temperature $\hat{T}_k$ by applying temperature scaling on validation data $\mathcal{D}_k = \{(x_{i,k}, y_{i,k})\}_{i=1}^{n_k}$ from k-th domain, i.e., $\hat{T}_k = \texttt{TS-Alg}(\mathcal{D}_k, f)$ and `TS-Alg` denotes the TS algorithm.
2. **Learn linear regression of temperatures.** Extract the feature embeddings of the base deep model f on each domain. Use $\Psi(x_{i,k}) \in \mathbb{R}^p$ to denote the feature embedding of the i-th sample from k-th domain. Then we learn $\hat{\theta}$ by solving the following optimization problem,

$$\hat{\theta} = \operatorname*{argmin}_{\theta} \sum_{k=1}^{K}\sum_{i=1}^{n_k} \left(\langle \Psi(x_{i,k}), \theta\rangle - \hat{T}_k\right)^2.$$

3. **Predict temperature on unseen test samples.** Given an unseen test sample $\widetilde{x}$, we first compute the predicted temperature $\widetilde{T}$ using the learned linear model $\widetilde{T} = \langle \Psi(x_{i,k}), \hat{\theta}\rangle$. Then we output the confidence estimate for sample $\widetilde{x}$ as

$$\widetilde{\pi} = \max_{j} \left[\text{Softmax}\big(f(\widetilde{x})/\widetilde{T}\big)\right]_j.$$

We denote our proposed method by MD-TS (**M**ult-**D**omain **T**emperature **S**caling). A presentation of the algorithm in pseudocode can be found in Algorithm 1, Appendix A.

We pause to consider the basic concept in more detail. The goal of our proposed algorithm is to predict the best temperature for samples from different several domains. In an ideal setting where

[1] We use the penultimate layer outputs of model f as the feature embeddings by default.

the learned linear model $\hat{\theta}$ results in good calibration on *every* InD domain, we can expect that $\hat{\theta}$ will continue to yield good calibration on the OOD domain $\widetilde{\mathcal{P}}$ when $\widetilde{\mathcal{P}}$ is close to one or several InD domains. For example, $\widetilde{\mathcal{P}}$ will work well if $\widetilde{\mathcal{P}}$ is a mixture of the K domains, i.e., $\widetilde{\mathcal{P}} = \sum_{k=1}^{K} \alpha_k \mathcal{P}_k$ and $\alpha \in \Delta^{K-1}$. Regarding the algorithmic design, linear regression is one of the simplest models for solving the regression problem. It is computationally fast to learn such linear models as well as make predictions on new samples, making it attractive. We test alternative, more flexible, regression algorithms in Section 4 but do not observe significant gains over linear regression.

To illustrate how our proposed algorithm MD-TS performs differently from standard TS, we return to the ImageNet-C dataset. We compare the predicted temperature of our algorithm on new samples from domain k with the temperature that results from running TS on domain k alone. The results are summarized in Figure 2, where each circle corresponds to the mean predicted temperature on one InD domain. For each domain, we also visualize the standard deviation of the predicted temperatures for samples from that domain (the horizontal bar around each point). We find that our algorithm predicts the temperature quite well. Note that it does not have access to the domain index information of the fresh samples. By contrast, TS always uses the same temperature, regardless of the input point.

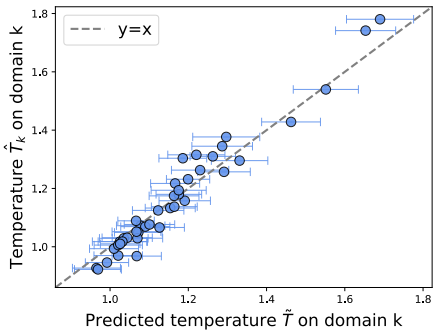


Figure 2: Compare the predicted temperature to the learned temperature $\hat{T}_k$ on the k-th domain.

4 Experiments

In this section, we present experimental results evaluating our proposed method, demonstrating its effectiveness on both in-distribution and out-of-distribution calibration. We focus on three real-world datasets, including ImageNet-C [14]—a widely used robustness benchmark image classification dataset, WILDS-RxRx1 [20]—an image of cells (by fluorescent microscopy) dataset in the domain generalization benchmark, and GLDv2 [42]—a landmark recognition dataset in federated learning. Additional experimental results and implementation details can be found in Appendix B.

