

Selective Prediction Under Distribution Shift: The Failure of Threshold Transfer

Shayam Ahmad

2301020102@cgu-odisha.ac.in

C.V. Raman Global University

Research Article

Keywords: selective prediction, distribution shift, uncertainty calibration, risk control, abstention, reject option

Posted Date: October 29th, 2026

DOI: <https://doi.org/10.21203/rs.3.rs-11212470/v1>

License:   This work is licensed under a Creative Commons Attribution 4.0 International License.

[Read Full License](#)

Additional Declarations: No competing interests reported.

Selective Prediction Under Distribution Shift: The Failure of Threshold Transfer

Abstract

A model can be allowed to say “I do not know” instead of giving an answer it may get wrong. This is called selective prediction. To make it work, someone picks a cut off value. If the model is less sure than that, it stays quiet. The cut off is picked on clean test data, so that the error rate among the answers it does give is at most some target, say 5%. This study shows the cut off does not survive when the input data changes. On CIFAR-10-C a cut off picked for 5% error gives 42% to 45% error at the worst corruption level. It gets worse when the target is stricter. At a 1% target the error is 19 times the promise. On CIFAR-10.1, a real new test set that only costs about 6 points of accuracy, the same cut off still doubles the promised error and stays quiet on nothing at all. The study then splits the system into two parts. One part sorts the inputs from most sure to least sure. The other part picks where to stop. Four different ways of sorting give almost the same result, within 1%. A perfect cut off reaches the target at every corruption level. So the sorting is fine. The stopping is broken. It lets in up to 77 extra points of data. Two published methods that guess accuracy without labels were tried as stopping rules. Both help, neither fixes it, and both lose to a very simple rule. A method with a proper statistical guarantee does fix it, but needs about 800 labelled examples from the new data under mild change and more than 4000 under strong change. The results hold for two model types, for four corruption types never used while building the study, and for real world change. Two versions of the guarantee method were built and thrown away, and both are reported with the reason.

Code Availability: The Jupyter notebooks implementing the experimental framework for selective prediction under distribution shift are publicly available at: https://github.com/shayamahmad/threshold-distribution_shift

Keywords: selective prediction, distribution shift, uncertainty calibration, risk control, abstention, reject option

1 Introduction

A model is trained on some data. It is then tested on data that looks like the training data. The accuracy number comes from that test. But the real world does not stay the same. Pictures get blurry. Cameras change. Light changes. Time passes. The input

data slowly stops looking like the training data. This is called distribution shift [28]. When it happens, the accuracy number no longer describes what the model is doing.

One common fix is to let the model stay quiet. Chow [1] studied this long ago and worked out the best trade off between how often a model answers and how often it is wrong. The idea was later brought into deep learning [2]. A recent survey in *Machine Learning* covers the whole area [3]. The method is simple. The model gives a score that says how sure it is. If the score is above a cut off, the answer is used. If not, the model stays quiet. The cut off is chosen so that the error rate among the given answers meets a target, for example 5%.

There is a problem hiding in this. To pick the cut off you need labels. The only labelled data you have is the validation set, and that came from the training data. The real data the model will see has no labels. So the cut off is chosen on one kind of data and used on another. Nothing says it will still work.

Other people have shown that a model's sense of how sure it is gets worse under shift. Ovadia et al. [4] tested many methods as shift got stronger and found this clearly. Large surveys say the same [5, 6]. Cattelan and Silva [7] showed that fixing the score after training does not behave the way people assumed. But none of this work says *which part* of the system breaks.

There are only two parts. Part one sorts the inputs, from the ones the model is most sure about to the ones it is least sure about. Part two picks where to draw the line. Almost all recent work tries to make part one better. Part two is usually treated as a small detail.

This study measures both parts on their own. The answer is one sided. Part one is already close to as good as it can get, and four different scores give nearly the same sorting. Part two is where the promise is lost. It gets worse as shift gets stronger. It also gets worse when the target is stricter, which is the opposite of what anyone would expect. The study then asks if the promise can be brought back, and what that costs.

Research gap

The gap is easy to state. Past work says the model's sense of sureness gets worse under shift, and that better scores help. It does not say which part of a working system loses the promised error rate, by how much, or what it costs to get it back using labels from the new data. This study measures all three.

Contributions

This work is a careful measurement, not a new algorithm. The claims are kept small on purpose.

1. A measurement of how badly the promise is broken. The broken factor gets worse as the target gets stricter: 4.4 at a 10% target, 8.4 at 5%, 14.4 at 2% and 19.1 at 1%.
2. Proof that the target was possible. A perfect cut off meets it at every corruption level, so the target was never impossible. The cut off was just in the wrong place.
3. A simple measure called coverage error, which separates the stopping part from the sorting part. Four scores are shown to sort almost the same way.

4. A negative result. Two published methods for guessing accuracy without labels do not fix the problem, and a one line rule beats both.
5. A method with a real guarantee, plus how many labels it needs, how wasteful it is, and two versions that failed.

Section 2 sets up the problem. Section 3 covers past work. Section 4 says exactly what was done. Section 5 gives results. Section 6 explains them. Section 7 lists what could be wrong. Section 8 lists limits and Section 9 ends.

2 Problem formulation

2.1 Selective classification

Let $\mathcal{X}$ be the set of possible inputs and $\mathcal{Y} = \{1, \dots, C\}$ the set of labels. A classifier $f : \mathcal{X} \rightarrow \Delta^{C-1}$ turns an input into a list of probabilities, one per class. Its answer is $\hat{y}(x) = \arg \max_c f_c(x)$, the class with the highest probability. A score $g : \mathcal{X} \rightarrow \mathbb{R}$ says how sure the model is. Bigger means more sure.

A selective classifier is f and g together with a cut off t . It works like this:

$$(f, g)_t(x) = \begin{cases} \hat{y}(x) & \text{if } g(x) \geq t, \\ \text{stay quiet} & \text{otherwise.} \end{cases} \quad (1)$$

For a test set $\{(x_i, y_i)\}_{i=1}^n$, two numbers matter. Coverage is the share of inputs the model answers. Selective risk is the error rate among only those answers:

$$\text{cov}(t) = \frac{1}{n} \sum_{i=1}^n \mathbf{1}[g(x_i) \geq t], \quad (2)$$

$$R(t) = \frac{\sum_{i=1}^n \mathbf{1}[g(x_i) \geq t] \mathbf{1}[\hat{y}(x_i) \neq y_i]}{\sum_{i=1}^n \mathbf{1}[g(x_i) \geq t]}. \quad (3)$$

Raise the cut off and coverage always goes down. Risk usually goes down too, but on a finite sample it can wobble, so the method in Section 4.6 does not assume it is smooth.

