Evaluating Large-Vocabulary Object Detectors: The Devil is in the Details

Achal Dave[1,2] Piotr Dollár[2] Deva Ramanan[1] Alexander Kirillov[2] Ross Girshick[2]

[1]Carnegie Mellon University [2]Facebook AI Research (FAIR)

Abstract

By design, average precision (AP) for object detection aims to treat all classes independently: AP is computed independently *per category and averaged. On the one hand, this is desirable as it treats all classes, rare to frequent, equally. On the other hand, it ignores cross-category confidence calibration, a key property in real-world use cases. Unfortunately, we find that on imbalanced,* large-vocabulary datasets*, the default implementation of AP is neither category independent, nor does it directly reward properly calibrated detectors. In fact, we show that the default implementation produces a* gameable *metric, where a simple, nonsensical re-ranking policy can improve AP by a large margin. To address these limitations, we introduce two complementary metrics. First, we present a simple fix to the default AP implementation, ensuring that it is truly independent across categories as originally intended. We benchmark recent advances in large-vocabulary detection and find that many reported gains do not translate to improvements under our new per-class independent evaluation, suggesting recent improvements may arise from difficult to interpret changes to cross-category rankings. Given the importance of reliably benchmarking cross-category rankings, we consider a pooled version of AP (AP^{Pool}) that rewards properly calibrated detectors by directly comparing cross-category rankings. Finally, we revisit classical approaches for calibration and find that explicitly calibrating detectors improves state-of-the-art on AP^{Pool} by 1.7 points.*

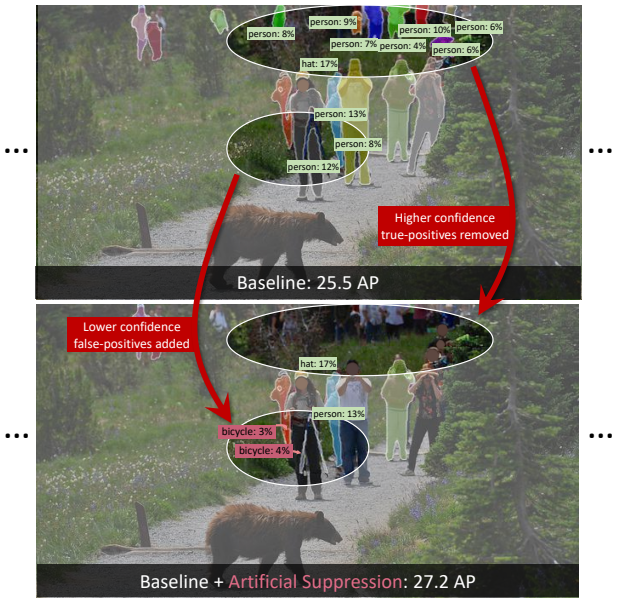


Figure 1. **The standard object detection average precision (AP) implementation can be gamed by a nonsensical re-ranking strategy.** Top: A detector normally outputs its top-k detections per image. Bottom: We discover that due to subtle implementation details in how AP is computed in practice, we can increase AP substantially by undesirably removing some accurate detections for frequent classes (*e.g.*, 'person') while promoting lower-confidence detections for rare classes, resulting in new false positives (*e.g.*, 'bicycle'). We analyze why this happens, how to fix it, and explore the consequences of the proposed solution.

1. Introduction

The task of object detection is commonly benchmarked using the mean of a per-category performance metric, usually average precision (AP) [4, 18]. This evaluation methodology is designed to treat all categories *independently*: the AP for each category is determined by the ranked list of detections for that category and is not influenced by the other categories. On the one-hand, this is a desirable property as it treats all classes, rare to frequent, equally. On the other hand, cross-category score calibration, a crucial property in real-world use cases, is ignored.

Surprisingly, in practice object detection benchmarking diverges from the goal of category-independent evaluation. Cross-category interactions enter into evaluation due to a seemingly innocuous implementation detail: the number of detections per image, across all categories, is limited to make evaluation tractable [18, 6]. If a detector would exceed this limit, then a policy must be chosen to reduce its output. The commonly used policy ranks all detections in an image by confidence and retains the top-scoring ones, up

to the limit. This policy naturally outputs the detections that are most likely to be correct according to the model.