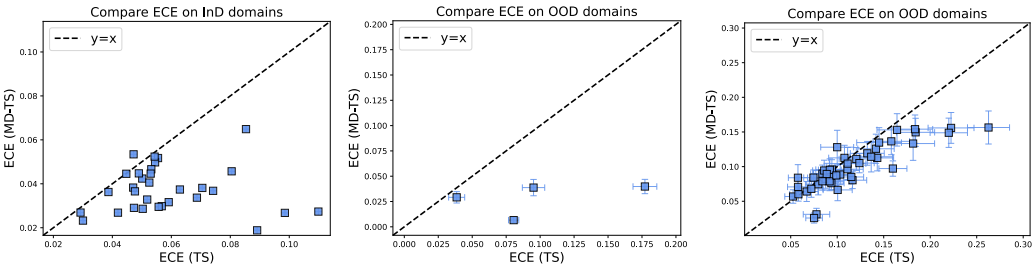
Datasets. We evaluate different calibration methods on three datasets, ImageNet-C, WILDS-RxRx1, and GLDv2. ImageNet-C contains 15 types of common corruptions where each corruption includes five severity levels. Each corruption with one severity is one domain, and there are 76 domains in total (including the standard ImageNet validation dataset). We partition the 76 domains into disjoint in-distribution domains and out-of-distribution by severity level or corruption type. WILDS-RxRx1 is a domain generalization dataset, and we treat each experimental domain as one domain. We adopt the default val/test split in Koh et al. [20]: use the four validation domains as in-distribution domains and the 14 test domains as the out-of-distribution domains. We also provide experimental results of other random splits in Appendix B. For GLDv2, each client corresponds to one domain, and there are 823 domains in total. We randomly select 500 domains for training the model, and then use the remaining 323 domains for evaluation denoted by validation domains. We further screen the validation domains by removing the domains with less than 300 data points. There are 44 domains after screening, and we use 30 domains as in-distribution domains and the remaining 14 domains as out-of-distribution domains. For all datasets, we randomly sample half of the data from in-distribution domains for calibrating models and use the remaining samples for InD ECE evaluation. We use all the samples from OOD domains for ECE evaluation.

Models and training setup. We consider multiple network architectures for evaluation, including ResNet-50 [13], ResNext-50 [44], DenseNet-121 [19], BiT-M-50 [21], Efficientnet-b1 [36], ViT-Small, and ViT-Base [8]. To evaluate on ImageNet-C, we directly evaluate models that are pre-trained on ImageNet [7]. For WILDS-RxRx1 and GLDv2, we use the ImageNet pre-trained models as initialization and apply SGD optimizer to training the models on training datasets.

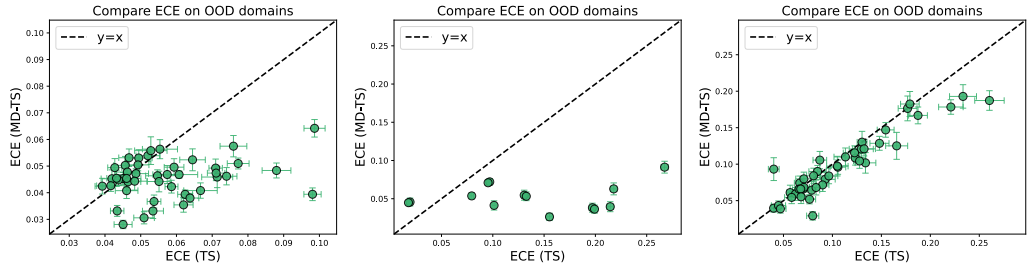
Evaluation metrics. We use the Expected Calibration Error (ECE) as the main evaluation metric. We set the bin size as 100 for ImageNet-C, and set bin size as 20 for WILDS-RxRx1 and GLDv2. We evaluate ECE on both InD domains and OOD domains. Specifically, we evaluate the ECE of each InD/OOD domain. Meanwhile, we also evaluate the ECE of the pooled InD/OOD domains, i.e.,

Table 1: Per-domain ECE (%) comparison on three datasets. We evaluate the per-domain ECE on InD and OOD domains. We report the mean and standard error of per-domain ECE on one dataset. Lower ECE means better performance.

