A Additional experimental details

A.1 About the evaluation metrics

All metrics are reported in percentage terms. The out-of-distribution detection metrics leverage the entropy. For the misclassification detection tasks, we use the confidence score (i.e. the maximum probability of the softmax) as uncertainty metric, as we find it to be the most effective for the task.

A.2 The impact of the input preprocessing pipeline

	Clean Data			**Domain-Shift**												**OOD**
	ImageNet-1K (Test)			**ImageNet-R**			**ImageNet-A**			**ImageNet-V2**			**ImageNet-Sk**			**ImageNet-O**
	Acc (↑)	**ECE** (↓)	**AdaECE** (↓)	**Acc** (↑)	**ECE** (↓)	**AdaECE** (↓)	**Acc** (↑)	**ECE** (↓)	**AdaECE** (↓)	**Acc** (↑)	**ECE** (↓)	**AdaECE** (↓)	**Acc** (↑)	**ECE** (↓)	**AdaECE** (↓)	**AUROC** (↑)
BiT-R50x1	80.05	1.58	1.76	38.98	10.01	10.01	26.89	19.73	19.67	67.98	1.75	1.74	24.72	18.39	18.39	67.01
BiT-R50x3	83.59	2.65	2.51	47.25	8.51	8.51	46.72	11.66	11.63	72.36	6.30	6.08	32.81	19.00	19.00	77.99
BiT-R101x1	82.04	1.16	1.06	43.65	7.49	7.49	38.32	15.79	15.72	70.97	4.33	4.29	29.10	18.28	18.28	73.62
BiT-R101x3	84.19	3.78	3.72	50.14	9.10	9.10	53.12	10.90	10.93	73.36	7.85	7.71	36.29	21.15	21.15	80.44
BiT-R152x2	84.17	2.96	2.71	51.02	8.51	8.51	52.97	10.37	10.13	73.46	6.30	6.12	36.96	19.22	19.22	80.72
BiT-R152x4	84.49	6.28	6.26	54.06	11.50	11.50	58.52	12.17	12.14	74.36	10.94	10.94	41.17	25.47	25.47	85.58
ConvNeXt-B	85.53	2.87	2.82	62.46	2.57	2.51	52.63	8.28	8.31	75.43	2.91	2.78	48.62	8.87	8.86	85.72
ConvNeXt-L	86.29	2.27	2.34	64.57	3.00	3.08	58.23	7.57	7.26	76.77	3.72	3.85	50.06	10.31	10.31	89.07
ConvNeXt-XL	86.58	2.40	2.29	66.01	2.92	2.90	61.11	7.54	7.21	77.20	4.00	4.24	52.67	11.15	11.15	90.04
ViT-B/16	77.85	1.39	1.38	43.09	5.28	5.28	23.31	23.51	23.51	65.94	4.67	4.53	18.33	12.74	12.74	79.93
ViT-L/16	84.33	1.72	1.70	61.75	2.88	2.88	46.36	12.55	12.39	74.15	5.52	5.43	46.21	10.56	10.56	90.63
Swin-B	84.81	8.52	8.52	59.81	2.11	2.14	49.88	8.57	8.40	75.07	5.11	5.06	45.43	7.50	7.50	83.94
Swin-L	85.95	5.65	5.65	64.44	2.29	2.19	58.96	6.82	6.83	76.49	3.24	3.02	49.06	8.73	8.72	87.66
Fine-tuned at resolution 384×384																
ConvNeXt-B-384	86.51	3.16	3.15	64.12	3.36	3.47	63.25	7.70	7.58	77.03	2.49	2.65	50.31	7.84	7.84	87.11
ConvNeXt-L-384	87.14	2.39	2.38	66.09	3.27	3.16	66.52	7.01	6.90	77.97	3.51	3.31	51.68	9.60	9.60	90.45
ConvNeXt-XL-384	87.45	2.37	2.49	67.24	3.22	3.35	69.59	7.28	7.29	78.34	3.03	2.87	53.80	8.69	8.67	91.12
ViT-B16-384	79.43	1.53	1.60	40.62	6.49	6.49	33.63	17.46	17.46	68.37	4.45	4.45	14.54	15.75	15.75	81.75
ViT-L16-384	85.80	2.09	1.93	63.26	3.31	3.31	63.07	6.11	5.86	76.47	5.29	5.25	46.10	12.38	12.38	92.42
SWIN-B-384	86.29	6.78	6.78	63.41	2.29	2.28	62.20	6.57	6.52	76.65	3.80	3.83	48.43	8.43	8.43	86.46
SWIN-L-384	87.01	6.58	6.58	66.40	3.40	3.50	67.92	7.37	7.29	77.51	3.89	3.79	50.29	7.62	7.62	89.25

Table 6: Analogous of Tables 3 and Table 4 but using the prepocessing pipeline suggested suggested by the timm library for each model. The conclusions of the main paper do not change.

