Training on Test Data with Bayesian Adaptation for Covariate Shift

Aurick Zhou, Sergey Levine
Department of Electrical Engineering and Computer Sciences
University of California, Berkeley
{aurick,svlevine}@berkeley.edu

Abstract

When faced with distribution shift at test time, deep neural networks often make inaccurate predictions with unreliable uncertainty estimates. While improving the robustness of neural networks is one promising approach to mitigate this issue, an appealing alternate to robustifying networks against all possible test-time shifts is to instead directly adapt them to unlabeled inputs from the particular distribution shift we encounter at test time. However, this poses a challenging question: in the standard Bayesian model for supervised learning, unlabeled inputs are conditionally independent of model parameters when the labels are unobserved, so what can unlabeled data tell us about the model parameters at test-time? In this paper, we derive a Bayesian model that provides for a well-defined relationship between unlabeled inputs under distributional shift and model parameters, and show how approximate inference in this model can be instantiated with a simple regularized entropy minimization procedure at test-time. We evaluate our method on a variety of distribution shifts for image classification, including image corruptions, natural distribution shifts, and domain adaptation settings, and show that our method improves both accuracy and uncertainty estimation.

1 Introduction

Modern deep learning methods can provide high accuracy in settings where the model is evaluated on data from the same distribution as the training set, but accuracy often degrades severely when there is a mismatch between the training and test distributions [Hendrycks and Dietterich, 2019, Taori et al., 2020]. In safety-critical settings, effectively deploying machine learning models requires not only high accuracy, but also requires the model to reliably quantify uncertainty in its predictions in order to assess risk and potentially abstain from making dangerous, unreliable predictions. Reliably estimating uncertainty is especially important in settings with distribution shift where inaccurate predictions are more prevalent, but the reliability of uncertainty estimates often also degrades along with the accuracy as the shifts become more severe [Ovadia et al., 2019]. In real-world applications, distribution mismatch at test time is often inevitable, thus necessitating methods that can robustly handle distribution shifts, both in terms of retaining high accuracy but also in providing meaningful uncertainty estimates.

As it can be difficult to train a single model to be robust to all potential distribution shifts we might encounter at test time, we can instead robustify models by allowing for *adaptation* at test time, where we finetune the network on unlabeled inputs from the shifted target distribution, thus allowing the model to specialize in the particular shift it encounters. Since test-time distribution shifts often cannot be anticipated during training time, we restrict the adaptation procedure to operate without any further access to the original training data. Prior work on test-time adaptation [Wang et al., 2020a, Sun et al., 2019b] focused on improving accuracy, and found that simple objectives like entropy minimization capable of providing substantial improvements under distribution shift [Wang et al.,

35th Conference on Neural Information Processing Systems (NeurIPS 2021), Sydney, Australia.

2020a]. However, these prior works do not consider uncertainty estimation, and an objective like entropy minimization can quickly make predictions overly confident and less calibrated, leaving us without reliable uncertainty estimates and thus unable to quantify risks when making predictions. Our goal in this paper is to design a test-time adaptation procedure that can not only improve predictive accuracy under distribution shift, but also provide reliable uncertainty estimates.

While a number of prior papers have proposed various heuristic methods for test-time adaptation [Wang et al., 2020a, Sun et al., 2019a], it remains unclear what precisely unlabeled test data under covariate shift can actually tell us about the optimal classifier. In this work, we take a Bayesian approach to this question, and explicitly formulate a Bayesian model that describes how unlabeled test data from a different domain can be related to the classifier parameters. Such a model requires introducing an additional explicit assumption, as the classifier parameters are conditionally independent of unlabeled data in the standard model for discriminative classification [Seeger, 2000]. The additional assumption we introduce intuitively states that the data generation process at test-time, though distinct from the one at training time (hence, under covariate shift) is still more likely to produce inputs that have a single unambiguous labeling, even if that labeling is not known. We argue that this assumption is reasonable in practice, and leads to an appealing graphical model where approximate inference corresponds to a Bayesian extension of entropy minimization.

We propose a practical test-time adaptation strategy, Bayesian Adapatation for Covariate Shift (BACS), which approximates Bayesian inference in this model and outperforms prior adaptive methods both in terms of increasing accuracy and improving calibration under distribution shift. Our adaptation strategy is simple to implement, requires minimal changes to standard training procedures, and outperforms prior test-time adaptation techniques on a variety of benchmarks for robustness to distribution shifts.

2 Bayesian Adaptation for Covariate Shift

In this section, we will devise a probabilistic graphical model that describes how unlabeled data in a new test domain can inform our posterior about the model, and then describe a practical deep learning algorithm that can instantiate this model in a system that enables test-time adaptation. We will begin by reviewing standard probabilistic models for supervised and discuss why such models are unable to utilize unlabeled data. We then discuss a probabilistic model proposed by Seeger [2000] that does incorporate unlabeled data in a semisupervised learning (SSL) setting, and propose an extension to the model to account for distribution shift. Finally, we discuss the challenges in performing exact inference in our proposed model and describe the approximations we introduce in order to derive a tractable inference procedure suitable for test-time adaptation.

2.1 Probabilistic Model for Covariate Shift

In the standard Bayesian model for supervised learning (Figure 1), we assume that the inputs $\mathbf{X}$ are sampled i.i.d. from a generative model with parameters ϕ, while the corresponding labels are sampled from a conditional distribution $p(Y|\mathbf{X}, \theta)$ parameterized by θ. The parameters θ and ϕ are themselves random variables, sampled independently from prior distributions $p(\phi)$ and $p(\theta)$. We observe only a dataset $\mathcal{D} = \{\mathbf{X}_i, Y_i\}_{i=1}^n$, and then perform inference over the parameters θ using Bayes rule:

$$p(\theta|\mathcal{D}) \propto p(\theta) \prod_{i=1}^{n} p(Y_i|\mathbf{X}_i, \theta). \tag{1}$$

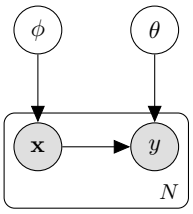

Figure 1: Probabilistic model for standard supervised learning, observing N labeled data points.

To make predictions on a new input $\mathbf{x}_{n+1}$, we marginalize over the posterior distribution of the classifier parameters θ to obtain the predictive distribution

$$p(Y_{n+1}|\mathbf{X}_{n+1}, \mathcal{D}) = \int p(Y_{n+1}|\mathbf{X}_{n+1}, \theta) p(\theta|\mathcal{D})\, d\theta.$$

Note that observing the (unlabeled) test input $\mathbf{x}_{n+1}$ (or more generally, any number of test inputs) does not affect the posterior distribution of parameters θ, and so within this probabilistic model, there is no benefit to observing multiple unlabeled datapoints before making predictions.

Inference for semi-supervised learning. By treating the parameters θ, ϕ as a-priori independent, the standard model assumes there is no relationship between the input distribution and the process that generates labels, leading to the inability to utilize unlabeled data in inferring θ. To utilize unlabeled data, we can introduce assumptions about the relationship between the labeling process and input distribution using a model of the form in Figure 2 [Seeger, 2000].

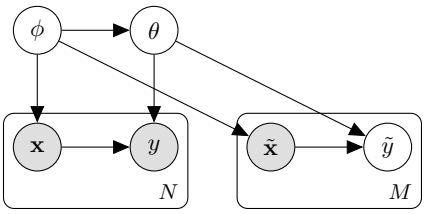

Figure 2: Model for SSL [Seeger, 2000], observing N labeled pairs and M unlabeled inputs, both generated from the same distribution. Observing unlabeled inputs $\tilde{\mathbf{x}}$ provides information about the input distribution ϕ, and thus about θ.

The assumption we use has a simple intuitive interpretation: we assume that the inputs that will be shown to our classifier have an unambiguous and clear labeling. This assumption is reasonable in many cases: if the classifier discriminates between automobiles and bicycles, it is reasonable that it will be presented with images that contain either an automobile or bicycle. Of course, this assumption is not necessarily true in all settings, but this intuition often agrees with how discriminatively trained models are actually used. This simple intuition can be formalized in terms of a prior belief about the *conditional entropy* of labels conditioned on the inputs.

Similar to Grandvalet and Bengio [2004], we can encode this belief using the additional factor

$$p(\theta|\phi) \propto \mu(\theta) \exp(-\alpha H_{\theta,\phi}(Y|\mathbf{X})) \tag{2}$$

$$= \mu(\theta) \exp\left(\alpha \mathbb{E}_{\mathbf{X}\sim p(X|\phi), Y\sim p(Y|X,\theta)}[\log p(Y|\mathbf{X},\theta)]\right), \tag{3}$$

where $\mu(\theta)$ is a prior over the parameters θ that is agnostic of the input distribution parameter ϕ.

Now, observing unlabeled test inputs $\mathcal{U} = \{\tilde{\mathbf{x}}_1, \ldots, \tilde{\mathbf{x}}_m\}$ provides information about the parameter ϕ governing the input distribution, which then allows us to update our belief over the learned parameters θ through Equation 2 and thus allows inference to utilize unlabeled data within a Bayesian framework.

Extension to covariate shift. The previous probabilistic model for SSL assumes all inputs were drawn from the same distribution (given by ϕ). However, our goal is use unlabeled data to adapt our model to a *different* test distribution, so we extend the model to incorporate *covariate shift*.

We now assume we have two input-generating distributions; ϕ specifies the input distribution for our labeled training set, and $\tilde{\phi}$ specifies the shifted distribution of unlabeled test inputs which we aim to adapt to. Under the assumption of covariate shift, the same classifier θ is used to generate labels in both the train and test domains, leading to the model in Figure 3.

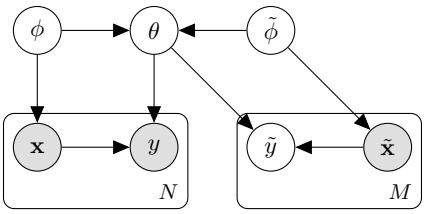

Figure 3: Our proposed probabilistic model for adaptation for covariate shift, observing a training set with N labeled pairs and M unlabeled inputs from a shifted test distribution.

We argue that our prior belief of low aleatoric uncertainty is reasonable for *both* the distribution induced by ϕ and the one induced by $\tilde{\phi}$; even if there is some distribution shift from the training set, we can still often expect the test data we are shown to have unambiguous labels. We can then incorporate both ϕ and $\tilde{\phi}$ into our belief over θ with the factor

$$p(\theta|\phi, \tilde{\phi}) \propto \mu(\theta) \exp(-\alpha H_{\theta,\phi}(Y|\mathbf{X})) \exp(-\tilde{\alpha} H_{\theta,\tilde{\phi}}(Y|\mathbf{X})), \tag{4}$$

with the two scalar hyperparameters $\alpha, \tilde{\alpha}$ controlling how to weight the entropies from each distribution with the likelihoods of the labeled training set. While we can extend this model to include more labeled or unlabeled input distributions, we focus on simply having one training distribution and one test distribution as it is the most relevant for our problem setting of adapting to a particular distribution shift at test time.

