On the Out-of-Distribution Coverage of Combining Split Conformal Prediction and Bayesian Deep Learning

Anonymous authors
Paper under double-blind review

Abstract

Bayesian deep learning and conformal prediction are two methods that have been used to convey uncertainty and increase safety in machine learning systems. We focus on combining Bayesian deep learning with split conformal prediction and how this combination effects out-of-distribution coverage; particularly in the case of multiclass image classification. We suggest that if the model is generally underconfident on the calibration set, then the resultant conformal sets may exhibit worse out-of-distribution coverage compared to simple predictive credible sets. Conversely, if the model is overconfident on the calibration set, the use of conformal prediction may improve out-of-distribution coverage. We evaluate prediction sets as a result of combining split conformal methods and neural networks trained with (i) stochastic gradient descent, (ii) deep ensembles, and (iii) mean-field variational inference. Our results suggest that combining Bayesian deep learning models with split conformal prediction can, in some cases, cause unintended consequences such as reducing out-of-distribution coverage.

1 Introduction

Bayesian deep learning and conformal prediction are two paradigms that have been used to represent uncertainty and increase trust in machine learning systems. Bayesian deep learning attempts to endow deep learning models with the ability to represent predictive uncertainty. These models often provide more calibrated outputs on both in-distribution and out-of-distribution data by approximating epistemic and aleatoric uncertainty. Conformal prediction is a method that takes (possibly uncalibrated) predicted probabilities and produces prediction sets that follow attractive guarantees; namely marginal coverage for exchangeable data (Vovk et al., 2005). On out-of-distribution data, however, this guarantee no longer holds unless knowledge of the distribution shift is known *a priori* (Tibshirani et al., 2019; Barber et al., 2023). It is natural then to consider combining conformal prediction with Bayesian deep learning models to try and enjoy in-distribution guarantees and better calibration on out-of-distribution data. In fact, combination of the two methods has been used to correct for misspecifications in the Bayesian modeling process (Dewolf et al., 2023; Stanton et al., 2023), thereby improving coverage and thus trust in the broader machine learning system. However, we suggest that application of both methods in certain scenarios may be counterproductive and worsen performance on out-of-distribution examples (see Figure 1). To investigate this suggestion we evaluate the combination of Bayesian deep learning models and split conformal prediction methods on image classification tasks, where some of the test inputs are out-of-distribution. Our primary contributions are as follows:

(i) Offer an explanation of how the under- or over-confidence of a model can predict when conformal prediction may worsen or improve out-of distribution coverage.

(ii) Evidence to support the explanation in (i) in the form of two empirical evaluations focusing on the out-of-distribution coverage of Bayesian deep learning combined with split conformal prediction.

(iii) Practical recommendations for those using Bayesian deep learning and conformal prediction to increase the safety of their machine learning systems.

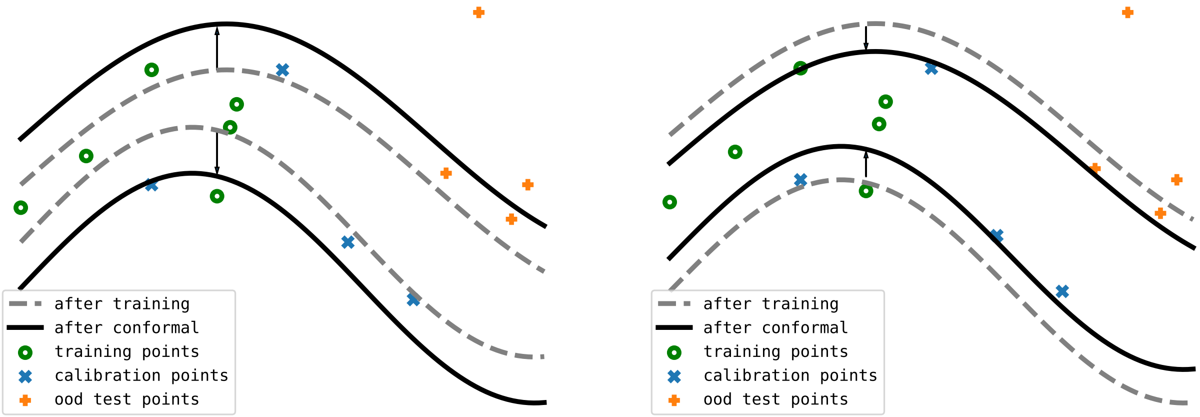


Figure 1: A conceptual illustration of how conformal prediction can help or harm out-of-distribution coverage for an error tolerance of 0.25. On the left is a conceptual illustration of how conformal prediction can make the overall machine learning system less confident after conformalizing a model that is overly confident on the calibration set. As a consequence, it gains coverage on out-of-distribution examples. The right conceptualizes the opposite direction and illustrates how conformal prediction can reduce coverage on out-of-distribution examples.

2 Preliminaries

2.1 Related Work

The analysis of combining more traditional Bayesian models with conformal prediction has been studied in Wasserman (2011) and Hoff (2021). Combining *full* (not *split*) Bayesian *deep learning* with conformal *regression* has been studied in the context of efficient computation of conformal Bayesian sets and Bayesian optimization (Fong and Holmes, 2021; Stanton et al., 2023). Both of these works consider settings with non-exchangeable data but assume *a priori* knowledge on the type of distribution shift. Theoretical foundations of using conformal prediction with non-exchangeable data can admit very powerful guarantees but have thus far assumed *a priori* knowledge about the distribution shift (Tibshirani et al., 2019; Angelopoulos et al., 2022; Fannjiang et al., 2022; Barber et al., 2023). Additionally, other powerful conformal algorithms have been posed to deal with out-of-distribution examples but require an additional model to detect those out-of-distribution examples (Angelopoulos et al., 2021). Dewolf et al. (2023) evaluate conformal prediction combined with Bayesian deep learning models but for regression tasks and with relatively low-dimensional inputs (no greater than 280 features). Additionally, Kompa et al. (2021) look at the coverage of prediction sets from various Bayesian deep learning methods and mention conformal prediction but do not include conformal prediction in their empirical analysis. We are not aware of a study examining the interaction between conformal prediction and Bayesian deep learning methods as it relates to the coverage of unknown, out-of-distribution examples for a task that deep learning models have excelled—image classification. We not only evaluate the combination of Bayesian deep learning and conformal prediction on image classification but also provide an intuitive explanation as to why, in certain scenarios, conformal prediction can actually *harm* out-of-distribution coverage we would otherwise see with non-conformal predictive sets.

2.2 Calibration

Modeling decisions factor into the *calibration* of the resultant predicted probabilities. Consider a vector of predicted probabilities produced by our model for input $\mathbf{x}$,

$$\hat{\mathbf{p}}(\mathbf{x}) = (\hat{p}_1(\mathbf{x}), ..., \hat{p}_K(\mathbf{x}))\,, \quad \sum_{k=1}^{K} \hat{p}_i(\mathbf{x}) = 1 \;\; \forall \mathbf{x} \in \boldsymbol{\mathcal{X}}$$

where $\boldsymbol{\mathcal{X}}$ is the sample space for $\mathbf{x}$ and K is the number of possible labels y can assume (i.e. the labels that can be assigned to $\mathbf{x}$). For a model to be *well-calibrated* (Zadrozny and Elkan, 2002; Vasilev and D'yakonov, 2023) the following must hold,

$$\mathbb{P}(y = i | \hat{p}_i(\mathbf{x}) = p) = p, \;\; \forall i \in \boldsymbol{\mathcal{Y}}, \forall p \in [0, 1],$$

where the probability is taken over the joint data distribution $p(\mathbf{x}, y)$ and $\boldsymbol{\mathcal{Y}}$ is the sample space for labels y. This says that on average over $p(\mathbf{x}, y)$, the predicted probability assigned to each class (not just the one assigned highest probability) represents the true probability of that class being the true label. If a model is well-calibrated, then we should be able to create prediction sets that achieve marginal coverage for an error tolerance α by creating a predictive credible set which we later abbreviate as *cred*. To do so, for each model output $\hat{\mathbf{p}}(\mathbf{x})$, we order the probabilities therein from greatest to least and continue adding the corresponding labels until the cumulative probability mass just exceeds $1 - \alpha$.

2.3 Bayesian Deep Learning

One important reason for miscalibration is the abstention of representing epistemic uncertainty (Wilson and Izmailov, 2020). In order to create more calibrated predicted probabilities for neural networks, a popular approach is Bayesian deep learning, where a major goal is to faithfully represent parameter uncertainty (a type of epistemic uncertainty). Parameter uncertainty[1] arises due to many different configurations of the model weights that can explain the training data, which happens especially in deep learning where we have highly expressive models (Wilson, 2020). Hence, Bayesian deep learning is an attractive approach to helping achieve calibration in practice, even on out-of-distribution examples. However, in order for the Bayesian approach to perform well a few important assumptions are required:

- Our observation model relating weights $\mathbf{w}$ and inputs $\mathbf{x}$ to labels y, $p(y|\mathbf{x}, \mathbf{w})$, is well-specified. This means that our observation model has the ability to produce the true data generating function.
- Our prior over weights $p(\mathbf{w})$ is well-specified. This means that when it is paired with the observation model, we induce a distribution over functions $p(f)$ that places sufficient probability on the true data generating function (MacKay, 2003).
- We often need to *approximate* the posterior distribution of the weights with respect to the training dataset, $p(\mathbf{w}|\mathcal{D})$, for many interesting observation models, including neural networks.

In this study we use a class of observation models (convolutional neural networks) and priors (zero-mean Gaussians) that have been shown to produce good inductive biases for image classification tasks (Wilson and Izmailov, 2020; Izmailov et al., 2021); and focus on varying the method of approximating the posterior over parameters.

2.4 Conformal Prediction

We restrict our attention to a subset of conformal prediction methods: *split* (or inductive) conformal prediction (Vovk, 2012). Split conformal prediction requires an extra held-out calibration set (which we denote $\mathcal{D}_{\text{cal}}$) to be used during a "calibration step". Importantly, split conformal prediction assumes exchangeability of the calibration and test set data. By allowing our final output to be a prediction set $\mathbb{Y} \subseteq (1, ..., K)$, conformal prediction guarantees the true class y to be included on average with probability (confidence) $1 - \alpha$:

$$\Pr(y \in \mathbb{Y}) \geq 1 - \alpha, \tag{3}$$

where α is a user-chosen error tolerance and Pr reads "probability that". This guarantee is *marginal* in the sense that it is guaranteed on average with respect to the data distribution $p(\mathbf{x}, y)$ as well as the distribution over possible calibration sets we could have selected. Not only does split conformal prediction guarantee

---

[1] *Parameter* is a loaded term, and context is needed to precisely understand what it means. In this case, we mean the *weights* of the neural network; not the parameters that may govern the distribution over such weights.

the inequality in (3) but also upper bounds the coverage given that the scores s_i have a continuous joint distribution. In particular, let n_{cal} be the number of data pairs in the calibration set, then

$$\Pr(y \in \mathbb{Y}) \leq 1 - \alpha + \frac{1}{n_{\text{cal}} + 1}, \tag{4}$$

which is proved in Lei et al. (2018). Split conformal prediction methods work by first defining a *score* function that measures the disagreement between output probabilities $\hat{\mathbf{p}}(\mathbf{x})$ of a model and a label y. We denote a general score function as $s(\mathbf{x}, y)$, but it should be noted that the outputted scores depend on the underlying fitted model through the $\hat{\mathbf{p}}(\mathbf{x})$ it produces. The conformal method computes the scores on the held-out calibration set and then takes the

$$\lceil (1-\alpha)(1+\frac{1}{|\mathcal{D}_{\text{cal}}|}) \rceil\text{-quantile}$$

of those scores which we denote τ. Then, during test time, prediction sets are constructed by computing the score for each possible y_i, and including y_i into the prediction set if its score is less than or equal to τ. Informally, if the candidate label y_i produces a score that conforms to what we have seen on the calibration set, it is included.