2.2 The operator's guarantee

Someone running the system picks a target error rate α . They want $R(t) \leq \alpha$. The cut off is found by solving

$$\hat{t}_{\text{clean}} = \min\{t : R_{\text{val}}(t) \leq \alpha\}, \quad (4)$$

where R_{val} is measured on held out validation data from the training distribution P . Taking the smallest t that works means the model answers as often as possible, which is what anyone would want.

The system is then used on data from some other distribution Q . The number that matters is $R_Q(\hat{t}_{\text{clean}})$. Eq. 4 says nothing about it. This paper measures it.

2.3 Diagnostics

Three numbers are used everywhere in this paper. The *risk ratio* is $R_Q(\hat{t})/\alpha$. It says how many times worse the real error is than the promise. The *perfect cut off*, or oracle, is

$$t_Q^* = \min\{t : R_Q(t) \leq \alpha\}, \quad (5)$$

found using the true labels of the new data. Nobody has those labels in real life. It answers a different question: was the target even possible? *Coverage error* is

$$\Delta_{\text{cov}}(\hat{t}) = \text{cov}_Q(\hat{t}) - \text{cov}_Q(t_Q^*), \quad (6)$$

and it is positive when the system answered more often than it should have.

The split between these three is the key idea of the paper. Risk ratio shows the symptom. Coverage error shows the cause, because every method here sorts the same way and only stops in a different place. The area under the risk coverage curve, written AURC, measures how good the sorting is on its own:

$$\text{AURC} = \frac{1}{n} \sum_{k=1}^n R(g_{(k)}), \quad (7)$$

where $g_{(k)}$ is the k th largest score. If two methods use the same score, they have the same AURC and sit on the same curve. Then the only thing left that can differ is Δ_{cov} .

3 Related work

3.1 Letting a model stay quiet

The idea goes back to Chow [1], who worked out the best trade off when the true probabilities are known. It was later used with margin based classifiers [8] and given a proper theory for the noise free case [29]. Geifman and El-Yaniv [2] brought it to deep networks and gave a bound on a held out set. The survey by Hendrickx et al. [3] in *Machine Learning* sorts this work into groups and points out that people test it in inconsistent ways. Others have used the same idea to reduce unfair treatment of groups [9], which links to Section 5.9.

All of this work assumes one thing. The data used to pick the cut off and the data seen later come from the same place. Chow needs the true probabilities. Geifman and El-Yaniv's bound holds on held out data from the same source. Neither is wrong. Neither covers the case studied here.

3.2 Calibration and where it stops helping

Deep networks are usually too sure of themselves. Temperature scaling fixes much of this by fitting one number [10]. Older ways include Platt scaling [11] and isotonic regression [12]. Calibration error is usually measured with bins [13].

Ovadia et al. [4] tested these under growing shift and found calibration gets worse, with ensembles holding up best [15]. The models here show something a bit more

exact. Temperature scaling helps at *every* corruption level, cutting calibration error by about 76% at level 0 and still by about 25% at level 5. What fails is the size of what is left. After scaling, the error grows from 0.007 to 0.223, about thirty times bigger. So calibration helps everywhere and is enough nowhere. This matters because it means bad calibration is not the whole story, which is why the stopping rule is studied on its own.

Wider surveys of uncertainty in deep learning [5] and of the two kinds of uncertainty [6], both from Springer, give the background. Ensembles and Bayesian approximations are the usual alternatives to a single confidence score [15, 30]. Neither looks at whether a cut off moves across distributions.

3.3 Distribution shift and how it is tested

Shift is usually studied with benchmarks. CIFAR-10-C and ImageNet-C add corruptions at set strengths [16], so shift becomes a dial instead of a yes or no. CIFAR-10.1 [17] does something different. A brand new test set was collected from the same source as the original, on purpose, to make the two as alike as possible, and models still lose several points. WILDS [18] collects shift that comes from real changes of hospital, camera or population. The gap between fake and real shift matters here, because a result that only shows up under fake noise is not very useful. Section 5.7 therefore repeats the main measurement on CIFAR-10.1.

3.4 Guessing accuracy without labels

If accuracy on the new data could be guessed without labels, the cut off could be fixed. Two methods matter. Average Thresholded Confidence [19] learns a score cut off on validation data so that the share of validation points below it equals the validation error, then reuses that cut off on new unlabelled data. Difference of Confidences [20] guesses the accuracy drop from the drop in average confidence.

Both were built and tested for one job: guessing accuracy over a *whole* unlabelled set. Picking a cut off is a different job. The set whose error must be guessed is the set of accepted points, and that set was chosen using the score itself. Nobody has reported whether these methods still work in that case. Section 5.6 tests it. They do not.

3.5 Guarantees without assumptions

Conformal prediction gives guarantees on finite samples without assuming a distribution [21, 31]. Learn then Test [22] extends this to controlling risk, by treating the choice of setting as a set of hypothesis tests. The method in Section 4.6 is one case of that idea, using an exact binomial bound [23] and a Bonferroni correction over a fixed list of candidate cut offs. The new part here is not the method. It is the measurement of how many labels it costs at different strengths of shift.

3.6 Where this study sits

Past work shows three things. Sureness gets worse under shift. Better scores help. Given labels, risk can be controlled. It does not show which part of a working system

loses the promise, or what getting it back costs. This study measures both, and reports a negative result for the no labels route.

4 Methods

4.1 Data

CIFAR-10 [24] has 50,000 training images and 10,000 test images, ten classes, all the same size, at 32×32 pixels. The training set is cut once into 45,000 for training and 5,000 for validation, using a fixed shuffle with seed 12345. Every model and every experiment reuses the exact same cut. This was checked by code, by comparing validation labels across runs, and any later step was blocked if the check failed. It matters because Eq. 4 fits the cut off on the validation half. A different split would quietly ruin every comparison.

Three kinds of shift are used.