Datasets	Architectures	InD-domains			OOD-domains		
		MSP [15]	TS [12]	MD-TS	MSP [15]	TS [12]	MD-TS
ImageNet-C	ResNet-50	7.36±0.28	5.80±0.10	3.84±0.05	6.87±0.16	5.70±0.06	**4.55±0.04**
	Efficientnet-b1	6.78±0.07	6.12±0.15	3.99±0.07	6.54±0.06	4.87±0.05	**4.05±0.03**
	BiT-M-R50	6.93±0.27	6.99±0.25	3.86±0.06	6.32±0.16	6.50±0.16	**4.30±0.04**
	ViT-Base	4.77±0.16	4.34±0.12	3.76±0.07	4.09±0.06	4.01±0.05	**3.86±0.04**
WILDS-RxRx1	ResNet-50	26.22±0.38	9.83±0.57	2.85±0.17	26.22±0.38	13.78±0.43	**5.25±0.11**
	ResNext-50	25.30±0.76	9.39±0.58	3.13±0.19	20.71±0.30	11.80±0.37	**5.07±0.09**
	DenseNet-121	32.37±0.91	8.91±0.60	2.94±0.18	24.49±0.35	13.08±0.41	**5.38±0.13**
GLDv2	ResNet-50	12.56±0.08	11.61±0.09	9.90±0.06	11.36±0.15	10.75±0.14	**9.76±0.12**
	BiT-M-R50	14.86±0.12	11.31±0.07	9.78±0.06	13.91±0.21	9.83±0.11	**9.16±0.10**
	ViT-Small	12.44±0.11	11.12±0.07	9.75±0.05	11.00±0.18	9.65±0.11	**9.01±0.10**



(a) (InD) ImageNet-C. (b) (InD) WILDS-RxRx1. (c) (InD) GLDv2.



(d) (OOD) ImageNet-C. (e) (OOD) WILDS-RxRx1. (f) (OOD) GLDv2.

Figure 3: Per-domain ECE of MD-TS and TS on both in-distribution domains and out-of-distribution domains. Each plot is shown with ECE of TS (X-axis) and ECE of MD-TS (Y-axis). Top: per-domain ECE evaluated on InD domains. Bottom: per-domain ECE evaluated on OOD domains. Lower ECE is better.

the ECE evaluated on all samples from InD/OOD domains. We use unseen samples from the InD domain to measure the per-domain ECE. We also measure the averaged per-domain ECE results (i.e., per-domain ECE averaged across domains).

4.1 Main results

We summarize the ECE results of different methods on three datasets in Table 1 and Figure 3. We use TS to denote temperature scaling [12], and use MSP to denote applying the maximum softmax probability [15] of the model output (i.e., without calibration). In Table 1, we use the ImageNet validation dataset and ImageNet-C datasets with severity level $\mathsf{s} \in \{1, 5\}$ as the InD domains and use the remaining datasets as OOD domains. We present the averaged per-domain ECE results in Table 1, and visualize the ECE of each domain in Figure 3. As shown in Table 1 and Figure 3(a)-3(c), we find that our proposed approach achieves much better InD per-domain calibration compared with

Table 2: Model performance prediction comparison results of different methods on three datasets. Lower MAE indicates better performance.

Datasets	Architectures	InD-domains MAE			OOD-domains MAE		
		MSP [15]	TS [12]	MD-TS	MSP [15]	TS [12]	MD-TS
ImageNet-C	ResNet-50	5.88	4.74	1.28	5.15	3.96	**1.70**
	BiT-M-R50	6.08	6.16	1.33	4.97	5.23	**1.66**
WILDS-RxRx1	ResNet-50	33.65	9.61	1.61	26.20	13.66	**4.76**
	ResNext-50	25.32	8.55	1.39	20.72	12.88	**4.78**
GLDv2	ResNet-50	9.60	9.17	7.11	9.72	9.40	**8.08**
	BiT-M-R50	12.67	7.18	4.64	12.30	7.34	**6.37**

baselines. Also, TS does not significantly improve over MSP on ImageNet-C InD domains in Table 1, but our proposed method largely improve the ECE compared with MSP and TS. For instance, the ECE results of MSP and TS on Efficientnet-b1 are 6.93 and 6.99, and our method achieves 3.84. Intuitively, when there are a diverse set of domains in the calibration dataset, a single temperature cannot provide well-calibrated confidences. In contrast, our proposed method is able to produce much better InD confidence estimates by leveraging the multi-domain structure of the data.

