Classification with Conceptual Safeguards

Hailey Joren HJOREN@UCSD.EDU
University of California, San Diego

Charlie T. Marx CMARX@STANFORD.EDU
Stanford University

Berk Ustun BERK@UCSD.EDU
University of California, San Diego

Abstract: Machine learning models are often used to automate routine tasks. In settings where mistakes are costly, we can trade off accuracy for coverage by abstaining from making a prediction on instances for which the model is uncertain. In this work, we present a new approach to selective classification in deep learning with concepts. Our approach constructs a concept bottleneck model where the front-end model can make predictions given soft concepts and leverage concept confirmation to improve coverage and performance under abstention. We develop techniques to propagate uncertainty and identify concepts for confirmation. We evaluate our approach on real-world and synthetic datasets, showing that it can improve performance and coverage across a range of tasks.

1. Introduction

One of the most promising applications of modern machine learning is the ability to automate routine tasks. These opportunities include tasks such as diagnosing lesions from dermatology images [13], detecting suspicious activity in credit card transactions [1], or species identification using wildlife footage [25]. Despite rapid progress in automated systems' predictive capabilities, deploying these systems in high-stakes settings remains challenging. Two requirements slow adoption:

1. *Performance*: Models are often used to automate tasks that a human can perform correctly. In these cases, maintaining high performance is expected given that a non-automated system should have near-perfect performance and because even small performance gaps can have a disproportionate impact. This is because automation scales, meaning that systems can compromise safety by rapidly generating a high volume of incorrect predictions. For instance, a small decrease in accuracy in skin cancer detection could result in delayed treatment, posing significant risks to human health and lives.

2. *Interpretability*: One of the barriers to adopting systems that can automate routine tasks is a lack of interpretability. Put simply, stakeholders would like to understand how a system makes its prediction, check that it is working correctly once it is deployed, and maintain the ability to override the prediction in an informed manner.

We struggle to build systems that meet these objectives, as the two goals often conflict with each other. For example, we might have the resources to create a highly accurate model, but this model may not be easily scrutinized. Conversely, we might build a model that is highly interpretable but fails to ensure a sufficient level of accuracy. In this work, we present

a new approach to building deep learning systems that address these challenges by drawing on ideas in selective classification and concept bottleneck models to build a versatile system for automation. Our approach seeks to build a classification model with *conceptual safeguards* (see Fig. 1): i.e., a model that can assign predictions on the basis of human-verifiable concepts, and that can avoid when these systems are not detected. Conceptual safeguards use selective classification to restrict automation to instances where the model is sufficiently confident – i.e., by allowing models to abstain from predicting when they are insufficiently confident. In tandem, conceptual safeguards address interpretability through a concept bottleneck model – which provides an architecture that allows end-users to understand how the system makes its predictions, check the prediction logic and improve coverage by verifying uncertain concepts at prediction time.

Concept bottleneck models will often perform poorly on datasets where we lack sufficiently rich concepts or when given uncertain concept predictions since they must be trained on a dataset of hard concept labels (see Table 1). Overcoming this challenge requires building concept bottleneck models that can handle uncertain predictions, perform well under selective classification, and are amenable to concept confirmation. To build models that can support selective classification, we must quantify the uncertainty in concept predictions and propagate this uncertainty end-to-end, including capturing uncertainty related to concept prediction. To enable concept confirmation, we must build the system in such a way that imputing the true concept value reliably improves performance.

The main contributions of this work include:

1. We introduce conceptual safeguards, a versatile approach to automating classification tasks that promotes safety and interpretability on deep learning tasks with concept labels.

2. We develop a technique to propagate the uncertainty from predicted concepts to predicted labels. The technique improves overall model performance by accounting for uncertainty and supports the development of reliable selective classification in concept bottleneck models.

3. We propose a method for flagging predicted concepts for human confirmation. In our proposed systems, concept confirmation can reliably improve model coverage while maintaining performance.

4. We conduct an empirical investigation of our proposed approach compared with baseline concept bottleneck models equipped with selective classification. Our results show that the proposed approach can improve coverage and performance without confirmation as well as lead to dramatic gains in coverage under concept confirmation.

Related Work

Selective Classification Our work builds on work in selective classification or classification with a reject option. Selective classification is a general-purpose family of methods to build models that abstain from prediction on low-confidence samples [3, 4, 10, 11]. The goal of selective classification is to improve performance while keeping prediction coverage as high as possible. Allowing for abstention also enables partial automation, in which prediction

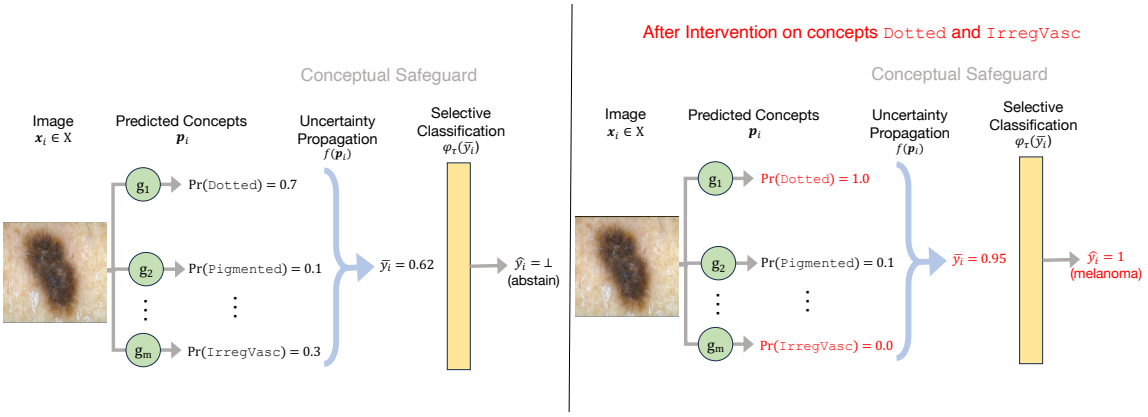

Figure 1: Overview of automation with a conceptual safeguard before confirmation (left) and after confirmation (right). We consider a model to predict melanoma $y = 1$ from an image x of a skin lesion. The model output $\hat{y} \in \{0, 1, \perp\}$ where 1 indicates melanoma and $\perp$ indicates abstention. The system concept predictors $g_1 \cdots g_m$ that output the predicted probability of m human-verifiable concepts such as (*IrregDots*), pigment (*IrregPigment*), vascular structures and (*IrregDots*). The conceptual safeguard takes these probabilities as input. It uses them to propagate uncertainty through the front-end model $f_{up}(\hat{c})$ and applies a selective classification. Under confirmation of concept *IrregDots* (right), the predicted probability is replaced with 1.0 as the concept is present. This in turn updates the predicted probability of the outcome $p(melanoma) = 0.91$, which relieves abstention $h(f_{up}(\hat{c})) = 1.0$ and improves coverage.

is automated only on samples with high certainty [8]. This is useful in applications with strict performance criteria due to the cost of mistakes, such as in cancer diagnosis [7], and in applications where it is difficult to train a model with consistently high performance, such as in species identification [21]. Recent work extends these benefits to deep learning settings [6, 12]. Applying selective classification in deep learning is challenging because most approaches require accurate uncertainty estimates to guide abstention decisions [2, 9]. We seek to overcome this challenge in concept bottleneck models by propagating uncertainty from the concept detectors to the front-end outcome predictor.