CIFAR-10-C [16] adds corruptions at five strengths. One detail is worth writing down because it trips people up. Each corruption file holds the same 10,000 test images five times over, one block per strength, so strength s sits at positions $[(s-1) \cdot 10000, s \cdot 10000)$. If the file is loaded whole and used as one test set, all five strengths get averaged and the trend vanishes. Fifteen standard corruptions are used while building the study: gaussian noise, shot noise, impulse noise, defocus blur, glass blur, motion blur, zoom blur, snow, frost, fog, brightness, contrast, elastic transform, pixelate and jpeg compression.

Held back corruptions. Four more, speckle noise, gaussian blur, spatter and saturate, were kept out of every choice made in this study. No cut off, setting, measure or method was picked while looking at them. They are used once, in Section 5.6, to check the result is not just an effect of looking at the same fifteen too many times.

CIFAR-10.1 version 6 [17] has 2,000 new images, 200 per class, gathered from the same source as CIFAR-10 years later, with a process made to copy the original distribution as closely as possible. It stands in for real world shift.

4.2 Models and training

Two model types are used, to check the effect is not about one design. The first is ResNet18 [25] changed for small images: the first layer is 3×3 with stride 1 and the pooling layer is dropped, which is normal for CIFAR and avoids throwing away detail straight away. The second is WideResNet-16-4 [26], 2.75 million parameters, shallower and wider.

Each is trained three times with different random seeds. Training uses stochastic gradient descent with Nesterov momentum 0.9, weight decay 5×10^{-4} , batch size 128 and a one cycle learning rate schedule peaking at 0.1 over 60 epochs [27]. Images are randomly cropped with padding 4 and randomly flipped left to right. Both models use batch normalisation. Training runs in mixed precision. Final training loss drops to between 0.002 and 0.014, so the models have learned the training set almost perfectly. Section 7 comes back to this, because models trained this far are known to be too sure of themselves.

Every result uses the three seed ensemble, made by averaging the probability outputs. One temperature is fitted per model type by minimising negative log likelihood of the ensemble on the clean validation set. The values are 1.242 for ResNet18 and 1.219 for WideResNet-16-4. Both are above one, which confirms the models were too sure before the fix. Clean test accuracy after fitting is 95.3% and 95.0%.

The raw outputs for every model, corruption and strength are computed once and saved as half precision numbers. Everything after that reads the saved file and never runs the models again. The whole file is about 100 megabytes. This is not just for speed. It means every experiment in Section 5 sees exactly the same model outputs, so any difference between two results comes from the method and never from a model being run again.

4.3 Metrics

Coverage, selective risk, risk ratio, oracle coverage, coverage error and AURC are all defined in Section 2. Calibration error uses 15 equal width bins [13]:

$$\text{ECE} = \sum_{m=1}^{15} \frac{|B_m|}{n} |\text{acc}(B_m) - \text{conf}(B_m)|, \quad (8)$$

where B_m holds the points whose confidence falls in bin m .

Every risk number comes with an error bar from bootstrap resampling, using 200 resamples for CIFAR-10-C and 2,000 for CIFAR-10.1, where there is less data.

4.4 Confidence scores

Four scores are compared. All come from the calibrated ensemble probability list $p \in \Delta^{C-1}$:

$$g_{\text{MSP}}(x) = \max_c p_c, \quad (9)$$

$$g_{\text{margin}}(x) = p_{(1)} - p_{(2)}, \quad (10)$$

$$g_{\text{ent}}(x) = \sum_c p_c \log p_c, \quad (11)$$

$$g_{\text{ens}}(x) = - \sum_c \text{Var}_s \left[p_c^{(s)} \right], \quad (12)$$

where $p_{(1)}$ and $p_{(2)}$ are the two biggest entries and $p^{(s)}$ is the output of seed s . Eq. 9 is the usual baseline [14]. Eq. 12 measures how much the three models disagree, which is a common stand in for model uncertainty [15].

4.5 Threshold selectors

Five rules are compared. All five accept in order of g_{MSP} and differ only in where they stop. By the argument in Section 2.3 they all sit on one risk coverage curve, so the only thing separating them is Δ_{cov} .

Clean transfer uses Eq. 4 on the clean validation set and keeps that cut off. This is what people do now, and it is the baseline.

Mean confidence guesses the error of the accepted set as one minus its average confidence, then picks the largest coverage whose guess meets the target. It needs no labels. It is included as a deliberately simple reference.

ATC-MC and **ATC-NE** follow Garg et al. [19]. A cut off c is fixed on validation data so that the share of validation points scoring below c equals the validation error. The score is g_{MSP} for ATC-MC and g_{ent} for ATC-NE. On new data, the error of an accepted set A is guessed as $|\{i \in A : s_i < c\}|/|A|$.

Oracle uses Eq. 5 with the true labels of the new data. Nobody can do this in real life. It is shown as the best possible answer.

Difference of Confidences [20] was also built. It picked almost the same cut offs as the mean confidence rule in early runs, so it is left out of the main tables to avoid repeating a row.

4.6 Label based risk control

The final method is Algorithm 1. It uses a small labelled sample from the new data. A list of $K = 20$ candidate cut offs is fixed ahead of time from quantiles of the *validation* confidences, so the list never depends on the new data. For each candidate, an exact binomial upper bound [23] on the selective risk is worked out from the labelled sample at level $\alpha_c = 0.10/K$. The list is walked from the loosest candidate to the strictest, and the first one whose bound meets the target is returned. That gives the most coverage that can be proved safe. Dividing the level by K is a Bonferroni correction, which keeps the overall failure chance at 0.10 or less, following Learn then Test [22]. If no candidate passes, the method stays quiet on everything. That counts as coverage zero and stays in the averages instead of being thrown away.

Algorithm 1 Picking a cut off with a proved risk bound

Require: labelled sample S from the new data; target α ; level $\delta = 0.10$; validation confidences V

```

1:  $T \leftarrow K = 20$  quantiles of  $V$ , smallest first
2:  $\delta_c \leftarrow \delta/K$ 
3: for  $t \in T$  (loosest first) do
4:    $A \leftarrow \{i \in S : g(x_i) \geq t\}$ 
5:   if  $|A| = 0$  then
6:     continue
7:   end if
8:    $k \leftarrow |\{i \in A : \hat{y}(x_i) \neq y_i\}|$ 
9:    $U \leftarrow$  exact binomial upper bound on  $k/|A|$  at level  $\delta_c$ 
10:  if  $U \leq \alpha$  then
11:    return  $t$ 
12:  end if
13: end for
14: return  $\infty$  ▷ stay quiet on everything

```

Two earlier versions of this method were built and thrown away. They are reported here because the reasons are useful and because the same mistakes are easy to make again.