2.2 Approximations for Tractable Inference

Performing inference over θ in the above model can be challenging in our test-time adaptation setting for several reasons. First of all, inference over θ would also require performing inference over the

parameters of the generative models ϕ and $\tilde{\phi}$ to evaluate likelihoods in Equation 4. This is difficult in practice for two reasons: First, the inputs x might be high-dimensional and difficult to model, as in the case of images. Second, the amount of unlabeled data might be fairly small, and insufficient to accurately estimate the parameters $\tilde{\phi}$ if we employ a highly expressive generative model. As such, we would much prefer to avoid explict generative modeling and to only perform inference over the discriminative model parameters θ instead.

Another issue is that performing inference would require access to both the labeled training data and the unlabeled test data at the same time, while our test-time adaptation setting assumes that we can no longer look at the training set when it is time to adapt to the test distribution. We will discuss how to address these issues and describe our method, Bayesian Adaptation for Covariate Shift (BACS), which provides a practical instantiation of inference in this Bayesian model in a computationally tractable test-time adaptation setting.

Plug-in approximation with empirical Bayes. We first propose to avoid explicit generative modeling by using with a plug-in empirical Bayes approach. Empirical Bayes is a common approximation for hierarchical Bayesian models that simply uses a point estimate of ϕ rather than marginalizing over ϕ's (similarly for $\tilde{\phi}$), which would reduce the computation to estimating only a single generative model for each input distribution given by parameters ϕ^* and $\tilde{\phi}^*$. To eliminate the need to train the parameters ϕ^* of a generative model of the inputs altogether, we note that $p(\theta|\phi, \tilde{\phi})$ only depends on ϕ and $\tilde{\phi}$ through the input distributions $p(x|\phi), p(\tilde{x}|\tilde{\phi})$. We can then approximate Equation 4 by plugging in the empirical distributions of x and $\tilde{x}$ in place of $p(x|\phi^*), p(\tilde{x}|\tilde{\phi}^*)$, resulting in

$$p(\theta|\hat{\phi}, \tilde{\phi}) \propto \mu(\theta) \exp\left(-\frac{\alpha}{n}\sum_{i=1}^{n} H(Y_i|\mathbf{x}_i, \theta)\right) \exp\left(-\frac{\tilde{\alpha}}{m}\sum_{i=1}^{m} H(Y_i|\tilde{\mathbf{x}}_i, \theta)\right).$$

Now, given a labeled training set $\mathcal{D}$ and unlabeled test points $\mathcal{U} = \{\tilde{\mathbf{x}}_i\}_{i=1}^m$, the new posterior distribution over parameters now has log probabilities (up to an additive normalizing constant)

$$\log p(\theta|\mathcal{D}, \mathcal{U}) = \log \mu(\theta) + \sum_{i=1}^{n} \log p(y_i|\mathbf{x}_i, \theta) - \frac{\alpha}{n}\sum_{i=1}^{n} H(Y|\mathbf{x}_i, \theta) - \frac{\tilde{\alpha}}{m}\sum_{j=1}^{m} H(Y|\tilde{\mathbf{x}}_j, \theta). \quad (5)$$

For convenience, we simply set $\alpha = 0$, as additionally minimizing entropy on the labeled training set is unnecessary when we are already maximizing the likelihood of the given labels with highly expressive models. Now, to infer θ given the observed datasets $\mathcal{D}$ and $\mathcal{U}$, the simplified log-density is

$$\log p(\theta|\mathcal{D}, \mathcal{U}) = \log \mu(\theta) + \sum_{i=1}^{n} \log p(y_i|\mathbf{x}_i, \theta) - \frac{\tilde{\alpha}}{m}\sum_{j=1}^{m} H(Y|\tilde{\mathbf{x}}_j, \theta). \quad (6)$$

Computing the maximum-a-posteriori (MAP) solution of this model corresponds to simply optimizing model parameters θ that on the supervised objective on the labeled training set in addition a minimum entropy regularizer on the unlabeled input. However, we should not necessarily expect the MAP solution to provide reasonable uncertainty estimates, as the learned model is being encouraged to make confident predictions on the test inputs, and so any single model will likely provide overconfident predictions. Marginalizing the predictions over the posterior distribution over parameters is thus essential to recovering meaningful uncertainty estimates, as the different models, though each being individually confident, can still express uncertainty through their combined predictions when the models predict different labels.

Approximate inference for test-time adaptation. We now discuss how to perform inference in the above model in a way suitable for *test-time adaptation*, where we adapt to the test data without any further access to the original training set. To enable this, we propose to learn an approximate posterior

$$q(\theta) \approx p(\theta|\mathcal{D}) = \log \mu(\theta) + \sum_{i=1}^{n} \log p(y_i|\mathbf{x}_i, \theta) \quad (7)$$

during training time, and then use this approximate training set posterior $q(\theta)$ in place of the training set when performing inference on the unlabeled test data. This gives us an approximate posterior with log density

$$\log p(\theta|\mathcal{D}, \mathcal{U}) = \log q(\theta) - \frac{\tilde{\alpha}}{m}\sum_{j=1}^{m} H(Y|\tilde{\mathbf{x}}_j, \theta) \quad (8)$$

Algorithm 1 Bayesian Adaptation for Covariate Shift (BACS)

Input: Ensemble size k, Entropy weight $\tilde{\alpha}$, Training Data: $\mathbf{x}_{1:n}, y_{1:n}$, Test Data: $\tilde{\mathbf{x}}_{1:m}$
Output: Predictive distributions $p(y|\tilde{\mathbf{x}}_j)$ for each test input $\tilde{\mathbf{x}}_j$
Training: For each $i \in (1, \ldots, k)$ compute an approximate density $q_i(\theta) \approx p(\theta|\mathbf{x}_{1:n}, y_{1:n})$
Adaptation:
for all ensemble members $i \in (1, \ldots, k)$ **do**
  Compute adapted parameters $\hat{\theta}_i = \arg\max_\theta \frac{\tilde{\alpha}}{m} \sum_{j=1}^{m} -H(Y|\tilde{\mathbf{x}}_j, \theta) + \log q_i(\theta)$
end for
For each test input $\tilde{\mathbf{x}}_j$, marginalize over ensemble $p(y|\tilde{\mathbf{x}}_j) = \frac{1}{k} \sum_{i=1}^{k} p(y|\tilde{\mathbf{x}}_j, \hat{\theta}_i)$.

Unlike the full training set, the learned approximate posterior density $q(\theta)$, which can be as simple as a diagonal Gaussian distribution, can be much easier to store and optimize over for test-time adaptation, and the training time procedure would be identical to any approximate Bayesian method that learns a posterior density. In principle, we can now instantiate test-time adaptation by running any approximate Bayesian inference algorithm, such as variational inference or MCMC, to sample θ's from the density in Equation 8, and average predictions from these samples to compute the marginal probabilities for each desired test label.

Practical instantiation with ensembles. As previously mentioned, marginalizing over different models that provide diverse labelings of the test set is crucial to providing uncertainty estimates after adaptation via entropy minimization. We thus propose to use an ensembling approach [Lakshminarayanan et al., 2016] as a practical method to adapt to the test distribution while maintaining diverse labelings. Deep ensembles simply train multiple models from different random initializations, each independently optimizing the target likelihoods, and averages together the models' predicted probabilities at test time. They are able to provide effective approximations to Bayesian marginalization due to their ability to aggregate models across highly distinct modes of the loss landscape [Fort et al., 2019, Wilson and Izmailov, 2020].

Our method, BACS, summarized in Algorithm 1, trains an ensemble of k different models on the training set, each with their approximate posterior $q_i(\theta)$ that captures the local loss landscape around each mode in the ensemble. Then at test time, we independently optimize each of the k models by minimizing Equation 8 (using the corresponding $q_i(\theta)$ for each ensemble member), and then average the predictions across all adapted ensemble members.

3 Related Work

Entropy minimization. Entropy minimization has been used as a self-supervised objective in many settings, including domain adaptation [Saito et al., 2019, Carlucci et al., 2017], semisupervised learning [Grandvalet and Bengio, 2004, Berthelot et al., 2019, Lee and Lee, 2013], and few-shot learning [Dhillon et al., 2015]. Grandvalet and Bengio [2004] proposed a probabilistic model incorporating entropy minimization for semisupervised learning (without distribution shift), but only use the probabilistic model to motivate entropy minimization as a regularizer for a MAP solution in order improve accuracy, which does not capture any epistemic uncertainty. In contrast, we are concerned with test-time adaptation under distribution shift, which requires introducing a model of separate training-time and test-time input distributions, and with providing reliable epistemic uncertainty estimates, which we obtain via Bayesian marginalization.We also devise an approximate inference scheme to allow for efficient adaptation without access to the training data.

Test time entropy minimization (TENT) [Wang et al., 2020a] uses entropy minimization as the sole objective when adapting to the test data (though without an explicit Bayesian interpretation) and adapts without any further access to the training data, but only aims to improve accuracy and not uncertainty estimation. Similarly to Grandvalet and Bengio [2004], TENT only learns a single model using entropy minimization, whereas we show that explicitly performing Bayesian inference and marginalizing over multiple models is crucial for effective uncertainty estimation. TENT also heuristically proposes to only adapt specific parameters in the networks at test time for stability reasons, while our usage of a learned posterior density to account for the training set allows us to adapt the whole network, improving performance in some settings and eliminating the need for the heuristic design decision.

Domain Adaptation. Unsupervised domain adaptation [Ganin et al., 2015, Wang and Deng, 2018] tackles a similar problem of learning classifiers in the presence of distribution shift between our train and test distributions. Most unsupervised domain adaptation works assume access to both the labeled training data as well as unlabeled test data at the same time, while we restrict adaptation to occur without any further access to the training data. One recent line of work, known as source-free domain adaptation [Liang et al., 2020, Kundu et al., 2020, Li et al., 2020] also restricts the adaptation procedure to not have access to the training data together with the unlabeled data from the test distribution. In contrast to these algorithms, we are concerned with using adaptation to improve both uncertainty estimation as well as accuracy, and our algorithm is additionally amenable to an online setting, where prediction and adaptation occur simultaneously without needing to see the entirety of the test inputs.