However, this natural policy is not necessarily the best policy given the objective of maximizing AP. We will demonstrate a counter-intuitive result: there exists a policy, which can achieve higher AP, that discards a well-chosen set of higher-confidence detections in favor of promoting lower-confidence detections; see Figure 1. We first derive this result using a simple toy example with a perfectly calibrated detector. Then, we show that given a real-world detection model, we can employ this new ranking policy to improve AP on the LVIS dataset [6] by a non-trivial margin. This policy is unnatural because it directly contradicts the model's confidence estimates (even when they are perfectly calibrated) and thus indicates that AP, as implemented in practice, can be vulnerable to gaming-by-re-ranking.

This analysis reveals that the default AP implementation neither achieves the goal of being independent per class nor, to the extent that it involves cross-category interactions, does it measure cross-category score calibration with a principled methodology. Further, the metric can be gamed. To address these limitations, first we fix AP to make it truly independent per class, and second, given the practical importance of calibration, we consider a complementary metric, AP^{Pool}, that directly measures cross-category ranking.

Our fix to the standard AP implementation removes the detections-per-image limit and replaces it with a per-class limit over the entire evaluation set. This simple modification leads to tractable, class-independent evaluation. We examine how recent advances on LVIS fare under the new evaluation by benchmarking recently proposed loss functions, classifier head modifications, data sampling strategies, network backbones, and classifier retraining schemes. Surprisingly, we find that many gains in AP stemming from these advances do not translate into improvements for the proposed category-independent AP evaluation. This finding shows that the standard AP is sensitive to changes in cross-category ranking. However, this behavior is an unintentional side-effect of the detection-per-image limit, making it difficult to characterize, and it may not reliably measure improvements given its vulnerability to gaming.

To enable reliable benchmarking, we propose to directly measure improvements to cross-category ranking with a complementary metric, AP^{Pool}. AP^{Pool} pools detections from all classes and computes a single precision-recall curve—the detection equivalent of micro-averaging from the information retrieval community [20]. To optimize AP^{Pool}, true positives for *all* classes must rank ahead of false positives for *any* class, making it a principled measure of cross-category ranking. We extend simple score calibration approaches to work for large-vocabulary object detection and demonstrate significant AP^{Pool} improvements that result in state-of-the art performance.

2. Related work

Large-vocabulary detection. Object detection research has largely focused on small-to-medium vocabularies (*e.g.*, 20 [4] to 80 [18] classes), though notable exceptions exist [2, 10]. Recent detection benchmarks with hundreds [34, 15] to over one-thousand classes [6] have renewed interest in large-vocabulary detection. Most approaches re-purpose models originally designed for small vocabularies, with modifications aimed at class imbalance. Over-sampling images with rare classes to mimic a balanced dataset [6] is simple and effective. Another strategy leverages advances from the long-tail *classification* literature, including classifier retraining [12, 33] and using a normalized classifier [19, 29]. Finally, recent work proposes several new loss functions to reduce the penalty for predicting rare classes, *e.g.*, equalization loss (EQL) [19], balanced group softmax (BaGS) [16] or the CenterNet2 Federated loss [35]. We analyze these advances in large-vocabulary detection, finding that a number of them do not show improvements under our fixed, independent per-class AP evaluation, indicating that they improve existing AP by modifying cross-category rankings.

Detection evaluation. Average precision (AP) is the most common object detection metric, used by PASCAL [4], COCO [18], OpenImages [15], and LVIS [6]. Conceptually, AP evaluates detectors independently for each class. We show that common implementations deviate from this conceptual goal in important ways, and propose potential fixes and alternatives. Prior work analyzing AP focuses on comparisons across classes, *e.g.* Hoiem *et al*. [9] present a normalized average precision (AP_N) and Zhang *et al*. [33] propose 'sampled AP', but does not expose the issues covered in this paper. Our procedural fix for AP computation removes the impact of cross-category scores on evaluation, and thus we propose a variant, AP^{Pool}, which explicitly rewards better cross-category rankings. From an information retrieval perspective, AP^{Pool} is the micro-averaging counterpart to AP [20], which evaluates macro-averaged performance, and has been used as a diagnostic in prior work [3].