Deep Learning with Concepts Our work is related to work on using concept labels in deep learning. In particular, concept bottleneck models seek to improve model interpretability by leveraging dataset annotations to learn concepts and outcome predictions sequentially [17]. Methods for learning with concepts derived from raw inputs are motivated by their ability to provide an unprecedented degree of interpretability, such as allowing observation of counterfactuals or correcting concepts to improve performance. One challenge for learning with concepts includes a need for annotations in training data, which limits the performance of these types of models compared to alternative approaches that do not require concept labels [22, 27, 28].

Performance is further hampered by the requirement that the front-end model be trained on and accept hard concept labels, or else risk label leakage and preclude concept confirmation [14, 20]. While concept bottleneck models can be trained on soft labels, we are specifically interested in concept models that allow confirmation to improve performance and coverage. This challenge limits the overall performance of these types of models [17] and makes it difficult to estimate model uncertainty. We seek to meet the requirements for

confirmation and avoid label leakage while leveraging uncertainty to improve overall model performance.

Model	$\hat{C}$	$\hat{y}$	Intervenability	Performance	Predictions
Independent	$\{0,1\}^m$	$\{0,1\}$	✓	P_I	n
Sequential	$[0,1]^m$	$\{0,1\}$	✗	$P^S \geq P_I$	n
Joint	$[0,1]^m$	$\{0,1\}$	✗	$P^J \geq P_I$	n
Conceptual Safeguards	$[0,1]^m$	$\{0,1,\perp\}$	✓	$P^U \geq P^I$	$\leq n$

Table 1: Conceptual safeguards use an independent model design in which the front-end model is trained on true concepts. This architecture is required if we wish to ensure that *confirmation* – i.e., setting concept predictions with their true values – will improve performance. Our systems enhance an independent concept bottleneck so that it can make use of uncertainty in the concept predictions to improve performance and abstain effectively,

2. Problem Statement

We consider a classification task with the goal of predicting a label using a set of complex features and simple concepts. We start with a dataset of n i.i.d. training instances $\{(\mathbf{x}_i, \mathbf{c}_i, y_i)\}_{i=1}^n$. Each example consists of a vector of d *features* $\mathbf{x}_i \in \mathcal{X} \subseteq \mathbb{R}^d$ (e.g., pixels in an x-ray image), a *label* $y_i \in \mathcal{Y} = \{0,1\}$ (e.g., $y_i = 1$ if patient i has arthritis), and m *concepts* $\mathbf{c}_i \in \mathcal{C} = \{0,1\}^m$ (e.g., $c_{i,k} = 1$ if image i contains a bone spur).

We use the dataset to build a *concept bottleneck model* to predict the label y. A concept bottleneck model consists of two components:

- A *concept detector* $g : \mathcal{X} \to [0,1]^m$, which maps the features to a vector of probabilistic concept predictions $\mathbf{q}_i := g(\mathbf{x}_i) \in [0,1]^m$. The kth entry in this prediction $q_{i,k}$ estimates the probability that the corresponding concept $c_{i,k} = 1$.
- A *front-end model* $f : \mathcal{C} \to \mathcal{Y}$ takes as input a vector of *hard* concepts $\mathbf{c}_i$ and outputs a predicted probability $\overline{y}_i := f(\mathbf{c}_i)$ that $y_i = 1$.

We train the concept detector and front-end model using supervised learning with the datasets $\{(\mathbf{x}_i, \mathbf{c}_i)\}_{i=1}^n$ and $\{(\mathbf{c}_i, y_i)\}_{i=1}^n$, respectively.

- *Decoupled Concept Detectors*: Training the concept detectors separately ensures that we can train each concept detector using as much data as possible. In contrast, training a single model that predicts all concepts forces us to restrict our attention to the subset of training instances where all concepts are labeled. We train the concept detectors $g_1, \ldots, g_m$ to output probability predictions (e.g., ERM with the cross-entropy loss). We calibrate the models using a post-hoc calibration technique (e.g., isotonic temperature scaling [29] or Platt's Scaling [23]).
- *Independent Front-End*: We train the front-end model f independently from the concept detectors – i.e., using the true concepts $\mathbf{c} \in \{0,1\}$ rather than the predicted concepts q. Independent training is a key requirement for any concept bottleneck model where humans intervene on concepts. A front-end model that is trained using the predicted concepts may require incorrect concept predictions to assign accurate label predictions [14, 18, 19].

Model Pipeline We view the concept detector and front-end model as an end-to-end predictor $h : \mathcal{X} \to \{0, 1, \perp\}$, whose outputs $\hat{y}_i := h(\mathbf{x}_i) \in \{0, 1, \perp\})$ includes an *abstention* option $\hat{y}_i = \perp$. The requirements for the predictor are twofold: We control *selective accuracy* – i.e., the accuracy on instances outputs a prediction $\text{Accuracy}(h) := \Pr(y = \hat{y} \mid \hat{y} \neq \perp)$ and we maximize *coverage* – i.e., the proportion of instances on which the model does not abstain $\text{Coverage}(h) := \Pr(\hat{y} \neq \perp)$. Our goal is to maximize coverage subject to the constraint that accuracy is at least $1 - \alpha$, for a user-specified error tolerance $\alpha \in [0, 1]$. We control accuracy and coverage through *abstention* and *concept confirmation* (see Fig. 2).

Abstention To avoid making predictions on instances where confidence is low and errors are likely, we incorporate the option to *abstain*. Given a confidence threshold $\tau \in [0, 1]$, an abstention function $\varphi_\tau : [0, 1] \to \mathcal{Y} \cup \{\perp\}$ takes as input a probabilistic prediction $\overline{y}_i \in [0, 1]$, and returns a prediction:

$$\hat{y}_i = \varphi_\tau(\overline{y}_i) = \begin{cases} 1 & \text{if } \overline{y}_i > 1 - \tau \\ 0 & \text{if } \overline{y}_i < \tau \\ \perp & \text{otherwise} \end{cases}$$

Intuitively, the abstention function rounds the probabilistic prediction to a hard prediction if there is sufficient confidence and otherwise abstains.

Confirmation We let users *confirm*[1] the predicted concepts for any given instance. For example, in the x-ray screening task, we could ask a human to confirm that concept k is present in x-ray i. We assume that all concepts are *human-verifiable* [5]. Given any concept that we choose to confirm. we replace the soft predicted concepts $q_{i,k}$ with the ground truth value $c_{i,k}$.

We write the confirmation process as a function $\psi_S : [0, 1]^m \to [0, 1]^m$ where $S \subseteq [m]$ denotes a subset of concepts to confirm. The function takes as input the vector of concept prediction and outputs $\boldsymbol{p}_i = [p_{i,1}, \ldots, p_{i,m}] \in [0, 1]^m$. where k in S where:

$$p_{i,k} := \begin{cases} c_{i,k} & \text{if } k \in S \\ q_{i,k} & \text{if } k \notin S \end{cases} \tag{1}$$

In general, we will assume that the subset of concepts that we confirm S should depend on the concept predictions, in which case we write $S(\mathbf{q}_i)$ instead of S.

Since our goal is to reduce the need for human oversight where possible, we assume a fixed confirmation budget $B \in [0, m]$ and limit ourselves to an average of at most B concept confirmations per example. If we set $0 \leq B \leq 1$, then the proportion of instances confirmed is at most B.