The standard pre-processing pipeline For the results reported in the main paper, we apply the standard ImageNet-1K test pre-processing pipeline: we first rescale the image at resolution 256 × 256 then extract the center crop of 224 × 224 and normalise with respect to the mean and standard deviation of the training set.

Model-specific pre-processing pipelines However, it should be noticed that the timm library suggests using a different pre-processing pipeline for each architecture. We do not follow this procedure for the results in the main paper as fine-tuning the test pre-processing pipeline hyperparameters would require a cross-validation procedure to not overfit the test set and we want to have a fair comparison using the same evaluation procedure for all models. We report the results applying the timm-proposed preprocessing pipelines in Tables 6 and 7. All the conclusions drawn in the main paper about ConvNeXts, ViTs and SwinTranformers do not change. The only case in which altering the pipeline dramatically changes the performance is on BiT models. With respect to the performance with the default pre-processing pipeline, BiT models become:

- **significantly more accurate on in-distribution data**. For instance, BiT-R152×4's accuracy jumps from 78.16% to 84.49%.

	Clean Data	**Domain-Shift**			
	ImageNet-1K (Test)	**ImageNet-A**	**ImageNet-R**	**ImageNet-SK**	**ImageNet-V2**
	PRR (↑)				
BiT-R50x1	72.48	23.31	-25.84	56.68	68.08
BiT-R50x3	73.41	-19.39	-8.67	62.41	67.39
BiT-R101x1	74.04	16.70	-22.27	60.64	68.12
BiT-R101x3	73.39	15.32	**-8.64**	62.54	66.61
BiT-R152x2	73.24	48.97	-20.76	61.36	66.81
BiT-R152x4	71.82	23.89	-35.15	62.15	64.54
ConvNeXt-B	73.43	16.03	-39.91	67.48	69.84
ConvNeXt-L	73.48	40.56	-23.60	69.04	69.50
ConvNeXt-XL	**74.36**	35.96	-19.32	69.29	70.07
ViT-B16	74.12	**49.46**	-33.53	64.59	70.89
ViT-L16	76.24	5.92	-31.09	**69.70**	**72.61**
Swin-B	71.99	17.10	-16.70	63.98	67.61
Swin-L	72.70	-10.78	-25.43	63.83	68.91
	Fine-tuned at resolution 384×384				
ConvNeXt-B-384	74.06	36.79	-22.39	67.37	68.75
ConvNeXt-L-384	74.12	32.74	**-10.88**	68.47	69.16
ConvNeXt-XL-384	74.71	55.16	-12.21	69.05	70.27
ViT-B/16-384	74.35	46.47	-32.97	66.18	71.13
ViT-L/16-384	**76.89**	-9.48	-20.06	**69.41**	**72.76**
Swin-B-384	72.53	**69.93**	-42.21	63.73	67.58
Swin-L-384	71.73	27.26	-17.02	63.04	66.71

Table 7: Analogous of Table 5 but using the preprocessing pipeline suggested by the timm library for each model. The conclusions of the main paper do not change.

- **significantly more accurate on covariate shifted inputs**. Particularly remarkable is the improvement when exposed to ImageNet-A. For instance, the accuracy of BiT-R50×1 jumps from 10.97 to 38.98 (which renders the smallest BiT model better performing than ViT-B/16!). Similarly, larger capacity BiT models can outperform ViT-L/16 and BiT-R152×4 is as competitive as the top-performing transformer (Swin-L). It is important to recall that ImageNet-A samples were selected to produce low accuracy on ResNets. This selection bias obviously makes comparisons between ResNets and any other architecture unfair. However, already changing the pre-processing pipeline at test time is enough to significantly weaken the adversarial effectiveness of the selection process on ResNet inspired architectures. Similarly, on other data-shift datasets, BiTs become extremely more competitive, and can outperform or be almost comparable to smallest transformer variants in many cases.
- **significantly better at out-of-distribution detection** (e.g. the minimum gap between BiT models and ViT-L/16 passes from almost 11% to less than 7%)
- **significantly more calibrated on both in-domain and covariate shifted inputs**. (e.g. the ECE is approximately halved in most cases on in-distribution data)