Uncertainty estimation under distribution shift (no adaptation). Various methods have been proposed for uncertainty estimation with distribution shift that do not incorporate test time adaptation. Ovadia et al. [2019] measured the calibration of various models under various distribution shift without any adaptation, finding that deep ensembles [Lakshminarayanan et al., 2016] and some other Bayesian methods that marginalize over multiple models perform well compared to methods that try to recalibrate predictions using only the source data. Beyond ensembles, other Bayesian techniques [Dusenberry et al., 2020, Maddox et al., 2019] have also demonstrated improved uncertainty estimation under distribution shift compared to standard models. Various other techniques have been found to improving calibration under distribution shift through significant changes to the training procedure, for example utilizing different loss functions [Padhy et al., 2020, Tomani and Buettner, 2020], extensive data-augmentations [Hendrycks et al., 2019], or extra pre-training [Xie et al., 2019b].

Uncertainty estimation with both train and test distributions. We now discuss prior work that considers uncertainty estimation assuming access to both train and test data simultaneously. Recent work studying calibration for domain adaptation algorithms [Pampari and Ermon, 2020, Park et al., 2020, Wang et al., 2020b] found that predictions are poorly calibrated in the target domain even if the models were well-calibrated on the labeled source domain. These works all propose methods based on importance weighting between the target domain and the labeled source domain in order to recalibrate target domain predictions using only labels in the source domain. They are not directly applicable in our test-time adaptation setting, since they require estimates of density ratios between target and source distributions, which we cannot obtain without either a generative model of training inputs, or access to the training data during adaptation. Our method also differs in how we approach uncertainty estimation. Instead of using extra labeled data and post-hoc recalibration techniques for classifiers, our method uses Bayesian inference to provide meaningful uncertainty estimates.

For uncertainty estimation in regression problems, Chan et al. [2020] also propose to adapt Bayesian posteriors to unlabeled data by optimizing the predictive variance of Bayesian neural network at an input to serve as a binary classifier of whether the point is in-distribution or not. thus encouraging higher variance for out-of-distribution points. Their method is again not applicable in our test-time setting because they require access to both the train and test data at once.

Uncertainty estimation with test time adaptation. Nado et al. [2020] evaluate various techniques for uncertainty estimation in conjunction with adapting batch-norm statistics to the shifted test domain, and again find that deep ensembles provide well-calibrated prediction in addition to improved accuracy. While our method similarly utilizes ensembles and adapts batch norm statistics, and we show that additionally adapting via entropy minimization at test time further improves predictive accuracy without sacrificing calibration.

Bayesian semi-supervised methods: Seeger [2000] proposed a probablistic model for incorporating unlabeled data in semi-supervised learning to motivate regularization for the classifier that depends on the input distribution in MAP inference. However, they do not tackle uncertainty estimation and their model does not account for any distribution shift like ours does. Gordon and Hernández-Lobato [2020] perform Bayesian semi-supervised learning combining both generative and discriminative models. In contrast, our method does not need to learn a generative model of the data, and explicitly tackles the problem of distribution shift instead of assuming the labeled and unlabeled data come from the same distribution. Another line of work [Ng et al., 2018, Ma et al., 2019, Walker and Glocker, 2019, Liu et al., 2020] propose Bayesian methods for semisupervised learning specialized graph-structured data.

Method	CIFAR10-C				CIFAR100-C			
	Acc	NLL	Brier	ECE	Acc	NLL	Brier	ECE
Vanilla	59.90	1.892	0.6216	0.2489	35.72	4.271	0.9797	0.3883
BN Adapt	82.36	0.8636	0.2909	0.1204	57.58	2.294	0.6401	0.2377
TENT (1 epoch)	84.29	0.7862	0.2629	0.1119	62.46	2.047	0.5828	0.2280
TENT (5 epoch)	85.16	0.8483	0.2603	0.1191	63.46	2.199	0.5987	0.2592
BACS (MAP) (1 epoch)	84.82	0.7808	0.2585	0.1119	63.05	2.090	0.5908	0.2411
BACS (MAP) (5 epochs)	85.20	0.8075	0.2575	0.1144	63.53	2.175	0.6009	0.2551
Vanilla Ensemble	61.72	1.535	0.5431	0.1684	38.66	3.343	0.8439	0.2462
Ensemble BN Adapt	85.99	0.4722	0.2043	0.03229	64.22	1.464	0.4793	**0.0515**
Ensemble TENT (1 epoch)	87.28	0.4351	0.1867	**0.02868**	67.83	**1.318**	0.4392	0.05909
BACS (ours) (1 epoch)	**87.77**	**0.4260**	**0.1809**	0.02986	**68.33**	1.324	**0.4360**	0.06519

Table 1: **CIFAR-10/100 Corrupted** results at the highest level of corruption, averaged over all corruption types. With one epoch of adaptation, BACS consistently outperforms all baselines in terms of accuracy, NLL and Brier score, and can further improve with more training. In terms of ECE, all ensembled methods with adaptation performs similarly, and substantially outperforming the non-adaptive or non-ensembled baselines.

4 Experiments

In our experiments, we aim to analyze how our test-time adaptation procedure in BACS performs when adapting to various types of distribution shift, in comparison to prior methods, in terms of *both* the accuracy of the adapted model, and its ability to estimate uncertainty and avoid over-confident but incorrect predictions. We evaluate our method and prior techniques across a range of distribution shifts, including corrupted datasets, natural distribution shifts, and domain adaptation settings.

Architectures and implementation. For our ImageNet [Deng et al., 2009] experiments, we use the ResNet50v2 [He et al., 2016b] architecture, while for other datasets, we use ResNet26 [He et al., 2016a]. For all methods utilizing ensembles, we use ensembles of 10 models, and report results averaged over the same 10 seeds for the non-ensembled methods. While adapting our networks using the entropy loss, we also allow the batch normalization statistics to adapt to the target distribution. To obtain approximate posteriors for each ensemble member, we use SWAG-D [Maddox et al., 2019], which estimates a Gaussian posterior with diagonal covariance from a trajectory of SGD and requires minimal changes to standard training procedures. During adaptation, we initialize each model from the corresponding posterior mean, corresponding to the solution obtained by Stochastic Weight Averaging [Izmailov et al., 2018]. For methods that optimize on the test distribution, we report results after one epoch of adaptation unless otherwise stated.

Comparisons. We compare our method against two state-of-the-art prior methods for test-time adaptation: TENT [Wang et al., 2020a], which simply minimizes entropy on the test data with a single model, without the additional posterior term accounting for the training set that we use in BACS, and ensembles adapted using the batch norm statistics of the shifted test set, as discussed by Nado et al. [2020]. We also compare to deep ensembles [Lakshminarayanan et al., 2016] without any adaptation as a baseline for uncertainty estimation under distribution shift, as well as ensembles of models each adapted using TENT.

Metrics. In addition to accuracy, we also evaluate uncertainty estimation using the negative log likelihood (NLL), Brier score [Brier, 1950] and expected calibration error (ECE) [Naeini et al., 2015] metrics. NLL and Brier score are both proper scoring rules [Gneiting and Raftery, 2007], and are minimized if and only if the predicted distribution is identical to the true distribution. ECE measures calibration by binning predictions according to the predicted confidence and averaging the absolute differences between the average confidence and empirical accuracy within each bin.

Corrupted images. We first evaluate our method on CIFAR-10-C, CIFAR-100-C, and ImageNet-C [Hendrycks and Dietterich, 2019], where distribution-shifted datasets are generated by applying different image corruptions at different intensities to the test sets of CIFAR10, CIFAR100 [Krizhevsky, 2012], and ImageNet [Deng et al., 2009] respectively. In Table 1, we show comparisons at the most severe level of corruption for CIFAR10-C and CIFAR100-C. With just a single epoch of adaptation at test time, both our method BACS and TENT ensembles substantially outperform other baselines in accuracy, NLL, and Brier score, with BACS improving slightly over TENT ensembles, showing that combining test-time entropy minimization with Bayesian marginalization can lead to strong improvements over either alone. We also note that simply adapting for more epochs with entropy

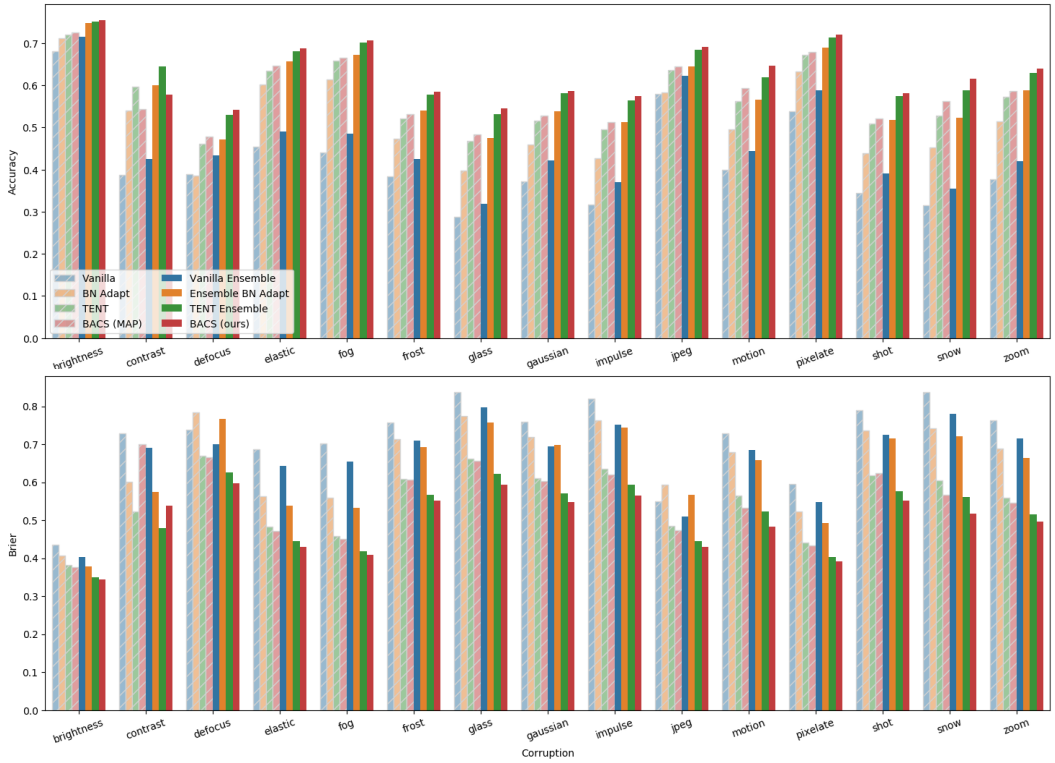

Figure 4: **ImageNet-C Results by Corruption**. For each corruption type, we show the results for each method averaged over all levels of corruption. BACS improves over baselines for every corruption type except for contrast, where BACS has close to 0 accuracy at the most severe level of corruption.

minimization (as seen in TENT (5 epochs)), can further improve accuracy of a single model, but can actually lead to *worse* uncertainty estimates as measured by NLL or ECE as predictions become excessively overconfident. In contrast, combining ensembling with entropy minimization consistently improves accuracy much more than simply adapting a single model for more epochs, while also substantially improving uncertainty estimates.