3 Evaluation & Method Details

3.1 When Might Conformal Prediction Harm Out-of-Distribution Coverage?

While conformal prediction guarantees marginal coverage when the calibration data and test data are exchangeable, it loses its guarantees when encountering out-of-distribution data at test time[2]. Bayesian deep learning, on the other hand, has no such guarantees but has been shown to improve calibration (and thus coverage) on out-of-distribution inputs; a symptom of trying to quantify epistemic uncertainty. A natural desire, then, is to combine conformal prediction with Bayesian deep learning models in order to enjoy in-distribution guarantees while reaping the benefits of out-of-distribution calibration.

However, we suggest that blindly combining these methods can, in some cases, result in unintended consequences that hurt the coverage on out-of-distribution examples. The reasoning goes as follows: conformal prediction does whatever it needs to do to guarantee the *desired coverage* on in-distribution data; meaning it provides a lower bound (3) and an upper-bound (4). This means that if the model is generally overconfident on the calibration dataset (as is often the case with stochastic gradient descent methods; see Figure 2b), then conformal prediction creates a threshold τ that ultimately makes the prediction sets larger than they would otherwise be without conformal prediction. In this case, conformal prediction makes the overall machine learning system *less confident*, and it will actually assist when encountering out-of-distribution data. Conversely, if we have a model that is generally underconfident on the calibration dataset (as is often the case with mean-field variational inference; see Figure 2b), conformal prediction creates a threshold τ that makes the prediction sets *smaller* than they would otherwise be to bound the desired coverage tightly on in-distribution data. However, this could hinder coverage on out-of-distribution data by making the overall machine learning system more confident than it should be. Both cases are demonstrated in Figure 1. We next describe the particular methods used in our evaluation and how we measure the performance of their respective combinations.

3.2 Bayesian Deep Learning Methods

Consider a probabilistic conditional model $p(y, \mathbf{w}|\mathbf{x}) = p(y|\mathbf{x}, \mathbf{w})p(\mathbf{w})$ where $\mathbf{w}$ are the weights, $p(\mathbf{w})$ is a prior distribution, and $p(y|\mathbf{x}, \mathbf{w})$ is the probability of y given our weights $\mathbf{w}$ and fixed inputs $\mathbf{x}$. Denote the training dataset as $\mathcal{D}$. Then the posterior predictive distribution can be written as

$$p(y|\mathbf{x}, \mathcal{D}) = \int p(y|\mathbf{x}, \mathbf{w})p(\mathbf{w}|\mathcal{D}). \tag{5}$$

---

[2] This is true, unless, as mentioned before, knowledge is known *a priori* about the out-of-distribution data (Tibshirani et al., 2019; Barber et al., 2023)

We implement three methods for approximating (5), which is sometimes referred to as the *Bayesian model average* (Wilson and Izmailov, 2020).

Stochastic Gradient Descent (SGD) typically involves finding the maximum a posteriori (MAP) estimate for $\mathbf{w}$. In the context of (1), this means we approximate $p(\mathbf{w}|\mathcal{D}) \approx \hat{\mathbf{w}}_{\text{MAP}}$ where

$$\begin{aligned}\hat{\mathbf{w}}_{\text{MAP}} &= \underset{\mathbf{w}}{\operatorname{argmax}} \log p(\mathbf{w}|\mathcal{D}) \\ &= \underset{\mathbf{w}}{\operatorname{argmax}} \left(\log p(\mathcal{D}|\mathbf{w}) + \log p(\mathbf{w}) + \text{constant}\right)\end{aligned}$$

Neural networks trained via stochastic gradient descent have been found to often be uncalibrated by being overly confident in its predictions (see Figure 4), especially on out-of-distribution examples (Guo et al., 2017).

Deep ensembles (ENS) works by combining the outputs of multiple neural networks with different initializations (Lakshminarayanan et al., 2017). The idea is that the variation in their respective outputs can represent epistemic uncertainty. In this case, we implicitly approximate the posterior and simply average the hypotheses generated by each model in the ensemble:

$$p(y|\mathbf{x},\mathcal{D}) \approx \frac{1}{J}\sum_{j=1}^{J} p(y|\mathbf{x},\hat{\mathbf{w}}_{\text{MAP}_j})$$

where $\hat{\mathbf{w}}_{\text{MAP}_j}$ is the stochastic gradient solution for model j in the ensemble of J models.

Mean-field Variational Inference (MFV) seeks to approximate the posterior with a variational distribution (Blei et al., 2017):

$$p(\mathbf{w}|\mathcal{D}) \approx q(\mathbf{w}|\hat{\boldsymbol{\theta}}) \overset{(i)}{=} \prod_{i=1}^{m} q(w_i|\hat{\theta}_i) \tag{6}$$

where (i) comes from the mean-field assumption, and q is usually a simple distribution (e.g. a Gaussian). To make the approximation in (6), we maximize with respect to $\boldsymbol{\theta}$ a function equivalent to the KL divergence between $p(\mathbf{w}|\mathcal{D})$ and $q(\mathbf{w}|\boldsymbol{\theta})$ up to a constant. It is usually termed the *evidence lower bound* (ELBO). We can write the objective as

$$\begin{aligned}\hat{\boldsymbol{\theta}} &= \underset{\boldsymbol{\theta}}{\operatorname{argmax}} \text{ ELBO}(\boldsymbol{\theta},\mathcal{D}) \\ &= \underset{\boldsymbol{\theta}}{\operatorname{argmax}} \left(\underbrace{\mathbb{E}_{\mathbf{q}(\mathbf{w}|\boldsymbol{\theta})}[\log p(\mathcal{D}|\mathbf{w})]}_{\text{Expected log likelihood of data}} + \underbrace{\text{KL}[q(\mathbf{w}|\boldsymbol{\theta}) \,||\, p(\mathbf{w})]}_{\text{KL between variational and prior}} \right).\end{aligned} \tag{7}$$

The objective (7) is a tradeoff between two terms. The expected log likelihood of the data prefers $q(\cdot)$ place its mass on the maximum likelihood estimate while the KL divergence term prefers $q(\cdot)$ stay close to the prior (Blei et al., 2017). Bayesian neural networks for classification have been shown to be generally underconfident by overestimating aleatoric uncertainty (Kapoor et al., 2022). Indeed, we find this is the case for both our experiments (see Figure 4).

3.3 Conformal Prediction Methods

We implement two common split conformal prediction methods.

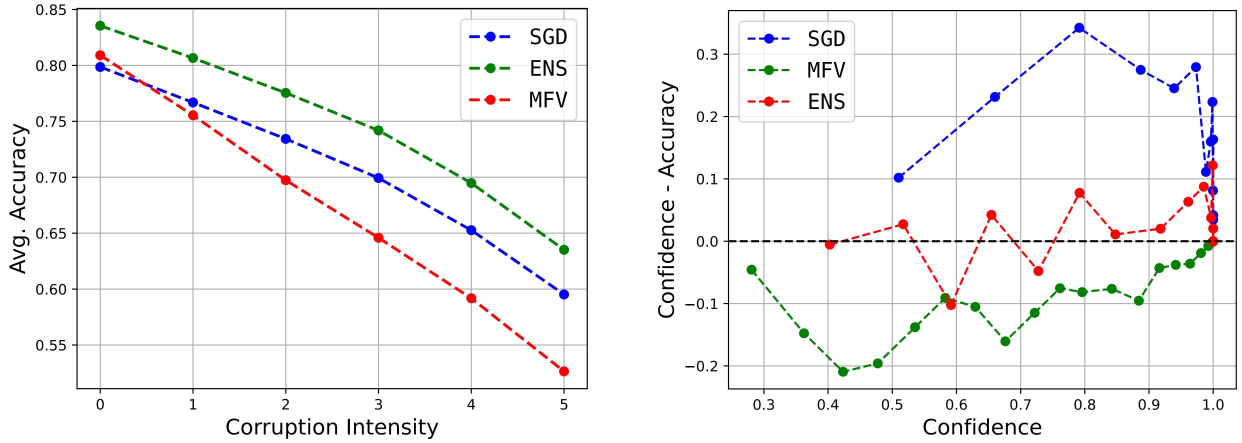
Threshold prediction sets (*thr*) uses the following score function (Sadinle et al., 2019):

$$s(\mathbf{x},y) = 1 - \hat{p}_y(\mathbf{x}).$$

That is, the score for an input $\mathbf{x}$ with true label y is one minus the probability mass the model assigns to the true label y. This procedure only takes into account the probability mass assigned to the *correct label.*

Adaptive prediction sets (*aps*) uses the following score function (Romano et al., 2020):

$$s(\mathbf{x},y) = \hat{p}_1(x) + \cdots + U\hat{p}_y(x),$$



(a) **CIFAR10** *accuracy* plot. The accuracy plot shows, for each corruption intensity, the average accuracy over all corrupted datasets at that intensity.

(b) **CIFAR10** *reliability* plot. The reliability plot is a result of evaluating on a calibration set which is taken from the original CIFAR10 test set.

Figure 2: The accuracy and reliability plots for the CIFAR10 experiment.

where $\hat{p}_1(x) \geq \cdots \geq \hat{p}_y(x)$ and U is a uniform random variable in $[0, 1]$ to break ties. That is, we order the probabilities in $\hat{\mathbf{p}}(\mathbf{x})$ from greatest to least and continue adding the probabilities, stopping after we reach the probability associated with the correct label y.

3.4 Evaluation Measures

A prediction set is any subset of possible labels. We would like a collection of prediction sets to have certain minimal properties. We would first like the collection to be *marginally covered*: given a user-specified error tolerance α, the average probability that the true class y is contained in the given prediction sets is $1 - \alpha$. We would also like each prediction set in the collection to be *small*: the prediction sets should contain as few labels as possible without losing coverage. Hence, we evaluate on the basis of the marginal coverage and average size of a collection of prediction sets emitted by applying split conformal prediction to Bayesian deep learning model outputs.