The first used a longer list of 100 candidates with the same correction. It was safe but far too careful. It stayed quiet on everything when given 100 labels or fewer, and reached a real error of about 0.008 against a 0.05 target, wasting most of the coverage it could have had. A check showed the correction was not needed at all: without it, the target was broken in only 1% to 4% of runs against an allowance of 10%. Dividing the level by 100 bought nothing.

The second used fixed sequence testing, walking from the strictest candidate downwards and stopping at the first one that could not be proved safe. That is a valid idea in general and needs no correction. Here it gave coverage that *fell* as more labels were added, from 0.57 at 500 labels down to 0.20 at 4000. No correct method can do that. The reason is structural. The strictest candidates accept only a few labelled points, on average 1.7 points when given 100 labels, and no bound can prove a 5% error rate from two or three points. So the walk stopped on random noise, not on evidence. Fixed sequence testing only works when the first tests in the list can actually be passed, and here they cannot be.

Only Algorithm 1 passes a check that coverage always grows with more labels, so only its results are reported in Section 5.8.

4.7 Experimental protocol

Every measurement on CIFAR-10-C is averaged over the fifteen standard corruptions at a given strength. Label budget experiments draw the labelled sample 20 times per corruption. The cut off is picked on the drawn sample and tested on the rest, which never overlap, so no point is used twice. On CIFAR-10.1, where there are only 2,000 points, 200 draws are used.

All training ran on one NVIDIA T4. Training all six models took about 65 minutes. Everything after the saved outputs runs on a normal processor in a few minutes, so the study is cheap to repeat.

5 Results

5.1 Confidence drops slower than accuracy

Figure 1 and Table 1 show accuracy and average confidence for one ResNet18 seed over the fifteen standard corruptions. Accuracy falls 39 points from level 0 to level 5. Average confidence falls only 12 points. The gap grows at every single step, from 3.0 points to 29.7 points. At the worst level the model is wrong on 45% of inputs while still saying it is 85% sure.

This matches what Ovadia et al. [4] found, now on the models used here, and shows the problem exists before any staying quiet is added.

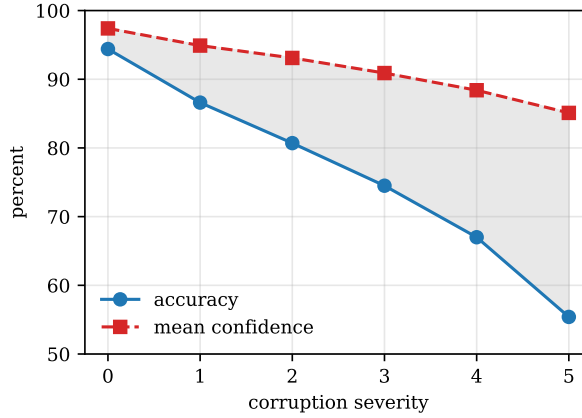


Fig. 1 Accuracy and average confidence against corruption strength. ResNet18, one seed, averaged over the fifteen standard corruptions. The grey area is the gap. Confidence falls much slower than accuracy, so the model gives no sign that its inputs got harder.

Table 1 Accuracy and average confidence against corruption strength. ResNet18, one seed, mean over fifteen corruptions.

Strength	Accuracy	Confidence	Gap
0	94.4%	97.4%	+3.0
1	86.6%	94.9%	+8.3
2	80.7%	93.1%	+12.4
3	74.5%	90.9%	+16.4
4	67.0%	88.4%	+21.4
5	55.4%	85.1%	+29.7

5.2 The clean cut off does not carry over

A cut off was picked by Eq. 4 on clean validation data for $\alpha = 0.05$ and then used unchanged on every corruption and strength. Table 2 gives the result for the ResNet18 ensemble. The target is broken at every strength and by every corruption. Coverage hardly moves, going from 0.973 down to only 0.890, which means the rule is rejecting almost nothing even while accuracy collapses. Figure 2 puts all five rules on one chart.

The error bars are small, so this is not luck. Real error at strength 5 is 0.423 with a 95% range of [0.413, 0.431]. At strength 3 it is 0.233 with [0.226, 0.241]. WideResNet-16-4 is worse at strength 5, at 0.452. The worst single corruption at strength 5 is impulse noise, at 0.745.

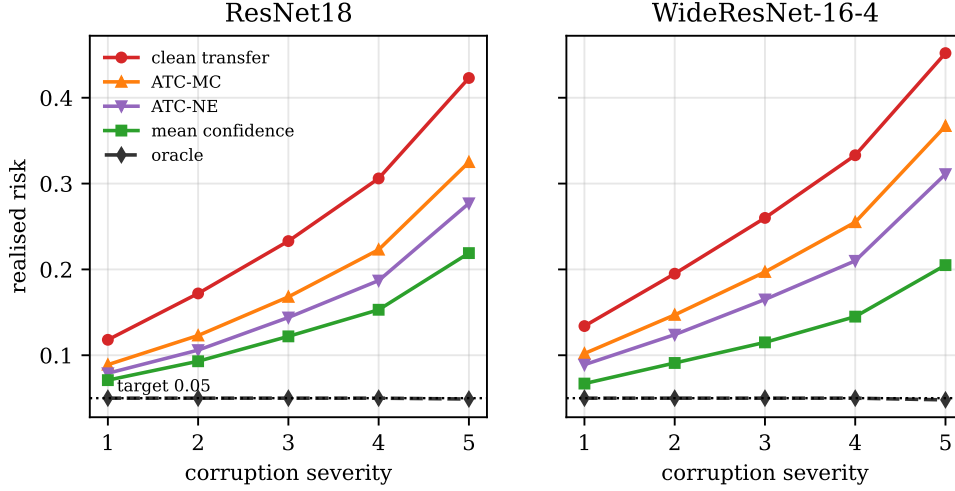


Fig. 2 Real error of five cut off rules against corruption strength, averaged over the fifteen standard corruptions. The dotted line is the 0.05 target. Every rule sits above it at every strength. The order of the five is the same for both model types.

Table 2 Real error and coverage using a cut off carried over from clean data. ResNet18 ensemble, target 0.050.