In Table 2, we show comparisons on ImageNet-C averaged over all corruption levels and corruption types. BACS is able to outperform all prior methods in terms of accuracy, NLL, and Brier score. We note that ensembling and batch norm adaptation actually *worsen calibration*, as measured by ECE, compared to the single model or non-adapted baselines respectively, despite each technique providing substantial improvements on all other metrics. We see that an ablation of our method that only uses a single model, BACS (MAP), outperforms the other non-ensemble methods in accuracy, NLL, and Brier score.

Method	Acc	NLL	Brier	ECE
Vanilla	41.79	3.127	0.7152	0.06439
BN Adapt	51.54	2.676	0.6564	0.1709
TENT	57.06	2.035	0.5536	**0.0342**
BACS (MAP)	58.06	2.036	0.5528	0.04270
Vanilla ensemble	46.03	2.788	0.6670	0.07256
Ensemble BN Adapt	58.30	2.428	0.6333	0.2594
Ensemble TENT	62.40	1.788	0.5131	0.1010
BACS (ours)	**63.04**	**1.726**	**0.4962**	0.08308

Table 2: **ImageNet-C** results averaged over all corruption types and levels. BACS outperforms all baselines in accuracy, NLL, and Brier scores.

We also show results for accuracy and Brier score at each level of corruption in Figure 4. BACS (ours) consistently outperforms baselines at each level of corruption (with one exception being accuracy at the highest level of corruption, where a single corruption with much lower accuracy drags down the mean to be slightly below to that of Ensemble TENT).

ImageNet-R. In Table 3, we further evaluate robustness using ImageNet-R [Hendrycks et al., 2021], which consists of images that are abstract renditions of 200 of the ImageNet classes. Similar to our Imagenet-C results, we find that BACS performs the best across accuracy, NLL, and Brier score, while being slightly outperformed in ECE by vanilla ensembles.

Impact of posterior term. In this section, we discuss the effects of using the training set posterior density in our objective (Equation 8) when adapting at test time. Intuitively, the training posterior density ensures that our adapted classifiers, while minimizing entropy on the target distribution, remain constrained to still perform well on the training set and stay near the initial solution found during training. We empirically find that the posterior term can be important for preventing entropy minimization from finding degenerate solutions: adapting the whole network on several ImageNet-C corruptions without the posterior term can lead to poor models that achieve close to 0 accuracy on many corruptions.

Method	Acc	NLL	Brier	ECE
Vanilla	36.40	3.288	0.7602	0.05667
BN Adapt	38.05	3.236	0.7646	0.09744
TENT	41.13	2.967	0.7167	0.02666
BACS (MAP)	43.55	2.909	0.7183	0.1074
Vanilla ensemble	40.38	3.011	0.7180	**0.02339**
Ensemble BN Adapt	42.82	3.019	0.7408	0.1696
Ensemble TENT	45.75	2.726	0.6815	0.1025
BACS (ours)	**47.31**	**2.565**	**0.6625**	0.04270

Table 3: **ImageNet-R** results. BACS outperforms all baselines in accuracy, NLL, and Brier scores.

While TENT, which adapts via entropy minimization without any term accounting for the training data, uses an ad-hoc solution that restricts adaptation to only the learnable scale and shift parameters in the batch norm layers, our use of the training set posterior density is motivated directly from our proposed probabilistic model and does not require any heuristic choices over which parameters should or should not be adapted. In our ImageNet experiments (Tables 2 and 3), we see that an ablation of our method without ensembles, corresponding to the MAP solution in our proposed model, outperforms TENT. The difference between our ablation BACS (MAP) and TENT is precisely that BACS (MAP) adapts the whole network, but with an additional regularization term, while TENT adapts only a small part of the network without any regularizer.

However, we find the approximate posterior density can also be limiting in certain situations, which we can observe in experiments transferring from SVHN to MNIST (Appendix B.4). Here, there is a large discrepancy between the training and test domains, and we find that adaptation with the posterior is unable to adjust the parameters enough and underperforms compared to an ablation of our method that simply removes the posterior term while still adapting the whole network.

5 Discussion

We presented Bayesian Adaptation for Covariate Shift (BACS), a Bayesian approach for utilizing test-time adaptation to obtain both improved accuracy and well-calibrated uncertainty estimates when faced with distribution shift. We have shown that adapting via entropy minimization without Bayesian marginalization can lead to overconfident uncertainty estimates, while our principled usage of an approximate training posterior during adaptation can outperform previous heuristic methods. These observation support our hypothesis that framing entropy minimization within a well-defined Bayesian model can lead to significantly more effective test-time adaptation techniques. Our method is straightforward to implement, requires minimal changes to standard training procedures, and improves both accuracy and uncertainty estimation for a variety of distribution shifts in classification.

Limitations and future work. One limitation of our approach is that it requires effective techniques for estimating the parameter posterior from the training set. While the study of such methods, and Bayesian neural networks more broadly, is an active area of research, it remains a significant practical challenge. It is likely that the simple Gaussian posteriors we employ provide a poor estimation of the true posterior and can overly constrain the network during adaptation. Therefore, a relevant direction for future work is to integrate BACS with more sophisticated Bayesian neural network methods.

Another promising direction for future work is to explore other objectives that have had success in semi-supervised learning settings, such as consistency based losses [Sajjadi et al., 2016, Miyato et al., 2017, Xie et al., 2019a, Sohn et al., 2020] or information maximization [Gomes et al., 2010], which can be straightforwardly incorporated into our method as suitable priors on the relationship between the data distribution and classifier. More broadly, we hope that our work will spur further research into test-time adaptation techniques based on well-defined Bayesian models that describe how unlabeled test data should inform our posterior estimates of the model parameters, even in the presence of distributional shift.

Acknowledgements

We thank the anonymous reviewers for their extremely valuable feedback and discussions, which have greatly improved our paper. This research was supported by the DARPA Assured Autonomy program and DARPA LwLL, with compute support from Google Cloud and the Tensorflow Research Cloud (TFRC) program.

References

D. Berthelot, G. Research, N. Carlini, I. Goodfellow, A. Oliver, N. Papernot, and C. Raffel. MixMatch: A Holistic Approach to Semi-Supervised Learning. Technical report, 2019. URL https://github.com/google-research/mixmatch.

G. W. Brier. Verification of forecasts expressed in terms of probability. *Monthly Weather Review*, 78 (1), 1 1950. ISSN 0027-0644. doi: 10.1175/1520-0493(1950)078<0001:VOFEIT>2.0.CO;2.

F. M. Carlucci, L. Porzi, B. Caputo, E. Ricci, and S. R. Bulò. AutoDIAL: Automatic DomaIn Alignment Layers. *Proceedings of the IEEE International Conference on Computer Vision*, 2017-October:5077–5085, 4 2017. URL http://arxiv.org/abs/1704.08082.

A. J. Chan, A. M. Alaa, Z. Qian, and M. Van Der Schaar. Unlabelled Data Improves Bayesian Uncertainty Calibration under Covariate Shift. Technical report, 11 2020. URL http://proceedings.mlr.press/v119/chan20a.html.

A. Coates, A. Ng, and H. Lee. An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 215–223, 2011.

J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255, 2009.

G. S. Dhillon, P. Chaudhari, A. Ravichandran, and S. Soatto. A Baseline for Few-Shot Image Classification. In *International Conference on Learning Representations*. arXiv, 9 2015. URL http://arxiv.org/abs/1909.02729.

M. W. Dusenberry, G. Jerfel, Y. Wen, Y.-a. Ma, J. Snoek, K. Heller, B. Lakshminarayanan, and D. Tran. Efficient and Scalable Bayesian Neural Nets with Rank-1 Factors. *arXiv preprint arXiv:2005.07186*, 2020.

N. Ford, J. Gilmer, N. Carlini, and D. Cubuk. Adversarial Examples Are a Natural Consequence of Test Error in Noise. *36th International Conference on Machine Learning, ICML 2019*, 2019-June: 4115–4139, 1 2019. URL https://arxiv.org/abs/1901.10513v1.

S. Fort, H. Hu, and B. Lakshminarayanan. Deep Ensembles: A Loss Landscape Perspective. *arXiv*, 12 2019. URL http://arxiv.org/abs/1912.02757.

Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky. Domain-Adversarial Training of Neural Networks. *Advances in Computer Vision and Pattern Recognition*, 17(9783319583464):189–209, 5 2015. URL http://arxiv.org/abs/1505.07818.

T. Gneiting and A. E. Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association*, 102(477):359–378, 3 2007. ISSN 01621459. doi: 10.1198/016214506000001437. URL https://www.tandfonline.com/doi/abs/10.1198/016214506000001437.

R. Gomes, A. Krause, and P. Perona. Discriminative Clustering by Regularized Information Maximization. Technical report, 2010. URL http://www.cs.ubc.ca/.

J. Gordon and J. M. Hernández-Lobato. Combining deep generative and discriminative models for Bayesian semi-supervised learning. *Pattern Recognition*, 100:107156, 4 2020. ISSN 00313203. doi: 10.1016/j.patcog.2019.107156.

Y. Grandvalet and Y. Bengio. Semi-supervised Learning by Entropy Minimization. In *Advances in Neural Information Processing Systems*, volume 17, 2004.

K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2016-December, pages 770–778. IEEE Computer Society, 12 2016a. ISBN 9781467388504. doi: 10.1109/CVPR.2016.90. URL http://image-net.org/challenges/LSVRC/2015/.

K. He, X. Zhang, S. Ren, and J. Sun. Identity Mappings in Deep Residual Networks. *ECCV*, 2016b. URL https://arxiv.org/abs/1603.05027v3.

D. Hendrycks and T. Dietterich. Benchmarking Neural Network Robustness to Common Corruptions and Perturbations. *7th International Conference on Learning Representations, ICLR 2019*, 3 2019. URL http://arxiv.org/abs/1903.12261.

D. Hendrycks, N. Mu, E. D. Cubuk, B. Zoph, J. Gilmer, and B. Lakshminarayanan. AugMix: A Simple Data Processing Method to Improve Robustness and Uncertainty. *arXiv*, 12 2019. URL http://arxiv.org/abs/1912.02781.

