UNCERTAINTY SETS FOR IMAGE CLASSIFIERS USING CONFORMAL PREDICTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Convolutional image classifiers can achieve high predictive accuracy, but quantifying their uncertainty remains an unresolved challenge, hindering their deployment in consequential settings. Existing uncertainty quantification techniques, such as Platt scaling, attempt to calibrate the network's probability estimates, but they do not have formal guarantees. We present an algorithm that modifies any classifier to output a *predictive set* containing the true label with a user-specified probability, such as 90%. The algorithm is simple and fast like Platt scaling, but provides a formal finite-sample coverage guarantee for every model and dataset. Furthermore, our method generates much smaller predictive sets than alternative methods, since we introduce a regularizer to stabilize the small scores of unlikely classes after Platt scaling. In experiments on both Imagenet and Imagenet-V2 with ResNet-152 and other classifiers, our scheme outperforms existing approaches, achieving exact coverage with sets that are often factors of 5 to 10 smaller.

1 INTRODUCTION

Imagine you are a doctor making a high-stakes medical decision based on diagnostic information from a computer vision classifier. What would you want the classifier to output in order to make the best decision? This is not a casual hypothetical; such classifiers are already used in medical settings (e.g., Razzak et al., 2018; Lundervold & Lundervold, 2019; Li et al., 2014). A maximum-likelihood diagnosis with an accompanying probability may be useful, but it may not be the most essential piece of information. As a doctor, your primary objective is to ensure the health of your patient, which requires ruling in or ruling out harmful diagnoses. In other words, even if the most likely diagnosis is a stomach ache, it is equally or more important to rule out stomach cancer. Therefore, you would want the classifier to give you—in addition to an estimate of the most likely outcome—actionable uncertainty quantification, such as a set of predictions that provably covers the true diagnosis with a high probability (e.g., 90%). This is called a *prediction set* (see Figure 1). Our paper describes a novel method for constructing prediction sets from any pre-trained image classifier that are provably exact, relatively small, and practical to implement.

Formally, for a discrete response $Y \in \mathcal{Y} = \{1, \ldots, K\}$ and a feature vector $X \in \mathbb{R}^d$, we develop a method for taking any fitted classifier and creating an uncertainty set function, $\mathcal{C}(X)$, mapping a feature vector to a subset of $\{1, \ldots, K\}$ such that

$$P(Y \in \mathcal{C}(X)) \geq 1 - \alpha, \tag{1}$$

for a pre-specified confidence level α such as 5%. We introduce a general technique that modifies any black-box classifier to output predictive sets that are rigorously guaranteed to satisfy the desired *coverage* property shown in Eq. (1). For evaluations, we focus on Imagenet classification using convolutional neural networks (CNNs) as the base classifiers, since this is a particularly challenging testbed. In this setting, X would be the image and Y would be the class label.

One approach might be to assemble the set by including classes from highest to lowest probability (e.g., after Platt scaling and a softmax function; see Platt et al., 1999; Guo et al., 2017) until their sum just exceeds the threshold $1 - \alpha$. We call this strategy `naive` and formulate it precisely in Algorithm 1. There are two problems with `naive`: first, the probabilities output by CNNs are

Figure 1: **Prediction set examples on Imagenet.** We show three examples of the class `fox squirrel` and the 95% prediction sets generated by `RAPS` to illustrate how the size of the set changes as a function of the difficulty of a test-time image.

known to be incorrect (Nixon et al., 2019), so the sets from `naive` do not achieve coverage. Second, for examples where the model is not confident, `naive` must select many classes before it reaches the desired confidence level, leading to a large set size.

The coverage problem can be solved by picking a new threshold using holdout samples. For example, with α =10%, if choosing sets that contain 93% estimated probability achieves 90% coverage on the holdout set, we use the 93% cutoff instead. We refer to this algorithm, introduced in Romano et al. (2020), as *Adaptive Prediction Sets* (`APS`). The `APS` procedure provides coverage but produces large sets because it is sensitive to the noisy probability estimates assigned to unlikely classes—see Section 2.3 for details. To fix this, we introduce a regularization technique that tempers the influence of these noisy estimates, leading to smaller, more stable sets. We describe our proposed algorithm, *Regularized Adaptive Prediction Sets* (`RAPS`), in Algorithms 2 and 3 (with `APS` as a special case). As we will see in Section 2, both `APS` and `RAPS` are always guaranteed to satisfy Eq. (1)—regardless of model and dataset. Furthermore, we show that `RAPS` is guaranteed to have better performance than choosing a fixed-size set. Both methods impose negligible computational requirements in both training and evaluation, and require only 1000 held-out data points in practice.

We will also see in Section 3 that `RAPS` provides significant practical gains. Conducting the most extensive evaluation of conformal prediction in deep learning to date, on Imagenet and Imagenet-V2, we find that `RAPS` is uniformly preferable to `naive` and `APS`. For example, using a ResNeXt-101, `naive` does not achieve coverage, while `APS` and `RAPS` achieve it exactly. However, `APS` sets have an average size of 19, while `RAPS` sets have an average size of 2 at $\alpha = 10\%$ (Figure 2 and Table 1). We will provide an accompanying codebase that implements our method as a wrapper for any PyTorch classifier, along with code to exactly reproduce all of our experiments.

1.1 Related work

Reliably estimating predictive uncertainty for neural networks is an unsolved problem. Historically, the standard approach has been to train a Bayesian neural network to learn a distribution over network weights (Quinonero-Candela et al., 2005; MacKay, 1992; Neal, 2012; Kuleshov et al., 2018; Gal, 2016). This approach incurs significant computational expense and implementational complexity, which more recent approaches address with model ensembles (Lakshminarayanan et al., 2017; Jiang et al., 2018) or approximations of Bayesian inference for the parameters of the last layer only (Riquelme et al., 2018; Sensoy et al., 2018). These methods also have major practical limitations; for example, ensembling requires training many copies of a neural network adversarially. Therefore, the most widely used approach to quantifying uncertainty is the ad-hoc "calibration" of the softmax scores with Platt scaling (Platt et al., 1999; Guo et al., 2017; Nixon et al., 2019). We will refer to this approach as *traditional calibration*.

