Fractal Calibration for long-tailed object detection

Supplementary Material

6. Background: Classification Calibration

We theoretically derive the classification calibration for image classification. Let $p_s(y|x)$ and $p_t(y|x)$ be the source and target conditional distributions. Using the Bayes theorem, we write the source and target conditional distributions as:

$$p_s(y|x) = \frac{p_s(x|y)p_s(y)}{p_s(x)}, p_t(y|x) = \frac{p_t(x|y)p_t(y)}{p_t(x)} \quad (12)$$

Dividing them, we write the target conditional distribution:

$$p_t(y|x) = \frac{1}{\kappa(x)} \frac{p_t(y)}{p_s(y)} p_s(y|x) \frac{p_t(x|y)}{p_s(x|y)} \quad (13)$$

where $\kappa(x) = \frac{p_t(x)}{p_s(x)}$. During training, we approximate $p_s(y|x)$ by model $f_y(x;\theta) = z$ and a scorer function $s(x) = e^x$ for multiple category classification. Thus, the learned source conditional distribution is $p_s(y|x) \propto e^{f_y(x;\theta)}$. Substituting it inside Eq. 13, we rewrite the target condition distribution as:

$$\begin{aligned} p_t(y|x) \propto & \frac{1}{\kappa(x)} \frac{p_t(y)}{p_s(y)} e^{f_y(x;\theta)} \frac{p_t(x|y)}{p_s(x|y)} \\ = & d(x,y) \cdot e^{f_y(x;\theta) + \log(p_t(y)) - \log(p_s(y)) - \log(\kappa(x))} \end{aligned} \quad (14)$$

where we assume that $d(x,y) = \frac{p_t(x|y)}{p_s(x|y)} = 1$. This is a reasonable assumption, in cases where both train and test generating functions come from the same dataset, as it is in our benchmarks. In inference, we calculate the prediction $\bar{y}$ by taking the maximum value of Eq. 14:

$$\begin{aligned} \bar{y} &= \arg\max_y e^{(f_y(x;\theta) + \log(p_t(y)) - \log(p_s(y)) - \log(\kappa(x)))} \\ &= \arg\max_y (f_y(x;\theta) + \log(p_t(y)) - \log(p_s(y))) \end{aligned} \quad (15)$$

where $\kappa(x)$ is simplified because it is a function of x and it is invariant to $\arg\max_y$. Eq. 15 is the post-calibration method [28, 65]. It can be used during inference to achieve balanced performance by injecting prior knowledge inside the model's predictions, via $p_t(y)$ and $p_s(y)$, in order to align the source with the target label distribution and compensate for the label shift problem.

7. Fractal Dimension Variants

We explore various ways for computing the fractal dimension using the box-counting method [80], the information dimension [78] (Info), and a smooth variant (SmoothInfo). The information variant is defined as:

Dimension	AP^m	AP^m_r	AP^b
Info	**28.6**	23.2	28.3
SmoothInfo	**28.6**	**23.4**	28.3
Box	**28.6**	23.0	**28.4**

Table 9. Fractal Dimension Variants using MaskRCNN with ResNet50 and RFS on LVISv1. All of the are robust and we have chosen the Box variant in the main paper.

$$\text{Info-}\Phi(y) = \lim_{G\to\infty} \frac{\log \sum_{j=0}^{G-1} \sum_{i=0}^{G-1} \frac{\mathbb{1}(n_y(\mathbf{u}))}{G}}{\log(G)} \quad (16)$$

It is the similar to the box-counting dimension, except for the box count which is normalised by the grid size G. This way, the information dimension is represented by the growth rate of the probability $p = \frac{\mathbb{1}(n_y(\mathbf{u}))}{G}$ as G grows to infinity.

In practise, the quantity $\mathbb{1}(n_y(\mathbf{u}))$ can be frequently zero for many locations $\mathbf{u}$ especially for rare classes that have few samples and are sparsely located. For this reason, we also proposed a smooth information variant defined as:

$$\text{Smooth-}\Phi(y) = \lim_{G\to\infty} \frac{1 + \log \sum_{j=0}^{G-1} \sum_{i=0}^{G-1} \frac{1+\mathbb{1}(n_y(\mathbf{u}))}{G}}{\log(G)} \quad (17)$$

This Equation is inspired by the smooth Inverse Document Frequency [79] used in natural language processing and its purpose is to smooth out zero values in $\mathbb{1}(n_y(\mathbf{u}))$ calculation.