Strength	1	2	3	4	5
Error	0.118	0.172	0.233	0.306	0.423
Coverage	0.973	0.960	0.944	0.923	0.890

5.3 Stricter targets break by bigger factors

Table 3 runs the same test at four targets. The actual error goes down as the target gets stricter, because a stricter target means a higher cut off. But the *ratio* goes the other way, from 4.4 at a 10% target up to 19.1 at a 1% target.

This is the most surprising result here, so it is worth explaining why. A loose target gives a low cut off, near the point where almost everything is accepted. There the accepted set is nearly all the data, so the error is close to the overall error whatever the distribution. A strict target gives a high cut off, up in the part of the score range where the link between score and being right is exactly what shift breaks most. The bigger the promise, the more it depends on the very top of the score range behaving like it did on clean data. That is the part that does not.

Table 3 Effect of the target at strength 5, ResNet18 ensemble. Ratio is real error divided by target.

Target	Cut off	Error	Ratio	Coverage
0.10	0.313	0.443	4.4	0.986
0.05	0.457	0.420	8.4	0.890
0.02	0.829	0.288	14.4	0.520
0.01	0.941	0.191	19.1	0.362

5.4 The target was possible all along

Table 4 gives the perfect cut off from Eq. 5. Its error stays between 0.047 and 0.050 at every strength, so the target can be met everywhere. What changes is what it costs. Coverage drops from 1.000 to 0.264. Meanwhile the carried over cut off keeps coverage at 0.995. At strength 5 the working system answers 99.5% of inputs when the right answer was 26.4%. Figure 3 shows the gap.

This changes how everything above should be read. The operator did not ask for something impossible. The cut off was simply put in the wrong place, and a right place existed.

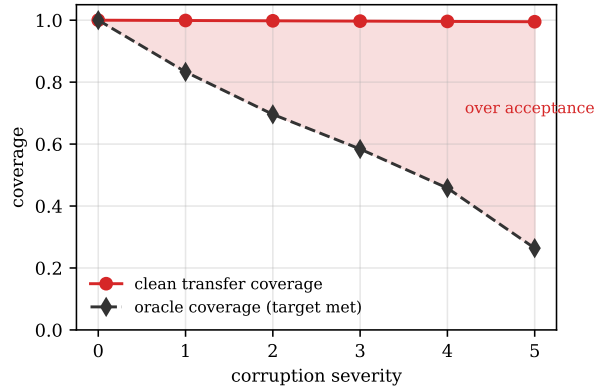


Fig. 3 Coverage chosen by the carried over cut off against the coverage where the target is really met. ResNet18 ensemble. The shaded area is the data that should have been rejected and was not. At strength 5 the gap is 73 points.

5.5 The sorting is not what is holding things back

Table 5 gives AURC for the four scores of Eqs. 9 to 12, all from the same ensemble. Lower means better sorting. Maximum probability, margin and negative entropy are within 1% of each other at every strength. Model disagreement is a little worse, by about 5%, which is worth noting since disagreement between ensemble members is often sold as a better measure of uncertainty [15].

Table 4 Perfect coverage against carried over coverage. ResNet18 three seed ensemble, target 0.050.

Strength	0	1	2	3	4	5
Perfect coverage	1.000	0.833	0.696	0.584	0.458	0.264
Carried over coverage	1.000	0.999	0.998	0.997	0.996	0.995
Perfect error	0.047	0.050	0.050	0.050	0.050	0.049

Coverage reachable at a 5% target is the same to three decimal places for maximum probability and margin, at every strength. So swapping the score does not move the best possible answer at all. Put together with Section 5.4, where a perfect cut off over plain maximum probability already meets the target everywhere, this says the sorting is close to as good as it gets for this problem.

Table 5 AURC times 1000 for four scores. ResNet18 ensemble, lower is better.

Score	Strength 1	Strength 3	Strength 5
Maximum probability, Eq. 9	32.1	96.3	255.6
Margin, Eq. 10	32.3	97.3	257.6
Negative entropy, Eq. 11	32.3	96.2	254.7
Model disagreement, Eq. 12	34.8	103.3	269.1

5.6 Coverage error points at the stopping rule

Table 6 gives coverage error at strength 5 for both model types and both corruption sets. The carried over cut off accepts 67 to 77 points too much. Both ATC versions cut that down a lot but never remove it, and both lose to the mean confidence rule, which is one line of arithmetic with nothing fitted.

The order of the five rules is the same in all four settings: oracle, then mean confidence, then ATC-NE, then ATC-MC, then clean transfer. That this order survives a change of model type and a change of corruption set is the strongest internal check in the study.

Figure 4 compares the two corruption sets across all strengths. On the held back corruptions, clean transfer reaches 0.386 for ResNet18 and 0.421 for WideResNet-16-4 at strength 5. Since nothing in the study was chosen by looking at these, the effect is not just an artefact of using the same fifteen over and over.

5.7 Real world shift

CIFAR-10.1 costs 6.4 accuracy points for ResNet18 and 6.9 for WideResNet-16-4. This is a small change, on a test set built on purpose to look like the original. Table 7 shows

Table 6 Coverage error at strength 5, target 0.050. Positive means too much was accepted. Standard means the fifteen corruptions used while building the study. Held back means the four that were not.

Rule	ResNet18		WideResNet-16-4	
	Standard	Held back	Standard	Held back
Clean transfer	+0.731	+0.675	+0.769	+0.692
ATC-MC	+0.415	+0.406	+0.469	+0.427
ATC-NE	+0.311	+0.310	+0.330	+0.298
Mean confidence	+0.227	+0.228	+0.170	+0.155
Oracle	0.000	0.000	0.000	0.000