In this work, we develop an alternative method for uncertainty quantification that is based on *conformal prediction*. Originating in the online learning literature, conformal prediction is a general approach for generating predictive sets that satisfy the coverage property in Eq. (1) (Vovk et al., 1999; 2005). Although the original approach to conformal prediction requires special algorithmic development for each predictor under study, a convenient data-splitting version known as *split conformal prediction* enables conformal prediction methods to be deployed for essentially any predictor (Papadopoulos et al., 2002; Lei et al., 2018). While mechanically very different from traditional

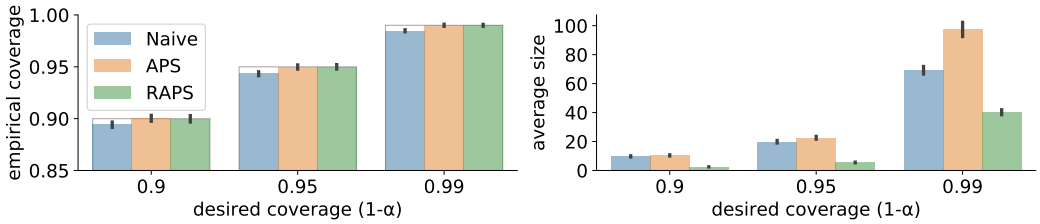


Figure 2: **Coverage and average set size on Imagenet for prediction sets from three methods.** All methods use a ResNet-152 as the base classifier, and results are reported for 100 random splits of Imagenet-Val, each of size 20K. See Section 3.1 for full details.

calibration as discussed above, we will refer to our approach as *conformal calibration* to highlight that the two methodologies have overlapping but different goals.

Note that conformal prediction is a general framework, not a specific algorithm—there are many conformal algorithms and important design decisions must be made to achieve the best performance for each context. To this end, Romano et al. (2020) and Cauchois et al. (2020) introduce techniques aimed at achieving coverage that is similar across regions of feature space, whereas Hechtlinger et al. (2018) introduce a technique aimed at achieving equal coverage for each class. While these methods have conceptual appeal, thus far there has been limited empirical evaluation of this general approach for state-of-the-art CNNs. Concretely, the only works that we are aware of that include some evaluation of conformal methods on ImageNet—the gold standard for benchmarking computer vision methods—are Hechtlinger et al. (2018), Park et al. (2019), Cauchois et al. (2020), and Messoudi et al. (2020), although in all four cases further experiments are needed to more fully evaluate their operating characteristics for practical deployment.

2 METHODS

In developing uncertainty set methods to improve upon `naive`, we are guided by three desiderata. First and most importantly, the ***coverage desideratum*** says the sets must provide exactly $1-\alpha$ coverage, as discussed above. Secondly, the ***size desideratum*** says we want sets of small size, since these convey more detailed information and may be more useful in practice. Lastly, the ***adaptiveness desideratum*** says we want the sets to communicate instance-wise uncertainty: they should be smaller for easy test-time examples than for hard ones; see Figure 1 for an illustration. Coverage and size are obviously competing objectives, but size and adaptiveness are also often in tension. The size desideratum seeks small sets, while the adaptiveness desideratum seeks larger sets for instances where the classifier is very uncertain. For example, always predicting a set of size five could achieve coverage, but it is not adaptive. We will investigate the behavior of various prediction sets along all three axes in our experiments.

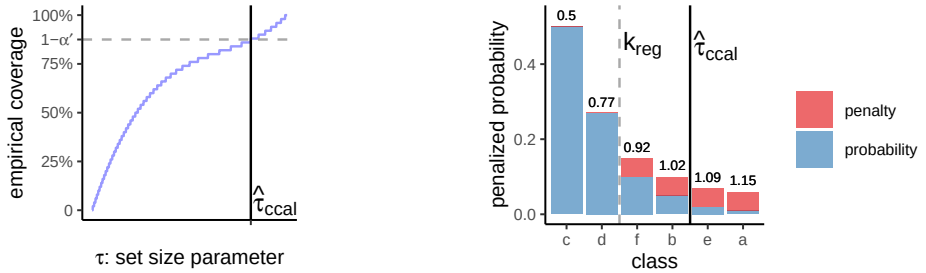
We now turn to the specifics of our proposed method. We begin in Subsection 2.1 by describing an abstract data-splitting procedure called *conformal calibration* that enables the near-automatic construction of valid predictive sets (that is, sets satisfying Eq. (1)). Subsequently, in Subsection 2.2, we provide a detailed presentation of a concrete procedure for performing conformal calibration. In

Algorithm 1 Naive Prediction Sets

Input: α, sorted scores $scores$, associated permutation of classes I, $rand$

procedure NAIVE($\alpha, scores, I, rand$)
  $L \leftarrow 0$
  while $\text{sum}(scores[0:L]) < 1-\alpha$ **do** ▷ Stop if $1-\alpha$ probability exceeded
    $L \leftarrow L+1$
  if $rand$ **then** ▷ Break ties randomly (explained in Appendix C)
    $U \leftarrow \text{Unif}(0,1)$
    $V \leftarrow (scores[0:L] - (1-\alpha))/scores[L]$ ▷ Vectors are zero-indexed in pseudocode
    if $U \leq V$ **then**
      $L \leftarrow L-1$
  return $I[0:L]$

Output: The $1-\alpha$ prediction set, $I[0:L]$



(a) Conformal calibration

(b) A RAPS prediction set

Figure 3: **Visualizations of conformal calibration and RAPS sets.** In the left panel, the y-axis shows the empirical coverage on the conformal calibration set, and $1-\alpha' = \lceil (n+1)(1-\alpha) \rceil / n$. In the right panel, the printed numbers indicate the cumulative probability plus penalty mass. For the indicated value $\hat{\tau}_{\text{ccal}}$, the RAPS prediction set is $\{\text{c, d, f, b}\}$.

Subsection 2.3 we discuss the optimality of our procedure, proving that it is strictly better than any procedure that returns sets of a fixed size, unlike alternative approaches.

2.1 Conformal calibration

We first review a general technique for producing valid prediction sets. The idea is intuitive: consider a procedure that outputs a predictive set for each observation, and further suppose that this procedure has a tuning parameter τ that controls the size of the sets, as elaborated in Gupta et al. (2019). (In RAPS, τ is the cumulative sum of the sorted, penalized classifier scores.) We take a small independent *conformal calibration set* of data, and then choose the tuning parameter τ such that the predictive sets are large enough to achieve $1-\alpha$ coverage on this conformal calibration set. See Figure 3 for an illustration. This calibration step yields a choice of τ, and the resulting set is formally guaranteed to have coverage $1-\alpha$ on a future test point from the same distribution; see Theorem 1 below.