- **significantly better at performing in-domain misclassification detection and most distribution-shift experiments**. It increases (in most cases) on ImageNet-R, ImageNet-SK and ImageNet-V2. On ImageNet-A the performance decreases. This is another interesting case in which the calibration and misclassification detection provide complementary information: while the calibration error decreases on ImageNet-A, the misclassification detection performance gets worse, indicating the problem of being overconfidently wrong becomes more pronounced.

Models fine-tuned at resolution 384×384 It should also be noticed that variants fine-tuned at resolution 348×384 exist (see the lower parts of Table 6 and Table 7). These variants generally outperform the variants fine-tuned at lower resolution in terms of accuracy, but generally exhibit worse uncertainty properties. The final conclusions of our paper do not change when considering these variants. Since we could not find BiT checkpoints fine-tuned at this resolution in the timm library, we reported the performance for models fine-tuned at 224×224 to have a fair comparison.

A.3 The impact of pre-training

It would be interesting to study the robustness and reliability of models without pre-training on ImageNet-21K. Unfortunately, checkpoints training solely on ImageNet-1K are often not included in the timm library or in general not publicly available, mostly because some of the models considered do not produce good performance if trained from scratch on ImageNet-1K.

For completeness, we report the performance results on ConvNeXt-B/L and Swin-B in Tables 8 and 9 . Notice, in this case the out-of-distribution detection results are reported using the negative confidence score as a form of uncertainty, as we find it to be the most effective in this case.

As it can be seen in Table 8, ConvNeXt-B is typically more accurate and better calibrated than Swin-B except on ImageNet-A and ImageNet-V2 (where Swin-B is more calibrated). Swin-B produces better out-of-distribution detection performance. As seen in Table 9, ConvNeXt-L outperforms all other models at misclassification detection except in one case. However, we cannot draw conclusions from these two tables given the lack of comparison with other strong Transformer architectures and CNNs.

We can however evaluate the difference between with and without pretraining as follows:

- in all cases, in-distribution accuracy is significantly improved by pre-training
- for ConvNeXt models, the lack of pre-training harms calibration on in-domain data and under covariate shift. For Swin-B, the lack of pretraining improves the calibration on in-distribution data, but harms it under covariate shift.
- the lack of pretraining significantly damages the out-of-distribution detection performance of all models

- the lack of pretraining harms the misclassification detection performance except in the case of ImageNet-R for ConvNeXt-B and Swin-B. Swin-B performance drops more significantly than ConvNeXt-L models without pretraining.

	Clean Data			**Domain-Shift**												**OOD**
	ImageNet-1K (Test)			**ImageNet-R**			**ImageNet-A**			**ImageNet-V2**			**ImageNet-Sk**			**ImageNet-O**
	Acc (↑)	**ECE** (↓)	**AdaECE** (↓)	**Acc** (↑)	**ECE** (↓)	**AdaECE** (↓)	**Acc** (↑)	**ECE** (↓)	**AdaECE** (↓)	**Acc** (↑)	**ECE** (↓)	**AdaECE** (↓)	**Acc** (↑)	**ECE** (↓)	**AdaECE** (↓)	**AUROC** (↑)
ConvNeXt-B	83.73	3.33	3.43	51.72	8.18	8.14	35.79	22.55	22.51	73.69	5.55	6.30	38.27	22.78	22.78	62.64
ConvNeXt-L	84.16	3.86	3.95	53.93	8.50	8.47	40.54	21.42	21.40	74.01	5.74	6.47	40.14	23.40	23.40	62.68
Swin-B	83.08	5.08	5.01	47.20	8.72	8.71	34.39	20.43	20.45	72.10	5.29	4.99	32.62	22.83	22.83	64.01

Table 8: Analogous of Table 6, but checkpoints are not pre-trainined on ImageNet-21K.