Next we study the performance of different methods on out-of-distribution domains. From Table 1, we find that MD-TS achieves the best performance on OOD domains arcoss all the settings. On ImageNet-C with BiT-M-R50, MD-TS improves the ECE from 6.54 (MSP) to 4.05, while the performance of TS is similar to MSP. Moreover, MD-TS significantly outperforms MSP and TS on WILDS-RxRx1, where MD-TS improves over TS by around 5.00 measured in ECE. Figure 3(d)-3(f) display the per-domain ECE performance on out-of-distribution domains. MD-TS improves over TS on more than half of the domains in all three datasets. For the remaining domains, MD-TS performs slightly worse than TS. Furthermore, on those domains that TS performs poorly (ECE > 8), MD-TS largely improves over TS by large margins. Further comparisons in Appendix B.7 show that these improvements continue to hold when relative to two other calibration techniques: MC dropout [9] and deep ensembles [24].

4.2 Predicting generalization

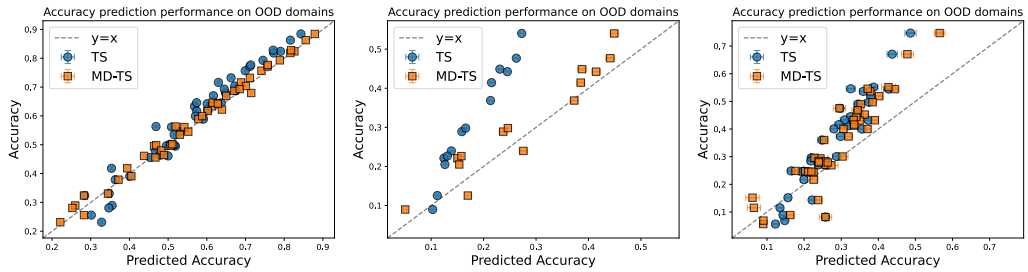
Suppose a model can produce calibrated confidences on unseen samples, in which case we could leverage the calibrated confidence to predict the model performance. Specifically, based on the definition of ECE in Eq. (1), when the model is well-calibrated, the average of the calibrated confidence is close to the average accuracy, i.e., $\mathsf{Conf}(\mathcal{D}) \approx \mathsf{Acc}(\mathcal{D})$.[2] Meanwhile, predicting model performance accurately is an essential ingredient in developing reliable machine learning systems, especially under distributional shifts [11]. As shown in Table 1, we find that our proposed method produces well-calibrated confidence values on both InD and OOD domains. We now measure its performance on predicting model performance and compare with existing methods. We measure the performance using mean absolute error (MAE), $\text{MAE} = (1/K) \cdot \sum_{k=1}^{K} |\mathsf{Conf}(\mathcal{D}_k) - \mathsf{Acc}(\mathcal{D}_k)|$ where S_k is the dataset from the k-th domain.

We show the predicting model accuracy results in Table 2. MD-TS significantly improves over existing methods on predicting model performance across all three datasets. For example, on ImageNet-C, calibrated confidence of MD-TS produces fairly accurate predictions on both InD and OOD domains (less than 2% measured in MAE), which largely outperforms MSP and TS. In Figure 4, we compared the prediction performance of TS and MD-TS on every OOD domain. We find that MD-TS achieves better prediction performance compared to TS on most of the domains. Refer to Appendix B.1 for more results in which other architectures are tested.

4.3 MD-TS ablations

To learn a calibration model that performs well per-domain, we apply linear regression on feature representations $\Phi(x_k)$ such that $\langle \Phi(x_k), \theta \rangle \approx \hat{T}_k$, where x_k is from domain k and $\hat{T}_k$ is the tem-

[2] $\mathsf{Conf}(\mathcal{D})$ denotes the average (calibrated) confidence on dataset $\mathcal{D}$, and $\mathsf{Acc}(\mathcal{D})$ denotes the average accuracy on dataset $\mathcal{D}$.



(a) (OOD) ImageNet-C. (b) (OOD) WILDS-RxRx1. (c) (OOD) GLDv2.

Figure 4: Predicting accuracy performance of MD-TS and TS on both out-of-distribution domains. Each plot is shown with predicted accuracy (X-axis) and accuracy (Y-axis). Each points corresponds to one domain. The network architecture is ResNet-50 for three datasets. Point closer to the $Y = X$ dashed line means better prediction performance.

Table 3: Per-domain ECE (%) results of MD-TS ablations on WILDS-RxRx1. We evaluate the per-domain ECE on InD and OOD domains, and report the mean and standard error of per-domain ECE. Lower ECE means better performance.