Formally, let $(X_i, Y_i)_{i=1,\ldots,n}$ be an independent and identically distributed (i.i.d.) set of variables that was not used for model training. Further, let $\mathcal{C}(x, u, \tau) : \mathbb{R}^d \times [0,1] \times \mathbb{R} \to 2^{\mathcal{Y}}$ be a set-valued function that takes a feature vector x to a subset of the possible labels. The second argument u is included to allow for randomized procedures; let $U_1, \ldots, U_n$ be i.i.d. uniform $[0,1]$ random variables that will serve as the second argument for each data point. Suppose that the sets are indexed by τ such that they are *nested*, meaning larger values of τ lead to larger sets:

$$\mathcal{C}(x, u, \tau_1) \subseteq \mathcal{C}(x, u, \tau_2) \quad \text{if} \quad \tau_1 \leq \tau_2. \tag{2}$$

To find a function that will achieve $1-\alpha$ coverage on test data, we select the smallest τ that gives at least $1-\alpha$ coverage on the conformal calibration set, with a slight correction to account for the finite sample size:

$$\hat{\tau}_{\text{ccal}} = \inf \left\{ \tau : \frac{|\{i : Y_i \in \mathcal{C}(X_i, U_i, \tau)\}|}{n} \geq \frac{\lceil (n+1)(1-\alpha) \rceil}{n} \right\}. \tag{3}$$

The set function $\mathcal{C}(x, u, \tau)$ with this data-driven choice of τ is guaranteed to have correct finite-sample coverage on a fresh test observation, as stated formally next.

Theorem 1 (Conformal calibration coverage guarantee)**.** *Suppose $(X_i, Y_i, U_i)_{i=1,\ldots,n}$ and $(X_{n+1}, Y_{n+1}, U_{n+1})$ are i.i.d. and let $\mathcal{C}(x, u, \tau)$ be a set-valued function satisfying the nesting property in Eq. (2). Suppose further that the sets $\mathcal{C}(x, u, \tau)$ grow to include all labels for large enough τ: for all $x \in \mathbb{R}^d$, $\mathcal{C}(x, u, \tau) = \mathcal{Y}$ for some τ. Then for $\hat{\tau}_{\text{ccal}}$ defined as in Eq. (3), we have the following coverage guarantee:*

$$P\Big(Y_{n+1} \in \mathcal{C}(X_{n+1}, U_{n+1}, \hat{\tau}_{\text{ccal}})\Big) \geq 1-\alpha.$$

This is the same coverage property as Eq. (1) in the introduction, written in a more explicit manner. The result is not new—a special case first appears in the regression setting in Papadopoulos et al. (2002), and subsequent work adapts the idea to classification (e.g., Hechtlinger et al., 2018).

As a technical remark, for most families of set-valued functions $\mathcal{C}(x, u, \tau)$ there is a matching upper bound:

$$P\Big(Y_{n+1} \in \mathcal{C}(X_{n+1}, U_{n+1}, \hat{\tau}_{\text{ccal}})\Big) \leq 1 - \alpha + \frac{1}{n+1}.$$

Roughly speaking, this will hold whenever the sets grow smoothly in τ. See Lei et al. (2018) for a formal statement of the required conditions.

2.2 Our method

Conformal calibration is a powerful general idea, allowing one to achieve the coverage desideratum for any choice of sets $\mathcal{C}(x, u, \tau)$. Nonetheless, this is not yet a full solution, since the quality of the resulting prediction sets can vary dramatically depending on the design of $\mathcal{C}(x, u, \tau)$. In particular, we recall the size and adaptiveness desiderata from Section 1—we want our uncertainty sets to be as small as possible while faithfully articulating the instance-wise uncertainty of each test point. In this section, we explicitly give our algorithm, which can be viewed as a special case of conformal calibration with the uncertainty sets $\mathcal{C}$ designed to extract information from CNNs.

Our algorithm has three main ingredients. First, for a feature vector x, the base model computes class probabilities $\hat{\pi}_x \in \mathbb{R}^k$, and we order the classes from most probable to least probable. Then, we add a regularization term to promote small predictive sets. Finally, we conformally calibrate the penalized prediction sets to guarantee exact coverage on future test points.

Formally, let $\rho_x(y) = \sum_{y'=1}^{K} \hat{\pi}_x(y') \mathbb{I}_{\{\hat{\pi}_x(y') > \hat{\pi}_x(y)\}}$ be the total probability mass of the set of labels that are more likely than y. These are all the labels that will be included before y is included. In addition, let $o_x(y) = |\{y' \in \mathcal{Y} : \hat{\pi}_x(y') \geq \hat{\pi}_x(y)\}|$ be the ranking of y among the label based on the probabilities $\hat{\pi}$. For example, if y is the third most likely label, then $o_x(y) = 3$.[1] We take

$$\mathcal{C}^*(x, u, \tau) := \Big\{ y \ : \ \rho_x(y) + \hat{\pi}_x(y) \cdot u + \underbrace{\lambda \cdot (o_x(y) - k_{reg})^+}_{\text{regularization}} \leq \tau \Big\}, \tag{4}$$

where $(z)^+$ denotes the positive part of z and $\lambda, k_{reg} \geq 0$ are regularization hyperparameters that are introduced to encourage small set sizes. See Figure 3 for a visualization of a `RAPS` predictive set and Appendix B for a discussion of how to select k_{reg} and λ.

Since this is the heart of our proposal, we carefully parse each term. First, the $\rho_x(y)$ term increases as y ranges from the most probable to least probable label, so our sets will prefer to include the y that are predicted to be the most probable. The second term, $\hat{\pi}_x(y) \cdot u$, is a randomized term to handle the fact that the value will jump discretely with the inclusion of each new y. The randomization term can never impact more than one value of y: there is at most one value of y such that $y \in \mathcal{C}(x, 0, \tau)$ but $y \notin \mathcal{C}(x, 1, \tau)$. These first two terms can be viewed as the CDF transform after arranging the classes from most likely to least likely, randomized in the usual way to result in a continuous uniform random variable (cf. Romano et al., 2020). We discuss randomization further in Appendix C.

Lastly, the regularization promotes small set sizes: for values of y that occur farther down the ordered list of classes, the term $\lambda \cdot (o_x(y) - k_{reg})^+$ makes that value of y require a higher value of τ before it is included in the predictive set. For example, if $k_{reg} = 5$, then the sixth most likely value of y has an extra penalty of size λ, so it will never be included until τ exceeds $\rho_x(y) + \hat{\pi}_x(y) \cdot u + \lambda$, whereas it enters when τ exceeds $\rho_x(y) + \hat{\pi}_x(y) \cdot u$ in the nonregularized version.