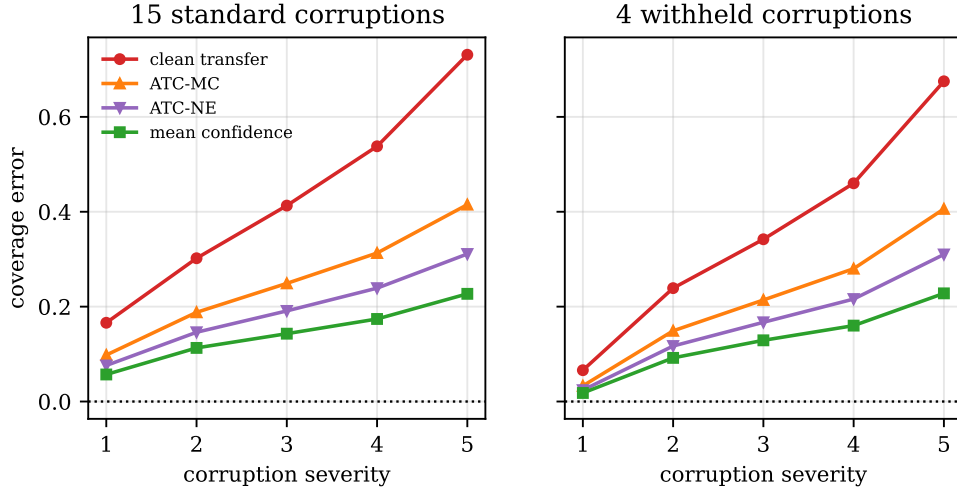


Fig. 4 Coverage error against corruption strength for ResNet18. Left is the fifteen standard corruptions. Right is the four held back from every choice made in the study. The shape and the order of the rules do not change.

the carried over cut off still misses the target by 2.2 and 2.4 times, with error bars that sit entirely above the target.

Coverage is exactly 1.000 for both model types. The system stays quiet on nothing at all. The perfect cut off meets the target at 0.828 and 0.845 coverage, so more than four fifths of the data could have been answered while keeping the promise. Instead the working rule answered everything and broke it.

The order of the five rules on CIFAR-10.1 matches the order on fake corruption exactly. For ResNet18 the errors are 0.111 for clean transfer, 0.086 for ATC-MC, 0.074 for ATC-NE, 0.066 for mean confidence and 0.050 for the oracle. Four separate settings, one order, no exceptions.

Table 7 Carried over cut off on CIFAR-10.1. Target 0.050. Ranges are 95% bootstrap over 2000 resamples.

Model	Error	95% range	Ratio	Coverage	Perfect cov.
ResNet18	0.111	[0.097, 0.125]	2.2	1.000	0.828
WideResNet-16-4	0.119	[0.105, 0.133]	2.4	1.000	0.845

5.8 How many labels a real guarantee costs

Table 8 gives Algorithm 1 on CIFAR-10-C. Efficiency means coverage divided by perfect coverage, so 1.0 would mean getting all the coverage that was possible while still keeping the promise. The target is broken in only 0% to 1% of runs against an allowance of 10%, and coverage grows with every extra label. About 2000 labels are needed to reach 80% efficiency at strength 3. More than 4000 are needed at strength 5.

Table 8 Efficiency, meaning coverage divided by perfect coverage, against label budget on CIFAR-10-C. Target 0.050, twenty draws per corruption, cut off picked and tested on separate halves.

Setting	250	500	1000	2000	4000
ResNet18, strength 3	0.39	0.64	0.76	0.82	0.85
ResNet18, strength 5	0.18	0.32	0.52	0.67	0.78
WideResNet, strength 3	0.33	0.59	0.74	0.80	0.86
WideResNet, strength 5	0.19	0.29	0.46	0.62	0.71

Table 9 repeats this on CIFAR-10.1, where much more coverage is possible in the first place. About 800 labels reach 80% efficiency, and the target is never broken. Figure 5 draws both curves.

Table 9 Efficiency and real error against label budget on CIFAR-10.1. Target 0.050, 200 draws per budget.

Setting	200	400	800	1200
ResNet18 error	0.007	0.015	0.023	0.028
ResNet18 efficiency	0.27	0.68	0.83	0.88
WideResNet efficiency	0.19	0.67	0.79	0.80

A simple guess at how many labels would be needed was tested and turned out wrong. It is reported because being wrong explains the mechanism. Proving a 5% error rate from a sample with no mistakes in it needs about 104 accepted points at level α_c , since $(1 - \alpha)^m \leq \alpha_c$ gives $m \geq \log \alpha_c / \log(1 - \alpha)$. Dividing that by the perfect

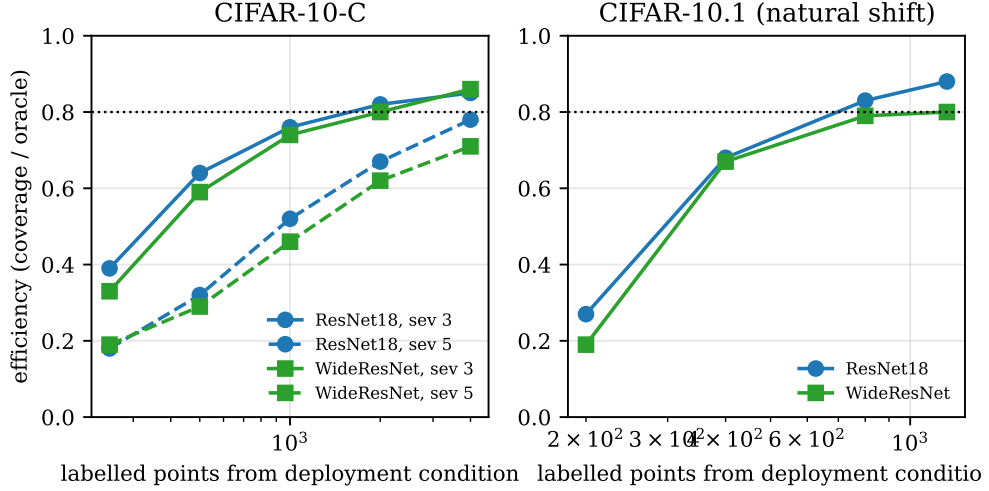


Fig. 5 Efficiency, meaning coverage divided by perfect coverage, against how many labelled points come from the new data. The dotted line marks 80%. The target was broken no more often than the 10% allowance at any point shown.

coverage at strength 3 predicts about 178 labels. The real answer is about 2000, more than ten times bigger.

The guess fails because the "no mistakes" part is wrong at the place that matters. At the perfect cut off the true error rate *equals* the target, so about 5% of accepted points are mistakes, and no finite sample can prove a 5% bound using data whose true rate is 5%. The method must back off to a safer cut off where the true error is clearly under the target. How far it must back off depends on how steeply error rises near that point, and that slope, not the no mistakes count, is what sets the cost. It also explains why strength 5 costs more: the curve is steeper there, so a bigger step back is needed for the same safety.