Architectures	**InD-domains**					**OOD-domains**				
	OLS	Ridge	Huber	KRR	KNN	OLS	Ridge	Huber	KRR	KNN
ResNet-50	2.85	2.88	2.90	2.85	3.00	5.25	5.26	5.29	4.99	5.44
ResNext-50	3.13	3.14	3.11	3.07	3.03	5.07	5.06	5.02	4.94	5.36
DenseNet-121	2.94	3.03	2.92	2.90	3.04	5.38	5.42	5.36	5.20	5.47

perature parameter for domain k. We investigate other methods for learning the map from feature representations to temperatures in a regression framework. Specifically, beside the ordinary least squares (OLS) used in Algorithm 1, we consider ridge regression (Ridge), robust regression with Huber loss (Huber), kernel ridge regression (KRR), and K-nearest neighbors regression (KNN). The implementations are mainly based on `scikit-learn` [32]. We use grid search (on InD domains) to select hyperparameters for Ridge, Huber, KRR, and KNN.

We summarize the comparative results for different regression algorithms in Table 3. Compared to OLS, other regression algorithms do not achieve significant improvement. Specifically, KRR achieves slightly better performance on OOD domains, while other algorithms have similar performance compared to OLS. Moreover, there are no hyperparameter in OLS, which makes it more practical in real-world problems. Meanwhile, the results suggest that our proposed MD-TS is stable to the choice of specific regression algorithms.

5 Theoretical analysis

In this section, we provide theoretical analysis to support our understanding of our proposed algorithm in the presence of distribution shifts. We use $h_k^\star(\cdot) = h(\cdot; f, \beta_k^\star) : \mathcal{X} \to [0, 1]$ to denote the best calibration map for the base model f on the k-th domain; this map *minimizes* the expected calibration error (ECE) $\mathbb{E}[|p - \mathbb{P}(\hat{y} = y|\hat{\pi} = p)|]$ over distribution $\mathcal{P}_k$. We also call $h_k^\star$ a hypothesis in the hypothesis class $\mathcal{H}$. Next, given the fixed base model f, we aim to learn $\hat{h}(\cdot) = h(\cdot; f, \hat{\beta})$ such that $\varepsilon(\hat{h}, \mathcal{P}_{k,X}) = \mathbb{E}_{X \sim \mathcal{P}_{k,X}}[|h_k^\star(X) - \hat{h}(X)|]$ is small for *every* domain k, where $\varepsilon_k(\hat{h})$ denotes the risk of $\hat{h}$ w.r.t. the the best calibration map $h_k^\star$ under domain $\mathcal{P}_k$. In addition, we are interested in generalizing to new domains: suppose there is an unseen OOD domain $\widetilde{\mathcal{P}}$ and its marginal feature distribution is different from existing domains, i.e., $\widetilde{\mathcal{P}}_X \neq \mathcal{P}_{k,X}$ for $k \in [K]$.

Our goal is to understand the conditions under which $\hat{h}$ can have similar calibration on OOD domains as the InD domains. For example, if the OOD domain is similar to the mixture distribution of InD domains, we would expect $\hat{h}$ performs similarly on InD and OOD domains. To quantify the distance

between two distributions, we first introduce the $\mathcal{H}$-divergence [4] to measure the distance between two distributions:

Definition 5.1 ($\mathcal{H}$-divergence)**.** *Given an input space $\mathcal{X}$ and two probability distributions $\mathcal{P}_X$ and $\mathcal{P}'_X$ on $\mathcal{X}$, let $\mathcal{H}$ be a hypothesis class on $\mathcal{X}$, and denote by $\mathcal{A}$ the collection of subsets of $\mathcal{X}$ which are the support of hypothesis $h \in \mathcal{H}$, i.e., $\mathcal{A}_{\mathcal{H}} = \{h^{-1}(1) \,|\, h \in \mathcal{H}\}$. The distance between $\mathcal{P}_X$ and $\mathcal{P}'_X$ is defined as*

$$\mathrm{d}_{\mathcal{H}}(\mathcal{P}_X, \mathcal{P}'_X) = \sup_{A \in \mathcal{A}_{\mathcal{H}}} \left| \Pr_{\mathcal{P}_X}(A) - \Pr_{\mathcal{P}'_X}(A) \right|.$$