D. Hendrycks, S. Basart, N. Mu, S. Kadavath, F. Wang, E. Dorundo, R. Desai, T. Zhu, S. Parajuli, M. Guo, D. Song, J. Steinhardt, and J. Gilmer. The Many Faces of Robustness: A Critical Analysis of Out-of-Distribution Generalization. *ICCV*, 2021. URL https://arxiv.org/abs/2006.16241v3.

P. Izmailov, D. Podoprikhin, T. Garipov, D. P. Vetrov, and A. G. Wilson. Averaging Weights Leads to Wider Optima and Better Generalization. In *UAI*, 2018.

A. Krizhevsky. Learning Multiple Layers of Features from Tiny Images. *University of Toronto*, 6 2012.

J. N. Kundu, N. Venkat, R. M V, and R. V. Babu. Universal Source-Free Domain Adaptation. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 4543–4552, 4 2020. URL https://arxiv.org/abs/2004.04393v1.

B. Lakshminarayanan, A. Pritzel, and C. Blundell. Simple and Scalable Predictive Uncertainty Estimation using Deep Ensembles. *Advances in Neural Information Processing Systems*, 2017-December:6403–6414, 12 2016. URL http://arxiv.org/abs/1612.01474.

Y. LeCun, C. Cortes, and C. J. Burges. MNIST handwritten digit database. *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist*, 2, 2010.

D.-h. Lee and D.-h. Lee. Pseudo-Label: The Simple and Efficient Semi-Supervised Learning Method for Deep Neural Networks. *ICML Workshop on challenges in representation learning*, 2013. URL http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.664.3543.

R. Li, Q. Jiao, W. Cao, H. S. Wong, and S. Wu. Model Adaptation: Unsupervised Domain Adaptation without Source Data. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 9638–9647, 2020. doi: 10.1109/CVPR42600.2020.00966.

J. Liang, D. Hu, and J. Feng. Do We Really Need to Access the Source Data? Source Hypothesis Transfer for Unsupervised Domain Adaptation. *arXiv*, 2 2020. URL http://arxiv.org/abs/2002.08546.

Z.-Y. Liu, S.-Y. Li, S. Chen, Y. Hu, and S.-J. Huang. Uncertainty Aware Graph Gaussian Process for Semi-Supervised Learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(04): 4957–4964, 4 2020. ISSN 2159-5399. doi: 10.1609/aaai.v34i04.5934. URL www.aaai.org.

J. Ma, W. Tang, J. Zhu, and Q. Mei. A Flexible Generative Framework for Graph-based Semi-supervised Learning. *arXiv*, 5 2019. URL http://arxiv.org/abs/1905.10769.

W. J. Maddox, P. Izmailov, T. Garipov, D. P. Vetrov, and A. G. Wilson. A simple baseline for bayesian uncertainty in deep learning. In *Advances in Neural Information Processing Systems*, pages 13132–13143, 2019.

T. Miyato, S.-i. Maeda, M. Koyama, and S. Ishii. Virtual Adversarial Training: A Regularization Method for Supervised and Semi-Supervised Learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(8):1979–1993, 4 2017. URL http://arxiv.org/abs/1704.03976.

Z. Nado, S. Padhy, D. Sculley, A. D'Amour, B. Lakshminarayanan, and J. Snoek. Evaluating Prediction-Time Batch Normalization for Robustness under Covariate Shift. *arXiv*, 6 2020. URL http://arxiv.org/abs/2006.10963.

M. P. Naeini, G. Cooper, and M. Hauskrecht. Obtaining well calibrated probabilities using bayesian binning. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng. Reading Digits in Natural Images with Unsupervised Feature Learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2011.

Y. C. Ng, N. Colombo, and R. Silva. Bayesian Semi-supervised Learning with Graph Gaussian Processes. *Advances in Neural Information Processing Systems*, 2018-December:1683–1694, 9 2018. URL http://arxiv.org/abs/1809.04379.

Y. Ovadia, E. Fertig, J. Ren, Z. Nado, D. Sculley, S. Nowozin, J. V. Dillon, B. Lakshminarayanan, and J. Snoek. Can You Trust Your Model's Uncertainty? Evaluating Predictive Uncertainty Under Dataset Shift. *arXiv*, 6 2019. URL http://arxiv.org/abs/1906.02530.

S. Padhy, Z. Nado, J. Ren, J. Liu, J. Snoek, and B. Lakshminarayanan. Revisiting One-vs-All Classifiers for Predictive Uncertainty and Out-of-Distribution Detection in Neural Networks. *arXiv*, 7 2020. URL http://arxiv.org/abs/2007.05134.

A. Pampari and S. Ermon. Unsupervised Calibration under Covariate Shift. *arXiv*, 6 2020. URL http://arxiv.org/abs/2006.16405.

S. Park, O. Bastani, J. Weimer, and I. Lee. Calibrated Prediction with Covariate Shift via Unsupervised Domain Adaptation. *arXiv*, 2 2020. URL http://arxiv.org/abs/2003.00343.

K. Saito, D. Kim, S. Sclaroff, T. Darrell, and K. Saenko. Semi-supervised Domain Adaptation via Minimax Entropy. *Proceedings of the IEEE International Conference on Computer Vision*, 2019-October:8049–8057, 4 2019. URL http://arxiv.org/abs/1904.06487.

M. Sajjadi, M. Javanmardi, and T. Tasdizen. Regularization With Stochastic Transformations and Perturbations for Deep Semi-Supervised Learning. *Advances in Neural Information Processing Systems*, pages 1171–1179, 6 2016. URL http://arxiv.org/abs/1606.04586.

M. Seeger. Input-dependent Regularization of Conditional Density Models. Technical report, 2000. URL https://infoscience.epfl.ch/record/175482.

K. Sohn, D. Berthelot, C.-L. Li, Z. Zhang, N. Carlini, E. D. Cubuk, A. Kurakin, H. Zhang, and C. Raffel. FixMatch: Simplifying Semi-Supervised Learning with Consistency and Confidence. *arXiv*, 1 2020. URL http://arxiv.org/abs/2001.07685.

S. Sun, G. Zhang, J. Shi, and R. Grosse. Functional Variational Bayesian Neural Networks. 3 2019a. URL http://arxiv.org/abs/1903.05779.

Y. Sun, X. Wang, Z. Liu, J. Miller, A. A. Efros, and M. Hardt. Test-Time Training with Self-Supervision for Generalization under Distribution Shifts. 9 2019b. URL http://arxiv.org/abs/1909.13231.

R. Taori, A. Dave, V. Shankar, N. Carlini, B. Recht, and L. Schmidt. Measuring Robustness to Natural Distribution Shifts in Image Classification. *arXiv*, 7 2020. URL http://arxiv.org/abs/2007.00644.

C. Tomani and F. Buettner. Towards Trustworthy Predictions from Deep Neural Networks with Fast Adversarial Calibration. 12 2020. URL http://arxiv.org/abs/2012.10923.

I. Walker and B. Glocker. Graph Convolutional Gaussian Processes. In *International Conference on Machine Learning*, 5 2019. URL http://arxiv.org/abs/1905.05739.

D. Wang, E. Shelhamer, S. Liu, B. Olshausen, and T. Darrell. Tent: Fully Test-time Adaptation by Entropy Minimization. *arXiv*, 6 2020a. URL http://arxiv.org/abs/2006.10726.

M. Wang and W. Deng. Deep Visual Domain Adaptation: A Survey. *Neurocomputing*, 312:135–153, 2 2018. doi: 10.1016/j.neucom.2018.05.083. URL https://arxiv.org/abs/1802.03601v4.

X. Wang, M. Long, J. Wang, and M. I. Jordan. Transferable Calibration with Lower Bias and Variance in Domain Adaptation. *arXiv*, 7 2020b. URL http://arxiv.org/abs/2007.08259.

A. G. Wilson and P. Izmailov. Bayesian Deep Learning and a Probabilistic Perspective of Generalization. *arXiv*, 2 2020. URL http://arxiv.org/abs/2002.08791.

Q. Xie, Z. Dai, E. Hovy, M.-T. Luong, and Q. V. Le. Unsupervised Data Augmentation for Consistency Training. *arXiv*, 4 2019a. URL http://arxiv.org/abs/1904.12848.

Q. Xie, M.-T. Luong, E. Hovy, and Q. V. Le. Self-training with Noisy Student improves ImageNet classification. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 10684–10695, 11 2019b. URL http://arxiv.org/abs/1911.04252.

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] Our contributions are in improving algorithms for robustness and adaptation, rather than any specific application. We expect the broader impacts of our work to be similar to those of prior works that are concerned with robustness and adaptation, and do not see any negative societal impacts specific to our work.
   (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? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
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] Included in supplemental materials.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See appendix A.1
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] See expanded results in appendix B with spread across different corruptions.
   (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.1
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] All datasets are cited in the text of the experiments section.
   (b) Did you mention the license of the assets? [Yes] See appendix A.3.
   (c) Did you include any new assets either in the supplemental material or as a URL? [No]
   (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? [No] We use only standard, widely used datasets in our experiments.
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]

A Experimental Details

A.1 Training Details

Model Training: Following TENT [Wang et al., 2020a], we use ResNet26 models [He et al., 2016a] for all experiments besides the ImageNet experiments. For ImageNet, we use ResNet50v2 [He et al., 2016b]. All methods utilizing ensembles use 10 different models in the ensemble.

For all ResNet26 experiments, we train all models and obtain posteriors using SWAG-D [Maddox et al., 2019] and use the same hyperparameters as provided by the authors for training a WideResNet28x10. For methods that do not include the posterior density term, we simply use the mean of the learned SWAG-D posterior, which simply corresponds to the solution obtained by Stochastic Weight Averaging (SWA) [Izmailov et al., 2018]. We train for 300 epochs on each training dataset, starting to collect iterates for SWA/SWAG starting from epoch 160. We use SGD with momentum for all experiments, with a learning rate schedule given by 0.1 for the first 150 epochs, decaying linearly down to 0.01 until epoch 270, then remaining at 0.01 for the remaining 30 epochs.

For ResNet50v2 experiments on ImageNet, we first train for 90 epochs using the example training script at `https://github.com/deepmind/dm-haiku/tree/master/examples/imagenet`. We then train for an additional 10 epochs with a constant learning rate of 0.0001 to collect iterates for SWAG, collecting 4 iterates per epoch.

Adaptation Details: For test time adaptation on the small scale experiments with ResNet26, we use the same hyperparameters described in TENT [Wang et al., 2020a]. using learning rate 0.001 and batch size 128, using SGD with momentum. For the ImageNet experiments with ResNet50v2, we tune the learning rate for accuracy using the validation corruptions in ImageNet-C, and keep the batch size fixed at 64. For TENT, we found a learning rate 0.001 to perform the best, while for our method BACS, we found a smaller learning rate of 0.0001 to perform the best.