5.9 Some classes get protected and some do not

At strength 3 with a carried over cut off, the error among accepted answers runs from 0.133 for the class frog up to 0.423 for the class dog, a factor of 3.2. Coverage stays near 0.99 for every class. So the system does not stay quiet more often on the classes it finds hardest. A single overall error number hides this completely. Anyone reading that one number would have no idea one class carries three times the error of another. This links to work using the reject option to reduce unfair treatment [9], and suggests that under shift it can fail in exactly the direction that matters.

6 Discussion

The results give a sharp answer to a broad worry. That models get less reliable under shift is already known [4]. What these measurements add is *where* the real failure

happens. The sorting stays close to the best possible and hardly changes across four scores. The stopping rule is where the promise is lost.

What this means in practice. A cut off fitted on clean validation data should not be shipped as though it enforced a target. In these tests it accepts nearly all the data, and the stated target does not hold at any strength, on either model type, on either corruption set, or on real world shift. The advice follows straight away: either pay for labels from the new data, or do not state a target at all. Stating one that cannot be kept is worse than stating none, because other parts of the system will trust it.

What this means for research. Sections 5.4 and 5.5 together show that a perfect cut off over plain maximum probability already meets the target everywhere, and that three other scores do not move the best possible answer. So work aimed at better scores will not close the gap in Section 5.6, because that gap is not caused by the score. This claim can be proved wrong: a score that clearly improves AURC would do it.

Two surprises. First, the bigger model is worse. WideResNet-16-4 matches ResNet18 on clean accuracy within 0.3 points but has a larger coverage error at strength 5 for every rule tested. More capacity did not give a cut off that travels better, and may even hurt. Second, two published methods lose to a one line rule. ATC and Difference of Confidences were built to guess accuracy over a whole unlabelled set, not to pick a cut off on a subset that was itself chosen using the score. Section 5.6 suggests those are not the same job. This is a note about where those methods apply, not a complaint about them on the job they were built for.

A small change is enough. The CIFAR-10.1 result deserves its own line. A drop of about 6 accuracy points is the kind of drift a live system could carry for months without anyone noticing, and that test set was built on purpose to be as close to the original as possible. That alone doubles the real error while the staying quiet part does nothing. So this is not only about the extreme corruption levels used in most robustness tests. It is a real deployment problem.

The cost is worst where it hurts most. Section 5.8 shows the number of labels needed grows as the coverage that is actually possible shrinks, which is to say as shift gets worse. Someone facing strong shift needs thousands of labelled examples to prove a 5% target. Under mild real world shift a few hundred are enough. Labels are hardest to get in exactly the situations that need the most of them. That is bad news, but useful to know before planning a deployment.

7 Threats to validity

Do the measures mean what they should? Coverage error is measured against a perfect cut off worked out on the same finite test set, so it carries that set's randomness. With 10,000 points per corruption and strength that randomness is small next to the effects reported, as the error bars in Section 5.2 show. On CIFAR-10.1 there are only 2,000 points, so it is larger, and the ranges in Table 7 should be read with that in mind.

Could something inside the study explain the result? Every experiment reads one saved set of model outputs made with one fixed split, so run to run luck cannot explain a difference between two methods. The split was checked, not assumed.

The four held back corruptions were used exactly once, which limits the risk of the effect coming from looking at the same data many times. Cut off picking and testing always use separate points, so nothing leaks.

How the models were trained. The models reach almost zero training loss, and models trained that far are known to be too sure of themselves. The size of the gap in Table 1 could be smaller for models stopped earlier or held back more. The direction is backed by earlier work, but the exact numbers should not be treated as universal.

Does it carry outside this setting? The study uses one dataset family at one image size. Real world shift is covered by one benchmark, and a mild one. The conclusions should be taken as settled for CIFAR sized image classification and as guesses anywhere else.

How wasteful the fix is. Algorithm 1 gives a real error of about 0.017 against a 0.050 target and breaks the target in 0% to 1% of runs against an allowance of 10%. Both say there is slack going unused. A tighter bound would need fewer labels, so the numbers in Tables 8 and 9 are upper limits on what a well built method needs, not tight requirements.

8 Limitations

The study covers CIFAR sized data, two model types and three seeds each. Nothing here is tested at ImageNet size, and the conclusions should not be carried there without checking.

Only one real world shift benchmark is used, and it is deliberately mild. Shift caused by a change of hospital, camera or population, of the kind collected in WILDS [18], was not tested, and that is the kind that matters most when the stakes are high.

CIFAR-10.1 has only 2,000 points. A budget of 1,200 labels means labelling 60% of the new data, and at that point staying quiet has almost no purpose left. The rows of Table 9 that mean anything in practice are 200 and 400. The bigger ones are shown only to display the shape of the curve.

Training is held fixed throughout. Methods that change training, such as learning when to stay quiet at the same time as learning to classify, or training on corrupted images, were not tested. Those could move the best possible answer itself and not just the stopping rule.

Only two label free methods were tested. The negative result in Section 5.6 covers those two. It does not prove that no label free method can ever work.

Finally, the fix assumes labels from the new data can be got at all. That assumption is exactly the hard part in the settings this work is about, and the study offers no way around it.

9 Conclusion

Selective prediction under distribution shift fails at the stopping rule, not at the score. A cut off picked on clean validation data for a 5% target gives up to 45% error under strong corruption and 11% under a mild real world change, while still answering nearly everything. The failure gets worse as the target gets stricter, reaching 19 times the promise at a 1% target. Four scores sort the data almost the same way, and a perfect

cut off meets the target at every strength, so the sorting is not the problem. Two published methods reduce the failure without removing it, and both lose to a simple mean confidence rule. A method with a proper guarantee does fix it, at a cost of about 800 labels under mild real world shift and more than 4000 under strong corruption, with the cost rising as the coverage that is possible falls.

The main limit is size. The next step is to repeat the measurement at ImageNet size and on shift from a real change of data source, and to test whether a tighter bound cuts the label cost enough to make the fix practical.

Declarations

Funding. No funding was received for this work.

Competing interests. The author declares no competing interests.

Ethics approval. Not applicable. The study uses public image benchmarks and involves no people or animals.

Data availability. All data used is public. CIFAR-10 [24], CIFAR-10-C [16] and CIFAR-10.1 version 6 [17] come from their original sources.

Code availability. The code is eleven notebooks covering training, saving model outputs and analysis, including the two thrown away methods in Section 4.6. It can be shared on request.