All variants are robust and SmoothInfo achieves slightly better AP^m_r because its calculation is more tolerant to few samples compared to the box-counting method. However, SmoothInfo and Info achieve slightly worse AP^b, thus we use the box-counting method in the main paper.

8. Object Distributions

We show that the object distribution $p_s(o,u)$ in the training set is similar to the object distribution $p_t(o,u)$ on the test set in the LVIS v1 dataset [22]. As shown in Figure 6, the distributions are close therefore we can safely assume that $p_s(o,u) \approx p_t(o,u)$. This explains the reason why the background logit should remain intact during calibration because there does not exist label shift for the generic object class (also for the background class) between the train and test sets.

Method	$dAP^b_{Cls}(\downarrow)$	$dAP^b_{Loc}(\downarrow)$	$dAP^b_{Both}(\downarrow)$	$dAP^b_{Dupe}(\downarrow)$	$dAP^b_{Bkg}(\downarrow)$	$dAP^b_{Miss}(\downarrow)$
Baseline	31.76	**6.16**	**0.45**	0.32	**1.8**	**6.82**
Cls calibration	20.49	6.96	1.02	**0.01**	3.26	6.46
Cls + Space calibration	**16.91**	6.42	0.85	**0.01**	2.84	6.84

Table 10. Error analysis using TIDE toolkit [6]. The class calibration reduces the misclassification error but it introduces more false background detections compared to the baseline. When adding the space calibration, it further reduces the misclassification error and also reduces the false background detections compared to the class calibration only.

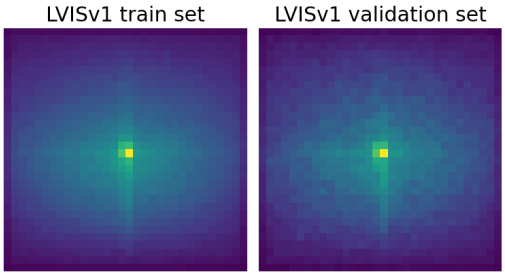

Figure 6. Comparison between the $p_s(o,u)$ (left) and $p_t(o,u)$ (right) in LVISv1 dataset. The distributions are similar, therefore we can safely assume that $p_s(o,u) \approx p_t(o,u)$.

9. Error Analysis

We perform an error analysis on the baseline method that uses MaskRCNN ResNet50 and our FRACAL with the same architecture, using the TIDE toolkit [6]. The TIDE toolkit, reports errors using the dAP^b metric which shows the AP^b loss due to a specific detection error. In more detail, dAP^b_{Cls} shows the AP^b loss due to misclassification, dAP^b_{Loc} shows the AP^b loss due to mislocalisation, dAP^b_{Both} indicates the loss due to both misclassification and mislocalisation, dAP^b_{Dupe} indicates the loss due to duplicate detections, dAP^b_{Bkg} shows the error due to background detections and dAP^b_{Miss} shows the errors due to miss-detections.

As Table 10 shows, by adding only the class calibration, the dAP^b_{Cls} is reduced significantly by 11.29 percentage points (pp) compared to the baseline. This shows that the class calibration method increases the correct rare class predictions. However, the class calibration sometimes overestimates the rare classes and predicts rare objects instead of background obtaining 1.44pp larger dAP^b_{Bkg} than the baseline. When the space calibration is added to the class calibration then it further reduces the misclassification rate dAP^b_{Cls} by 3.58pp showing that the space calibration classifies better the rare classes. At the same time, the space calibration slightly reduces the false background predictions by 0.42pp compared to the class calibration method. This analysis shows that both class and space calibration are important and the joint use can lead to better and more balanced detectors.

10. Confidence Calibration.

Many works in classification [21, 39, 67, 74] and object detection [40, 42, 73] study confidence calibration, which is a technique that allows the model to match its confidence score with its expected accuracy. Confidence calibration is important because it allows the detectors to output calibrated predictions that match the expected average precision. This leads to a safer deployment of detectors because the calibrated detectors provide reassurance regarding their detections which is a desired property in many safe-critical applications like autonomous vehicles [42].