The $\mathcal{H}$-divergence reduces to the standard total variation (TV) distance when $\mathcal{H}$ contains all measurable functions on $\mathcal{X}$, which implies that the $\mathcal{H}$-divergence is upper bounded by the TV-distance, i.e., $\mathrm{d}_{\mathcal{H}}(\mathcal{P}_X, \mathcal{P}'_X) \leq \mathrm{d}_{\mathsf{TV}}(\mathcal{P}_X, \mathcal{P}'_X)$. On the other hand, when the hypothesis class $\mathcal{H}$ has a finite VC dimension or pseudo-dimension, the $\mathcal{H}$-divergence can be estimated using finite samples from $\mathcal{P}_X$ and $\mathcal{P}'_X$ [4]. Next, we define the mixture distribution of the K in-distribution domains $\mathcal{P}^{\alpha}_{K,X}$ on input space $\mathcal{X}$ as follows:

$$\mathcal{P}^{\alpha}_{K,X} = \sum_{k=1}^{K} \alpha_k \mathcal{P}_{k,X}, \quad \text{where } \sum_{k=1}^{K} \alpha_k = 1 \text{ and } \alpha_k \geq 0.$$

Given multiple domains $\{\mathcal{P}_1, \ldots, \mathcal{P}_K\}$, we can optimize the combination parameters α such that $\mathcal{P}^{\alpha}_{K,X}$ minimizes the $\mathcal{H}$-divergence between $\mathcal{P}^{\alpha}_{K,X}$ and $\widetilde{\mathcal{P}}_X$. More specifically, we define $\hat{\alpha}$ as

$$\hat{\alpha} = \operatorname*{argmin}_{\alpha \in \Delta} \Big\{ \frac{1}{2} \mathrm{d}_{\bar{\mathcal{H}}}(\mathcal{P}^{\alpha}_{K,X}, \widetilde{\mathcal{P}}_X) + \lambda(\mathcal{P}^{\alpha}_{K,X}, \widetilde{\mathcal{P}}_X) \Big\}, \quad \lambda(\mathcal{P}^{\alpha}_{K,X}, \widetilde{\mathcal{P}}_X) = \varepsilon(h^{\star}, \mathcal{P}^{\alpha}_{K,X}) + \varepsilon(h^{\star}, \widetilde{\mathcal{P}}_X), \tag{3}$$

where $h^{\star} := \operatorname{argmin}_{h \in \mathcal{H}} \{\varepsilon(h, \mathcal{P}^{\alpha}_{K,X}) + \varepsilon(h, \widetilde{\mathcal{P}}_X)\}$ and $\bar{\mathcal{H}}$ is defined as $\bar{\mathcal{H}} := \{\mathrm{sign}(|h(x) - h'(x)| - t) \,|\, h, h' \in \mathcal{H}, 0 \leq t \leq 1\}$. We now give an upper bound on the risk on the unseen OOD domain. This result follows very closely those of Blitzer et al. [5], Zhao et al. [47], instantiated in our calibration setup. Details can be found in Appendix E.

Theorem 5.2. *Let $\mathcal{H}$ be a hypothesis class that contains functions $h : \mathcal{X} \to [0, 1]$ with pseudo-dimension $\mathrm{Pdim}(\mathcal{H}) = d$. Let $\{D_{k,X}\}_{k=1}^{K}$ denote the empirical distributions generated from $\{\mathcal{P}_{k,X}\}_{k=1}^{K}$, where $\mathcal{D}_{k,X}$ contains n i.i.d. samples from the marginal feature distribution $\mathcal{P}_{k,X}$ of domain k. Then for $\delta \in (0, 1)$, with probability at least $1 - \delta$, we have*

$$\varepsilon(\hat{h}, \widetilde{\mathcal{P}}_X) \leq \sum_{k=1}^{K} \hat{\alpha}_k \cdot \hat{\varepsilon}(\hat{h}, \mathcal{D}_{k,X}) + \frac{1}{2} \mathrm{d}_{\bar{\mathcal{H}}}(\mathcal{P}^{\hat{\alpha}}_{K,X}, \widetilde{\mathcal{P}}_X) + \lambda(\mathcal{P}^{\hat{\alpha}}_{K,X}, \widetilde{\mathcal{P}}_X) + \widetilde{O}\left(\frac{\mathrm{Pdim}(\mathcal{H})}{\sqrt{nK}} \right), \tag{4}$$

where $\hat{\alpha}$ and $\lambda(\mathcal{P}^{\hat{\alpha}}_{K,X}, \widetilde{\mathcal{P}}_X)$ are defined in Eq. (3), $\widetilde{\mathcal{P}}_X$ denotes the marginal distribution of the OOD domain, $\mathrm{Pdim}(\mathcal{H})$ is the pseudo-dimension of the hypothesis class $\mathcal{H}$, and $\hat{\varepsilon}(\hat{h}, \mathcal{D}_{k,X})$ is the empirical risk of the hypothesis $\hat{h}$ on $\mathcal{D}_{k,X}$.