	Clean Data	**Domain-Shift**			
	ImageNet-1K (Test)	**ImageNet-A**	**ImageNet-R**	**ImageNet-SK**	**ImageNet-V2**
	PRR (↑)				
ConvNeXt-B	70.45	-7.13	-11.43	65.16	65.31
ConvNeXt-L	69.71	36.21	-30.27	64.04	65.93
Swin-B	68.20	34.16	-2.43	59.58	63.18

Table 9: Analogous of Table 7, but checkpoints are not pre-trained on ImageNet-21K.

B Further discussion on why the practice of comparing models based on parameter count might be misleading

In this section we provide additional examples explaining why the parameter count is not really a representative of a model's capacity and its generalizability i.e., the ability of a model to capture better approximations of the function underlying the relationship between inputs and outputs that generalise better.

One might wonder whether ways to quantify this aspect of a model exist. For this reason, we resort to the known complexity measures in the literature and show that these are no better than parameter count for the purpose of comparing models belonging to different families of architectures. This advocates for the need of measures that allow to compare models independently of their kinship.

B.1 Practical examples indicating why parameter count is not a good proxy to compare model capacity and generalization

It is important to observe that all the considered models have significantly more parameters than the number of training samples (even when considering

ImageNet-21K as training set). Therefore, from a theoretical point of view, all the considered models can interpolate the training set. These models differ in the way in which their learning procedures can leverage the data and the available parameters to learn solutions that generalise better. How overparametrization is related to the extraordinary generalization properties of Neural Networks is still an open area of research, and out of the scope of this paper. Consider the following examples (refer Table 10 for parameter counts):

- consider BiT-R152×2 and BiT-R152×4. It is evident that although the latter has about 4 times the number of parameters of the former, the performance improvements observed in our tables are often marginal. This implies the training procedure is not capable of leveraging the additional number of parameters to boost the performance. It will be interesting if the future literature investigates how much BiT-R152×4 can be pruned before it looses its advantage over BiT-R152×2.
- consider ConvNeXt-B and BiT-R152×4. Although the first contains almost 10× less parameters than the latter, and both rely on convolutional inductive biases, ConvNeXt-B significantly outperforms BiT-R152×4 almost always. This comparison shows that parameter count is not representative of the generalisation properties of a model even when comparing models sharing convolutional inductive biases. Several other design choices that are often neglected in existing literature comparing the robustness and reliability of Transformers and CNNs (e.g. quantity and types of activations or normalization layers, kernel sizes, proportions between the block sizes etc.) can greatly influence the ability of a model to produce robust and reliable predictions.

B.2 Can complexity measures do better than parameter count?

A natural question that arises from the above observations is whether it is possible to find a measure that quantifies the generalization properties of a model as a function of its input-output behaviour, training dynamics, the properties of the mappings it has learned, or all these combined together. A recent study collected and compared the most popular measures in this regard [16]. We consider the two most popular ones and show how they cannot be used to compare the generalization properties of models belonging to different families, and therefore, for this purpose, are no more useful than parameter count.

- Path-Norm [26], defined as:

$$\mathrm{PN} = \sum_i f_{w^2}(\mathbf{1})[i]$$

  where f_{w^2} represents a network whose parameters have been squared, $\mathbf{1}$ indicates an input (of adequate shape, in this case we apply the same shape of ImageNet inputs) for which each entry is set to 1, $f_{w^2}(.)[i]$ represents the logit associated to class i.

- (logarithm of) Spec-Fro [27], defined as:

$$\log \mathrm{SF} = \log \prod_{i=1}^{L} ||\mathbf{W}_i||_2^2 \sum_{i=1}^{L} \mathrm{srank}(\mathbf{W}_i)$$

  where L is the total number of layers in the network, $\mathbf{W}_i$ represents the weight matrix of the i-th linear layer, $\mathrm{srank}(\mathbf{W}_i)$ represents the stable rank of $\mathbf{W}_i$, i.e. $\mathrm{srank}(\mathbf{W}_i) = ||\mathbf{W}_i||_F^2/||\mathbf{W}_i||_2^2$ [33]. The logarithm is taken for numerical stability reasons.