Model calibration. A well-calibrated model is one that provides accurate probabilistic confidence estimates. Calibration has been explored extensively in the classification setting, including parametric approaches, such as Platt scaling [23] and beta calibration [13], and non-parametric approaches, such as histogram binning [31], isotonic regression [32], bayesian binning into quantiles (BBQ) [21]. While small neural networks tend to be well-calibrated [22], Guo *et al*. [5] show that deep networks are heavily uncalibrated. Kuppers *et al*. [14] extend this analysis to deep network based object detectors and show that size and position of predicted boxes helps reduce calibration error. We also apply calibration strategies to object detectors, but find that *per-class* calibration is crucial for improving AP^{Pool}.

3. Pitfalls of AP on large-vocabulary detection

Through both toy and real-world examples, we show that cross-category prediction scores impact AP in subtle, counter-intuitive ways.

3.1. Background

The standard object detection evaluation aims to evaluate each class independently. In practice, however, this independence is broken due to an apparently harmless implementation detail: to evaluate efficiently, benchmarks limit the number of detections a model can output per image (*e.g.* to 100). In practice, this limit is set (hopefully) to be high enough that detections beyond it are unlikely to be correct. Importantly, this limit is shared across all classes, implicitly requiring models to rank predictions *across* classes.

Our analysis suggests that for long-tailed, large-vocabulary datasets this detections-per-image limit, when used with a class-balanced evaluation like AP, enables an undesirable ranking policy to perform better than the natural policy of ranking detections by their estimated confidence.

3.2. Analysis

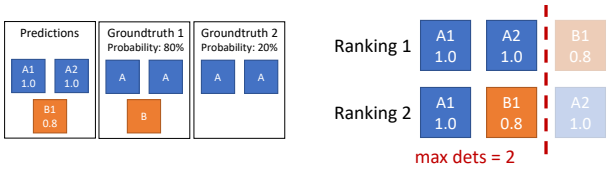
A toy example. Consider a toy evaluation on a dataset with 2 classes, as shown in Figure 2. For simplicity, suppose we have access to a detector that is perfectly calibrated: when the model outputs a prediction with confidence s (*e.g.* 0.3), the prediction is a true positive $100 \cdot s\%$ (*e.g.* 30%) of the time. We consider evaluating this model's outputs under two different rankings, using the standard class-balanced AP evaluation with a limit of 2 detections per image.

Under this setting, consider the predictions w.r.t. two possible groundtruth scenarios in Figure 2a. The model predicts two instances for class A (A1, A2) with confidence 1.0, and one instance for class B (B1), with confidence 0.8. Since the model is perfectly calibrated (by assumption), we know A1 and A2 are true positives 100% of the time, while B1 is a true positive 80% of the time.

With these predictions, consider the two potential rankings depicted in Figure 2b. Ranking 1 appears ideal: it ranks more confident detections before lower confident ones, as is standard practice. By contrast, Ranking 2 is arbitrary: B1 is ranked above A2, despite having lower confidence.

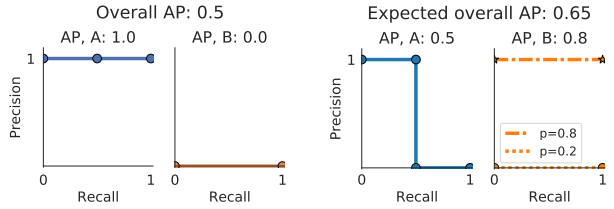
Surprisingly, Ranking 2 *outperforms* Ranking 1 under the AP metric with a limit of 2 detections per image, as shown in Figure 2c and Figure 2d. While Ranking 1 gets a perfect AP of 1.0 for class A, it gets 0 AP for class B, leading to an overall AP of 0.5. By contrast, while Ranking 2 leads to a lower AP for class A (0.5), it scores an expected AP of 0.8 for class B, yielding an overall AP of 0.65!

Of course, this is a toy scenario, concocted to highlight an evaluation pitfall using an artificially low detections-per-image limit of only two predictions. We now show that a similar effect exists for a real-world detection benchmark.



(a) Left: Predictions from a *perfectly calibrated* model: a prediction with confidence s is correct $100 \cdot s\%$ of the time. Middle, Right: the two possible groundtruth scenarios and their probabilities.

(b) Two potential rankings of the predictions. With detections-per-image limited to 2, the two rankings report different predictions (*i.e.* only those left of the dashed line).