Our method has the following coverage property:

Proposition 1 (`RAPS` coverage guarantee)**.** *Suppose $(X_i, Y_i, U_i)_{i=1,\ldots,n}$ and $(X_{n+1}, Y_{n+1}, U_{n+1})$ are i.i.d. and let $\mathcal{C}^*(x, u, \tau)$ be defined as in Eq. (4). Suppose further that $\hat{\pi}_x(y) > 0$ for all x and y. Then for $\hat{\tau}_{\text{ccal}}$ defined as in Eq. (3), we have the following coverage guarantee:*

$$1 - \alpha \leq P\Big(Y_{n+1} \in \mathcal{C}^*(X_{n+1}, U_{n+1}, \hat{\tau}_{\text{ccal}})\Big) \leq 1 - \alpha + \frac{1}{n+1}.$$

Note that the first inequality is a corollary of Theorem 1, and the second inequality is a special case of the remark in Section 2.1. The restriction that $\hat{\pi}_x(y) > 0$ is not necessary for the first inequality.

[1] For ease of notation, we assume the probabilities are distinct. Otherwise, ties should be broken randomly in the label ordering.

Algorithm 2 RAPS Conformal Calibration

Input: α; n tuples $(scores_i, I_i, y_i)_{i=1}^n$ containing sorted scores, the associated permutation of indexes, and labels for each example in the calibration set; k_{reg}; λ; $rand$

procedure RAPSC(α,$(scores_i, I_i, y_i, k_{reg})_{i=1}^n$,$\lambda$)
  for $i \in \{1, \cdots, n\}$ **do**
    $L_i \leftarrow \{\ j\ :\ I_i[j] = y_i\ \}$
    $E_i \leftarrow \Sigma_{j=0}^{L_i} scores_i[j] + \lambda(L_i - k_{reg} + 1)^+$
    if $rand$ **then**
      $U \sim \text{Unif}(0, 1)$
      $E_i \leftarrow E_i - scores_i[L_i] + U * scores_i[L_i]$
  $\hat{\tau}_{ccal} \leftarrow$ the $\lceil (1-\alpha)(1+n) \rceil$ largest value in $\{E_i\}_{i=1}^n$
  return $\hat{\tau}_{ccal}$

Output: The generalized quantile, $\hat{\tau}_{ccal}$ ▷ The value in Eq. (3)

Algorithm 3 RAPS Prediction Sets

Input: α, sorted $scores$ and the associated permutation of classes I for a test-time example, $\hat{\tau}_{ccal}$ from Algorithm 2, k_{reg}, λ, $rand$

procedure RAPS($\alpha, scores, I, \hat{\tau}_{ccal}, k_{reg}, \lambda, rand$)
  $L \leftarrow \mid j \in \mathcal{Y}\ :\ \Sigma_{i=0}^{j} scores[i] + \lambda(L - k_{reg})^+ \leq \hat{\tau}_{ccal} \mid + 1$
  $V \leftarrow (\hat{\tau}_{ccal} - \Sigma_{i=0}^{L-1} scores[i] - \lambda(L - k_{reg})^+ + scores[L-1])/scores[L-1]$
  if $rand$ & $V \leq U \sim \text{Unif}(0, 1)$ **then**
    $L \leftarrow L - 1$
  return $\mathcal{C} = I[0 : L]$ ▷ The L most likely classes

Output: The $1 - \alpha$ confidence set, $\mathcal{C}$ ▷ The set in Eq. (4)

2.3 Optimality considerations

In our experiments, the procedure that always returns the five most likely classes often has coverage above 90%, yet APS produces very large sets on average (see Table 1). To explain this phenomenon, note that small probability estimates (say, after the tenth most likely score) are dominated by noise. This leads to a *permutation problem* of the unlikely classes, whose ordering is determined mostly by random chance. If 1% of the true classes from the calibration set are deep in the tail due to the permutation problem, APS will choose large 99% predictive sets; see Figure 2. The inclusion of the RAPS regularization effectively eliminates this behavior; see Figure 4. To complement these experimental results, we now formally prove that RAPS with the correct regularization parameters will always dominate the simple procedure that returns a fixed set size. As we show in Appendix B, these parameters are easy to select and the method is not sensitive to their values.

For a feature vector x, let $\hat{y}_{(1)}(x)$ be the label with the highest predicted probability, $\hat{y}_{(2)}(x)$ be the label with the second highest predicted probability, and so on. We define the *top-k predictive sets* to be $\{\hat{y}_{(1)}(x), \ldots, \hat{y}_{(k)}(x)\}$.

Proposition 2 (RAPS dominates top-k sets)**.** *Suppose $(X_i, Y_i, U_i)_{i=1,\ldots,n}$ and $(X_{n+1}, Y_{n+1}, U_{n+1})$ are i.i.d. draws. Let k^* be the smallest k such that the top-k predictive sets have coverage at least $\lceil (n+1)(1-\alpha) \rceil / n$ on the conformal calibration points $(X_i, Y_i)_{i=1,\ldots,n}$. Take $\mathcal{C}^*(x, u, \tau)$ as in Eq. (4) with any $k_{reg} \leq k^*$ and $\lambda = 1$. Then with $\hat{\tau}_{\text{ccal}}$ chosen as in Eq. (3), we have*

$$\mathcal{C}^*(X_{n+1}, U_{n+1}, \hat{\tau}_{\text{ccal}}) \subset \{\hat{y}_{(1)}(x), \ldots, \hat{y}_{(k^*)}(x)\}.$$

In words, the RAPS procedure with heavy regularization will be strictly better than the top-k procedure in the sense that it has smaller sets while maintaining the desired coverage level. This is not true of either the naive baseline or the APS procedure; Figure 4 shows that these two procedures sometimes return predictive sets with size much larger than k^*.

