Fractal Calibration for long-tailed object detection

Konstantinos Panagiotis Alexandridis[1], Ismail Elezi[1], Jiankang Deng[2], Anh Nguyen[3] and Shan Luo[4]
[1]Huawei Noah's Ark Lab, [2]Imperial College London, [3]University of Liverpool, [4]King's College London

Summary

1. FRACAL is a space-aware logit adjustment method for imbalanced object detection;
2. It uses the fractal dimension to capture the occurrence patterns of classes in the training set, and fuses this information during testing via logit adjustment;
3. FRACAL surpasses the SOTA in long-tailed detection benchmarks and shows good performance in balanced benchmarks, without requiring training;

Motivation

In imbalanced classification, one can calibrate the logits $z_y \in \mathbf{R}^{\mathbf{C}}$ of a model, to achieve balanced test performance. This process is called logit adjustment and it is defined as:

$$C(z_y) = z_y + \log(p_t(y)) - \log(p_s(y)), \tag{1}$$

where $p_s(y)$, $p_t(y)$ are the class priors on train and test sets respectively.

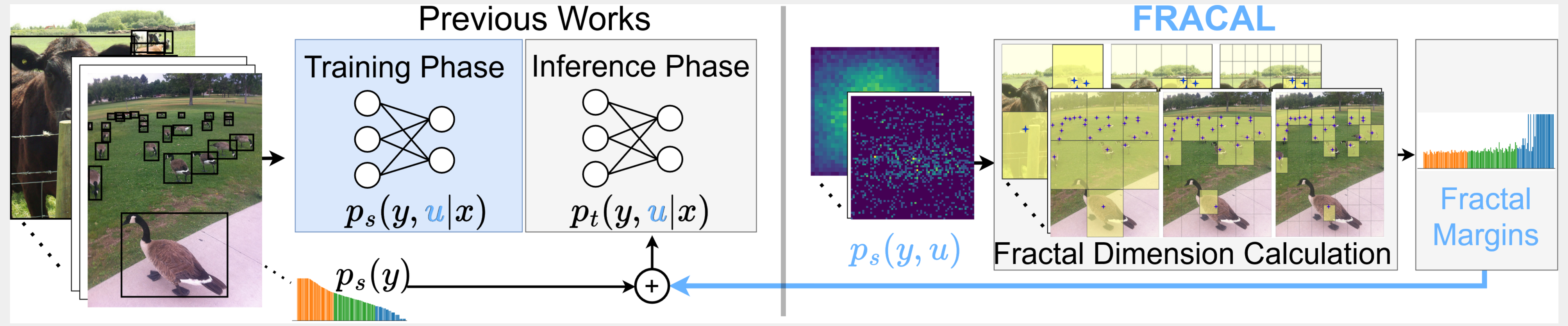

Figure 1: FRACAL embeds space-aware class information in the classification logits.

Eq.1 performs well in classification, however it does not consider the class-location relationship $p(y, u)$, which contains crucial information in object detection. To alleviate this, we propose FRACAL, a space-aware logit adjustment method, that embeds $p(y, u)$ information during inference via the fractal dimension, as shown in Figure 1.

Pipeline

FRACAL is applied during inference, in the classification logits of the detector. It makes spatially balanced detections for both the frequent classes like *hat* and rare classes like *tiara*.

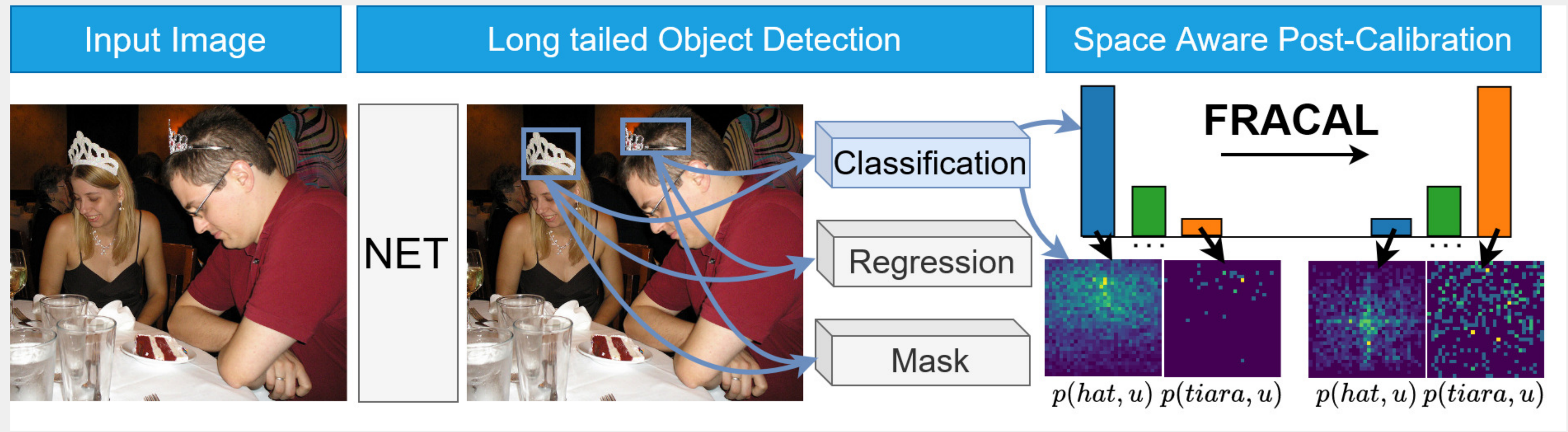

Figure 2: FRACAL pipeline.