Design Objectives Our task is to design these components so that they collectively define a model $h : \mathcal{X} \to \mathcal{Y} \cup \{\perp\}$ to maximize coverage subject to accuracy. Taken together, our end-to-end predictor has the following structure: given features, we infer concepts using g,

1. we use "confirmation", where the probability of a concept's presence is replaced with 0.0 or 1.0 by an oracle, to distinguish this process from "intervention", where a hard or soft concept prediction is replaced with a more accurate value

confirm a subset of the concepts using ψ, predict the label using f, and choose whether to abstain using φ:

$$h : \mathbf{x} \overset{g}{\mapsto} \mathbf{q} \overset{\psi}{\mapsto} \boldsymbol{p} \overset{f}{\mapsto} \overline{y} \overset{\varphi}{\mapsto} \hat{y}$$

We treat the concept detectors and front-end model as given. Thus, our main components for building conceptual safeguards consist of developing techniques for uncertainty propagation and confirmation.

- *Confirmation:* We wish to design methods that can select samples to confirm so that we maximize coverage, maintain accuracy, and adhere to a user-specific confirmation budget that is set to reflect the desired level of human intervention in a system.
- *Propagating Concept Uncertainty:* After concept detection and confirmation, we are left with concept predictions that represent probabilities $p_{i,k} \in [0, 1]$. However, the front-end model requires hard concepts in $\{0, 1\}$ as inputs. We require a method for propagating uncertainty to the front-end model f.

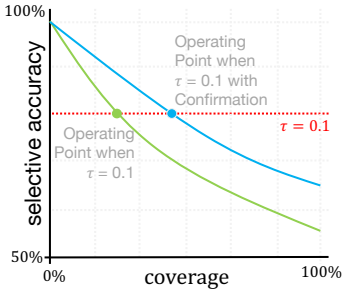

Figure 2: Coverage versus selective accuracy with and without concept confirmation. Coverage increases while selective accuracy decreases. Concept confirmation improves this tradeoff for a given threshold $1 - \alpha$, allowing for greater coverage while maintaining performance at or above the threshold.

3. Methodology

In this section, we describe how to build conceptual safeguards.

3.1 Uncertainty Propagation

We propagate the uncertainty in each concept through the front-end model. We make two key assumptions about the underlying data distribution to motivate our approach.

Assumption 1. *The label y and features $\boldsymbol{x}$ are conditionally independent given the concepts $\boldsymbol{c}$*

Assumption 2. *The concepts $\{c_1, \ldots, c_m\}$ are conditionally independent given the features $\boldsymbol{x}$.*

We note that these assumptions might not be met in practice, and include experiments in settings that violate these assumptions in Section 4. Given these assumptions, we can write the conditional distribution $p(y \mid \boldsymbol{x})$ in terms of quantities the concept bottleneck model

natively estimates:

$$\begin{aligned} p(y \mid \boldsymbol{x}) &= \sum_{\boldsymbol{c} \in \{0,1\}^m} p(y \mid \boldsymbol{c}, \boldsymbol{x})\, p(\boldsymbol{c} \mid \boldsymbol{x}) \\ &= \sum_{\boldsymbol{c} \in \{0,1\}^m} p(y \mid \boldsymbol{c})\, p(\boldsymbol{c} \mid \boldsymbol{x}) && \text{(Assumption 1)} \\ &= \sum_{\boldsymbol{c} \in \{0,1\}^m} p(y \mid \boldsymbol{c}) \prod_{k \in [m]} p(c_k \mid \boldsymbol{x}) && \text{(Assumption 2)} \end{aligned}$$

We can plug in estimates of these quantities to get our estimate for the probability that $y_i = 1$. Given a probabilistic prediction $\boldsymbol{p}_i = (p_{i,1}, \ldots, p_{i,k})$ for the concepts, we estimate

$$f(\boldsymbol{p}_i) := \sum_{\boldsymbol{c} \in \{0,1\}^m} f(\boldsymbol{c}) \prod_{k \in [m]} p_{i,k}^{c_k} (1 - p_{i,k})^{1 - c_k} \tag{2}$$

Computing (2) requires 2^m calls to the front-end model. In practice, this is negligible given that most models are trained with a few concepts. When m is large, and computation is prohibitively expensive, however, we can construct a Monte Carlo estimate using a sample of concept vectors.

Definition 3. Given a label $y \in \{0, 1\}$ and probabilistic prediction $p \in [0, 1]$, we say that p is a *calibrated* prediction for y if $\Pr(y = 1 \mid p = t) = t$ for all thresholds $t \in [0, 1]$.

Proposition 4. *Suppose that p is a calibrated prediction for y. Then a selective classification procedure $\varphi_\tau(p)$ that abstains when p has confidence below $1 - \tau$ achieves accuracy at least $1 - \tau$.*

3.2 Confirmation

We consider a method that greedily selects concepts to confirm based on the expectation of the gain in certainty, where we add concepts to S until we meet a confirmation budget.

One way to select concepts to confirm is to test the sensitivity of model output with respect to a concept. If confirming concept k leads to a large variance in the prediction, then confirming the concept will likely significantly impact the prediction. In our experiments, we use as our gain measure the variance of the result of confirmation, where $\bar{\boldsymbol{c}}_k$ is $\hat{\boldsymbol{c}}$ after confirming concept k:

$$\text{Gain}(\hat{\boldsymbol{c}}, k) := \text{Var}[f(\bar{\boldsymbol{c}}_k)] = \mathbb{E}((f(\bar{\boldsymbol{c}}_k))^2) - (\mathbb{E}(f(\bar{\boldsymbol{c}}_k)))^2 \tag{3}$$

Since we do not know $f(\bar{\boldsymbol{c}}_{S+\{k\}})$ before confirmation, we view $f(\bar{\boldsymbol{c}}_{S+\{k\}})$ as a random variable that takes on values $f(\bar{\boldsymbol{c}}_S[\bar{\boldsymbol{c}}_k = 1])$ with probability $\hat{\boldsymbol{c}}_k$ and $f(\bar{\boldsymbol{c}}_S[\bar{\boldsymbol{c}}_k = 0]$ with probability $(1 - \hat{\boldsymbol{c}}_k)$. Here $\bar{\boldsymbol{c}}_S[\bar{\boldsymbol{c}}_k = 0]$ means we have confirmed $\bar{\boldsymbol{c}}_S$ at index k with 0.

Algorithm 1 Greedy Concept Selection

Input: $\hat{\boldsymbol{C}}$, set of concept predictions that lead to abstention, $h(\hat{\boldsymbol{c}}) = 0$
Input: $B \geq 0$, confirmation budget
Input: $\gamma_1, \ldots, \gamma_m$, costs to confirm each concept
$S_1, \ldots, S_n \leftarrow \{\}$ *concepts to confirm for each instance*
repeat
  $i^*, k^* \leftarrow \arg\max_{i,k} \text{Gain}(\hat{\boldsymbol{c}}_i, k)$ s.t. $k \notin S_i$ *select next best concept that has not been chosen before*
  $S_i \leftarrow S_i \cup \{k\}$
  $B \leftarrow B - \gamma_k$
until $B < 0$ or $S_i = [m]$ for all i
Output: $S_1, \ldots, S_n$, confirmations for each sample

4. Experiments

We evaluate our method on synthetic and real-world datasets. Our goal is to benchmark the performance and coverage of our approach across prediction tasks with ablations to separate the impacts of confirmation and uncertainty propagation. We provide additional details on and results in Appendix A, and the code to reproduce these results in an anonymized repository.