In our main paper, we have focused on logit calibration, following the terminology of [69], which is different from confidence calibration, as the former aims in improving the rare class performance and the latter aims in reducing the expected calibration error. Despite that, we have analysed the calibration performance of our method compared to the baseline using the LVIS validation set, MaskRCNN ResNet50 and the newly proposed $LaECE_0$ and $LaACE_0$ metrics [42]. The $LaECE_0$ metric shows the mean absolute error between the detection confidence of all predictions and the respective IoU that matches the ground-truth aggregated over 25 confidence bins. $LaACE_0$ is similar to $LaECE_0$, however instead of using 25 bins, it uses an adaptive bin size. Both metrics jointly measure the localisation and classification quality and a low value indicates that the detection has an aligned localisation and classification estimate for the ground-truth. For more details on these metrics, please refer to [42].

Method	$LaECE_0$ (↓)	$LaACE_0$ (↓)
Baseline	16.8	19.8
FRACAL (ours)	**14.9**	**15.1**

Table 11. Calibration Errors using MaskRCNN ResNet50 on LVIS validation set. FRACAL reduces the calibration error compared to the baseline.

As Table 11 shows, FRACAL reduces the calibration error by 1.9 $LaECE_0$ points and by 4.7 $LaACE_0$ points, compared to the baseline. This shows that FRACAL logit calibration does not produce unreasonable class confidence estimates, in contrast, it enhances the confidence estimates for the rare classes making FRACAL a suitable logit adjustment method for long-tailed object detection.

11. Visualisations

11.1. Visualisation of the Φ distribution

As shown in the Figure 7, after applying our method, the detected objects have larger Φ values than the baseline, especially for the rare classes, which means that the detector makes more spatially balanced detections than the baseline.

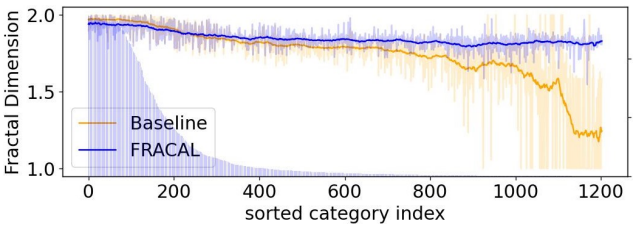

Figure 7. Φ distributions of the detected objects.

11.2. Detections using FRACAL

We provide more visualisations of our method in Figure 8. In (a) the ground-truth $p(y, u)$ of LVIS validation set is displayed using a grid of 64 by 64. In (b) the Baseline $\tilde{p}_b(y, u)$ is shown, with an one example prediction in (c) using a detection threshold of 0.01. In (d) our FRACAL $\tilde{p}_f(y, u)$ is shown using the same detection threshold and finally in (e) one example prediction of FRACAL is shown. In these subfigures, the name of the y-axis denotes the class name and the name of the x-axis indicates the average euclidean distance D between the coordinates of the predictions and the center point in image space. This distance measure shows how spread-out are the predictions and a larger distance indicates larger spatial uniformity because more predictions are further away from the image center. As the Figure suggests, our FRACAL predictions in (d) have larger distance D from the image center compared to the baseline predictions in (b), highlighting that FRACAL is more spatially balanced than the baseline. Furthermore, FRACAL makes more rare class-predictions as shown in (d) than the baseline predictions shown in (b), and correctly retrieves the rare classes *scarecrow*, *sobrero*, *crow*, *gargoyle* and *heron* in the (1-e), (2-e), (3-e), (4-e) and (5-e) respectively, in contrast to the baseline that fails to detect them.

12. FRACAL computational cost

FRACAL needs 28 seconds to compute the fractal dimension of all objects in LVIS dataset, using multiprocessing with 24 CPU cores. In comparison, LVIS inference costs 90 minutes, using 4-V100, thus our method only accounts for 0.5% of the total computation.