3 Experiments

In this section we report on experiments that study the performance of the predictive sets from naive, APS, and RAPS, evaluating each based on the three desiderata above. We begin with a brief

	Accuracy		Coverage α=0.05			Size α=0.05			Coverage α=0.10			Size α=0.10		
Model	Top-1	Top-5	Naive	APS	RAPS	Naive	APS	RAPS	Naive	APS	RAPS	Naive	APS	RAPS
ResNet152	.783	.940	.943	.950	.950	19.8	22.6	**5.43**	.894	.900	.901	9.72	10.3	**2.51**
ResNet101	.774	.935	.944	.950	.950	20.6	23.0	**6.78**	.896	.900	.900	10.2	10.9	**2.61**
ResNet50	.761	.929	.943	.950	.951	22.8	26.2	**7.31**	.895	.900	.900	11.6	12.3	**2.79**
ResNet18	.698	.891	.942	.950	.950	28.6	33.3	**15.3**	.896	.900	.900	15.4	16.1	**4.77**
ResNeXt101	.793	.945	.938	.950	.950	36.0	46.5	**5.18**	.889	.899	.901	17.1	19.4	**2.47**
DenseNet161	.771	.936	.942	.950	.950	23.7	28.0	**6.86**	.894	.899	.900	11.2	12.0	**2.66**
VGG16	.716	.904	.943	.950	.950	24.5	27.9	**12.2**	.896	.900	.900	13.4	14.1	**3.52**
InceptionV2	.695	.886	.938	.950	.950	142.	169.	**19.2**	.884	.900	.900	74.9	88.7	**5.52**
ShuffleNet	.694	.883	.940	.950	.950	58.9	71.3	**18.5**	.891	.901	.900	28.5	32.3	**5.58**

Table 1: **Results on Imagenet-Val.** We report coverage and size of `naive`, `APS`, and `RAPS` sets for nine different Imagenet classifiers. The median-of-means for each column is reported over 100 different trials. See Section 3.1 for full details.

preview of the experiments. In **Experiment 1**, we evaluated `naive`, `APS`, and `RAPS` on Imagenet-Val. Both `APS` and `RAPS` provided exact coverage, while `naive` sets had coverage slightly below the specified level. `APS` had larger sets on average than `naive` and `RAPS`. `RAPS` had a much smaller average set size than `APS` and `naive`. In **Experiment 2**, we repeated Experiment 1 on Imagenet-V2, and the conclusions still held. In **Experiment 3**, we produced histograms of set sizes for `naive`, `APS`, and `RAPS` for several different values of λ, illustrating a simple tradeoff between set size and adaptiveness. Finally, in **Experiment 4**, we computed histograms of `RAPS` sets stratified by image difficulty, showing that `RAPS` sets are smaller for easier images than for difficult ones.

In our experiments, we use nine standard, pretrained Imagenet classifiers from the `torchvision` open source repository (Paszke et al., 2019) with standard normalization, resize, and crop parameters, facilitating reproducibility and avoiding bias. Before applying `naive`, `APS`, or `RAPS`, we calibrated the classifiers using the standard temperature scaling/Platt scaling procedure described in Guo et al. (2017). Thereafter, `naive`, `APS`, and `RAPS` were applied, with `RAPS` using parameters $k_{reg} = 5$ and $\lambda = 0.2$. We used the randomized versions of these algorithms—see Appendix C for a discussion.

3.1 Experiment 1: coverage vs set size on Imagenet

In this experiment, we calculated the coverage and mean set size of each procedure for two different choices of α. Over 100 trials, we randomly sampled two subsets of Imagenet-Val: one conformal calibration subset of size 20K and one evaluation subset of size 20K. The median-of-means over trials for both coverage and set size are reported in Table 1. Figure 2 illustrates the performances of `naive`, `APS`, and `RAPS`; `RAPS` has much smaller sets than both `naive` and `APS`, while achieving exact coverage.

3.2 Experiment 2: coverage vs set size on Imagenet-V2

The same procedure as Experiment 1 was repeated on Imagenet-V2, with exactly the same normalization, resize, and crop parameters. The size of the calibration and evaluation sets was 5K, since Imagenet-V2 is a smaller dataset. The result shows that our method can still provide exact coverage under a significant distribution shift, as long as the conformal calibration set comes from the new distribution. The variance of the coverage is higher due to the smaller calibration and test set sizes.

3.3 Experiment 3: Set sizes of `naive`, `APS`, and `RAPS` on Imagenet

We demonstrate that regularization leads to smaller set sizes in more detail. For three values of λ, we collected the set sizes produced by each of `naive`, `APS`, and `RAPS` and plotted them as separate histograms on the same axis. The left panels of Figure 4 display the results of this experiment. As λ grows, the average set size decreases.

3.4 Experiment 4: Adaptiveness of `RAPS` on Imagenet

Lastly, we show that `RAPS` sets are smaller for easy images than hard ones, addressing the adaptiveness desideratum. We stratified the evaluation set into easy, medium, and hard images. Easy images are instances the classifier got correct. Medium difficulty images are examples where the

	Accuracy		Coverage α=0.05			Size α=0.05			Coverage α=0.10			Size α=0.10		
Model	Top-1	Top-5	Naive	APS	RAPS	Naive	APS	RAPS	Naive	APS	RAPS	Naive	APS	RAPS
ResNet152	.671	.876	.943	.950	.950	50.5	57.4	**20.4**	.896	.901	.900	25.9	27.5	**6.62**
ResNet101	.657	.860	.943	.950	.949	55.1	62.3	**24.6**	.894	.900	.899	28.3	30.3	**8.44**
ResNet50	.633	.846	.944	.950	.950	59.1	65.7	**29.2**	.894	.900	.901	30.5	32.6	**9.64**
ResNet18	.572	.802	.942	.950	.950	64.9	73.8	**48.2**	.894	.901	.901	35.1	37.7	**16.6**
ResNeXt101	.679	.875	.937	.949	.950	85.4	106.	**21.2**	.887	.901	.900	42.7	51.3	**6.77**
DenseNet161	.653	.861	.941	.950	.950	59.7	72.5	**25.2**	.894	.901	.900	59.7	72.5	**25.2**
VGG16	.588	.816	.943	.951	.940	57.4	65.1	**38.2**	.897	.900	.899	31.7	32.8	**14.4**
InceptionV2	.573	.797	.943	.950	.950	251.	275.	**72.4**	.893	.900	.900	145.	155.	**20.7**
ShuffleNet	.559	.781	.942	.950	.951	126.	144.	**80.2**	.893	.900	.900	66.2	71.8	**25.5**

Table 2: **Results on Imagenet-V2.** We report coverage and size of `naive`, `APS`, and `RAPS` sets for nine different Imagenet classifiers. This table is produced the same way as Table 1, except that each model is instead calibrated and tested on Imagenet-V2 over subsets of size 5K.