4.1 Setup

We consider six classification datasets with concept labels. The `melanoma` and `skincancer` datasets are image classification tasks to diagnose melanoma and skin cancer that are derived from the Derm7pt dataset [16]. The `warbler` and `flycatcher` datasets are image classification bird identification tasks derived from the CalTech-UCSD Birds dataset [26]. We process each dataset to binarize categorical concepts (e.g., `WingColor` or `VascularStructures`) and (`WingColorRed` or `IrregularVascularStructures`). We construct four systems for each dataset. Each system is an end-to-end selective classification model that allows for concept-based interventions. We build each system using the ame front-end model and concept detectors as described in Section 2. We train a front-end model f and concept detectors $g_1, \ldots, g_m$ for each dataset using logistic regression. We train concept detectors for each dataset – other than `synthetic` – using embeddings from a pre-trained model [i.e., InceptionV3 24]. The four systems include:

- Baseline: An independent concept bottleneck model build using concept detectors $g_1, \ldots, g_m$ and a front-end model f trained on the true concepts. This is a traditional concept bottleneck model without conceptual safeguards.
- CS + ImpactConf: Our proposed system uses uncertainty propagation and the targetted confirmation strategy as described in Section 3.
- Baseline + RandomConf & CS + RandomConf: The Baseline and CS systems paired with a baseline confirmation strategy that randomly selects concepts for confirmation. We report results for these systems as a baseline that we can use to assess the value added of our proposed confirmation strategy.

Evaluation We split each dataset into a training set to train models and their components (80%) and a test set (20%) to evaluate their performance. We construct an accuracy coverage curve for each method on each dataset by plotting the coverage and selective accuracy of the system for abstention thresholds of $\tau \in [0.5, \ldots, 0.95]$. (see Section 2). We evaluate these values when we confirm concepts on a subset of instances on which a model abstains (using the ImpactConf or the RandomConf strategies). We limit the number of confirmations to match a *confirmation budgets*. We then construct accuracy-coverage curves for *confirmation budgets* of 0%/10%/20%/50% which reflect the returns to no/low/medium/high human degrees of human intervention.

4.2 Results

Our results show that in general, our proposed methods CS + RandomConf and CS + ImpactConf outperform baselines Baseline and Baseline + RandomConf with regards to performance and coverage under selective classification. These gains vary across datasets. Given the same performance threshold, we achieve lower coverage on datasets `Melanoma` and `SkinCancer` than on datasets `CUBCommon` and `CUBRare`, likely as a result of the differences in the difficulty and quality of the concepts. In all cases, leveraging uncertainty across concepts improves this trade-off. These gains vary across datasets, where the difficulty of the task and the performance of the concept detectors impact the benefit of conceptual safeguards. Across all tasks, confirming concepts improves performance, with the greatest gains achieved by selecting the concepts to confirm based on their impact on the final prediction (CS + ImpactConf).

On the Gains of Accounting for Uncertainty The results summarized in Table Table 2 emphasize the advantages of incorporating uncertainty into the model, where the extent of these advantages varies based on the specific task and confirmation budget. For instance, when no confirmation is allowed (Confirmation Budget at 0%), the dataset `flycatcher` shows little to no benefit from accounting for uncertainty. In contrast, the `skincancer` dataset reveals a substantial performance gap between a model that includes uncertainty (CS + RandomConf, blue) and one that doesn't (Baseline + RandomConf, green). The impact of incorporating uncertainty also changes when a confirmation budget is introduced. Even though uncertainty made little difference in the `flycatcher` dataset at a 0% confirmation budget, allowing a 20% confirmation budget led to a significant increase in coverage, from 46.2% to 63.5%, when the threshold is set at $\tau = 0.05$ (see Table 3). This is noteworthy because it allows for the automation of predictions on more than an additional 17% of instances without requiring human intervention.

On Learning from Noisy Concepts We evaluate the effects of noisy concepts on model performance, we construct a synthetic dataset with varying levels of noise in the underlying concepts that generate the labels (see Table 6). When the probability of noise is 0%, the concept is fully deterministic and calculated via a parity function of its relevant features. As the probability of noise increases (e.g., 25% or 75%), the concept becomes increasingly stochastic, thus simulating real-world scenarios where underlying features may be subject to noise or uncertainty. Results from the synthetic datasets demonstrate that the benefits of conceptual safeguards increase with increasingly noisy concepts. With low noise $p(noise) = 25\%$, conceptual safeguards provide a real but small benefit to coverage and

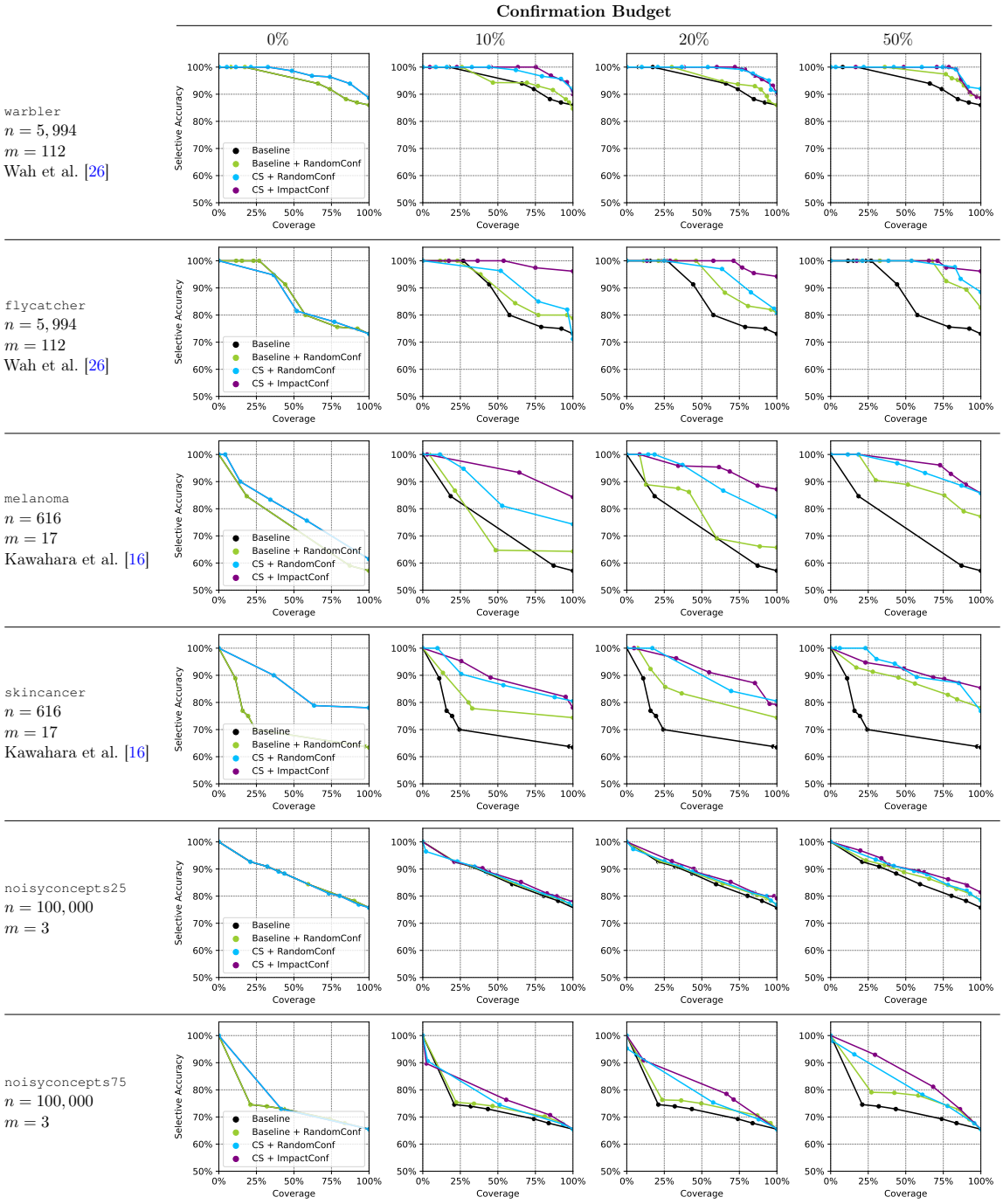

Table 2: Coverage vs. accuracy for all methods on all datasets. We include additional results in Appendix A for multiclass settings. Note that the methods Baseline (black) and Baseline + RandomConf (green) produce identical results when the confirmation budget is set to 0%. Similarly, the methods CS + RandomConf (blue) and CS + ImpactConf (purple) are also equivalent under the same condition.

performance. With high noise $p(noise) = 25\%$, conceptual safeguards provide a significant benefit. See Appendix A.1 for details on synthetic data construction and experiments.