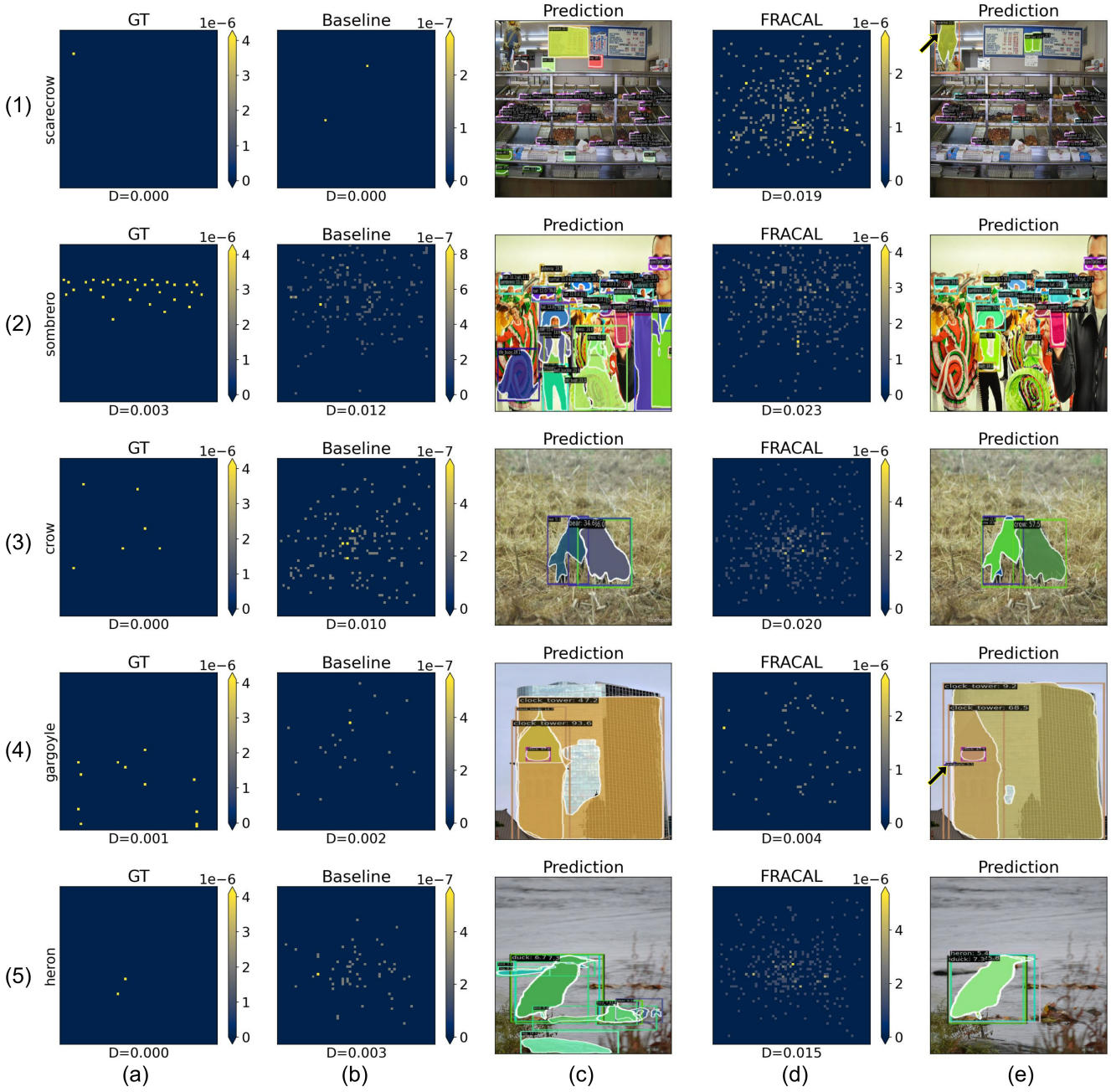

Figure 8. Visualisations on LVIS validation set using MaskRCNN ResNet 50 and mmdetection framework. It is recommended to zoom in for better visualisation of the detections. Our FRACAL, makes more spatially balanced predictions indicated by the larger distance D, and it detects more rare classes compared to the baseline.