As shown in Table 10, both these metrics are inadequate in comparing models belonging to different families (e.g. Path-Norm and Spec-Fro of BiT are evidently at another scale with respect to those of other models; also, no inter-family consistent sorting based on generalization on the in-domain test set seems to emerge). Also for the same architecture, the behaviour of these metrics is inconsistent when comparing models pre-trained on ImageNet-21K and then fine-tuned on ImageNet-1K with respect to those trained only on ImageNet-1K. For instance, in the case of ConvNeXt these metrics remain almost unchanged, while for Swin-B the change is dramatic. They also produce inconsistent behaviours within a family, for instance, they do not sort based on generalisation properties the ViT-B and L models with patch sizes 16 and 32. For these reasons, for the purposes of this analysis, these metrics are no more useful than the parameter count. Future research should address this issue.

	# params	Path-Norm	log-Spec-Fro
BiT-R50×1	25	71.90	101.179
BiT-R50×3	217	211.21	103.10
BiT-R101×1	44	75.43	197.44
BiT-R101×3	387	224.96	199.62
BiT-R152×2	232	151.50	295.13
BiT-R152×4	936	298.22	296.47
ConvNeXt-B	88	0.51	2.23
ConvNeXt-L	196	0.75	55.60
ConvNeXt-XL	348	-0.28	80.75
ViT-B/16	86	0.34	46.77
ViT-L/16	304	-0.44	118.72
ViT-B/32	88	0.17	46.20
ViT-L/32	306	0.86	118.17
Swin-B	87	0.04	34.11
Swin-L	195	-0.95	84.72
Trained on ImageNet-1K only			
ConvNeXt-B	88	0.50	2.22
ConvNeXt-L	196	0.76	55.60
Swin-B	87	-314.96	381.30

Table 10: **Path-Norm and Spec-Fro Complexity measures** for each of the considered models (checkpoints pre-trained on ImageNet-21K and fine-tuned on ImageNet-1K, except for the bottom part of the table

C Samples of the ImageNet-9 and Cue-Conflict dataset

To provide better context, in Figures 3 and 4 we show a few samples from the ImageNet-9 mixed-same and mixed-random splits. In Figure 5 we show samples from the Cue-Conflict dataset.

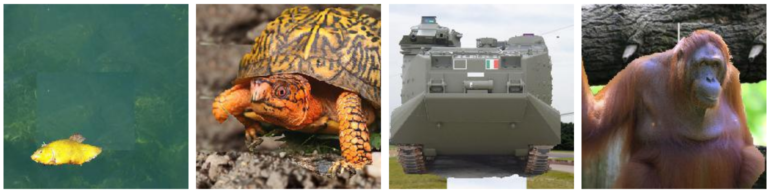

Fig. 3: Samples from the ImageNet-9 mixed-same split, in which the foreground of a class is mixed with a background from the same class.

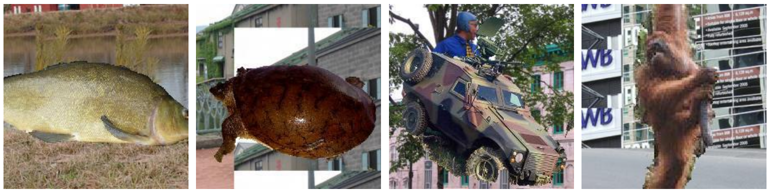

Fig. 4: Samples from the ImageNet-9 mixed-random split, in which the foreground of a class is mixed with a background from another class.

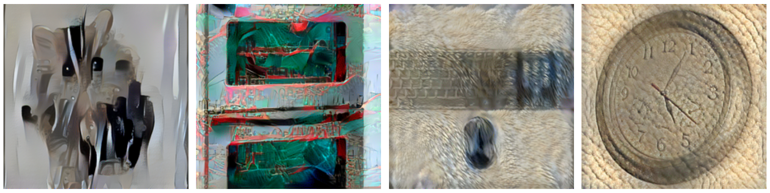

Fig. 5: Samples from the Cue-Conflict dataset where style transfer is used to alter the texture of an image using the image from another class as the style source.

D Proof that AUROC is invariant to the choice of positive and negative classes

Here we provide a simple proof to show that for the binary threshold classifier, AUROC does not vary depending on the choice of positive and negative classes. We would like to mention that we do not claim any technical novelty here. This proof is entirely for the purpose of completeness and to theoretically support our empirical findings in Table 2.