(c) Ranking 1 precision and recall. Since A1, A2 have a precision of 1.0, AP for class A is 1.0. Class B has no predictions, so the AP is 0.0, leading to an overall AP of 0.5.

(d) Ranking 2 precision and recall. AP for A is 0.5. For B: B1 is either a true positive (AP 1.0) or not (AP 0.0). On average, this results in AP 0.8 for B, and overall AP of 0.65.

Figure 2. **Limiting detections-per-image rewards undesirable rankings.** A toy scenario showing the interplay between a class-balanced AP evaluation and a limit on the number of detections per image. A perfectly calibrated model should output 'Ranking 1' in (b) since it ranks detections that are more likely correct first. However, given a detections-per-image limit of 2, 'Ranking 2' yields a higher AP even though it ranks a detection that is more likely incorrect (B1) ahead of one that is more likely correct (A2). Note that by removing the limit, the rankings across categories become fully independent and both rankings would result in an equal overall AP for the two rankings (0.75 in expectation; not visualized).

A real-world example. The LVIS [6] dataset uses the evaluation described above, with a limit of 300 detections per image. We investigate whether an artificial ranking policy, as in Figure 2, can lead to improved AP on this dataset. Concretely, we evaluate a simple policy: we first discard all but the top k scoring detections per class across the entire evaluation dataset. Given the predictions in Figure 2a, applying this policy with $k = 1$ leads to 'Ranking 2' from Figure 2b: an arbitrary ranking which, nevertheless, leads to a higher AP than the baseline 'Ranking 1.'

This ranking policy, combined with the detections-per-image limit, is undesirable: it explicitly discards high-scoring predictions for many classes in order to fit low-scoring predictions from other classes into the detections-per-image limit, as shown by our toy example in Figure 2b and with real-world detections in Figure 1. Using a baseline Mask R-CNN model [7] (see appendix for details), we find that this strategy, with $k = 10,000$, improves LVIS AP by 1.2 points, and AP_r by 2.9 points, as shown in Table 1.

dets/class	dets/im	AP	AP_r	AP_c	AP_f
∞ ('Ranking 1')	300	22.6	12.6	21.1	28.6
10,000 ('Ranking 2')	300	**23.8** (+1.2)	15.5 (+2.9)	22.7	28.5

Table 1. **Undesirable 'Ranking 2' (Figure 2b) improves LVIS AP.** Artificially limiting the number of detections per class across the entire validation set leads to *higher* LVIS AP when using the standard limit of 300 detections per image, perhaps paradoxically. In Figure 1 we show how this ranking policy (which, again, improves AP) suppresses some higher-confidence detections in favor detections that the model estimates are more likely incorrect.

	LVIS				COCO
dets/**im**	AP	AP_r	AP_c	AP_f	AP
100	18.2	6.5	15.8	26.1	37.4
300	22.6	12.6	21.1	28.6	37.5 (+0.1)
1,000	25.0 (+2.4)	16.8 (+4.2)	24.1	29.7	37.5 (+0.1)
2,000	25.6 (+3.0)	18.1 (+5.5)	24.6	29.9	37.5 (+0.1)
5,000	26.0 (+3.4)	19.7 (+7.1)	24.9	30.0	37.5 (+0.1)
10,000	26.1 (**+3.5**)	19.8 (**+7.2**)	25.0	30.1	37.5 (**+0.1**)

Table 2. **Increasing the limit on detections per image *significantly* improves LVIS AP.** AP_r improves by over 7 points (over 50% relative), indicating many accurate rare class predictions are ignored due to the default limit of 300 detections per image. By contrast, this limit does not significantly impact COCO, which contains a significantly smaller vocabulary.

Note that this results purely from a modified *ranking policy*, without any changes to the evaluation or model. This non-trivial improvement is roughly the magnitude achieved by a typical new method published at CVPR (*e.g.* [27, 16]).

The relatively larger improvement to AP_r suggests that under the standard confidence-based ranking, accurate predictions for rare classes are crowded out by frequent class predictions due to the detections-per-image limit.

Although this limit appears high (at 300 detections-per-image), LVIS contains over a thousand object classes: even outputting a single prediction for each class is impossible under the limit. The assumption is that detections beyond the first 300 are likely to be false positives. Table 2 verifies that this assumption is incorrect: increasing the limit on detections per image leads to significantly higher results on the LVIS dataset. In particular, the AP for rare categories improves *drastically* from 12.6 to 19.5 with a higher limit.