Method

To compute $p(y, u)$ we could count the class occurrences $n_y(\mathbf{u})$ of object locations that fall inside the cell $\mathbf{u} = [i, j]$, as shown below. This grid-based method, however, makes $p(y, \mathbf{u})$ sparse and noisy, especially for large grid sizes G as shown in Figure 3-a.

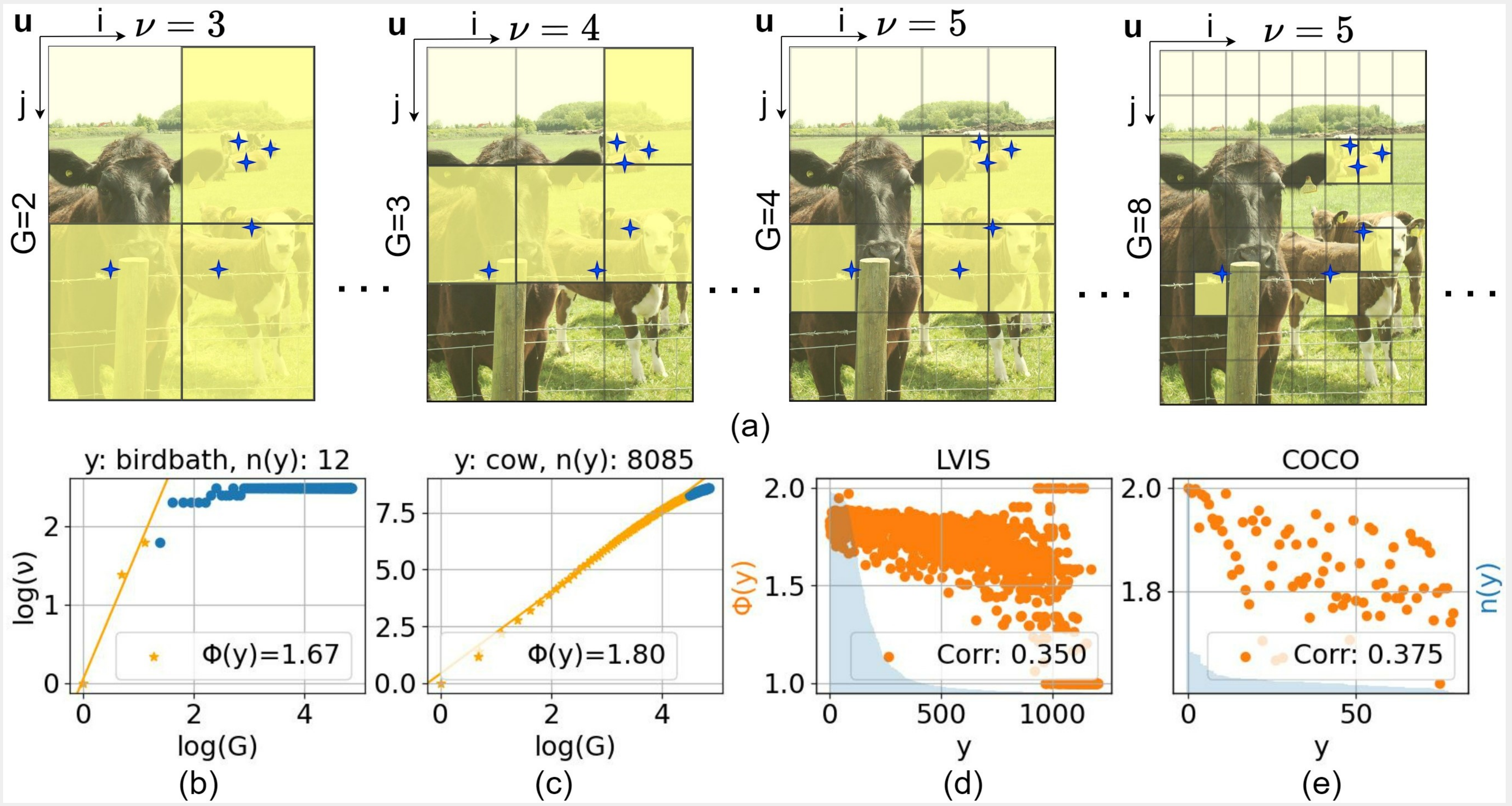

Figure 3: We iteratively count ν_y for various grid-sizes G and do a linear interpolation to find Φ.