References

- [1] Chow, C.K.: On optimum recognition error and reject tradeoff. *IEEE Transactions on Information Theory* **16**(1), 41–46 (1970)
- [2] Geifman, Y., El-Yaniv, R.: Selective classification for deep neural networks. In: *Advances in Neural Information Processing Systems*, vol. 30 (2017)
- [3] Hendrickx, K., Perini, L., Van der Plas, D., Meert, W., Davis, J.: Machine learning with a reject option: a survey. *Machine Learning* **113**(5), 3073–3110 (2024)
- [4] Ovadia, Y., Fertig, E., Ren, J., Nado, Z., Sculley, D., Nowozin, S., Dillon, J., Lakshminarayanan, B., Snoek, J.: Can you trust your model’s uncertainty? Evaluating predictive uncertainty under dataset shift. In: *Advances in Neural Information Processing Systems*, vol. 32 (2019)
- [5] Gawlikowski, J., Tassi, C.R.N., Ali, M., Lee, J., Humt, M., Feng, J., Kruspe, A., Triebel, R., Jung, P., Roscher, R., Shahzad, M., Yang, W., Bamler, R., Zhu, X.X.: A survey of uncertainty in deep neural networks. *Artificial Intelligence Review* **56**, 1513–1589 (2023)
- [6] Hüllermeier, E., Waegeman, W.: Aleatoric and epistemic uncertainty in machine learning: an introduction to concepts and methods. *Machine Learning* **110**(3), 457–506 (2021)

- [7] Cattelan, L.F.P., Silva, D.: How to fix a broken confidence estimator: evaluating post hoc methods for selective classification with deep neural networks. In: Proceedings of the Conference on Uncertainty in Artificial Intelligence (2024)
- [8] Homenda, W., Luckner, M., Pedrycz, W.: Classification with rejection based on various SVM techniques. In: International Joint Conference on Neural Networks, pp. 3480–3487 (2014)
- [9] Kamiran, F., Mansha, S., Karim, A., Zhang, X.: Exploiting reject option in classification for social discrimination control. *Information Sciences* **425**, 18–33 (2018)
- [10] Guo, C., Pleiss, G., Sun, Y., Weinberger, K.Q.: On calibration of modern neural networks. In: Proceedings of the International Conference on Machine Learning, pp. 1321–1330 (2017)
- [11] Platt, J.: Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In: Smola, A., Bartlett, P., Schölkopf, B., Schuurmans, D. (eds.) *Advances in Large Margin Classifiers*, pp. 61–74. MIT Press, Cambridge (1999)
- [12] Zadrozny, B., Elkan, C.: Transforming classifier scores into accurate multi-class probability estimates. In: Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 694–699 (2002)
- [13] Naeini, M.P., Cooper, G., Hauskrecht, M.: Obtaining well calibrated probabilities using Bayesian binning. In: Proceedings of the AAAI Conference on Artificial Intelligence, pp. 2901–2907 (2015)
- [14] Hendrycks, D., Gimpel, K.: A baseline for detecting misclassified and out of distribution examples in neural networks. In: International Conference on Learning Representations (2017)
- [15] Lakshminarayanan, B., Pritzel, A., Blundell, C.: Simple and scalable predictive uncertainty estimation using deep ensembles. In: *Advances in Neural Information Processing Systems*, vol. 30 (2017)
- [16] Hendrycks, D., Dietterich, T.: Benchmarking neural network robustness to common corruptions and perturbations. In: International Conference on Learning Representations (2019)
- [17] Recht, B., Roelofs, R., Schmidt, L., Shankar, V.: Do CIFAR-10 classifiers generalize to CIFAR-10? arXiv preprint arXiv:1806.00451 (2018)
- [18] Koh, P.W., Sagawa, S., Marklund, H., Xie, S.M., Zhang, M., Balsubramani, A., Hu, W., Yasunaga, M., Phillips, R.L., Gao, I., Lee, T., David, E., Stavness, I., Guo, W., Earnshaw, B., Haque, I., Beery, S., Leskovec, J., Kundaje, A., Pierson,

- E., Levine, S., Finn, C., Liang, P.: WILDS: a benchmark of in the wild distribution shifts. In: Proceedings of the International Conference on Machine Learning, pp. 5637–5664 (2021)
- [19] Garg, S., Balakrishnan, S., Lipton, Z.C., Neyshabur, B., Sedghi, H.: Leveraging unlabeled data to predict out of distribution performance. In: International Conference on Learning Representations (2022)
 - [20] Guillory, D., Shankar, V., Ebrahimi, S., Darrell, T., Schmidt, L.: Predicting with confidence on unseen distributions. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 1114–1124 (2021)
 - [21] Vovk, V., Gammerman, A., Shafer, G.: Algorithmic Learning in a Random World. Springer, New York (2005)
 - [22] Angelopoulos, A.N., Bates, S., Candès, E.J., Jordan, M.I., Lei, L.: Learn then test: calibrating predictive algorithms to achieve risk control. arXiv preprint arXiv:2110.01052 (2021)
 - [23] Clopper, C.J., Pearson, E.S.: The use of confidence or fiducial limits illustrated in the case of the binomial. *Biometrika* **26**(4), 404–413 (1934)
 - [24] Krizhevsky, A.: Learning multiple layers of features from tiny images. Technical report, University of Toronto (2009)
 - [25] He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)
 - [26] Zagoruyko, S., Komodakis, N.: Wide residual networks. In: Proceedings of the British Machine Vision Conference (2016)
 - [27] Smith, L.N., Topin, N.: Super convergence: very fast training of neural networks using large learning rates. In: Artificial Intelligence and Machine Learning for Multi Domain Operations Applications. SPIE (2019)
 - [28] Quiñonero-Candela, J., Sugiyama, M., Schwaighofer, A., Lawrence, N.D. (eds.): Dataset Shift in Machine Learning. MIT Press, Cambridge (2009)
 - [29] El-Yaniv, R., Wiener, Y.: On the foundations of noise free selective classification. *Journal of Machine Learning Research* **11**, 1605–1641 (2010)
 - [30] Gal, Y., Ghahramani, Z.: Dropout as a Bayesian approximation: representing model uncertainty in deep learning. In: Proceedings of the International Conference on Machine Learning, pp. 1050–1059 (2016)
 - [31] Shafer, G., Vovk, V.: A tutorial on conformal prediction. *Journal of Machine Learning Research* **9**, 371–421 (2008)