Dataset	Prediction Thresholds	Baseline	Baseline + RandomConf	CS + RandomConf	CS + ImpactConf
`warbler`	$\tau = 0.05$	17.3%	30.0%	94.67%	90.0%
$n = 5,994$	$\tau = 0.1$	74.0%	89.3%	100.00%	100.00%
$m = 112$	$\tau = 0.15$	100.00%	100.00%	100.00%	100.00%
Wah et al. [26]	$\tau = 0.2$	100.00%	100.00%	100.00%	100.00%
`flycatcher`	$\tau = 0.05$	26.9%	46.2%	63.5%	84.62%
$n = 5,994$	$\tau = 0.1$	44.2%	46.2%	63.5%	100.00%
$m = 112$	$\tau = 0.15$	44.2%	65.4%	82.7%	100.00%
Wah et al. [26]	$\tau = 0.2$	57.7%	100.00%	100.00%	100.00%
`melanoma`	$\tau = 0.05$	0.0%	8.6%	37.1%	61.43%
$n = 616$	$\tau = 0.1$	0.0%	8.6%	37.1%	68.57%
$m = 17$	$\tau = 0.15$	0.0%	41.4%	64.3%	100.00%
Kawahara et al. [16]	$\tau = 0.2$	18.6%	41.4%	64.3%	100.00%
`skincancer`	$\tau = 0.05$	0.0%	7.3%	17.1%	32.93%
$n = 616$	$\tau = 0.1$	0.0%	15.9%	17.1%	54.88%
$m = 17$	$\tau = 0.15$	11.0%	25.6%	17.1%	85.37%
Kawahara et al. [16]	$\tau = 0.2$	11.0%	36.6%	100.00%	85.4%
`noisyconcepts25`	$\tau = 0.05$	0.0%	0.0%	4.32%	0.0%
$n = 100,000$	$\tau = 0.1$	32.3%	33.9%	36.3%	44.66%
$m = 3$	$\tau = 0.15$	43.6%	45.8%	54.6%	69.16%
	$\tau = 0.2$	80.4%	86.8%	83.8%	93.31%
`noisyconcepts75`	$\tau = 0.05$	0.0%	0.0%	0.10%	0.0%
$n = 100,000$	$\tau = 0.1$	0.0%	0.0%	6.4%	11.17%
$m = 3$	$\tau = 0.15$	0.0%	0.0%	6.4%	11.17%
	$\tau = 0.2$	0.0%	0.0%	6.4%	11.17%

Table 3: Coverage for varying performance thresholds τ with confirmation budget 20%. We include results for additional datasets and confirmation budgets in Appendix A.2.

On Confirmation Confirming concepts improves performance across all datasets. On dataset `flycatcher`, confirmation with a budget of 20% improves coverage from 26.92% (Baseline) to 46.15% (Baseline + RandomConf) at threshold $\tau = 0.5$, leading to nearly twice as many samples that can be automated. These gains compound with conceptual safeguards, where coverage reaches 63.46% with uncertainty propagation alone, without sacrificing performance or requiring additional human intervention (CS + RandConf). For conceptual safeguards overall, this coverage improves to reach 84.62% (CS + ImpactConf). This additional gain is a result of ImpactConf. Instead of intervening randomly on concepts up to the confirmation budget, ImpactConf considers the variance of the confirmation with respect to the outcome prediction and intervenes first on the highest impact concepts. This weighting allows humans to confirm the concepts with the greatest benefit first, leading to considerable gains in coverage without sacrificing safety. Confirming concepts can also lower automation tasks in applications where the concepts require a lower level of expertise than the label. For example, non-expert users may be able to confirm concepts such as `WingColorRed`, while confirming the final species prediction to `RedFacedCormorant` may require greater expertise.

On Multiclass Prediction Tasks While we limit our theoretical exposition to binary prediction tasks, conceptual safeguards can also be applied in multiclass settings. The main adaption required is in the uncertainty propagation component of the conceptual safeguards. Specifically, rather than propagating uncertainty about the prediction in general, we extend the prediction vector to include probabilities for each of the classes using the possible concepts $\boldsymbol{c} \in \{0,1\}^m$. For the selective classification component φ_τ, we estimate uncertainty based on the likelihood of the most probable class and threshold prediction accordingly. We include results on multiclass datasets in Appendix A.3.

5. Concluding Remarks

In this work, we introduce conceptual safeguards, a versatile and interpretable approach for automating classification tasks. Empirical investigation substantiates the effectiveness of our approach in various applications, establishing it as a promising avenue for safe and interpretable automation. One limitation of our approach is its potential to exacerbate existing performance disparities across diverse groups [15]. This stems from the nature of selective classification, which while improving performance and safety, could inadvertently magnify biases. A secondary limitation is the model's reliance on well-defined concepts to capture all relevant information about the label from the input. If the chosen concepts do not encapsulate all necessary information, the model's performance will necessarily be compromised. These limitations highlight important areas for future research, centered on improving equitable outcomes and refining the utility of concept bottleneck models within our conceptual safeguards framework.

References

[1] Siddhartha Bhattacharyya, Sanjeev Jha, Kurian Tharakunnel, and J Christopher Westland. Data mining for credit card fraud: A comparative study. *Decision support systems*, 50(3): 602–613, 2011.

[2] Nontawat Charoenphakdee, Zhenghang Cui, Yivan Zhang, and Masashi Sugiyama. Classification with rejection based on cost-sensitive classification. In *International Conference on Machine Learning*, pages 1507–1517. PMLR, 2021.

[3] C Chow. On optimum recognition error and reject tradeoff. *IEEE Transactions on information theory*, 16(1):41–46, 1970.

[4] Chi-Keung Chow. An optimum character recognition system using decision functions. *IRE Transactions on Electronic Computers*, pages 247–254, 1957.

[5] Katherine Maeve Collins, Matthew Barker, Mateo Espinosa Zarlenga, Naveen Raman, Umang Bhatt, Mateja Jamnik, Ilia Sucholutsky, Adrian Weller, and Krishnamurthy Dvijotham. Human uncertainty in concept-based ai systems. In *Proceedings of the 2023 AAAI/ACM Conference on AI, Ethics, and Society*, pages 869–889, 2023.

[6] Ran El-Yaniv et al. On the foundations of noise-free selective classification. *Journal of Machine Learning Research*, 11(5), 2010.

[7] Andre Esteva, Brett Kuprel, Roberto A Novoa, Justin Ko, Susan M Swetter, Helen M Blau, and Sebastian Thrun. Dermatologist-level classification of skin cancer with deep neural networks. *nature*, 542(7639):115–118, 2017.