During adaptation with BACS, according to equation 8, we would optimize each model with the objective

$$\max_{\theta} \log q_i(\theta) - \frac{\tilde{\alpha}}{m} \sum_{j=1}^{m} -H(Y|\tilde{\mathbf{x}}_j, \theta). \tag{9}$$

where $q_i(\theta)$ is a diagonal Gaussian posterior density obtained by SWAG-D, and $\tilde{\alpha}$ is a hyperparameter controlling how much to weight the entropy minimization objective against the posterior term. We initialize from the mean of the SWAG posterior before adaptation. In practice, we also rewrite the objective as

$$\max_{\theta} \beta \log q_i(\theta) - \frac{1}{m} \sum_{j=1}^{m} -H(Y|\tilde{\mathbf{x}}_j, \theta), \tag{10}$$

where $\beta = \frac{1}{\tilde{\alpha}}$, and use minibatches of test inputs for the entropy minimization term. For all small-experiments with ResNet26, we use $\beta = 0.0001$. For the ImageNet experiments, we jointly tuned β with the learning rate using the extra validation corruptions and selected $\beta = 0.0003$.

Compute Resources: The initial training for each ResNet50 model on ImageNet used a Google Cloud TPU v3 instance with 8 cores, each taking approximately 20 hours to train. In total, as we needed to train 10 models for the ensemble, this utilized 200 hours of compute with TPU v3 instances with 8 cores each.

All other model training and all the adaptation were run using local GPU servers with Nvidia Titan RTX GPUs. We estimate model training time to be around 200 hours in total for all ResNet26 models (10 models for each of CIFAR10, CIFAR100, and SVHN and an estimated 6 hour training time with on one GPU). For adaptation and evaluation, we estimate the total time needed for the ImageNet-Corrupted experiments, as these would take up the bulk of the time spent on adaptation. For each of the 15 corruption types and 5 corruption levels, we estimate the total time it takes to adapt each model using entropy minimization for one epoch, evaluate the model without any adaptation, evaluate the model with adapted batchnorm statistics, and evaluate each model after entropy minimization to be 10 minutes. With 10 models for each corruption, this totals 125 hours of GPU time to compile the ImageNet Corrupted results, which take up the bulk of the time needed for adaptation and evaluation in our experiments.

	CIFAR10-C				CIFAR100-C			
Method	Acc	NLL	Brier	ECE	Acc	NLL	Brier	ECE
TENT	84.52	0.7248	0.2289	0.09914	64.37	3.265	0.6111	0.2759
BACS (ours)	87.43	0.4242	0.1845	0.02882	67.28	1.323	0.4435	0.0500

Table 4: **CIFAR-10/100 Corrupted Online** results at the highest level of corruption, averaged over all corruption types at severity level 5 (the most severe level). While online adaptation performs slightly worse than offline adaptation, our method BACS still provides substantial improvements over TENT, ensembles with BN adaptation and other baselines.

A.2 Metrics

We compute Brier score for a probability vector $p(y|x_n)$ and true label y_n as

$$\sum_{y \in \mathcal{Y}} (p(y|x_n) - \delta(y - y_n))^2. \tag{11}$$

To compute expected calibration error (ECE), we use 20 bins. We order all predictions by confidence and aggregate them into 20 bins, each with the same number of data points, instead of using fixed windows for the buckets. Given the bins B_i, we then compute ECE as

$$\text{ECE} = \frac{1}{20} \sum_{i=1}^{20} |\text{acc}(B_i) - \text{conf}(B_i)|, \tag{12}$$

where acc and conf compute the average accuracies and confidences within the predictions in each bucket.

A.3 Dataset details

Models were trained on CIFAR10/100 [Krizhevsky, 2012], ImageNet [Deng et al., 2009], CIFAR10/100-Corrupted and Imagenet-Corrupted [Hendrycks and Dietterich, 2019], STL10 [Coates et al., 2011], SVHN [Netzer et al., 2011], MNIST [LeCun et al., 2010].

For the corrupted datasets, we report results averaged over all 15 standard corruption types in [Hendrycks and Dietterich, 2019], using the 4 extra corruption types for validation.

CIFAR10-Corrupted, STL10, SVHN, and MNIST were downloaded using Tensorflow Datasets. The Imagenet training data was directly downloaded from `www.image-net.org`), while CIFAR-100-Corrupted and Imagenet-Corrupted were downloaded from the author-released images at `zenodo.org/record/3555552` and `https://zenodo.org/record/2235448` respectively.

For Imagenet-C, we initially ran experiments and performed hyperparameter tuning using the datasets generated through Tensorflow Datasets, which recomputed the the corrupted images instead of simply downloading the images released by the original authors. However, performance on the TFDS version of ImageNet-C (see appendix B.3) is not directly comparable to that using the official released dataset, with overall results being substantially stronger with the TFDS version. This discrepancy is also noted in Ford et al. [2019], who postulated the performance differences are due to the extra JPEG compression used for the officially released dataset making the tasks more difficult.

We were unable to find licensing information for CIFAR10, CIFAR100 or STL10. Imagenet is released under a custom license stipulating non-commercial research and educational use only (see `www.image-net.org/download`). SVHN is released with a note stating it should be used for non-commercial uses only. MNIST is released under the CC BY-SA 3.0 license. CIFAR10/100-Corrupted and Imagenet-Corrupted are released under the CC BY 4.0 License.

B Additional Experimental Results

B.1 Online Evaluation

For the results in the main paper, all methods that adapted to the test distribution were evaluated in an offline setting, where each method could access the full test dataset before making any predictions.

	CIFAR10				CIFAR100			
Method	Acc	NLL	Brier	ECE	Acc	NLL	Brier	ECE
Vanilla	95.50	0.1715	0.07252	0.02549	77.88	1.023	0.3389	0.1198
BN Adapt	95.49	0.1767	0.07303	0.02588	77.88	1.071	0.3437	0.1266
TENT	95.48	0.1788	0.07662	0.03180	77.86	1.083	0.3458	0.1322
BACS (MAP)	95.40	0.1827	0.07734	0.02884	77.81	1.1388	0.3494	0.1412
Vanilla ensemble	96.07	**0.122**	**0.05835**	0.009644	80.34	**0.7182**	**0.2711**	**0.04254**
Ensemble BN Adapt	**96.09**	0.1244	0.05834	**0.009433**	80.46	0.7312	0.2720	0.04438
TENT Ensemble	96.08	0.1240	0.05869	0.01065	**80.68**	0.7341	0.2719	0.05352
BACS (ours)	95.98	0.1340	0.06154	0.01091	80.32	0.7428	0.2754	0.05051

Table 6: **CIFAR-10/100 In-distribution results.** We evaluate all methods on the standard (uncorrupted) test sets.

For BACS, this involves first making one pass through the test data to update the network for one epoch of optimization (or multiple epochs when specified), then making one more pass through the dataset to make predictions with the fully adapted network.

As there might be scenarios where we do not have access to all the test data from a particular distribution at once, we also include experiments on the corrupted datasets evaluating methods in an online setting, where we adapt the network and make predictions in a single pass through the dataset. The online procedure is more computationally efficient (requiring one fewer forward pass per batch) and does not require the model to wait for all the data to arrive before making predictions. Note that all adaptive methods we evaluate still require access to batches of test data to make updates, and we utilize the same adaptation hyperparameters as with the offline experiments. For online experiments, we fix the ordering on the test dataset for all methods.

We show results in the online setting for TENT and BACS in Tables 4 and 5. Compared to the offline results in Tables 1 and 2, online BACS generally performs worse than offline BACS, but still outperforms all baselines in accuracy, NLL, and Brier score.

Method	Acc	NLL	Brier	ECE
TENT	54.12	2.330	0.6005	0.09544
BACS (MAP)	56.13	2.171	0.5771	0.07055
TENT Ensemble	60.22	2.093	0.5702	0.1828
BACS (ours)	61.81	1.911	0.5369	0.1503

Table 5: **ImageNet-C Online** results averaged over all corruption types and levels. Again, online adaptation performs worse compared to offline adaptation, but online BACS still outperforms one TENT (as well as other baseliens) in accuracy, NLL, and Brier score.

B.2 In Distribution Results

At test time, it is also possible that the distribution we encounter is actually the same as training, though we would not necessarily know a priori. We thus also evaluate the performance of different adaptive methods when there is no distribution shift at test time in Table 6. We see that BACS does slightly underperform relative to ensembles without adaptation and ensembles with batch-norm adaptation.

B.3 Expanded ImageNet-C and CIFAR-C Results

In addition to measuring the average accuracy across the ImageNet-C corruptions, we also report results using the *mean corruption error* (mCE) [Hendrycks and Dietterich, 2019], which normalizes the per-corruption error rates using the performance of an AlexNet model as a baseline before averaging across corruption types. We include these results in Table 7.

We include expanded experimental results for ImageNet-Corrupted in Figure 7. For each metric, we now use box plots to show the variability of results over different corruption types, separating out results at each level of corruption.

Method	mCE
Vanilla	73.42
BN Adapt	61.24
TENT	54.37
BACS (MAP)	53.13
Vanilla Ensemble	67.98
Ensemble BN Adapt	52.70
TENT Ensemble	47.50
BACS (ours)	**46.78**

Table 7: **ImageNet-C mCE** results. BACS (ours) outperforms all other methods, while our ablation without ensembles BACS (MAP) outperforms all non-ensembled methods.

Across all corruption levels, BACS consistently performs the best in accuracy, NLL and Brier score (with the exception being the mean accuracy at the highest level of corruption, where a single corruption where BACS has very low accuracy drags down the mean, though median performance is still higher than all baselines).

finally, we include expanded experimental results for CIFAR10 and CIFAR100 Corrupted in Figures 5 and 6.

We also include experimental results using the TFDS version of ImageNet-C in Table 8, which we note has substantially better results overall than the officially released dataset. We also included boxplots for the Tensorflow Datasets version of ImageNet-Corrupted in Figure 8, where we see BACS performs the best in accuracy, NLL, and Brier scores at each level of corruption.

Method	Acc	NLL	Brier	ECE
Vanilla	45.64	2.867	0.6750	**0.0614**
BN Adapt	55.97	2.363	0.6050	0.1606
TENT	60.82	1.803	0.522	0.0313
BACS (MAP)	61.96	1.712	0.5022	0.0294
Vanilla Ensemble	50.48	2.519	0.6211	0.07878
Ensemble BN Adapt	62.18	2.137	0.5802	0.2438
TENT Ensemble	65.83	1.586	0.4726	0.0956
BACS (ours)	**66.64**	**1.492**	**0.4548**	0.0735

Table 8: **ImageNet-C (TFDS)** results averaged over all corruption types and levels. BACS again substantially outperforms all baselines in accuracy, NLL, and Brier score.