4 Experiments

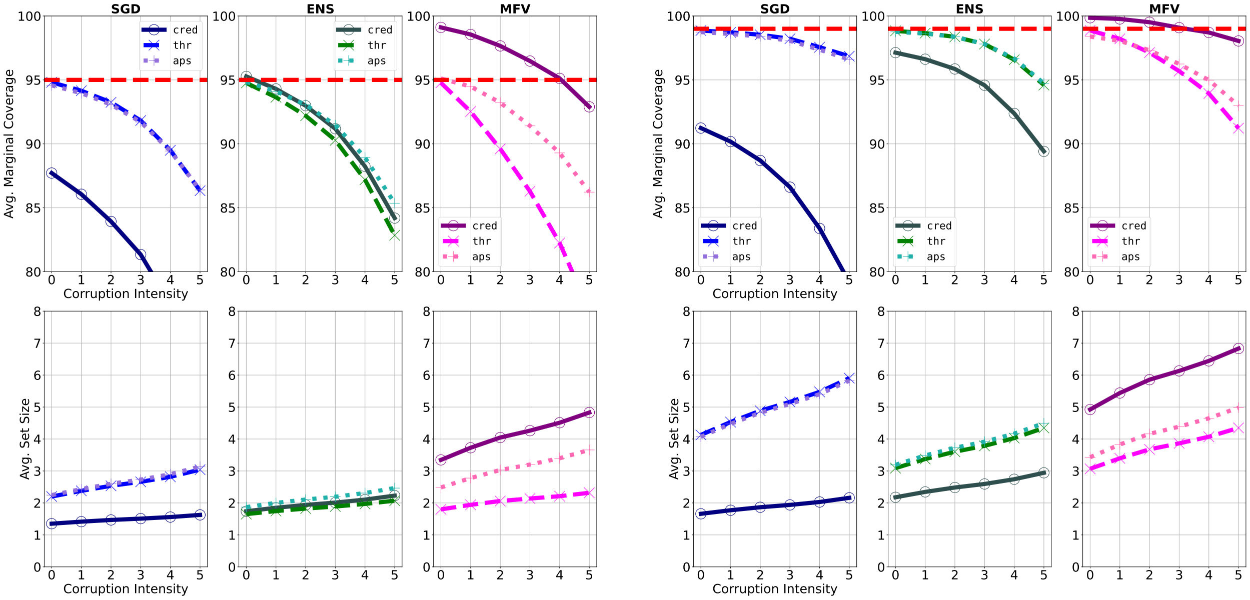
Our first experiments loosely follows the setup of Izmailov et al. (2021) and Ovadia et al. (2019). We train an AlexNet inspired model on CIFAR10 by means of stochastic gradient descent, deep ensembles, and mean-field variational inference (Krizhevsky et al., 2009; 2012). We then take 1000 examples (without replacement) from the CIFAR10 test set for a calibration set. We use this calibration set to determine thresholds τ that we use for conformal methods *thr* and *aps*, respectively. We then evaluate the marginal coverage and average size of the predictive credible sets (*cred*), *thr* sets, and *aps* sets that were produced for every single corrupted CIFAR10 dataset (Hendrycks and Dietterich, 2019) at every intensity[3].

4.1 CIFAR10-Corrupted

We first look at the reliability diagram of the three models on the calibration set in Figure 2b. According to this plot, stochastic gradient descent is generally overconfident and mean-field variational inference is generally underconfident. We thus suspect that conformal prediction set methods may *help* stochastic gradient descent in terms of out-of-distribution coverage, while *harm* mean-field variational inference in terms of out-of-distribution coverage. The average marginal coverage and average set size across datasets at each

---

[3] We take out those semantically similar images from the corrupted test set that we used for the calibration set to ensure no data leakage.



(a) An error tolerance of 0.05 (i.e. 0.95 desired coverage) which is denoted by the dashed horizontal red line.

(b) An error tolerance of 0.01 (i.e. 0.99 desired coverage) which is denoted by the dashed horizontal red line.

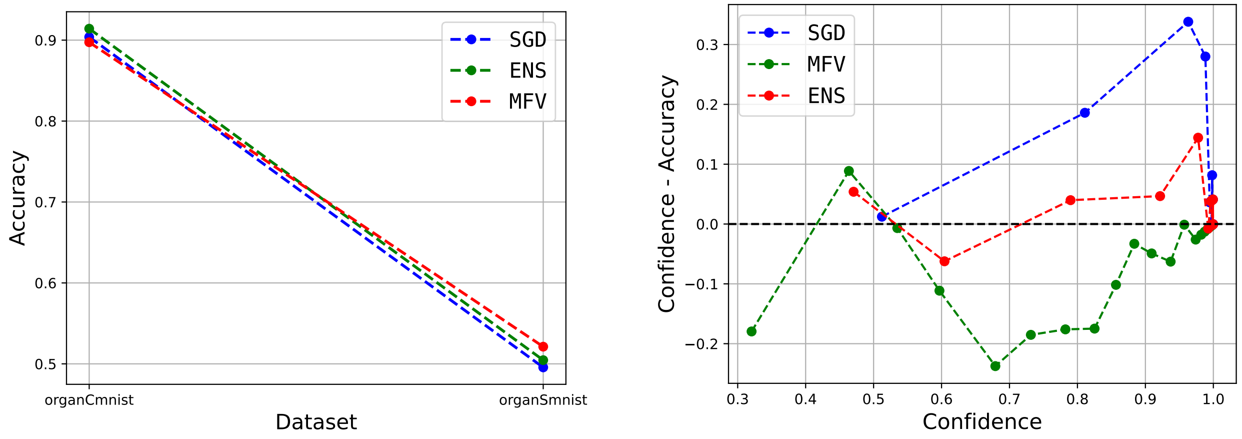
Figure 3: **CIFAR10**: Average marginal coverage and average set size across datasets at each corruption intensity from CIFAR10-Corrupted for error tolerances 0.05 and 0.01. For each figure, each column is for a particular method used to train the neural network – stochastic gradient descent, deep ensembles, and mean-field variational inference.

intensity is shown in Figure 3 for error tolerances 0.05 and 0.01. The average accuracies of each deep learning method are shown in Figure 2a.

Some general remarks are:

- Mean-field variational inference combined with simple predictive credible sets achieve better coverage than its combination with conformal methods. Moreover, it maintains coverage the best out of any other combination of methods for both error tolerances at the expense of slightly larger set sizes.
- The addition of conformal methods with stochastic gradient descent drastically improves its coverage on both in- and out-of-distribution examples.
- The addition of conformal methods with deep ensembles improves its coverage on both in- and out-of-distribution examples at the 0.01 error tolerance but only improves coverage slightly at the 0.05 error tolerance.
- The performance of the combination of methods varies at different error tolerances. For example, at the 0.01 error tolerance both the combination of stochastic gradient descent and conformal methods as well as deep ensembles with the conformal methods outperforms the combination of mean-field variational inference with the conformal methods. However, at the 0.05 error tolerance this is not the case.

Other conclusions can be made that are more tailored to a specific use case. For example, if one is looking for average marginal coverage ≥ 0.96 across all corrupted CIFAR10 datasets, then deep ensembles combined with *aps* at the 0.01 error tolerance gives us the best (smallest) sized sets for that coverage. This is an especially useful evaluation strategy when it comes to *standards* that machine learning models must abide by when deployed in high-stakes settings, and emphasizes the importance that the error tolerance we choose does not represent our *true* desires if we believe out-of-distribution data will be encountered. It is also important to note that while stochastic gradient descent and deep ensembles appear to be more robust to the



(a) **MedMNIST** *accuracy* plot. It is evaluated for the in-distribution organ**C**mnist dataset and the out-of-distribution organ**S**mnist dataset.

(b) **MedMNIST** *reliability* plot. The reliability plot is a result of evaluating on a calibration set which is taken from the organ**C**mnist test set.

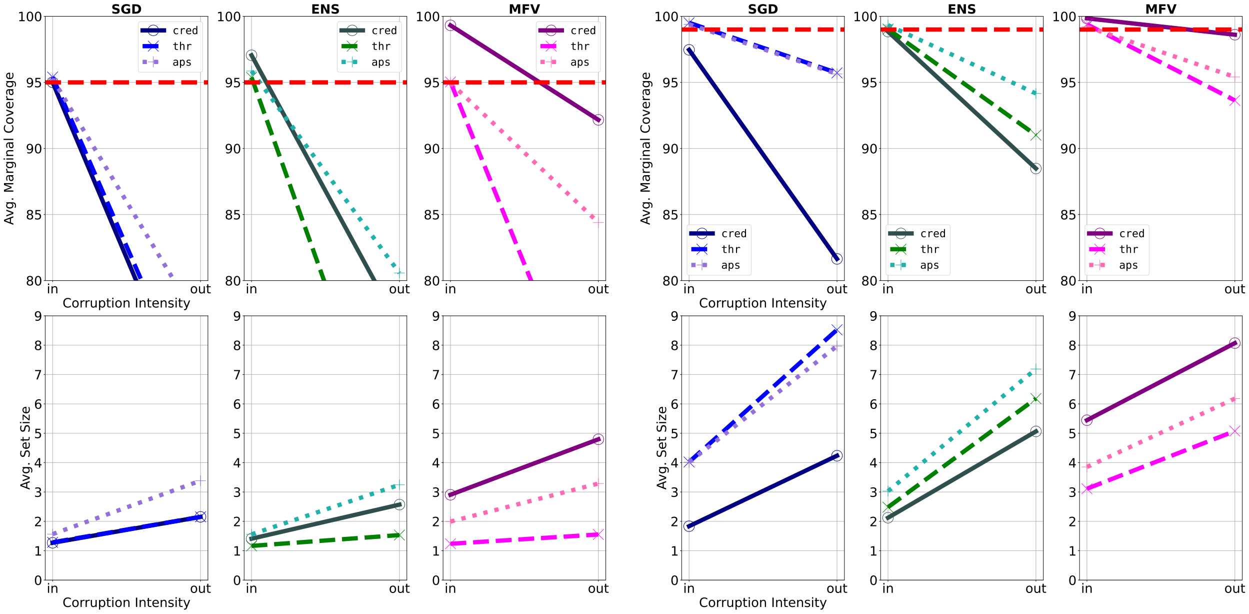
Figure 4: The accuracy and reliability plots for the MedMNIST experiment.

corruptions in terms of *accuracy* (see Figure 2a), this does not immediately translate to better coverage on out-of-distribution examples (see Figure 3.