[8] Jean Feng, Arjun Sondhi, Jessica Perry, and Noah Simon. Selective prediction-set models with coverage rate guarantees. *Biometrics*, 79(2):811–825, 2023.

[9] Adam Fisch, Tommi Jaakkola, and Regina Barzilay. Calibrated selective classification. *arXiv preprint arXiv:2208.12084*, 2022.

[10] Vaclav Voracek Vojtech Franc, Daniel Prusa, and Vaclav Voracek. Optimal strategies for reject option classifiers. *Journal of Machine Learning Research*, 24(11):1–49, 2023.

[11] Giorgio Fumera and Fabio Roli. Reject option with multiple thresholds. *Pattern recognition*, 33 (12):2099–2101, 2000.

[12] Yonatan Geifman and Ran El-Yaniv. Selective classification for deep neural networks. *Advances in neural information processing systems*, 30, 2017.

[13] Seung Seog Han, Myoung Shin Kim, Woohyung Lim, Gyeong Hun Park, Ilwoo Park, and Sung Eun Chang. Classification of the clinical images for benign and malignant cutaneous tumors using a deep learning algorithm. *Journal of Investigative Dermatology*, 138(7):1529–1538, 2018.

[14] Marton Havasi, Sonali Parbhoo, and Finale Doshi-Velez. Addressing leakage in concept bottleneck models. *Advances in Neural Information Processing Systems*, 35:23386–23397, 2022.

[15] Erik Jones, Shiori Sagawa, Pang Wei Koh, Ananya Kumar, and Percy Liang. Selective classification can magnify disparities across groups. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021.

[16] Jeremy Kawahara, Sara Daneshvar, Giuseppe Argenziano, and Ghassan Hamarneh. Seven-point checklist and skin lesion classification using multitask multimodal neural nets. *IEEE journal of biomedical and health informatics*, 23(2):538–546, 2018.

[17] Pang Wei Koh, Thao Nguyen, Yew Siang Tang, Stephen Mussmann, Emma Pierson, Been Kim, and Percy Liang. Concept bottleneck models. In *International Conference on Machine Learning*, pages 5338–5348. PMLR, 2020.

[18] Anita Mahinpei, Justin Clark, Isaac Lage, Finale Doshi-Velez, and Weiwei Pan. Promises and pitfalls of black-box concept learning models. *arXiv preprint arXiv:2106.13314*, 2021.

[19] Andrei Margeloiu, Matthew Ashman, Umang Bhatt, Yanzhi Chen, Mateja Jamnik, and Adrian Weller. Do concept bottleneck models learn as intended?, 2021. URL https://arxiv.org/abs/2105.04289.

[20] Andrei Margeloiu, Matthew Ashman, Umang Bhatt, Yanzhi Chen, Mateja Jamnik, and Adrian Weller. Do concept bottleneck models learn as intended? *arXiv preprint arXiv:2105.04289*, 2021.

[21] Mohammad Sadegh Norouzzadeh, Anh Nguyen, Margaret Kosmala, Alexandra Swanson, Meredith S Palmer, Craig Packer, and Jeff Clune. Automatically identifying, counting, and describing wild animals in camera-trap images with deep learning. *Proceedings of the National Academy of Sciences*, 115(25):E5716–E5725, 2018.

[22] Tuomas Oikarinen, Subhro Das, Lam M Nguyen, and Tsui-Wei Weng. Label-free concept bottleneck models. *arXiv preprint arXiv:2304.06129*, 2023.

[23] 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.

[24] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016.

[25] Michael A Tabak, Mohammad S Norouzzadeh, David W Wolfson, Steven J Sweeney, Kurt C VerCauteren, Nathan P Snow, Joseph M Halseth, Paul A Di Salvo, Jesse S Lewis, Michael D White, et al. Machine learning to classify animal species in camera trap images: Applications in ecology. *Methods in Ecology and Evolution*, 10(4):585–590, 2019.

[26] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset. Technical report, California Institute of Technology, 2011.

[27] Chih-Kuan Yeh, Been Kim, Sercan Arik, Chun-Liang Li, Tomas Pfister, and Pradeep Ravikumar. On completeness-aware concept-based explanations in deep neural networks. *Advances in neural information processing systems*, 33:20554–20565, 2020.

[28] Mert Yuksekgonul, Maggie Wang, and James Zou. Post-hoc concept bottleneck models. *arXiv preprint arXiv:2205.15480*, 2022.

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

Appendix A. Supporting Experimental Information

A.1 Datasets

Dataset	Reference	Outcome	n	m	$\lvert y_i = 1 \rvert$
`melanoma`	Kawahara et al. [16]	lesion is melanoma	616	17	151
`skincancer`	Kawahara et al. [16]	lesion is cancerous	616	17	177
`cubcommon`	Wah et al. [26]	bird in 10 most prevalent classes	5,994	112	2,907
`cubrare`	Wah et al. [26]	bird in 60 least prevalent classes	5,994	112	3,057
`warbler`	Wah et al. [26]	bird is type of warbler	5,994	112	2,907
`flycatcher`	Wah et al. [26]	bird is type of flycatcher	5,994	112	3,057
`cubspecies`	Wah et al. [26]	bird species	5,994	112	–
`cubtypes`	Wah et al. [26]	bird type	5,994	112	–
`noisy`	–	logistic function on $\mathcal{C}$	100,000	3	65,663

Table 4: Overview of datasets used in Section 4. Each dataset is publically available and de-identified where relevant.

`melanoma` & `skincancer` These datasets are derived from the Derm7pt dataset [16], which is de-identified and publicly available without patient information. We preprocess the dataset by splitting the original seven categorical image annotations (`pigment_network`, `streaks`, `pigmentation`, `regression_structures`, `dots_and_globules`, `blue_whitish_veil`, `vascular_structures`) into seventeen binary concepts. We consider two tasks: predicting `melanoma` and predicting `skincancer` (melanoma or basal cell carcinoma). We split the validation indices in the original dataset into a validation set and a hold-out test set for evaluation. We then balance the resulting classes by downsampling the majority class. To train the concept models, we augment the original training images with random color enhancements and random flipping to obtain 10x total training images.

`cubcommon`, `cubrare`, `warbler`, `flycatcher`, `cubspecies` & `cubtypes` These datasets are derived from the CUB 2011 dataset [26]. We follow the same preprocessing described in [17]. `cubcommon` predicts if a bird is "common" or in the top 10 largest groups. `cubrare` predicts if a bird is "rare" or in the bottom 60 groups. `warbler` classifies birds of type warbler and `flycatcher` classifies birds of type flycatcher. `cubspecies` and `cubtypes` are multiclass datasets for predicting bird species and bird types, respectively. To train the concept models, we augment the original training images with random color enhancements and random flipping to obtain 10x total training images.

`noisy` The `noisy` datasets are synthetic datasets that we primarily use to evaluate how their performance changes with respect to the quality of concept detectors. We sample these the examples

in these datasets $(\boldsymbol{x}_i, \boldsymbol{c}_i, y_i)$ from the following distribution:

$$\begin{aligned} x_1, \ldots, x_5 &= \text{Bernouilli}(0.7) \\ \xi_1, \xi_2, \xi_3 &= \text{Bernouilli}(p_\xi) \\ c_1 &= \text{parity}(x_1, x_2, x_4) \oplus \xi_1 \\ c_2 &= \text{parity}(x_1, x_2, x_3) \oplus \xi_2 \\ c_3 &= \text{parity}(x_1, x_2, x_5) \oplus \xi_3 \\ p &= \text{logistic}(1.0c_1 + 2.0c_2 + 3.0c_3 - 2.0) \\ y &\sim \text{Bernoulli}(p) \end{aligned} \tag{4}$$

The distribution shown in (4) includes an explicit noise parameter $p_\xi \in [0, 1]$ that we can set to control the noise in concept labels. When $p_\xi = 0$, the values of c_1, c_2, c_3 operate as parity functions, which can only be learned through a sufficiently complex model. When $p_\xi > 0$, we inject noise into the concept labels by randomly flipping the values of c_1, c_2, c_3 with probability p_ξ. Thus, the noise parameter sets an upper bound on the accuracy of concept labels – and larger values of p_ξ lead to less accurate concept models. In contrast to the real-world datasets, we train the concept detectors for the `noisy` datasets directly (i.e., without an embedding layer) by fitting multi-layer perceptron with a single hidden layer.