To alleviate this, we propose the fractal dimension Φ, which is a grid-independent metric and it can capture $p(y, u)$ more robustly:

$$\Phi(y) = \lim_{G \to \infty} \frac{\log(\nu_y)}{\log(G)}, \text{ where } \nu_y = \sum_{j=0}^{G-1} \sum_{i=0}^{G-1} \mathbb{1}(n_y(\mathbf{u})) \tag{2}$$

To calculate Eq.2, we compute $\nu_y - G$ pairs up to a threshold t and fit a line to these pairs as shown in Fig. 3(b) and Fig. 3(c). The slope of this line, approximates $\Phi(y)$. To select G we use the rule $G \leq t = \lfloor \sqrt{n_y} \rfloor$, which makes the calculation tractable.
We calculate Φ for all classes in the trainset and we fuse it into the model's predictions as:

$$\mathrm{S}(z_y) = \begin{cases} \frac{\sigma(z_y)}{\Phi(y)^\lambda}, & y \in \{1, ..., C\} \\ \sigma(z_y), & y = \text{background}, \end{cases} \tag{3}$$

where $\sigma()$ is the softmax function and λ a hyperparameter. This encourages balanced predictions and removes spatial bias. Finally, FRACAL is combined with Eq.1 as follows:

$$\text{FRACAL}(z_y) = \frac{\mathrm{S}(\mathrm{C}(z_y))}{\sum_{j=1}^{C+1} \mathrm{S}(\mathrm{C}(z_j))}. \tag{4}$$

Results

Results on LVIS dataset, using ResNets, Swin and Sigmoid based detectors. FRACAL outperforms the SOTA as shown in Tables 1 (a-d).

Method	AP^m
MRCNN	25.7
ECM	27.4
LogN	27.5
FRACAL	**28.6**

(1-a) w/ R50

Method	AP^m
MRCNN	27.0
ROG	28.8
LogN	29.0
FRACAL	**29.8**

(1-b) w/ R101

Method	AP^m
MRCNN-S	30.9
FRACAL-S	**33.6**
MRCNN-B	36.6
FRACAL-B	**38.5**

(1-c) w/ Swin

Method	AP^b
ATSS	25.3
w/ FRACAL	**26.7**
GFLv2	27.4
w/ FRACAL	**28.9**

(1-d) Sigmoid-based

Ablations using FRACAL with MaskRCNN-Resnet50. FRACAL increases the logit adjustment performance with both random samplers, in (2-a), and oversampling, in (2-b), and it outperforms the Grid based adjustment in (2-c).

C	S	AP^m	AP^m_r
		22.8	8.2
	✓	25.6	13.7
✓		26.3	16.5
✓	✓	**27.3**	**19.0**

(2-a) Random Sampler

C	S	AP^m	AP^m_r
		25.7	15.8
	✓	27.7	20.7
✓		28.0	22.4
✓	✓	**28.6**	**23.0**

(2-b) RFS

Method	AP^m	AP^m_r
G=1	28.0	22.4
G=2	27.1	17.5
G=4	25.0	10.5
ours	**28.6**	**23.0**

(2-c) Grid method

FRACAL generalises to other tasks like COCO, V3DET and OpenImages, using ResNet50.

Method	AP^m	AP^b
MRCNN	35.4	39.4
w/ FRACAL	**35.8**	**39.9**

(3-a) COCO

Method	AP^b
APA	29.9
w/ FRACAL	**30.3**

(3-b) V3DET

Method	top-1
CRCNN	65.8
w/ FRACAL	**67.5**

(3-c) OpenImages

FRACAL increases the fractal dimension of the distribution of the detections, in Figure 4. It reduces the calibration error and the misclassification error as shown in 4-a and 4-b respectively, however, it also increases the false positives.

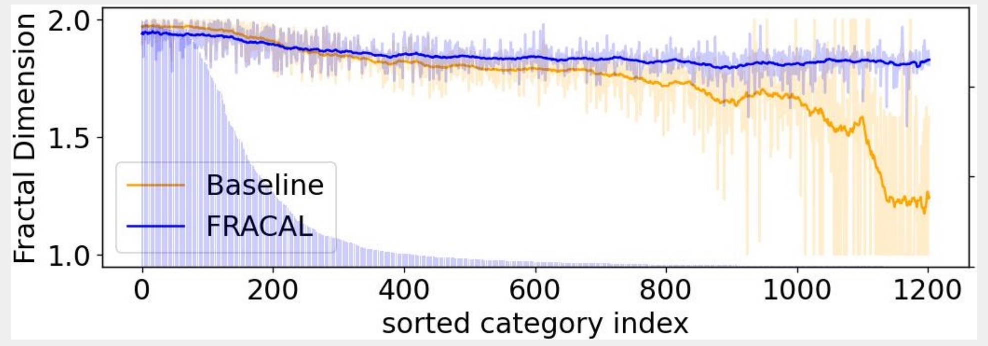

Figure 4: Φ values of class detections.

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

(4-a) Calibration Results

Method	dAP^b_{Cls}(↓)	dAP^b_{Bkg}(↓)
Baseline	31.76	**6.82**
FRACAL (ours)	**16.91**	2.84

(4-b) Detection Errors

Computational cost: The weights of FRACAL need only 28 seconds to be computed, in LVIS dataset, they need to be computed only once, and they are used only during inference.

Contact: alex.kostas@gmail.com