4.2 MedMNIST

Our second experiment provides a more realistic safety-critical scenario: 11-class classification of radiology scans. We train a ResNet18 model on the organ**C**mnist dataset of MedMNIST with the same three modeling methods (He et al., 2016; Yang et al., 2023). Conformal calibration is done using the first 500 examples from the test set of organ**C**mnist. We then evaluate the coverage and size on the remaining examples from the test set of organ**C**mnist for all three prediction set methods. The organ**S**mnist dataset has the same classes but is the result of different *views* of the subject of interest in the radiology scan. Thus, it is from a different distribution, and so we also evaluate the coverage and size on the test set of organ**S**mnist to see how the combination of conformal methods and Bayesian deep learning affects out-of-distribution coverage. As before, we first note the reliability diagram of each model on the calibration set shown in Figure 4b. Similar to the CIFAR10 experiment, stochastic gradient descent is generally overconfident and mean-field variational inference is generally underconfident while deep ensembles is neither. Therefore, we again suspect that conformal methods may *help* stochastic gradient descent in terms of out-of-distribution coverage, while *harm* mean-field variation inference in terms of out-of-distribution coverage. The average marginal coverage and average set size for each dataset is shown in Figure 5a for the 0.05 error tolerance and Figure 5b for the 0.01 error tolerance. The accuracies of the deep learning methods are shown in Figure 4a. The *in*-distribution dataset is organ**C**mnist and the *out*-of-distribution dataset is organ**S**mnist. The main conclusions are similar to the previous experiment:

- Mean-field variational inference combined with simple predictive intervals performs the best amongst all other combinations of methods in terms of maintaining the desired coverage; even compared to mean-field variational inference combined with conformal methods.
- Conformal prediction significantly helps stochastic gradient descent in terms of out-of-distribution coverage.
- Conformal prediction helps deep ensembles in terms of out-of-distribution coverage, but not nearly as significantly as it did with stochastic gradient descent.
- The error tolerance selected impacts the performance of each combination of methods.



(a) Error tolerance of 0.05 (i.e. 0.95 desired coverage) which is denoted by the dashed horizontal red line.

(b) Error tolerance of 0.01 (i.e. 0.99 desired coverage) which is denoted by the dashed horizontal red line.

Figure 5: **MedMNIST**: Average marginal coverage and average set size for the in-distribution organ**C**mnist dataset (denoted *in* on the plot) and for the out-of-distribution organ**S**mnist dataset (denoted *out* on the plot.)

This experiment provides further evidence for our explanation of how the under- or over-confidence of a model can predict when conformal prediction may worsen or improve out-of-distribution coverage, and importantly does so in a realistic safety-critical scenario. While conformal prediction may harm out-of-distribution coverage for mean-field variational inference (due to its underconfidence on the calibration set), it instead helps the over-confident stochastic gradient descent. Additionally, we see that the severity by which conformal prediction affects out-of-distribution coverage depends on the error tolerance we select.

5 Discussion

Limitations and Future Work: We recognize that we evaluated and compared only a few of the many conformal and Bayesian methods. Furthermore, although the results presented here add an important dimension to the practical considerations of combining conformal and Bayesian deep learning methods, there are many other questions that remain to be answered (e.g. adaptivity gains from using Bayesian deep learning with conformal prediction). We developed experiments that provide evidence for the explanation offered in Section 3.1 (see also Figure 1) which consequently demonstrated that certain modeling and data scenarios can seriously impact the benefit of conformal prediction. Future work may include developing diagnostics or practical checks that suggest one is in a particular scenario in which the utility of conformal prediction can be predicted. Future work might also include further evaluations with additional measures such as size-stratified coverage (Angelopoulos et al., 2020), and further mathematical analysis. Such analysis might provide additional insights into when and how to combine certain conformal prediction and Bayesian deep learning methods.

Conclusion: We demonstrated important scenarios in which conformal prediction can cause unintended consequences that affect the overall safety of a machine learning system. Evidence also suggests that we should choose an error tolerance α that takes into account the extent to which we will encounter out-of-distribution data. In some cases it is not realistic to think that the tolerance we select will be honored. We hope that this study also motivates the need for better evaluation strategies for Bayesian deep learning models. Echoing some of the arguments made in Kompa et al. (2021), frequentist coverage of both in-distribution and out-of-distribution examples for Bayesian deep learning models provides a nuanced and practical representation of both the calibration of the models and the benefits of using conformal prediction in realistic settings. These are all of immediate practical importance for a wide range of application areas, particularly in those where unsafe mistakes can incur a large cost. If strong guarantees of coverage are desired, then one may consider Bayesian deep learning, conformal prediction, or both, in an effort to provide those guarantees. Knowledge of the scope of application, an assessment to identify breaking important assumptions (e.g. out-of-distribution data), and expected use may help decide the methods that should be applied. Being aware of these results and using the conclusions will better equip engineers in creating safer machine learning systems.

References

TensorFlow Datasets, a collection of ready-to-use datasets. https://www.tensorflow.org/datasets.

A. Angelopoulos, S. Bates, J. Malik, and M. I. Jordan. Uncertainty sets for image classifiers using conformal prediction. *arXiv preprint arXiv:2009.14193*, 2020.

A. N. Angelopoulos, S. Bates, E. J. Candès, M. I. Jordan, and L. Lei. Learn then test: Calibrating predictive algorithms to achieve risk control. *arXiv preprint arXiv:2110.01052*, 2021.

A. N. Angelopoulos, S. Bates, A. Fisch, L. Lei, and T. Schuster. Conformal risk control. *arXiv preprint arXiv:2208.02814*, 2022.

R. F. Barber, E. J. Candes, A. Ramdas, and R. J. Tibshirani. Conformal prediction beyond exchangeability. *The Annals of Statistics*, 51(2):816–845, 2023.

P. Bilic, P. Christ, H. B. Li, E. Vorontsov, A. Ben-Cohen, G. Kaissis, A. Szeskin, C. Jacobs, G. E. H. Mamani, G. Chartrand, et al. The liver tumor segmentation benchmark (lits). *Medical Image Analysis*, 84:102680, 2023.

D. M. Blei, A. Kucukelbir, and J. D. McAuliffe. Variational inference: A review for statisticians. *Journal of the American statistical Association*, 112(518):859–877, 2017.

J. Bradbury, R. Frostig, P. Hawkins, M. J. Johnson, C. Leary, D. Maclaurin, G. Necula, A. Paszke, J. VanderPlas, S. Wanderman-Milne, and Q. Zhang. JAX: composable transformations of Python+NumPy programs, 2018. URL http://github.com/google/jax.

N. Dewolf, B. D. Baets, and W. Waegeman. Valid prediction intervals for regression problems. *Artificial Intelligence Review*, 56(1):577–613, 2023.

C. Fannjiang, S. Bates, A. N. Angelopoulos, J. Listgarten, and M. I. Jordan. Conformal prediction under feedback covariate shift for biomolecular design. *Proceedings of the National Academy of Sciences*, 119(43): e2204569119, 2022.

E. Fong and C. C. Holmes. Conformal bayesian computation. *Advances in Neural Information Processing Systems*, 34:18268–18279, 2021.

C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger. On calibration of modern neural networks. In *International conference on machine learning*, pages 1321–1330. PMLR, 2017.

C. R. Harris, K. J. Millman, S. J. Van Der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith, et al. Array programming with numpy. *Nature*, 585(7825):357–362, 2020.

K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

D. Hendrycks and T. Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. *arXiv preprint arXiv:1903.12261*, 2019.

T. Hennigan, T. Cai, T. Norman, L. Martens, and I. Babuschkin. Haiku: Sonnet for JAX, 2020. URL http://github.com/deepmind/dm-haiku.

P. Hoff. Bayes-optimal prediction with frequentist coverage control. *arXiv preprint arXiv:2105.14045*, 2021.

J. D. Hunter. Matplotlib: A 2d graphics environment. *Computing in Science & Engineering*, 9(3):90–95, 2007. doi: 10.1109/MCSE.2007.55.

P. Izmailov, S. Vikram, M. D. Hoffman, and A. G. G. Wilson. What are bayesian neural network posteriors really like? In *International conference on machine learning*, pages 4629–4640. PMLR, 2021.

S. Kapoor, W. J. Maddox, P. Izmailov, and A. G. Wilson. On uncertainty, tempering, and data augmentation in bayesian classification. *Advances in Neural Information Processing Systems*, 35:18211–18225, 2022.

B. Kompa, J. Snoek, and A. L. Beam. Empirical frequentist coverage of deep learning uncertainty quantification procedures. *Entropy*, 23(12):1608, 2021.

A. Krizhevsky, G. Hinton, et al. Learning multiple layers of features from tiny images. 2009.

A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.

B. Lakshminarayanan, A. Pritzel, and C. Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. *Advances in neural information processing systems*, 30, 2017.

J. Lei, M. G'Sell, A. Rinaldo, R. J. Tibshirani, and L. Wasserman. Distribution-free predictive inference for regression. *Journal of the American Statistical Association*, 113(523):1094–1111, 2018.

D. J. MacKay. *Information theory, inference and learning algorithms*. Cambridge university press, 2003.

Y. Ovadia, E. Fertig, J. Ren, Z. Nado, D. Sculley, S. Nowozin, J. Dillon, B. Lakshminarayanan, and J. Snoek. Can you trust your model's uncertainty? evaluating predictive uncertainty under dataset shift. *Advances in neural information processing systems*, 32, 2019.

Y. Romano, M. Sesia, and E. Candes. Classification with valid and adaptive coverage. *Advances in Neural Information Processing Systems*, 33:3581–3591, 2020.

M. Sadinle, J. Lei, and L. Wasserman. Least ambiguous set-valued classifiers with bounded error levels. *Journal of the American Statistical Association*, 114(525):223–234, 2019.

S. Stanton, W. Maddox, and A. G. Wilson. Bayesian optimization with conformal prediction sets. In *International Conference on Artificial Intelligence and Statistics*, pages 959–986. PMLR, 2023.

D. Stutz, K. D. Dvijotham, A. T. Cemgil, and A. Doucet. Learning optimal conformal classifiers. In *International Conference on Learning Representations*, 2021.

R. J. Tibshirani, R. Foygel Barber, E. Candes, and A. Ramdas. Conformal prediction under covariate shift. *Advances in neural information processing systems*, 32, 2019.

G. Van Rossum and F. L. Drake. *Python 3 Reference Manual*. CreateSpace, Scotts Valley, CA, 2009. ISBN 1441412697.

R. Vasilev and A. D'yakonov. Calibration of neural networks. *arXiv preprint arXiv:2303.10761*, 2023.

V. Vovk. Conditional validity of inductive conformal predictors. In *Asian conference on machine learning*, pages 475–490. PMLR, 2012.

V. Vovk, A. Gammerman, and G. Shafer. *Algorithmic learning in a random world*, volume 29. Springer, 2005.

M. L. Waskom. seaborn: statistical data visualization. *Journal of Open Source Software*, 6(60):3021, 2021. doi: 10.21105/joss.03021. URL https://doi.org/10.21105/joss.03021.

L. Wasserman. Frasian inference. 2011.

A. G. Wilson. The case for bayesian deep learning. *arXiv preprint arXiv:2001.10995*, 2020.

A. G. Wilson and P. Izmailov. Bayesian deep learning and a probabilistic perspective of generalization. *arXiv preprint arXiv:2002.08791*, 2020.

J. Yang, R. Shi, D. Wei, Z. Liu, L. Zhao, B. Ke, H. Pfister, and B. Ni. Medmnist v2-a large-scale lightweight benchmark for 2d and 3d biomedical image classification. *Scientific Data*, 10(1):41, 2023.