B.4 Domain Adaptation Results

We further evaluate BACS in additional small-scale experiments commonly evaluated for domain adaptation. We evaluate transferring from CIFAR10 to STL10 (using only the 9 overlapping classes), as well as a commonly used digit recognition task transferring from SVHN to MNIST.

CIFAR10 to STL10. We further evaluate all methods on a more natural distribution shift by considering evaluating a model trained CIFAR-10 and evaluated on STL-10 [Coates et al., 2011]. We only use the 9 overlapping classes in each dataset. In this setting, BACS and BN ensembles outperform the other methods.

Method	Acc	NLL	Brier	ECE
Vanilla	82.38	0.7825	0.2886	0.1185
BN adapt	83.72	0.7926	0.2709	0.1147
TENT	84.05	0.8831	0.2753	0.1216
Vanilla Ensemble	84.03	0.5360	0.2333	0.06561
Ensemble BN Adapt	85.40	0.5301	0.2195	**0.05949**
TENT Ensemble	85.10	0.5337	0.2270	0.06844
BACS (ours)	**85.47**	**0.5284**	**0.2184**	0.06114

Table 9: **CIFAR10 to STL10**: Source model trained on CIFAR10 and evaluated on STL10 test set the (with nonoverlapping classes removed during both training and test). Here, the ensembled approaches perform the best overall and perform similarly to one another in uncertainty estimation.

While TENT is able to improve accuracy slightly over the non-ensembled baselines, we again see that uncertainty estimates degrade as entropy is minimized (as seen by the increases in NLL, Brier, and ECE scores). In contrast, BACS still remains well-calibrated, again emphasizing the importance of Bayesian marginalization for reliable uncertainty estimation when adapting models via entropy minimization.

SVHN to MNIST transfer. We also evaluate our method on a digit classification task, transferring a model trained on SVHN [Netzer et al., 2011] to MNIST [LeCun et al., 2010], which is commonly studied as a the domain adaptation setting [Ganin et al., 2015, Liang et al., 2020]. We find that BACS is able to significantly outperform the models with no adaptation as well as ensembles with batch norm adaptation in accuracy, NLL, and Brier score.

We find that this severe distribution shift requires larger changes in parameters to effectively adapt, as seen by the substantial improvements between optimizing for one epoch and optimizing for 10. We also find that the approximate posterior term used in our method overly constrains the network during adaptation, as removing the regularizer from the method (denoted BACS-posterior in the table) results in the best performance in all metrics. We note that all ensemble methods perform much better than non-ensembled methods in terms of calibration, emphasizing the benefits of marginalizing over different models for uncertainty estimation when adapting via entropy minimization.

We note that our method and the compared baselines are focused on improving results in robustness settings, and do not necessarily obtain state-of-the-art performance in common domain adaptation

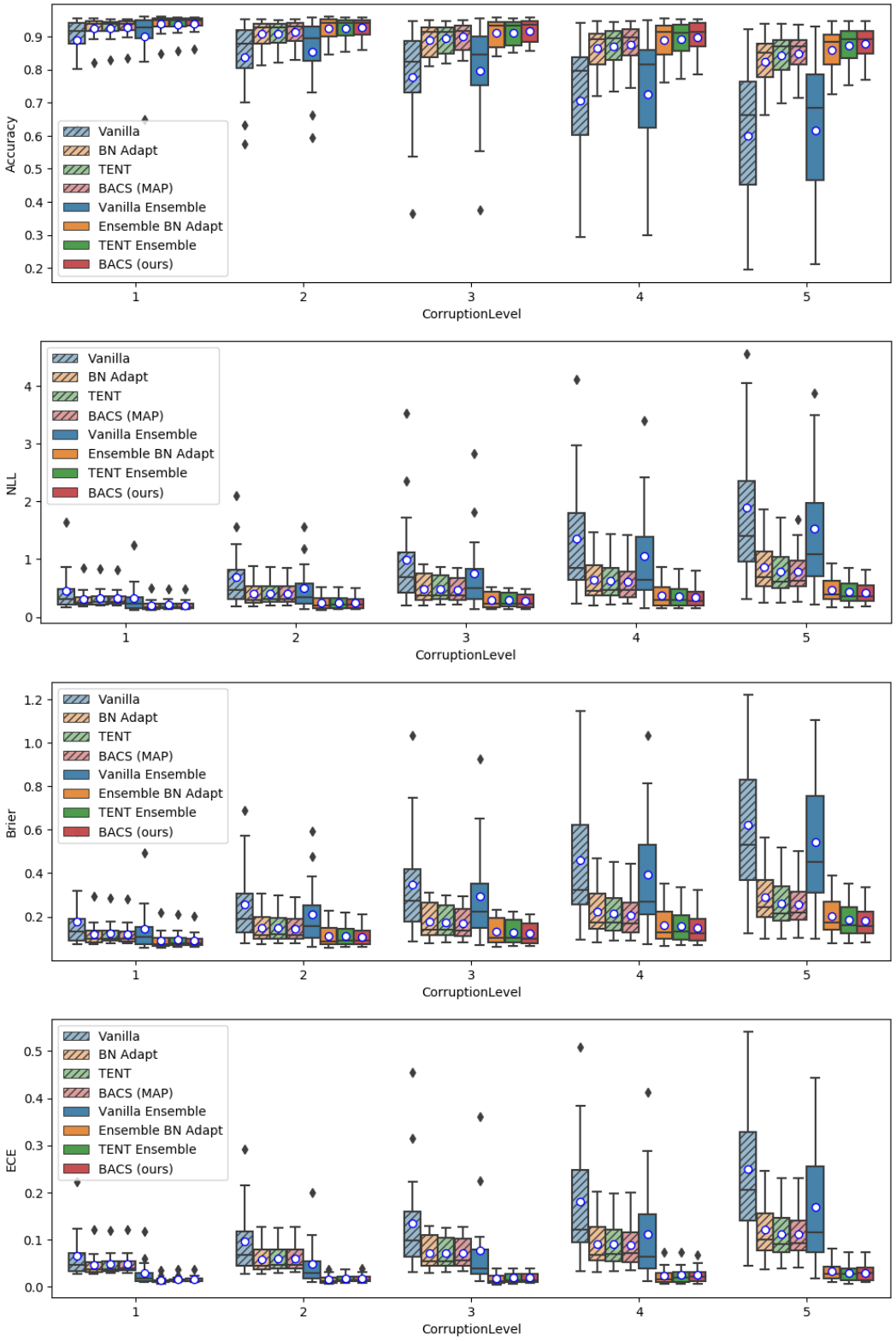

Figure 5: **CIFAR10 Corrupted Results**. For each corruption level, we use boxplots the spread of results over the different individual corruption types. Boxes are drawn at the 25th and 75th percentiles, with the median being drawn as a line in the middle of the box and the mean being shown with white dots. The ends of the whiskers show the min and max across corruption types at the level (with black diamonds for outliers).

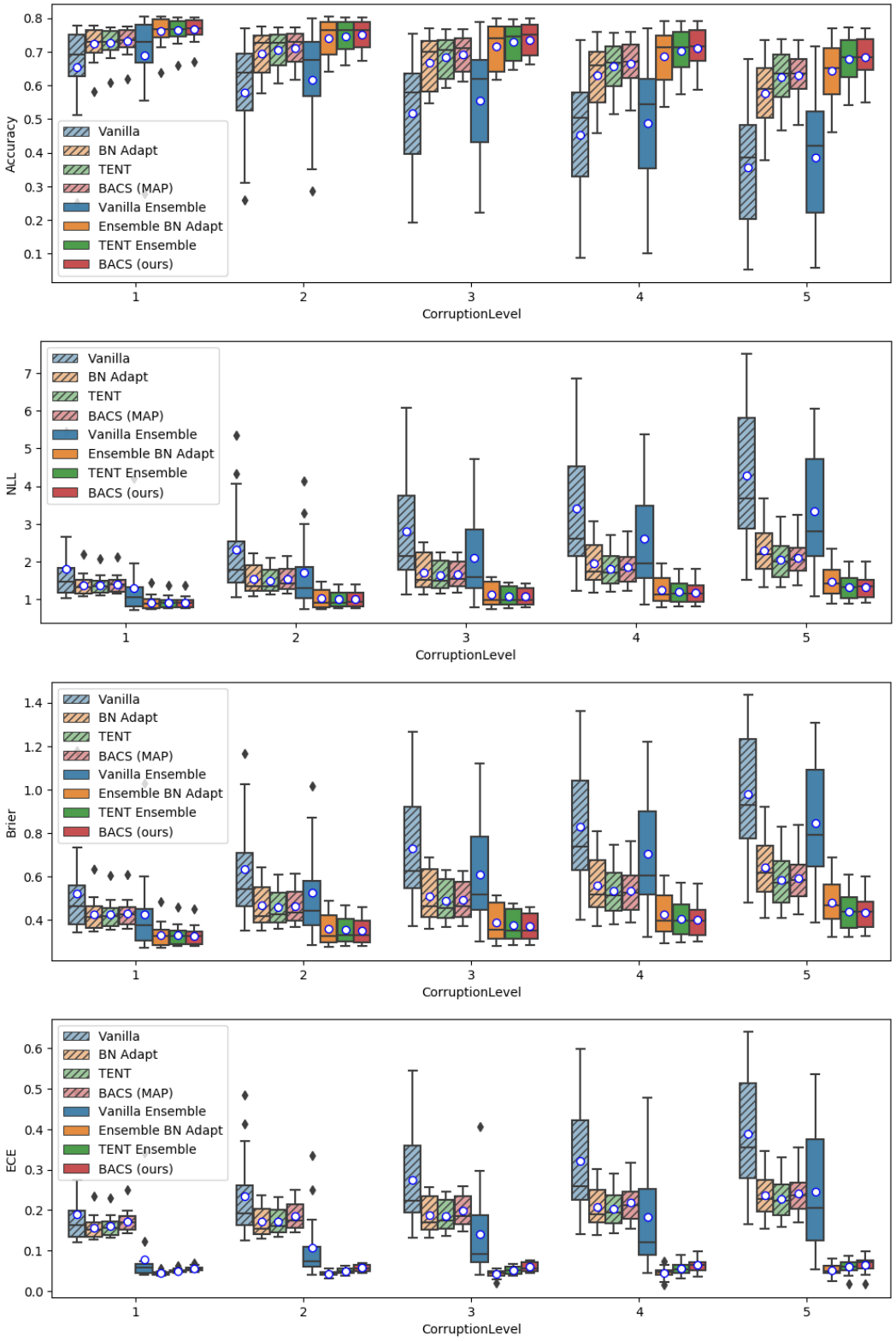

Figure 6: **CIFAR100 Corrupted Results**. For each corruption level, we use boxplots the spread of results over the different individual corruption types. Boxes are drawn at the 25th and 75th percentiles, with the median being drawn as a line in the middle of the box and the mean being shown with white dots. The ends of the whiskers show the min and max across corruption types at the level (with black diamonds for outliers).