This result means that if we can learn a hypothesis $\hat{h}$ that achieves small empirical risk $\hat{\varepsilon}(\hat{h}, \mathcal{D}_{k,X})$ on every domain, then $\hat{h}$ is able to achieve good performance on the OOD domain if distribution of the OOD domain is similar to the mixture distribution of InD domains measured by $\mathcal{H}$-divergence. In this case, if the learned calibration map $\hat{h}$ is well-calibrated on every domain $\mathcal{P}_k$, then $\hat{h}$ is likely to provide calibrated confidence for the OOD domain $\widetilde{\mathcal{P}}$. Recall from Section 4, we proposed an algorithm that performs well across InD domains. The upper bound in Eq. (4) provides insight into understanding why this algorithm is effective.

6 Discussion

We have developed an algorithm for robust calibration that exploits multi-domain structure in datasets. Experiments on real-world domains indicate that multi-domain calibration is an effective way to improve the robustness of calibration under distribution shifts. One interesting direction for future work would be to extend our algorithm to a scenario where no domain information is available. We hope the multi-domain calibration perspective in this paper can motivate further work to close the gap between in-distribution and out-of-distribution calibration.

References

[1] Anastasios N. Angelopoulos, Stephen Bates, Emmanuel J. Candès, Michael I. Jordan, and Lihua Lei. Learn then test: Calibrating predictive algorithms to achieve risk control. *arXiv preprint*, 2021. arXiv:2110.01052.

[2] Solon Barocas, Moritz Hardt, and Arvind Narayanan. *Fairness and Machine Learning*. fairmlbook.org, 2019. http://www.fairmlbook.org.

[3] Stephen Bates, Anastasios Angelopoulos, Lihua Lei, Jitendra Malik, and Michael I. Jordan. Distribution-free, risk-controlling prediction sets. *Journal of the ACM*, 68(6), September 2021. doi: 10.1145/3478535.

[4] Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. A theory of learning from different domains. *Machine learning*, 79(1): 151–175, 2010.

[5] John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman. Learning bounds for domain adaptation. *Advances in neural information processing systems*, 20, 2007.

[6] Maxime Cauchois, Suyash Gupta, Alnur Ali, and John C Duchi. Robust validation: Confident predictions even when distributions shift. *arXiv preprint arXiv:2008.04267*, 2020.

[7] 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.

[8] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2020.

[9] Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pages 1050–1059. PMLR, 2016.

[10] Yarin Gal, Jiri Hron, and Alex Kendall. Concrete dropout. *Advances in neural information processing systems*, 30, 2017.

[11] Devin Guillory, Vaishaal Shankar, Sayna Ebrahimi, Trevor Darrell, and Ludwig Schmidt. Predicting with confidence on unseen distributions. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1134–1144, 2021.

[12] 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.

[13] 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.

[14] Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. *arXiv preprint arXiv:1903.12261*, 2019.

[15] Dan Hendrycks and Kevin Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. *arXiv preprint arXiv:1610.02136*, 2016.

[16] Dan Hendrycks, Mantas Mazeika, Saurav Kadavath, and Dawn Song. Using self-supervised learning can improve model robustness and uncertainty. *Advances in Neural Information Processing Systems*, 32, 2019.

[17] Dan Hendrycks, Norman Mu, Ekin D Cubuk, Barret Zoph, Justin Gilmer, and Balaji Lakshminarayanan. Augmix: A simple data processing method to improve robustness and uncertainty. *arXiv preprint arXiv:1912.02781*, 2019.

[18] Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadavath, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Zhu, Samyak Parajuli, Mike Guo, Dawn Song, Jacob Steinhardt, and Justin Gilmer. The many faces of robustness: A critical analysis of out-of-distribution generalization. *ICCV*, 2021.

[19] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.

[20] Pang Wei Koh, Shiori Sagawa, Henrik Marklund, Sang Michael Xie, Marvin Zhang, Akshay Balsubramani, Weihua Hu, Michihiro Yasunaga, Richard Lanas Phillips, Irena Gao, et al. Wilds: A benchmark of in-the-wild distribution shifts. In *International Conference on Machine Learning*, pages 5637–5664. PMLR, 2021.

[21] Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Joan Puigcerver, Jessica Yung, Sylvain Gelly, and Neil Houlsby. Big transfer (bit): General visual representation learning. In *European conference on computer vision*, pages 491–507. Springer, 2020.