B. Zadrozny and C. 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.

APPENDIX

A TRAINING DETAILS

For both experiments we train stochastic gradient descent and mean-field variational inference for 5 different seeds. Deep ensembles is the result of combining 5 stochastic gradient descent model states from the different 5 seeds. We measure the validation accuracy every 10 epochs. For evaluation we then select the model state that attains the best validation accuracy. If the best validation accuracy is shared between multiple checkpoints, we use the model state from the earliest checkpoint amongst those that are tied.

A.1 CIFAR10

Dataset: The CIFAR10 dataset contains 60,000 $32 \times 32 \times 3$ RGB images in 10 classes, where each class contains 6,000 images each. There are 50,000 training images (5,000 images per class) and 10,000 test images (1,000 images per class). We take 5% of the original training dataset ($0.05 \times 50,000 = 2,500$) examples as a validation set, and leave the remaining 95% ($50,000 - 2,500 = 47,500$) examples for training. For preprocessing, we normalize the images with mean $(0.49, 0.48, 0.44)$ and standard deviation $(0.2, 0.2, 0.2)$ for each of the 3 channels. This is taken from the code repository of (Izmailov et al., 2021). We performed *no* data augmentation.

Base Model: We use an AlexNet inspired convolutional neural network as a base model, which is taken from the code repository of (Izmailov et al., 2021).

Training Hyperparameters: The following tables illustrate the training hyperparameters for stochastic gradient descent and mean-field variational inference. Our deep ensembles method consists of using $k = 5$ stochastic gradient descent model states corresponding to 5 different independent parameter initializations.

Name	**Value**
seeds	$\{1, \ldots, 5\}$
batch size	80
epochs	100
weight decay	5.0
temperature	1.0
learning rate schedule	cosine
checkpoint frequency	10

Table 1: Shared Training Hyperparameters

Name	**Value**
initial step size	8e-7
optimizer	sgd
momentum decay	0.9

Table 2: **SGD** Specific Training Hyperparameters

Name	**Value**
initial step size	4e-4
optimizer	Adam
initial σ	0.01
# samples	1

Table 3: **MFV** Specific Training Hyperparameters. The initial σ is the initial value of the standard deviation of the per-parameter Gaussians for mean-field variational inference.

A.2 MedMNIST

Dataset: MedMNIST contains many standardized datasets of biomedical images (Yang et al., 2023). We train on one of these datasets: organ**C**mnist. This dataset is part of a larger cohort of three datasets which are based on the 3D CT images from the Liver Tumor Segmentation Benchmark (Bilic et al., 2023). The larger cohort is {organ**A**mnist, organ**C**mnist, organ**S**mnist }, where **A**,**C**, and **S** are short for Axial, Coronal, and Sagittal. These describe different *views* of the CT scan (see Figure 6). We use the pre-specified training and validation sets provided by MedMNIST. These contain 13,000 training examples and 2,392 validation examples. Each image is grayscale. For preprocessing, we normalize the images with mean 0.49 and standard deviation 0.2 for the single channel. We performed *no* data augmentation.

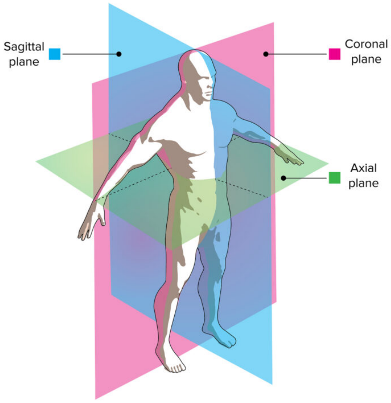


Figure 6: An illustration describing the axial, coronal, and sagittal views. `https://anatomytool.org/content/lecturio-drawing-sagittal-coronal-and-transverse-plane-english-labels`

Base Model: We use a ResNet18 neural network as a base model (He et al., 2016), which is a built-in model in the Haiku library (Hennigan et al., 2020).

Training Hyperparameters: The following tables illustrate the training hyperparameters for stochastic gradient descent and mean-field variational inference. Our deep ensembles method consists of using $k = 5$ stochastic gradient descent model states corresponding to 5 different independent parameter initializations.

Name	**Value**
seeds	$\{1, \dots, 5\}$
batch size	80
epochs	100
weight decay	10.0
temperature	1.0
learning rate schedule	cosine
checkpoint frequency	10

Table 4: Shared Training Hyperparameters

Name	**Value**
initial step size	6e-6
optimizer	sgd
momentum decay	0.9

Table 5: **SGD** Specific Training Hyperparameters

Name	Value
initial step size	1e-4
optimizer	Adam
initial σ	0.01
# samples	1

Table 6: **MFV** Specific Training Hyperparameters. The initial σ is the initial value of the standard deviation of the per-parameter Gaussians for mean-field variational inference.

B EVALUATION DETAILS

We first note that we use 30 samples to approximate the posterior predictive density when using mean-field variational inference. For both experiments, we create prediction sets in the same way. Given an error tolerance α, for each method (stochastic gradient descent, deep ensembles, and mean-field variational inference) and each evaluation dataset, we produce predicted probabilities $\hat{\mathbf{p}}(\mathbf{x})$ and then create...

Predictive Credible Sets: We order the probabilities $\hat{p}_i(\mathbf{x}) \in \hat{\mathbf{p}}(\mathbf{x})$ from greatest to least and continue adding the corresponding labels until the cumulative probability mass just exceeds $1-\alpha$. We sometimes abbreviate this method as *cred*.

Threshold Prediction Sets: Using a calibration set $\mathcal{D}_{\text{cal}}$ taken from the in-distribution test set, we compute scores for each example using the score function (Sadinle et al., 2019):

$$s(\mathbf{x}, y) = 1 - \hat{p}_y(\mathbf{x}).$$

Then we take the

$$[(1-\alpha)(1+\frac{1}{|\mathcal{D}_{\text{cal}}|})]\text{-quantile}$$

of these scores which we call τ. Then for each input we want to evaluate for, we create prediction sets as

$$\mathbb{Y}(\mathbf{x}) := \{y \in \boldsymbol{\mathcal{Y}} \mid s(\mathbf{x}, y) \leq \tau\}$$

where $\boldsymbol{\mathcal{Y}}$ is the sample space for labels y. We sometimes abbreviate this method as *thr*.

Adaptive Prediction Sets: Using a calibration set $\mathcal{D}_{\text{cal}}$ taken from the in-distribution test set, we compute scores for each example using the score function

$$s(\mathbf{x}, y) = \hat{p}_1(x) + \cdots + U\hat{p}_y(x),$$

where $\hat{p}_1(x) \geq \cdots \geq \hat{p}_y(x)$ and U is a uniform random variable in $[0, 1]$ to break ties (Romano et al., 2020). As in the case of threshold prediction, we take the

$$[(1-\alpha)(1+\frac{1}{|\mathcal{D}_{\text{cal}}|})]\text{-quantile}$$

of these scores which we call τ. Then for each input we want to evaluate for, we create prediction sets as

$$\mathbb{Y}(\mathbf{x}) := \{y \in \boldsymbol{\mathcal{Y}} \mid s(\mathbf{x}, y) \leq \tau\}$$

where $\boldsymbol{\mathcal{Y}}$ is the sample space for labels y. We sometimes abbreviate this method as *aps*.

Remark on Adaptive Prediction Sets: It is useful to note that for coverage to be tight (i.e. having the upper bound in equation (4) of the paper), adaptive prediction sets requires distinct conformity scores. To handle this, an additional standard uniform random variable is used. During the calibration phase, we take

$|\mathcal{D}_{\text{cal}}|$ random samples from the this variable and subtract it from the scores before computing the quantile τ. And during the prediction phase, we take $|\mathcal{D}_{\text{test}}|$ random samples and subtract it from the scores before checking if they are less than or equal to τ, where $\mathcal{D}_{\text{test}}$ is the test set. We use the implementation from Stutz et al. (2021), which allows one to input a random seed to do the above procedure.

B.1 CIFAR10 and CIFAR10-Corrupted

In the CIFAR10 experiment we evaluate the prediction sets on (i) the CIFAR10 test set (Krizhevsky et al., 2009), and (ii) all CIFAR10-Corrupted test sets (Hendrycks and Dietterich, 2019). The CIFAR10-Corrupted test sets contain 19 different corruptions, each with intensities ranging from 1 to 5.

Table 7: Different type of corruptions in CIFAR10-Corrupted.

	Corruption Type
1	brightness
2	contrast
3	defocus blur
4	elastic
5	fog
6	frost
7	frosted glass blur
8	gaussian blur
9	gaussian noise
10	impulse noise
11	jpeg compression
12	motion blur
13	pixelate
14	saturate
15	shot noise
16	snow
17	spatter
18	speckle noise
19	zoom blur

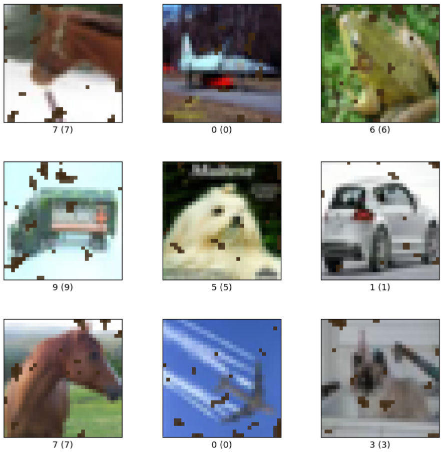


Figure 7: An example of the *spatter* corruption at intensity level 4. `https://www.tensorflow.org/datasets/catalog/cifar10_corrupted`

The $5 \times 19 = 95$ CIFAR10-Corrupted test sets are all corrupted versions of the original CIFAR10 test set. Thus, when we take the $1,000$ examples from the original CIFAR10 test set to use as a calibration set for our conformal methods, we also take the $1,000$ corresponding (semantically similar) examples from all the CIFAR10-Corrupted test sets. We evaluate the three prediction set methods on the remaining $9,000$ examples from the original CIFAR10 test set, and then all the trimmed CIFAR10-Corrupted test sets (which each contain $9,000$ examples). We run the procedure of taking a calibration set, finding τ, and computing prediction sets on all the test sets with three different seeds $\{1, 2, 3\}$. Then we take the average accuracy, marginal coverage, and set size across these three seeds. The accuracy results are presented in the main paper. The accuracy, marginal coverage, and set size results for each dataset are presented in section F.1. In the main paper we go further and take the average marginal coverage and average set size *across* data sets at each intensity and report those summarized results.

B.2 MedMNIST: organCmnist And organSmnist

In the MedMNIST experiment we only have two test sets. The organ**C**mnist test set (containing 8,268 examples) and the organ**S**mnist test set (containing 8,829 examples). The **C** in organ**C**mnist standards for *coronal* and the **S** in organ**S**mnist stands for *sagittal* (see Figure 6). We take 500 examples from the organ**C**mnist test set to use as a calibration set for our conformal methods. We then evaluate the three prediction set methods on the remaining examples from the organ**C**mnist test set as well as on the organ**S**mnist test set, the latter serving as our out-of-distribution test set. We run the procedure of taking a calibration set, finding τ, and computing prediction sets on all the test sets with three different seeds $\{1, 2, 3\}$. Then we take the average accuracy, marginal coverage, and set size across these three seeds. The results are presented in the main paper. The class proportions for all splits of both organ**C**mnist and organ**S**mnist are shown in Figure 8.