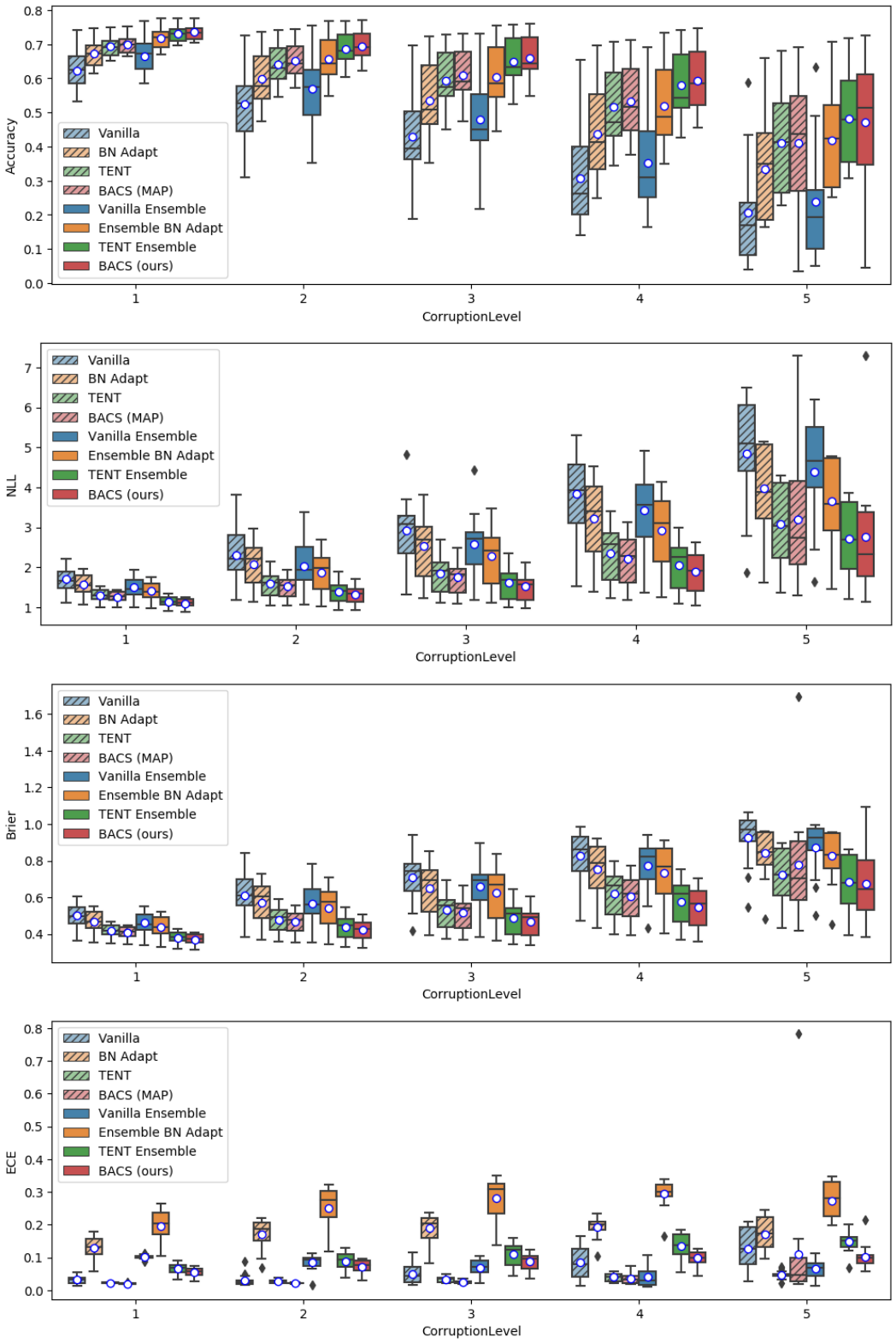

Figure 7: **ImageNet Corrupted Results**. For each corruption level, we use boxplots the spread of results over the different individual corruption types. Boxes are drawn at the 25th and 75th percentiles, with the median being drawn as a line in the middle of the box and the mean being shown with white dots. The ends of the whiskers show the min and max across corruption types at the level (with black diamonds for outliers).

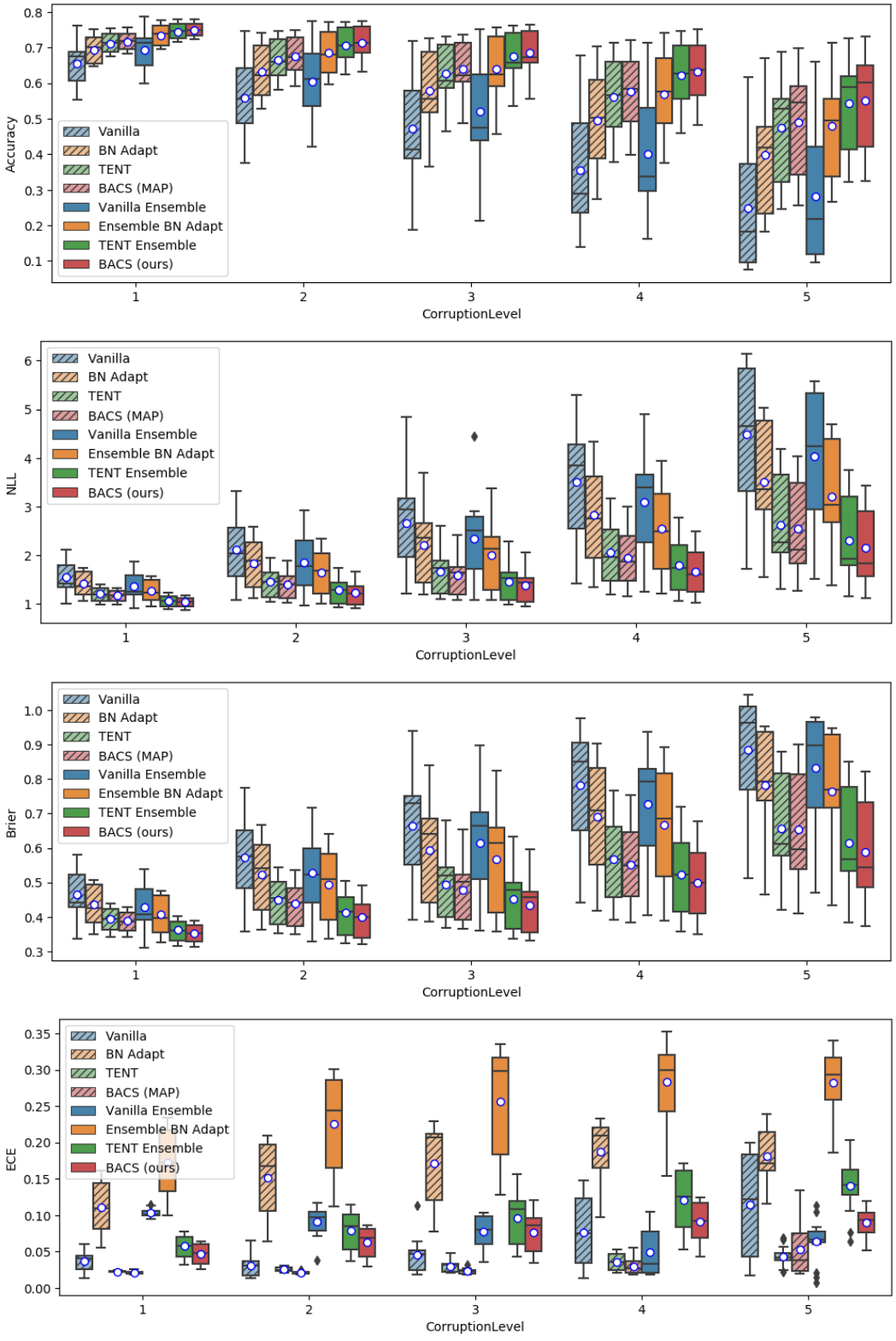

Figure 8: **ImageNet Corrupted (TFDS) Results**. For each corruption level, we use boxplots the spread of results over the different individual corruption types. Boxes are drawn at the 25th and 75th percentiles, with the median being drawn as a line in the middle of the box and the mean being shown with white dots. The ends of the whiskers show the min and max across corruption types at the level (with black diamonds for outliers). At each corruption level, BACS (ours) outperforms all baselines in accuracy, NLL, and Brier score.

Method	Acc	NLL	Brier	ECE
Vanilla	76.99	0.9967	0.3645	0.1290
Vanilla ensemble	79.52	0.7368	0.3024	0.03856
BN Adapt	73.43	1.4502	0.4453	0.1878
Ensemble BN Adapt	76.84	0.9444	0.3442	0.07855
TENT (1 epoch)	77.24	1.321	0.3896	0.1605
TENT (10 epochs)	85.53	0.9554	0.2035	0.1166
TENT Ensemble (1 epoch)	79.48	0.8869	0.3149	0.09254
TENT Ensemble (10 epochs)	86.89	0.5784	0.2035	0.06384
BACS (ours) (1 epoch)	84.14	0.6024	0.2285	0.05595
BACS (ours) (10 epochs)	86.32	0.5553	0.2094	0.05133
BACS - posterior (1 epoch)	87.28	0.4214	0.1650	0.02603
BACS - posterior (10 epochs)	**93.03**	**0.2371**	**0.0965**	**0.02404**

Table 10: **SVHN to MNIST**: In this domain adaptation setting, we find adapting batch norm statistics alone hurts performance compared to the unadapated models, but methods utilizing entropy minimization are able to improve substantially in accuracy.

settings when compared to algorithms specifically designed for these problems. For example, Liang et al. [2020] introduce a source-free domain adaptation algorithm (that also does not require access to the training data during adaptation) and report 99% accuracy transferring from SVHN to MNIST, though results are not directly comparable due to architecture and training differences. In contrast to typical source-free domain adaptation algorithms, our algorithm also focuses on improving uncertainty estimation, does not require multiple epochs of optimization during adaptation, and is amenable to online evaluations where predictions need to be made before seeing the entirety of the test data.