[22] Lingkai Kong, Jimeng Sun, and Chao Zhang. Sde-net: Equipping deep neural networks with uncertainty estimates. In *International Conference on Machine Learning*, pages 5405–5415. PMLR, 2020.

[23] Aviral Kumar and Sunita Sarawagi. Calibration of encoder decoder models for neural machine translation. *arXiv preprint arXiv:1903.00802*, 2019.

[24] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. *Advances in neural information processing systems*, 30, 2017.

[25] Donghwan Lee, Xinmeng Huang, Hamed Hassani, and Edgar Dobriban. T-cal: An optimal test for the calibration of predictive models. *arXiv preprint arXiv:2203.01850*, 2022.

[26] Matthias Minderer, Josip Djolonga, Rob Romijnders, Frances Hubis, Xiaohua Zhai, Neil Houlsby, Dustin Tran, and Mario Lucic. Revisiting the calibration of modern neural networks. *Advances in Neural Information Processing Systems*, 34, 2021.

[27] Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. *Foundations of machine learning*. 2018.

[28] Mahdi Pakdaman Naeini, Gregory Cooper, and Milos Hauskrecht. Obtaining well calibrated probabilities using bayesian binning. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

[29] Yaniv Ovadia, Emily Fertig, Jie Ren, Zachary Nado, David Sculley, Sebastian Nowozin, Joshua Dillon, Balaji Lakshminarayanan, and Jasper Snoek. Can you trust your model's uncertainty? evaluating predictive uncertainty under dataset shift. *Advances in neural information processing systems*, 32, 2019.

[30] Sangdon Park, Osbert Bastani, James Weimer, and Insup Lee. Calibrated prediction with covariate shift via unsupervised domain adaptation. In *International Conference on Artificial Intelligence and Statistics*, pages 3219–3229. PMLR, 2020.

[31] Sangdon Park, Edgar Dobriban, Insup Lee, and Osbert Bastani. Pac prediction sets under covariate shift. *arXiv preprint arXiv:2106.09848*, 2021.

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

[33] 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.

[34] Aleksandr Podkopaev and Aaditya Ramdas. Distribution-free uncertainty quantification for classification under label shift. In *Uncertainty in Artificial Intelligence*, pages 844–853. PMLR, 2021.

[35] Joaquin Quiñonero-Candela, Masashi Sugiyama, Anton Schwaighofer, and Neil D Lawrence. *Dataset shift in machine learning*. Mit Press, 2008.

[36] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pages 6105–6114. PMLR, 2019.

[37] Sunil Thulasidasan, Gopinath Chennupati, Jeff A Bilmes, Tanmoy Bhattacharya, and Sarah Michalak. On mixup training: Improved calibration and predictive uncertainty for deep neural networks. *Advances in Neural Information Processing Systems*, 32, 2019.

[38] Ryan J Tibshirani, Rina Foygel Barber, Emmanuel Candes, and Aaditya Ramdas. Conformal prediction under covariate shift. *Advances in neural information processing systems*, 32, 2019.

[39] Vladimir Vovk, Alexander Gammerman, and Glenn Shafer. *Algorithmic learning in a random world*. Springer Science & Business Media, 2005.

[40] Yoav Wald, Amir Feder, Daniel Greenfeld, and Uri Shalit. On calibration and out-of-domain generalization. *Advances in Neural Information Processing Systems*, 34, 2021.

[41] Ximei Wang, Mingsheng Long, Jianmin Wang, and Michael Jordan. Transferable calibration with lower bias and variance in domain adaptation. *Advances in Neural Information Processing Systems*, 33:19212–19223, 2020.

[42] Tobias Weyand, Andre Araujo, Bingyi Cao, and Jack Sim. Google landmarks dataset v2-a large-scale benchmark for instance-level recognition and retrieval. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2575–2584, 2020.

[43] Cihang Xie, Mingxing Tan, Boqing Gong, Jiang Wang, Alan L Yuille, and Quoc V Le. Adversarial examples improve image recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 819–828, 2020.

[44] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1492–1500, 2017.

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

[46] 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.

[47] Han Zhao, Shanghang Zhang, Guanhang Wu, José MF Moura, Joao P Costeira, and Geoffrey J Gordon. Adversarial multiple source domain adaptation. *Advances in neural information processing systems*, 31, 2018.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [No]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] See Theorem 5.2.
   (b) Did you include complete proofs of all theoretical results? [Yes] See Appendix E.
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] See supplemental material.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See Section 4 and Appendix A.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] See Section 4.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See Appendix A.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] See Section 4.
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]