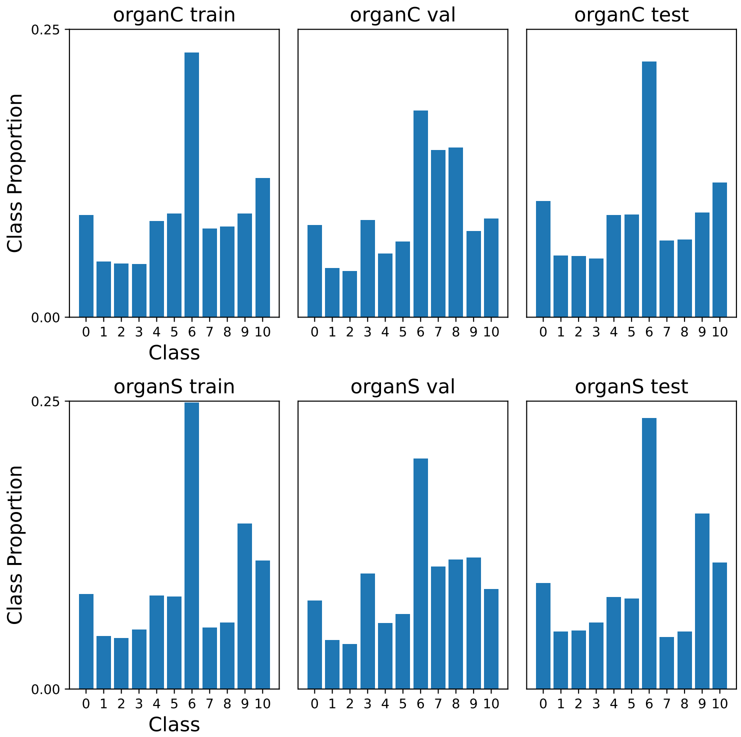


Figure 8: Class propotions for both the organ**C**mnist datasets and the organ**S**mnist datasets.

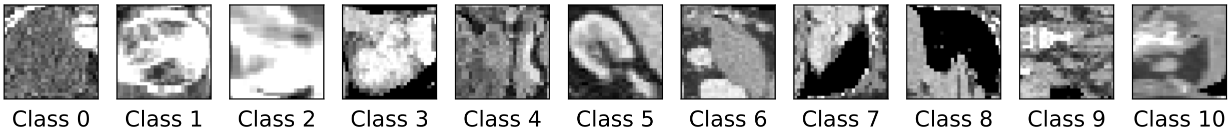


Figure 9: An example image from each of the 11 classes in the organ**C**mnist dataset.

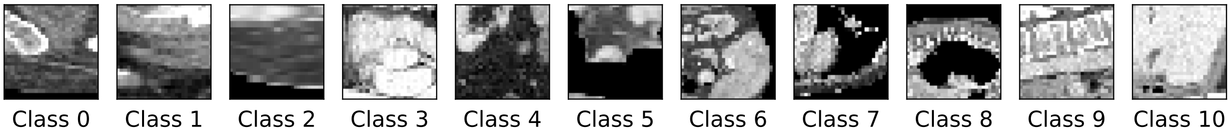


Figure 10: An example image from each of the 11 classes in the organ**S**mnist dataset.

C SPACE & COMPLEXITY

Space: If n is the number of learnable parameters for stochastic gradient descent, then mean-field variational inference requires $2n$ learnable parameters. This is due to treating each weight as a random variable from a Gaussian distribution, and instead having to fit the two parameters governing that distribution. If k is the number of models in the ensemble, then deep ensembles requires kn learnable parameters. We use $k = 5$.

Runtime: All methods have linear time complexity in the number of parameters $\mathcal{O}(n)$. However, there are some details to note: if m is the number of forward passes needed to predict using stochastic gradient descent, then mean-field variational inference requires pm forward passes where p is the number of samples to construct a Monte Carlo approximation for the Bayesian model average. During training we have $p = 1$ and during evaluation we have $p = 30$. If k is the number of models in the ensemble, then deep ensembles requires kn forward passes. We use $k = 5$.

D SOFTWARE PACKAGES

- Python 3, PSF License Agreement (Van Rossum and Drake, 2009).
- Matplotlib, Matplotlib License Agreement (Hunter, 2007).
- Seaborn, BSD License (Waskom, 2021).
- Numpy, BSD License (Harris et al., 2020).
- JAX, Apache 2.0 License (Bradbury et al., 2018).
- Haiku, Apache 2.0 License (Hennigan et al., 2020).
- Tensorflow Datasets, Apache 2.0 License (TFD).
- google-research/bnn_hmc, Apache 2.0 License (Izmailov et al., 2021).
- google-deepmind/conformal_training, Apache 2.0 License (Stutz et al., 2021).

E COMPUTE

We ran our experiments on an Ubuntu 18.04.6 system with a dual core 2.10GHz processor and 754 GiB of RAM. We also used a single Tesla V100-SXM2 GPU with 32 GiB of RAM.

CIFAR10 Experiment For *training*, stochastic gradient descent takes ≈ 3.6 minutes per seed and mean-field variational inference takes ≈ 4.6 minutes per seed. For *evaluation*, stochastic gradient descent takes ≈ .3 minutes per seed, mean-field variational inference takes ≈ .3 minutes per seed, and deep ensembles takes ≈ .4 minutes per seed. Assuming (i) you train stochastic gradient descent and mean-field variational inference with 5 different seeds, (ii) you evaluate using 3 seeds: the approximate time to run the CIFAR10 experiment is

$$\underbrace{((3.6+4.6)\times 5)}_{\text{training}} + (\underbrace{(0.3+0.3+0.4)\times 3}_{\text{evaluation}} \times \underbrace{96}_{\#\text{ datasets}}) \approx 5.48 \text{ hours}$$

MedMNIST Experiment For *training*, stochastic gradient descent takes ≈ 2.6 minutes per seed and mean-field variational inference takes ≈ 5 minutes per seed. For *evaluation*, stochastic gradient descent takes ≈ .4 minutes per seed, mean-field variational inference takes ≈ .8 minutes per seed, and deep ensembles takes ≈ .7 minutes per seed. Assuming (i) you train stochastic gradient descent and mean-field variational inference with 5 different seeds, (ii) you evaluate using 3 seeds: the approximate time to run the MedMNIST experiment is

$$\underbrace{((2.6+5)\times 5)}_{\text{training}} + (\underbrace{(0.4+0.8+0.7)\times 3}_{\text{evaluation}} \times \underbrace{2}_{\#\text{ datasets}}) \approx 0.82 \text{ hours}$$

F ADDITIONAL EXPERIMENTAL RESULTS

F.1 CIFAR10-Corrupted Per-Dataset Results

0.05 Error Tolerance

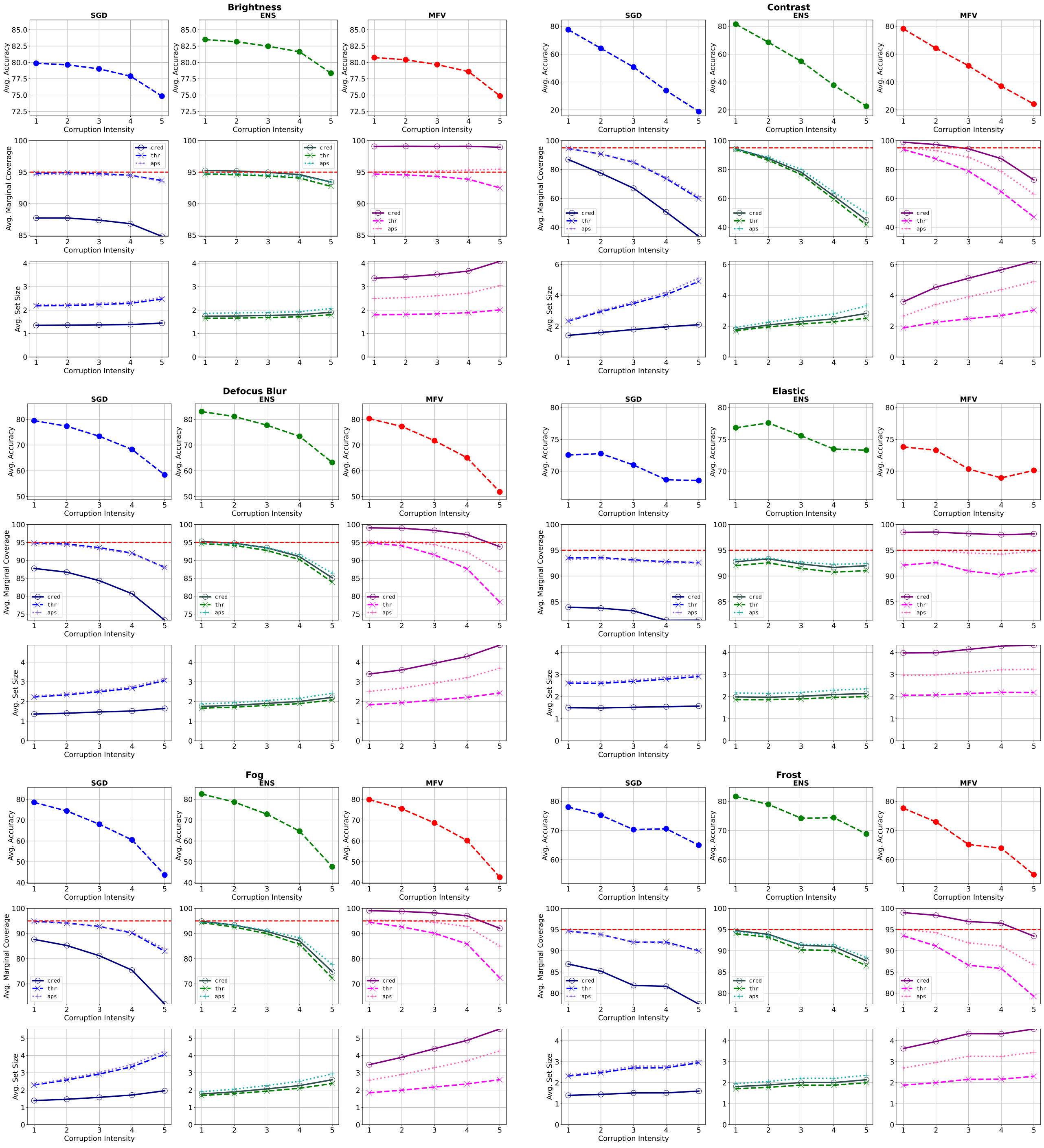


Figure 11: CIFAR10-Corrupted per-dataset results at the 0.05 Error Tolerance.