A.2 Additional Results

Dataset	Prediction Thresholds	Baseline	Baseline + RandomConf	CS + RandomConf	CS + ImpactConf
warbler	$\tau = 0.05$	17.3%	26.0%	92.00%	85.3%
$n = 5,994$	$\tau = 0.1$	74.0%	86.7%	100.00%	100.00%
$m = 112$	$\tau = 0.15$	100.00%	97.3%	100.00%	100.00%
Wah et al. [26]	$\tau = 0.2$	100.00%	100.00%	100.00%	100.00%
flycatcher	$\tau = 0.05$	26.9%	38.5%	51.9%	100.00%
$n = 5,994$	$\tau = 0.1$	44.2%	38.5%	51.9%	100.00%
$m = 112$	$\tau = 0.15$	44.2%	38.5%	76.9%	100.00%
Wah et al. [26]	$\tau = 0.2$	57.7%	96.2%	96.2%	100.00%
cubcommon	$\tau = 0.05$	2.2%	3.2%	10.2%	21.70%
$n = 5,994$	$\tau = 0.1$	12.5%	15.7%	10.2%	52.09%
$m = 112$	$\tau = 0.15$	22.4%	26.4%	39.1%	71.62%
Wah et al. [26]	$\tau = 0.2$	22.4%	26.4%	63.1%	91.82%
cubrare	$\tau = 0.05$	0.0%	0.0%	2.8%	32.72%
$n = 5,994$	$\tau = 0.1$	11.4%	13.7%	18.5%	32.72%
$m = 112$	$\tau = 0.15$	11.4%	24.4%	18.5%	68.61%
Wah et al. [26]	$\tau = 0.2$	31.9%	39.6%	35.6%	91.99%
melanoma	$\tau = 0.05$	0.0%	4.3%	11.43%	2.9%
$n = 616$	$\tau = 0.1$	0.0%	4.3%	27.1%	64.29%
$m = 17$	$\tau = 0.15$	0.0%	21.4%	27.1%	64.29%
Kawahara et al. [16]	$\tau = 0.2$	18.6%	21.4%	52.9%	100.00%
skincancer	$\tau = 0.05$	0.0%	0.0%	9.8%	25.61%
$n = 616$	$\tau = 0.1$	0.0%	13.4%	25.61%	25.61%
$m = 17$	$\tau = 0.15$	11.0%	13.4%	53.66%	45.1%
Kawahara et al. [16]	$\tau = 0.2$	11.0%	30.5%	100.00%	95.1%
noisyconcepts25	$\tau = 0.05$	0.0%	0.0%	2.12%	0.0%
$n = 100,000$	$\tau = 0.1$	32.3%	33.2%	34.5%	39.77%
$m = 3$	$\tau = 0.15$	43.6%	44.8%	49.2%	65.35%
	$\tau = 0.2$	80.4%	84.00%	80.2%	82.7%
noisyconcepts75	$\tau = 0.05$	0.0%	0.0%	0.0%	0.0%
$n = 100,000$	$\tau = 0.1$	0.0%	0.0%	3.24%	0.0%
$m = 3$	$\tau = 0.15$	0.0%	0.0%	3.24%	2.4%
	$\tau = 0.2$	0.0%	0.0%	3.24%	2.4%

Table 5: Coverage for varying performance thresholds τ with confirmation budget 10%

Dataset	Prediction Thresholds	Baseline	Baseline + RandomConf	CS + RandomConf	CS + ImpactConf
warbler	$\tau = 0.05$	17.3%	30.0%	94.67%	90.0%
$n = 5,994$	$\tau = 0.1$	74.0%	89.3%	100.00%	100.00%
$m = 112$	$\tau = 0.15$	100.00%	100.00%	100.00%	100.00%
Wah et al. [26]	$\tau = 0.2$	100.00%	100.00%	100.00%	100.00%
flycatcher	$\tau = 0.05$	26.9%	46.2%	63.5%	84.62%
$n = 5,994$	$\tau = 0.1$	44.2%	46.2%	63.5%	100.00%
$m = 112$	$\tau = 0.15$	44.2%	65.4%	82.7%	100.00%
Wah et al. [26]	$\tau = 0.2$	57.7%	100.00%	100.00%	100.00%
cubcommon	$\tau = 0.05$	2.2%	0.0%	11.7%	35.23%
$n = 5,994$	$\tau = 0.1$	12.5%	16.0%	22.7%	52.09%
$m = 112$	$\tau = 0.15$	22.4%	29.4%	42.2%	100.00%
Wah et al. [26]	$\tau = 0.2$	22.4%	29.4%	70.1%	100.00%
cubrare	$\tau = 0.05$	0.0%	0.0%	8.5%	33.56%
$n = 5,994$	$\tau = 0.1$	11.4%	15.0%	19.5%	51.09%
$m = 112$	$\tau = 0.15$	11.4%	15.0%	19.5%	70.12%
Wah et al. [26]	$\tau = 0.2$	31.9%	43.1%	65.9%	100.00%
melanoma	$\tau = 0.05$	0.0%	8.6%	37.1%	61.43%
$n = 616$	$\tau = 0.1$	0.0%	8.6%	37.1%	68.57%
$m = 17$	$\tau = 0.15$	0.0%	41.4%	64.3%	100.00%
Kawahara et al. [16]	$\tau = 0.2$	18.6%	41.4%	64.3%	100.00%
skincancer	$\tau = 0.05$	0.0%	7.3%	17.1%	32.93%
$n = 616$	$\tau = 0.1$	0.0%	15.9%	17.1%	54.88%
$m = 17$	$\tau = 0.15$	11.0%	25.6%	17.1%	85.37%
Kawahara et al. [16]	$\tau = 0.2$	11.0%	36.6%	100.00%	85.4%
noisyconcepts25	$\tau = 0.05$	0.0%	0.0%	4.32%	0.0%
$n = 100,000$	$\tau = 0.1$	32.3%	33.9%	36.3%	44.66%
$m = 3$	$\tau = 0.15$	43.6%	45.8%	54.6%	69.16%
	$\tau = 0.2$	80.4%	86.8%	83.8%	93.31%
noisyconcepts75	$\tau = 0.05$	0.0%	0.0%	0.10%	0.0%
$n = 100,000$	$\tau = 0.1$	0.0%	0.0%	6.4%	11.17%
$m = 3$	$\tau = 0.15$	0.0%	0.0%	6.4%	11.17%
	$\tau = 0.2$	0.0%	0.0%	6.4%	11.17%

Table 6: Coverage for varying performance thresholds τ with confirmation budget 20%