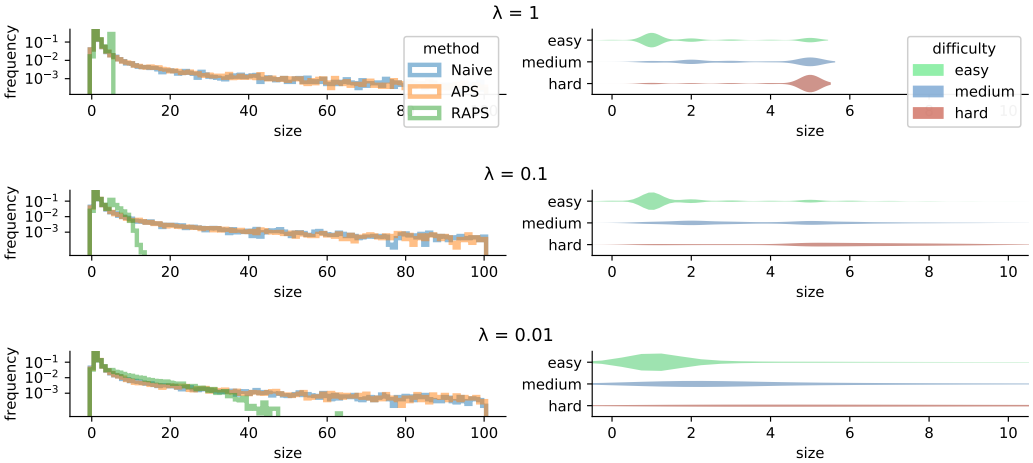


Figure 4: **Set sizes produced with ResNet-152.** Left: size of the prediction sets from three methods across all test examples. Right: size of the `RAPS` prediction sets stratified by image difficulty. Fatter regions of the violins indicate more examples with that size. See Section 3.4 for details.

classifier's second or third highest classes were correct. Hard images are those remaining. Then, we ran `RAPS` over each of these strata and plotted them as violins in the right panels of Figure 4. When λ is small, `RAPS` allows sets to be large. But when λ is very large, `RAPS` essentially clips sets to be a maximum of size 5. Experiments 3 and 4 together illustrate the tradeoff between the adaptiveness and size desiderata: as the average set size decreases, the `RAPS` procedure truncates sets larger than the smallest fixed set that provides coverage. Loosely speaking, this happens because hard examples are undercovered; since large sets are penalized, the conformal calibration procedure must increase set sizes on easy examples to ensure the $1-\alpha$ coverage guarantee is satisfied over the dataset as a whole. However, the size only increases slightly for a small fraction of easy images, since they are more common than hard ones, and the total probability mass can often exceed $\hat{\tau}_{ccal}$ by including only one more class. We further explore this experimentally in Tables 4 and 5 (see Appendix D).

4 Discussion

Our method enables a researcher to take any base classifier and return predictive sets guaranteed to achieve a pre-specified error level, such as 90%, while retaining small average size. It is simple to deploy, so it is an attractive, automatic way to quantify the uncertainty of image classifiers—an essential task in such settings as medical diagnostics, self-driving vehicles, and flagging dangerous internet content. Predictive sets in computer vision have many further uses, since they systematically identify hard test-time examples. Finding such examples is useful in active learning where one only has resources to label a small number of points. In a different direction, one can improve efficiency of a classifier by using a cheap classifier outputting a prediction set first, and an expensive one only when the cheap classifier outputs a large set (a *cascade*; see, e.g., Li et al. 2015). Finally, one can use predictive sets during model development to identify failure cases and outliers. This helps probe the model's weaknesses, suggesting strategies for improving its performance.

REFERENCES

Maxime Cauchois, Suyash Gupta, and John Duchi. Knowing what you know: valid and validated confidence sets in multiclass and multilabel prediction. 2020. arXiv:2004.10181.

Yarin Gal. Uncertainty in Deep Learning. *University of Cambridge*, 1(3), 2016.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. 2017. arXiv:1706.04599.

Chirag Gupta, Arun K Kuchibhotla, and Aaditya K Ramdas. Nested conformal prediction and quantile out-of-bag ensemble methods. *arXiv*, pp. arXiv–1910, 2019.

Yotam Hechtlinger, Barnabás Póczos, and Larry Wasserman. Cautious deep learning, 2018. arXiv:1805.09460.

Heinrich Jiang, Been Kim, Melody Guan, and Maya Gupta. To trust or not to trust a classifier. In *Advances in Neural Information Processing Systems*, pp. 5541–5552, 2018.

Volodymyr Kuleshov, Nathan Fenner, and Stefano Ermon. Accurate uncertainties for deep learning using calibrated regression. 2018. arXiv:1807.00263.

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Advances in Neural Information Processing Systems*, pp. 6402–6413, 2017.

Jing Lei, Max G'Sell, Alessandro Rinaldo, Ryan J. Tibshirani, and Larry Wasserman. Distribution-free predictive inference for regression. *Journal of the American Statistical Association*, 113 (523):1094–1111, 2018. doi: 10.1080/01621459.2017.1307116.

Haoxiang Li, Zhe Lin, Xiaohui Shen, Jonathan Brandt, and Gang Hua. A convolutional neural network cascade for face detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5325–5334, 2015.

Qing Li, Weidong Cai, Xiaogang Wang, Yun Zhou, David Dagan Feng, and Mei Chen. Medical image classification with convolutional neural network. In *2014 13th International Conference on Control Automation Robotics & Vision (ICARCV)*, pp. 844–848. IEEE, 2014.

Alexander Selvikvåg Lundervold and Arvid Lundervold. An overview of deep learning in medical imaging focusing on MRI. *Zeitschrift für Medizinische Physik*, 29(2):102–127, 2019.

David JC MacKay. *Bayesian methods for adaptive models*. PhD thesis, California Institute of Technology, 1992.

Soundouss Messoudi, Sylvain Rousseau, and Sebastien Destercke. Deep conformal prediction for robust models. In *Information Processing and Management of Uncertainty in Knowledge-Based Systems*, pp. 528–540. Springer, 2020.

Radford M Neal. *Bayesian Learning for Neural Networks*. Springer Science & Business Media, 2012.

Jeremy Nixon, Michael W Dusenberry, Linchuan Zhang, Ghassen Jerfel, and Dustin Tran. Measuring calibration in deep learning. In *CVPR Workshops*, pp. 38–41, 2019.

Harris Papadopoulos, Kostas Proedrou, Vladimir Vovk, and Alex Gammerman. Inductive confidence machines for regression. In *Machine Learning: European Conference on Machine Learning ECML 2002*, pp. 345–356, 2002.

Sangdon Park, Osbert Bastani, Nikolai Matni, and Insup Lee. PAC confidence sets for deep neural networks via calibrated prediction. 2019. arXiv:2001.00106.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, pp. 8026–8037, 2019.

John Platt et al. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advances in Large Margin Classifiers*, 10(3):61–74, 1999.

Joaquin Quinonero-Candela, Carl Edward Rasmussen, Fabian Sinz, Olivier Bousquet, and Bernhard Schölkopf. Evaluating predictive uncertainty challenge. In *Machine Learning Challenges Workshop*, pp. 1–27. Springer, 2005.