0.05 Error Tolerance

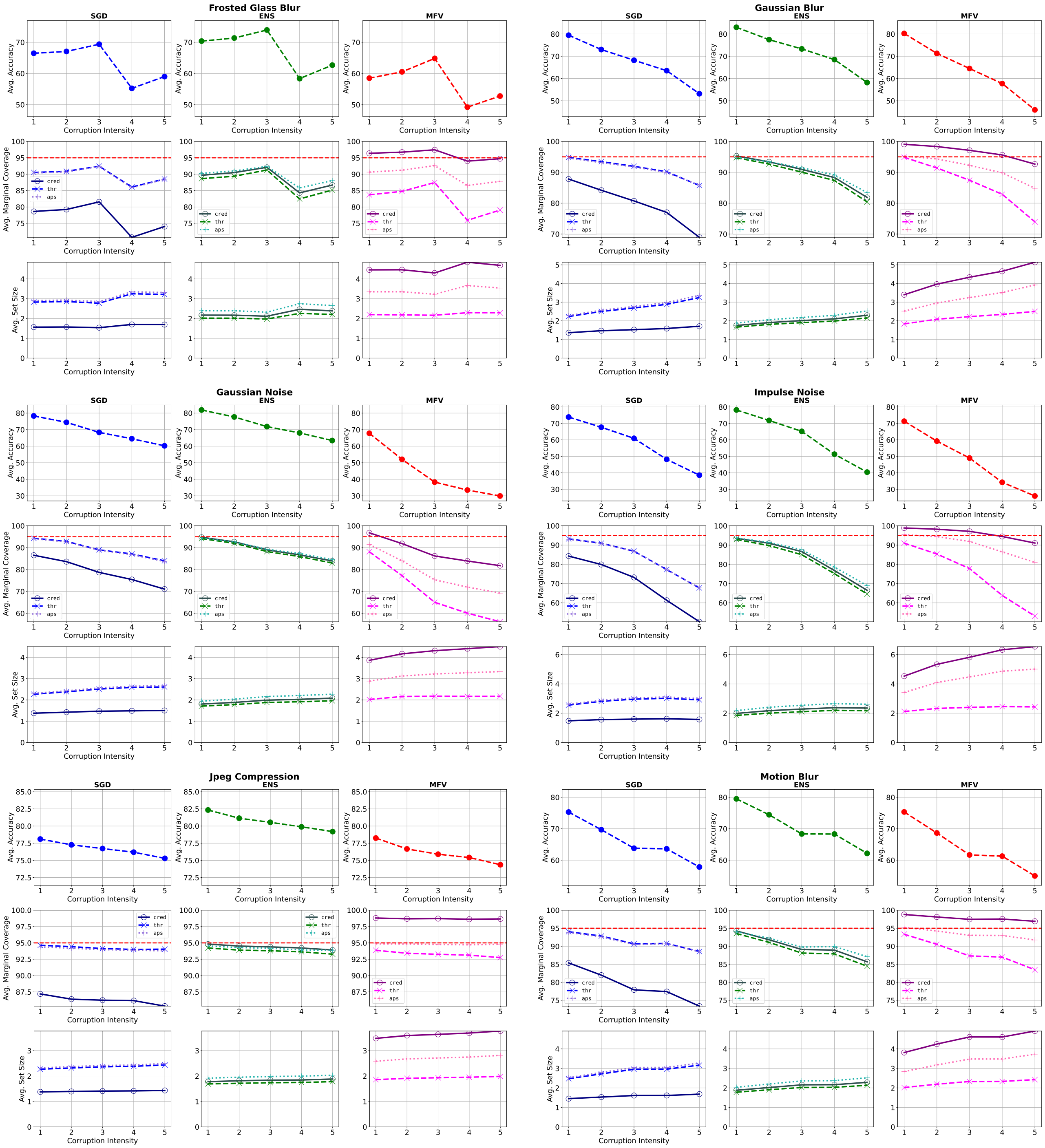


Figure 12: CIFAR10-Corrupted per-dataset results at the 0.05 Error Tolerance.

0.05 Error Tolerance

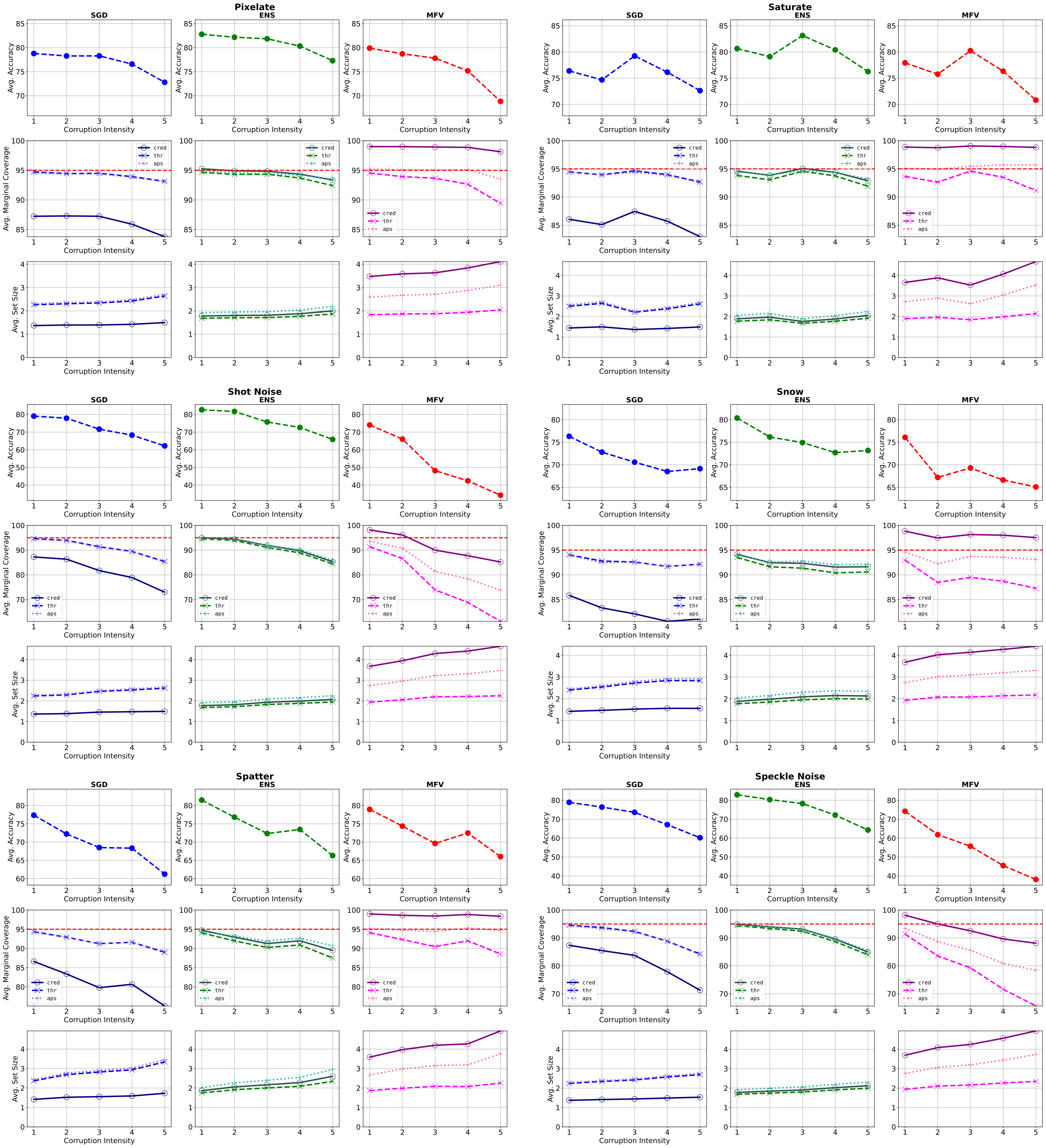


Figure 13: CIFAR10-Corrupted per-dataset results at the 0.05 Error Tolerance.

0.05 Error Tolerance

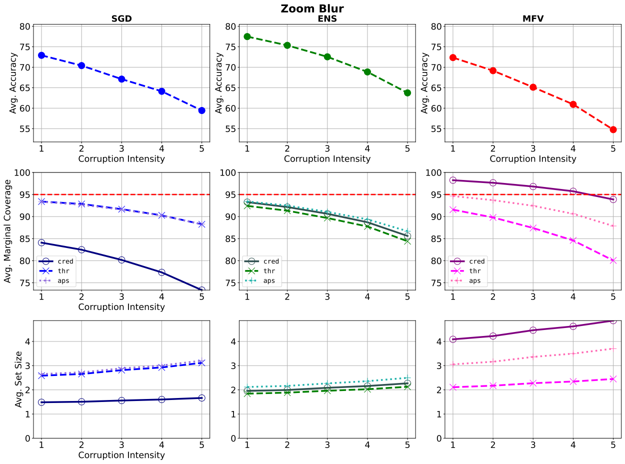
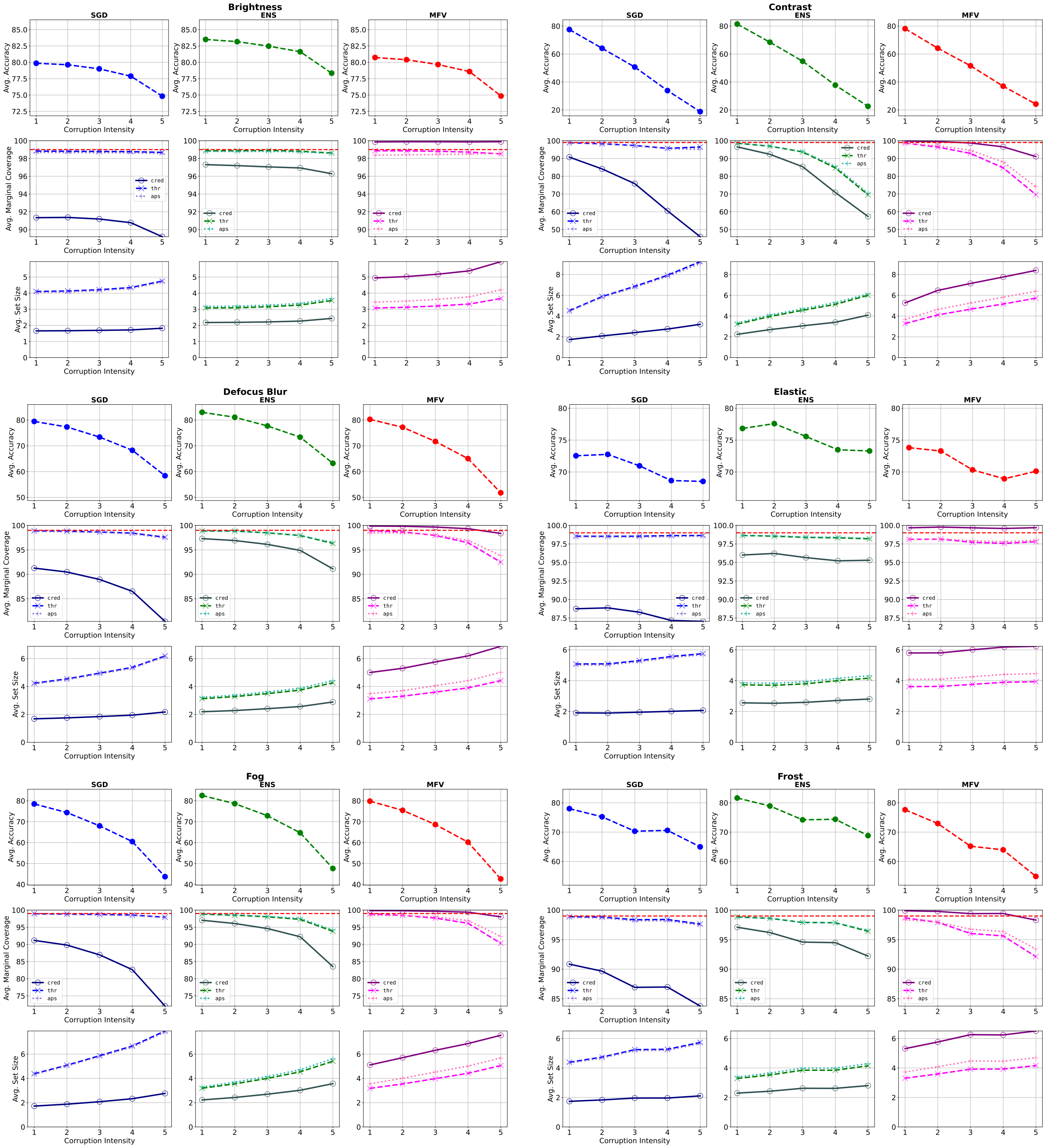