Dataset	Prediction Thresholds	Baseline	Baseline + RandomConf	CS + RandomConf	CS + ImpactConf
warbler	$\tau = 0.05$	17.3%	84.7%	83.3%	86.67%
$n = 5,994$	$\tau = 0.1$	74.0%	93.3%	100.00%	92.7%
$m = 112$	$\tau = 0.15$	100.00%	100.00%	100.00%	100.00%
Wah et al. [26]	$\tau = 0.2$	100.00%	100.00%	100.00%	100.00%
flycatcher	$\tau = 0.05$	26.9%	67.3%	82.7%	100.00%
$n = 5,994$	$\tau = 0.1$	44.2%	76.9%	86.5%	100.00%
$m = 112$	$\tau = 0.15$	44.2%	90.4%	100.00%	100.00%
Wah et al. [26]	$\tau = 0.2$	57.7%	100.00%	100.00%	100.00%
cubcommon	$\tau = 0.05$	2.2%	10.7%	17.0%	100.00%
$n = 5,994$	$\tau = 0.1$	12.5%	19.7%	51.6%	100.00%
$m = 112$	$\tau = 0.15$	22.4%	69.8%	81.0%	100.00%
Wah et al. [26]	$\tau = 0.2$	22.4%	80.8%	81.0%	100.00%
cubrare	$\tau = 0.05$	0.0%	3.0%	14.5%	100.00%
$n = 5,994$	$\tau = 0.1$	11.4%	18.5%	28.4%	100.00%
$m = 112$	$\tau = 0.15$	11.4%	67.4%	77.5%	100.00%
Wah et al. [26]	$\tau = 0.2$	31.9%	67.4%	77.5%	100.00%
melanoma	$\tau = 0.05$	0.0%	18.6%	44.3%	72.86%
$n = 616$	$\tau = 0.1$	0.0%	30.0%	62.9%	80.00%
$m = 17$	$\tau = 0.15$	0.0%	51.4%	100.00%	100.00%
Kawahara et al. [16]	$\tau = 0.2$	18.6%	75.7%	100.00%	100.00%
skincancer	$\tau = 0.05$	0.0%	0.0%	30.49%	0.0%
$n = 616$	$\tau = 0.1$	0.0%	28.0%	42.7%	48.78%
$m = 17$	$\tau = 0.15$	11.0%	56.1%	85.4%	100.00%
Kawahara et al. [16]	$\tau = 0.2$	11.0%	84.1%	85.4%	100.00%
noisyconcepts25	$\tau = 0.05$	0.0%	0.0%	11.0%	19.75%
$n = 100,000$	$\tau = 0.1$	32.3%	36.0%	41.99%	38.7%
$m = 3$	$\tau = 0.15$	43.6%	65.6%	64.1%	78.19%
	$\tau = 0.2$	80.4%	91.6%	92.8%	100.00%
noisyconcepts75	$\tau = 0.05$	0.0%	0.0%	1.67%	0.0%
$n = 100,000$	$\tau = 0.1$	0.0%	0.0%	15.8%	29.55%
$m = 3$	$\tau = 0.15$	0.0%	0.0%	15.8%	29.55%
	$\tau = 0.2$	0.0%	0.0%	15.8%	68.25%

Table 7: Coverage for varying performance thresholds τ with confirmation budget 50%

A.3 Multiclass Experiments

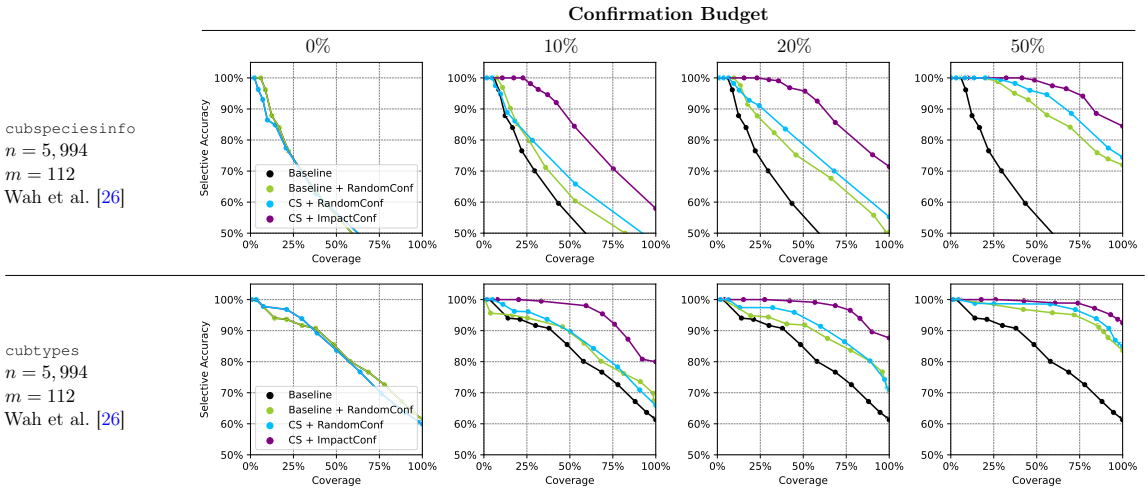

Table 8: Coverage vs. accuracy for all methods on multiclass datasets.

Dataset	Prediction Thresholds	Baseline	Baseline + RandomConf	CS + RandomConf	CS + ImpactConf
cubtypes	$\tau = 0.05$	3.3%	16.7%	25.7%	68.95%
$n = 5,994$	$\tau = 0.1$	37.9%	45.7%	36.7%	75.96%
$m = 112$	$\tau = 0.15$	48.4%	58.3%	50.3%	83.81%
Wah et al. [26]	$\tau = 0.2$	57.9%	68.1%	63.8%	92.15%

Table 9: Coverage for varying performance thresholds τ with confirmation budget 10%

Dataset	Prediction Thresholds	Baseline	Baseline + RandomConf	CS + RandomConf	CS + ImpactConf
cubspeciesinfo	$\tau = 0.05$	8.7%	13.4%	12.7%	50.92%
$n = 5,994$	$\tau = 0.1$	8.7%	17.7%	24.4%	58.10%
$m = 112$	$\tau = 0.15$	12.4%	23.2%	24.4%	68.61%
Wah et al. [26]	$\tau = 0.2$	16.7%	33.1%	39.6%	68.61%
cubtypes	$\tau = 0.05$	3.3%	3.7%	44.7%	77.30%
$n = 5,994$	$\tau = 0.1$	37.9%	50.9%	60.1%	82.97%
$m = 112$	$\tau = 0.15$	48.4%	64.1%	74.0%	100.00%
Wah et al. [26]	$\tau = 0.2$	57.9%	88.6%	88.8%	100.00%

Table 10: Coverage for varying performance thresholds τ with confirmation budget 20%

Dataset	Prediction Thresholds	Baseline	Baseline + RandomConf	CS + RandomConf	CS + ImpactConf
`cubspeciesinfo`	$\tau = 0.05$	8.7%	37.1%	46.1%	67.11%
$n = 5,994$	$\tau = 0.1$	8.7%	45.1%	55.9%	76.79%
$m = 112$	$\tau = 0.15$	12.4%	55.9%	70.1%	84.64%
Wah et al. [26]	$\tau = 0.2$	16.7%	69.4%	70.1%	100.00%
`cubtypes`	$\tau = 0.05$	3.3%	71.8%	72.5%	92.99%
$n = 5,994$	$\tau = 0.1$	37.9%	86.1%	92.0%	100.00%
$m = 112$	$\tau = 0.15$	48.4%	91.5%	99.8%	100.00%
Wah et al. [26]	$\tau = 0.2$	57.9%	100.00%	100.00%	100.00%

Table 11: Coverage for varying performance thresholds τ with confirmation budget 50%