Given a classifier $f : \mathbf{x} \mapsto \mathbb{R}^k$ and a scoring function $g : \mathbb{R}^k \mapsto \mathbb{R}$ (e.g. entropy), let the binary threshold classifier be such that $g(f(\mathbf{x})) \geq t$ for a given threshold t implies that the sample $\mathbf{x}$ belongs to 'positive' class, otherwise negative. Therefore, given a dataset with M samples, one could simply sort these samples using the $g(.)$ scores and find an index beyond which all the samples belong to the positive class. Now, let us define TP as the number of true positives (similarly, FN, FP, and TP are defined). Total number of positive samples can then be calculated as $P = TP + FN$. Similarly, total number of negatives $N = TN + FP$. Using these notations, following rates can be defined

- ○ TPR $= TP/(TP + FN)$ (True Positive Rate, also called Recall[5])
- ○ FPR $= FP/(FP + TN)$ (False Positive Rate)
- ○ TNR $= TN/(TN + FP)$ (True Negative Rate)
- ○ FNR $= FN/(FN + TP)$ (False Negative Rate)

By definition, AUROC is the area under the TPR and FPR curve, where each (TPR(t_i), FPR(t_i)) point on the curve is specific to a particular threshold t_i that is used for the classification using the score $g(.)$.

Let $t_i > t_j$ if $i > j$ then, it is simple to observe that the ROC is a monotonically increasing (not strictly) stair function whose step occur in correspondence of an input $\mathbf{x}$ in the dataset. Since the dataset size is fixed (say M samples), one can identify at most M values of t at which the ROC value could increase. Let (TPR(t_i), FPR(t_i)) be the element of this ordered set. Since the dataset size is fixed, there are at most M increasing values of the threshold t. Then, the area under ROC curve can be obtained as

$$\text{AUROC} = \sum_{i=1}^{M} (b_i - b_{i-1}) h_i$$

where, $b_i = FPR(t_i)$ and $h_i = TPR(t_i)$.

Now, let us flip the labels and $-g(f(\mathbf{x})) \geq t$ is considered as the 'positive' class[6] (note, this way scoring function reverts the order with which the dataset can be sorted). Let TP' now denote the true-positive in this new scenario. Similarly, let's apply the same convention to the other elements of the confusion matrix in this scenario. The relation between the confusion matrices of this scenario and the previous is:

[5] Precision $= TP/(TP + FP)$

[6] Any strictly monotonically decreasing function applied to g will not change what follows.

- ∘ $TP' = TN$
- ∘ $FP' = FN$
- ∘ $FN' = FP$
- ∘ $TN' = TP$

Therefore,

$$\begin{aligned}\text{TPR} &= TP/(TP+FN) = TN'/(TN'+FP') = \text{TNR}' = 1 - \text{FPR}',\\ \text{FPR} &= FP/(FP+TN) = FN'/(FN'+TP') = \text{FNR}' = 1 - \text{TPR}'. \qquad (1)\end{aligned}$$

The AUROC can now be computed as:

$$\text{AUROC}' = \sum_{i=1}^{M}(b'_i - b'_{i-1})h'_i.$$

It is easy to observe that since the sorting of the samples based on their scoring function values is reversed, harmonising the indexing across these two sorted sets, then $b'_i = 1 - h_{M-i}$ and $h'_i = 1 - b_{M-i}$ (applying Eq. (1)). Replacing these values in AUROC$'$ we obtain

$$\begin{aligned}\text{AUROC}' &= \sum_{i=1}^{M}(1 - h_{M-i} - (1 - h_{M-i+1}))(1 - b_{M-i})\\ &= \sum_{i=1}^{M}(h_{M-i+1} - h_{M-i})(1 - b_{M-i}) = \text{AUROC}\end{aligned}$$

The last equality is obvious with geometric reasoning: while the AUROC partitions the ROC with vertical rectangles (one for each step) and summates the area of the rectangles obtained this way, AUROC$'$ partitions the same ROC stair using horizontal lines (one for each step) and summates the area of these rectangles (which is obviously the same). This is further validated by reasoning geometrically on the mapping implied by the swap of the positive class: this mapping moves the origin of the original space to $(1, 1)$, rotates the coordinate axes of 90° anti-clockwise and flips what used to be the horizontal axis orientation.

Additionally, it is straightforward to observe that when the minority class is sampled multiple times for rebalancing, TPR and FPR are unchanged, therefore, AUROC is unchanged.