Figure 14: CIFAR10-Corrupted per-dataset results at the 0.05 Error Tolerance.

0.01 Error Tolerance



Brightness

SGD ENS MFV

Contrast

SGD ENS MFV

Defocus Blur

SGD ENS MFV

Elastic

SGD ENS MFV

Fog

SGD ENS MFV

Frost

SGD ENS MFV

Figure 15: CIFAR10-Corrupted per-dataset results at the 0.01 Error Tolerance.

0.01 Error Tolerance

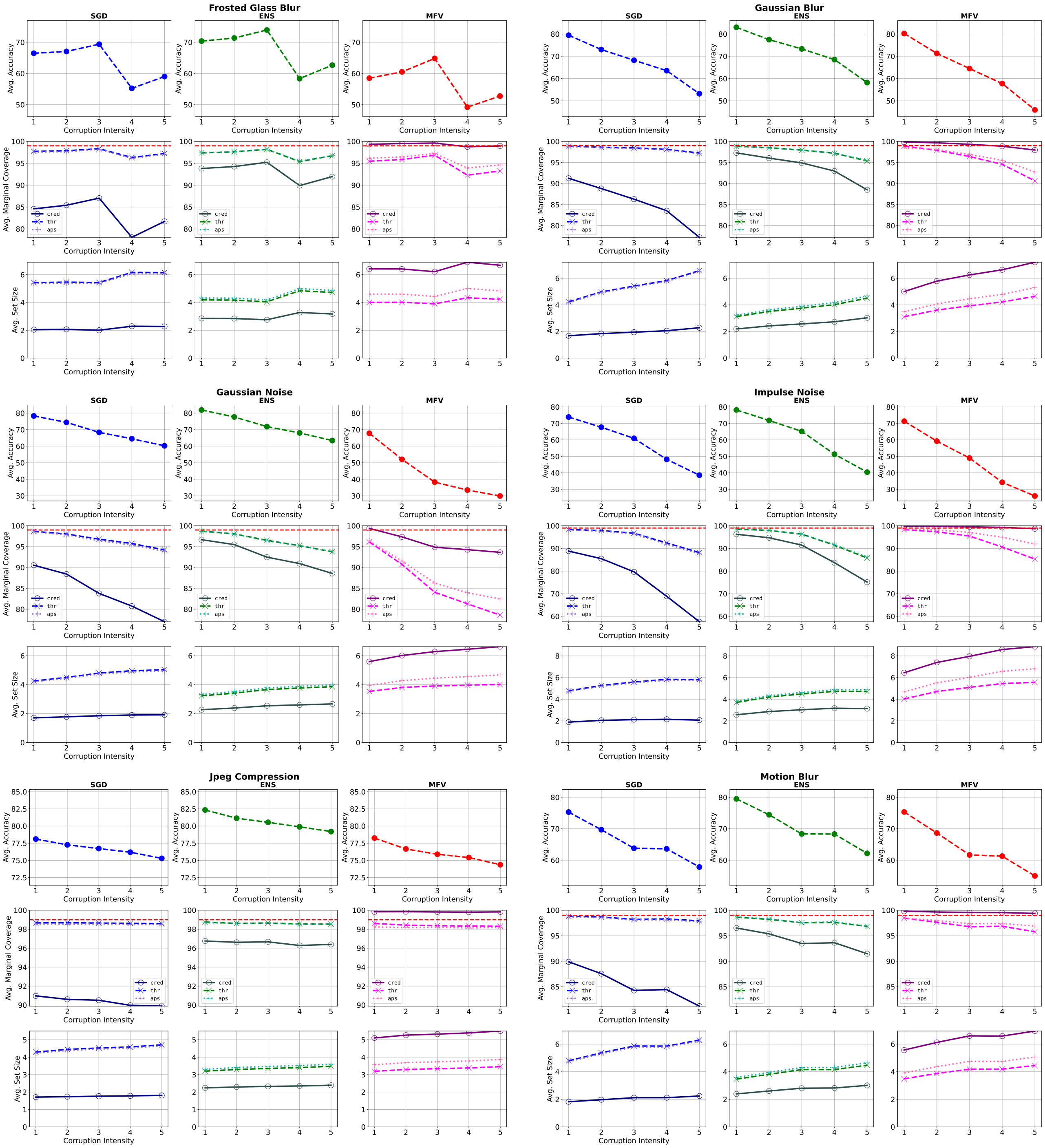


Figure 16: CIFAR10-Corrupted per-dataset results at the 0.01 Error Tolerance.

0.01 Error Tolerance

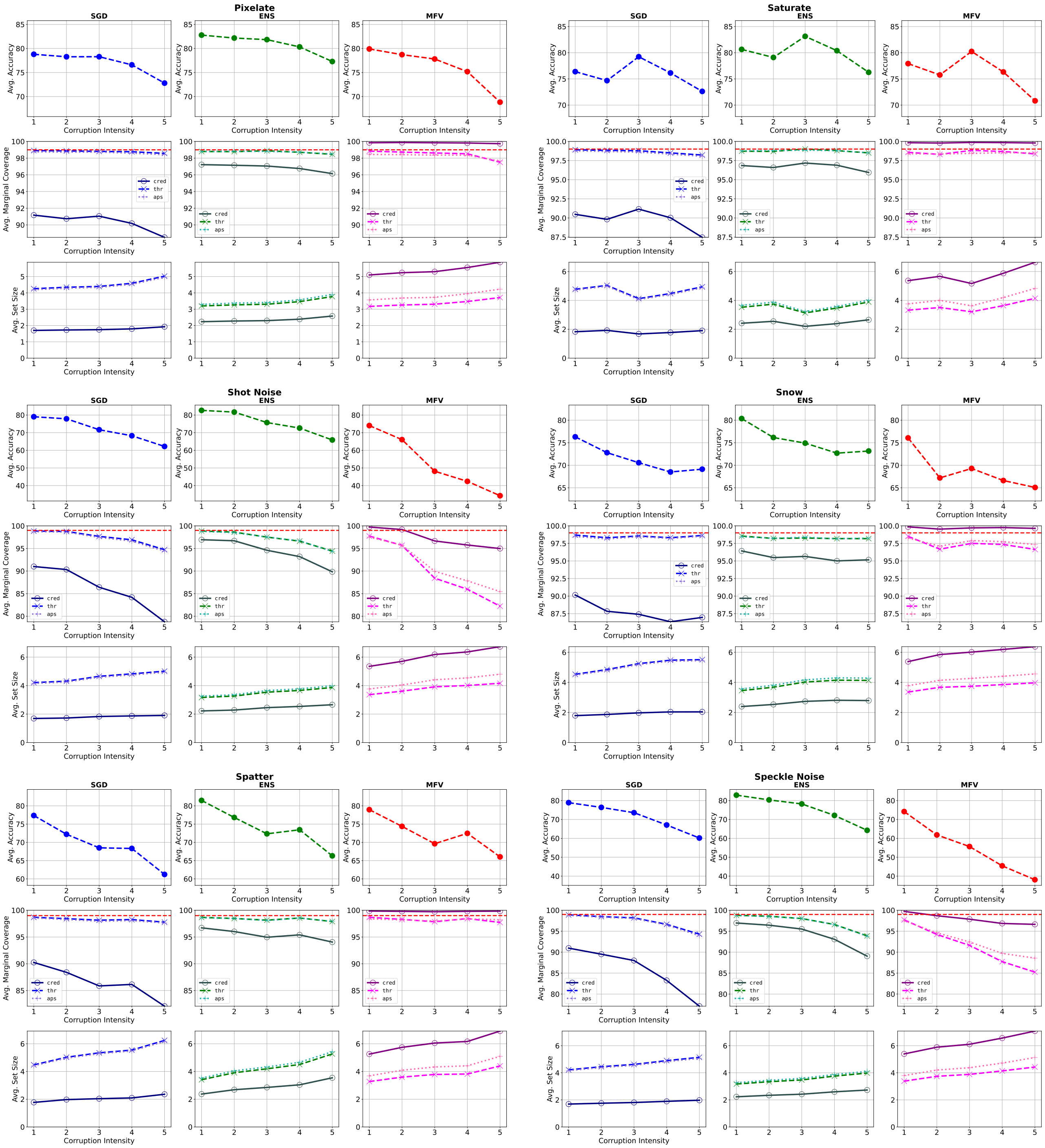


Figure 17: CIFAR10-Corrupted per-dataset results at the 0.01 Error Tolerance.

0.01 Error Tolerance

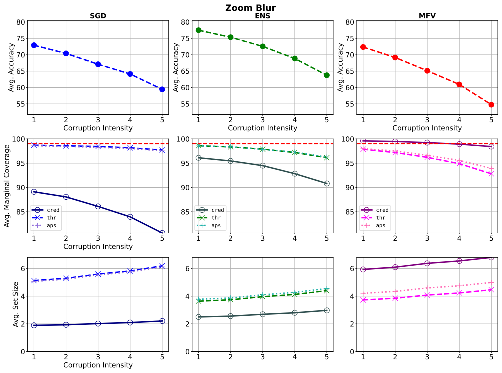


Figure 18: CIFAR10-Corrupted per-dataset results at the 0.01 Error Tolerance.