Muhammad Imran Razzak, Saeeda Naz, and Ahmad Zaib. Deep learning for medical image processing: Overview, challenges and the future. In *Classification in BioApps*, pp. 323–350. Springer, 2018.

Carlos Riquelme, George Tucker, and Jasper Snoek. Deep Bayesian bandits showdown: An empirical comparison of Bayesian deep networks for Thompson sampling. 2018. arXiv:1802.09127.

Yaniv Romano, Matteo Sesia, and Emmanuel J. Candès. Classification with valid and adaptive coverage. 2020. arXiv:2006.02544.

Murat Sensoy, Lance Kaplan, and Melih Kandemir. Evidential deep learning to quantify classification uncertainty. In *Advances in Neural Information Processing Systems*, pp. 3179–3189, 2018.

Vladimir Vovk, Alexander Gammerman, and Craig Saunders. Machine-learning applications of algorithmic randomness. In *International Conference on Machine Learning*, pp. 444–453, 1999.

Vladimir Vovk, Alex Gammerman, and Glenn Shafer. *Algorithmic Learning in a Random World*. Springer, 2005. doi: 10.1007/b106715.

A PROOFS

Theorem 1. Let $s(x, u, y) = \inf_\tau\{y \in \mathcal{C}(x, u, \tau)\}$, and let $s_i = s(X_i, U_i, Y_i)$ for $i = 1, \ldots, n$. Then

$$\{y : s(x, u, y) \leq \tau\} = \{y : y \in \mathcal{C}(x, u, \tau)\}$$

because $\mathcal{C}(x, u, \tau)$ is a finite set growing in τ by the assumption in Eq. (2). Thus,

$$\{\tau : |\{i : s_i \leq \tau\}| \geq \lceil (1-\alpha)(n+1) \rceil\} = \left\{\tau : \frac{|\{i : Y_i \in \mathcal{C}(X_i, U_i, \tau)\}|}{n} \geq \frac{\lceil (n+1)(1-\alpha) \rceil}{n}\right\}.$$

Considering the left expression, the infimum over τ of the set on the left hand side is the $\lceil (1-\alpha)(n+1) \rceil$ smallest value of the s_i, so this is the value of $\hat{\tau}_{\text{ccal}}$. Since $s_1, \ldots, s_n, s(X_{n+1}, U_{n+1}, Y_{n+1})$ are exchangeable random variables, $|\{i : s(X_{n+1}, U_{n+1}, Y_{n+1}) > s_i\}|$ is stochastically dominated by the discrete uniform distribution on $\{0, 1, \ldots, n\}$. We thus have that

$$\begin{aligned}
P\left(Y_{n+1} \notin \mathcal{C}(X_{n+1}, U_{n+1}, \hat{\tau}_{\text{ccal}})\right) &= P\left(s(X_{n+1}, U_{n+1}, Y_{n+1}) > \hat{\tau}_{\text{ccal}}\right) \\
&= P\left(|\{i : s(X_{n+1}, U_{n+1}, Y_{n+1}) > s_i\}| \geq \lceil (n+1)(1-\alpha) \rceil\right) \\
&= P\left(\frac{|\{i : s(X_{n+1}, U_{n+1}, Y_{n+1}) > s_i\}|}{n+1} \geq \frac{\lceil (n+1)(1-\alpha) \rceil}{n+1}\right) \\
&\leq \alpha.
\end{aligned}$$

□

Proposition 1. The lower bound follows from Theorem 1. To prove the upper bound, using the result from Theorem 2.2 of Lei et al. (2018) it suffices to show that the variables $s(X_i, U_i, Y_i) = \inf\{\tau : Y_i \in \mathcal{C}(X_i, U_i, \tau)\}$ are almost surely distinct. To this end, note that that

$$s(X_i, U_i, Y_i) = \rho_{X_i}(Y_i) + \hat{\pi}_{X_i}(Y_i) \cdot U_i + \lambda(o_{X_i}(Y_i) - k_{reg})^+,$$

and due to the middle term of the sum, these values are distinct almost surely provided $\hat{\pi}_{X_i}(Y_i) > 0$. □

$k_{reg} \mid \lambda$	0	0.0001	0.001	0.01	0.02	0.05	0.2	0.5	0.7	1.0
1	11.212	10.155	7.022	3.594	2.936	2.265	2.09	2.256	2.246	2.246
2	11.212	10.162	7.078	3.652	3.03	2.42	2.108	2.256	2.246	2.246
3	11.212	10.174	7.127	3.762	3.149	2.594	2.194	2.161	2.173	2.182
4	11.212	10.185	7.175	3.851	3.285	2.756	2.393	2.291	2.285	2.281
5	11.212	10.198	7.211	3.938	3.408	2.921	2.577	2.502	2.489	2.486
6	11.212	10.206	7.261	4.056	3.516	3.077	2.744	2.689	2.694	2.694
10	11.212	10.245	7.389	4.464	3.975	3.648	3.387	3.359	3.355	3.355
20	11.212	10.343	7.747	5.334	4.991	4.71	4.566	4.544	4.544	4.544
50	11.212	10.599	8.731	7.234	7.044	6.914	6.871	6.866	6.866	6.866
100	11.212	10.875	9.926	9.261	9.192	9.15	9.132	9.132	9.132	9.132
500	11.212	11.212	11.212	11.212	11.212	11.212	11.212	11.212	11.212	11.212

Table 3: Set sizes of RAPS with parameters k_{reg} and λ, a ResNet-152, and coverage level 90%.

Proposition 2. We first show that $\hat{\tau}_{\text{ccal}} \leq 1 + k^* - k_{reg}$. Note that since at least $\lceil (1-\alpha)(n+1) \rceil$ of the conformal calibration points are covered by a set of size k^*, at least $\lceil (1-\alpha)(n+1) \rceil$ of the E_i in Algorithm 2 are less than or equal to $1 + k^* - k_{reg}$. Thus, by the definition of $\hat{\tau}_{\text{ccal}}$, we have that it is less than or equal to $1 + k^* - k_{reg}$. Then, note that by the definition of $\mathcal{C}^*$ in Eq. (4), we have that

$$|\mathcal{C}^*(X_{n+1}, U_{n+1}, \hat{\tau}_{\text{ccal}})| \leq k^*.$$

as long as $\hat{\tau}_{\text{ccal}} \leq 1 + k^* - k_{reg}$, since for the $k^* + 1$ most likely class, the sum in Eq. (4) will exceed $\lambda \cdot (1 + k^* - k_{reg}) = (1 + k^* - k_{reg}) \geq \hat{\tau}_{\text{ccal}}$, and so the $k^* + 1$ class will not be in the set. □

B PICKING k_{reg} AND λ

While any value of the tuning parameters λ and k_{reg} lead to exact coverage (Proposition 1), some values will lead to smaller sets. We proved that with $k_{reg} = k^*$ and $\lambda \geq 1$, RAPS outputs sets no larger than k^* (Proposition 2). In practice, the sets are often much smaller, as we show across our experiments. Furthermore, in our experiments we chose $k_{reg} = 5$ and $\lambda = 0.2$, which achieved strong results for many models (which have varying accuracy) and choices of the coverage level, illustrating our method's robustness to these parameter choices. We report on a wider set of parameter values in Table 3, and find that the performance is strong for many parameter choices; indeed, our fixed choice of parameters presented in Table 1 is not even the optimal set of parameters, it was just taken as a reasonably good representative value.

To build intuition for the role of k_{reg}, consider the following scenarios where we set $\lambda = 1$. If $k_{reg} \leq k^* - 2$, the sets output by RAPS will always be of size k^* or $k^* - 1$; these sets are still small, but not adaptive. If $k_{reg} = k^* + m$, $m \in \mathbb{N}$, then RAPS would only penalize sets of size $k^* + m$ and above, losing the guarantee described in Proposition 2. Therefore, it is natural to pick $k_{reg} \approx k^*$. For example, if the desired coverage is 90% and the top-5 accuracy of the model is around 90%, we would pick $k_{reg} = 5$. Turning to λ, RAPS with a larger λ tends to cover fewer difficult examples and more easy ones. Small values of λ like 0.01 will work reasonably well in many settings, which makes them a attractive default choice, but tuning both k^* and λ can lead to even better performance.

C RANDOMIZED PREDICTORS

The reader may wonder why we choose to use a randomized procedure. The randomization is needed to achieve $1 - \alpha$ coverage exactly, which we will explain via an example. Note that the randomization is of little practical importance, since the predictive set output by the randomized procedure will differ from the that of the non-randomized procedure by at most one element.

Turning to an example, assume for a particular input image we expect a set of size k to have 91% coverage, and a set of size $k - 1$ to have 89% coverage. In order to achieve our desired coverage of 90%, we randomly choose size k or $k - 1$ with equal probability. In general, the probabilities will not be equal, but rather chosen so the weighted average of the two coverages is exactly 90%. If a user of our method desires deterministic sets, it is easy to turn off this randomization with a single flag, resulting in slightly conservative sets.

		$\lambda = 0$		$\lambda = 0.001$		$\lambda = 0.01$		$\lambda = 0.05$		$\lambda = 0.1$		$\lambda = 0.2$		$\lambda = 1$	
difficulty	count	cvg	sz	cvg	sz	cvg	sz	cvg	sz	cvg	sz	cvg	sz	cvg	sz
1	15668	0.95	5.2	0.95	3.8	0.96	2.5	0.97	2.1	0.97	2.0	0.98	2.0	0.98	2.0
2 to 3	2578	0.78	15.7	0.78	10.5	0.80	6.0	0.83	4.3	0.84	3.9	0.85	3.7	0.86	3.6
4 to 6	717	0.68	31.7	0.68	19.7	0.70	9.7	0.71	6.1	0.71	5.3	0.70	4.8	0.64	4.4
7 to 10	334	0.63	41.0	0.63	24.9	0.60	11.6	0.39	6.9	0.22	5.7	0.06	5.1	0.00	4.5
11 to 100	622	0.55	57.8	0.51	34.1	0.26	14.7	0.02	7.9	0.00	6.4	0.00	5.4	0.00	4.6
101 to 1000	81	0.23	96.7	0.00	51.6	0.00	19.1	0.00	9.1	0.00	7.1	0.00	5.8	0.00	4.7

Table 4: **Coverage and size conditional on difficulty.** We report coverage and size of RAPS sets using ResNet-152 with $k_{reg} = 5$ and varying λ (recall that $\lambda = 0$ is the APS procedure). The desired coverage level is 90%. The 'difficulty' is the ranking of the true class's estimated probability.

	$\lambda = 0$			$\lambda = 0.001$			$\lambda = 0.01$			$\lambda = 0.1$			$\lambda = 1$		
size	cnt	cvg	diff	cnt	cvg	diff	cnt	cvg	diff	cnt	cvg	diff	cnt	cvg	diff
0 to 1	11627	0.88	1.4	11539	0.88	1.4	11225	0.89	1.3	10476	0.92	1.2	10027	0.93	1.2
2 to 3	3687	0.91	2.2	3702	0.91	2.2	3741	0.92	2.1	3845	0.93	2.3	3922	0.94	2.2
4 to 6	1239	0.91	3.0	1290	0.91	3.1	1706	0.92	3.3	4221	0.89	4.1	6051	0.83	6.8
7 to 10	688	0.93	4.1	765	0.93	4.2	1314	0.91	4.7	1436	0.71	15.2	0		
11 to 100	2207	0.94	8.1	2604	0.93	10.7	2014	0.86	13.3	22	0.59	42.5	0		
101 to 1000	552	0.97	23.8	100	0.90	27.4	0			0			0		

Table 5: **Coverage and difficulty conditional on set sizes produced by RAPS.** This table is produced the same way as Table 4, but reported differently.

D COVERAGE AND SIZE CONDITIONAL ON IMAGE DIFFICULTY

In order to probe the adaptiveness properties of APS and RAPS we stratified coverage and size by image difficulty (the position of the true label in the list of most likely to least likely classes, based on the classifier predictions) in Table 4. With increasing λ, coverage decreases for more difficult images and increases for easier ones. In the most difficult regime, even though APS can output large sets, those sets still rarely contain the true class. This suggests regularization is a sensible way to stabilize the sets. As a final word on Table 4, notice that as λ increases, coverage improves for the more common medium-difficulty examples, although not for very rare and difficult ones.

In Table 5, we stratified coverage and image difficulty by the set size produced by RAPS. Turning our attention to the $\lambda = 0$ column, we see that when APS outputs a set of size $101 - 1000$, APS has higher than $1 - \alpha$ coverage. If the probabilities output by the CNN were correct, APS would have $1 - \alpha$ coverage in each stratum, so this indicates the probabilities output by the CNN are wrong. This provides further evidence for the the permutation problem discussed in the main text. The APS procedure still achieves coverage by outputting excessively large sets for certain examples (as in the last row of the table). Alternatively, RAPS can be used to regularize the set sizes—for $\lambda = .001$ to $\lambda = .01$ the coverage stratified by set size is more balanced.