Is this an issue for COCO? Given the impact of the detections-per-image limit on LVIS evaluation, a natural question is whether this also affects the widely used COCO dataset. Table 2 shows that increasing this limit does not significantly change COCO AP for a baseline model. We hypothesize that this is likely due to the significantly smaller vocabulary in the COCO dataset relative to the detections per image limit: with only 80 classes, detections beyond the top 100 per image are unlikely to meaningfully impact AP. Fortunately, this suggests that the limit on detections per image has not negatively impacted COCO evaluation.

dets/**class**	dets/**im**	AP	AP_r	AP_c	AP_f
1,000	∞	21.9	17.7	22.2	23.5
5,000	∞	25.0 (+3.1)	19.5 (+1.8)	24.4	28.2
10,000	∞	25.6 (+3.7)	19.7 (+2.0)	24.7	29.1
30,000	∞	26.0 (**+4.1**)	19.8 (+2.1)	24.9	29.8
50,000	∞	26.0 (**+4.1**)	19.9 (**+2.2**)	25.0	30.0

Table 3. **LVIS AP evaluation with varying limits on the number of detection/class, with no limit on detections/image.** Increasing the limit beyond 10,000 does not significantly affect AP.

4. AP without cross-category dependence

To address this interaction between LVIS AP and cross-category scores, we consider two alternative implementations of average precision.

Higher detections-per-image limit. Our first option is to reduce the impact of this issue by increasing the detections-per-image limit. At a high enough limit, predictions outside the limit are exceedingly unlikely to be correct, and thus unlikely to affect the evaluation. Note that this is, implicitly, the strategy used by COCO. But what should this limit be? To answer this, we can analyze AP at varying limits, and choose the limit beyond which AP does not significantly change. Table 2 suggests that increasing the limit beyond 5,000 detections per image does not significantly affect AP. Unfortunately, increasing the detections-per-image limit leads to practical challenges for evaluation. At 5,000 detections per image, the outputs of a baseline model on the LVIS validation set are over $15\times$ larger than at the default limit of 300 detections (37GB *vs*. 2.4GB), leading to a prohibitively slower evaluation. Moreover, the LVIS test sets can only be evaluated by submitting results to an evaluation server and such an increase in file size is likely impractical.

Limit detections-per-class. We now present an alternative, tractable implementation. One natural strategy might be to limit detections per-class, per-image. But even a modest limit of, say, 10 detections results in an intractable number of detections per image (10,000+) for LVIS-sized vocabularies. Instead, we remove the limit on detections-per-image entirely, and limit the number of detections per class across the whole dataset. Specifically, models are limited to k predictions for each of the n classes across the entire evaluation set. The model can choose to output all of these predictions on a single image, or spread them over the entire dataset.

This evaluation appears similar to the undesirable 'Ranking 2' in Figure 2. However, while 'Ranking 2' is an undesirable *strategy* for resolving competition across classes, our evaluation *removes* this competition altogether by providing an independent detection budget for each class.

Similar to before, we desire a value of k beyond which AP does not meaningfully change. Table 3 shows that increasing the limit beyond 10,000 does not significantly affect AP. This limit does increase the file size and evaluation

speed, but only by a factor of 2×, keeping evaluation time tractable. In principle, this limit should be viewed as depending on the evaluation set size. For practical purposes, the validation and test sets in LVIS all contain 20,000 images and thus a single limit suffices.

We note that this evaluation also has a natural appeal when viewing detection as an information retrieval task, the field of study from which AP evaluation originates: the detector is allowed to 'retrieve' up to k detections (or 'documents') for each class from the entire evaluation set (or 'corpus'). In practice, various strategies exist for efficiently selecting the top k detections over a large set of images.

We recommend limiting detections per class, with no limit per image. In the remainder of the paper we refer to the version of LVIS evaluation that uses a detections-per-image limit as 'AP$^{\text{Old}}$' and our new, recommended version that limits detections per class as 'AP$^{\text{Fixed}}$'.

5. Impact on long-tailed detector advances

We have shown that the current AP evaluation introduces subtle, undesirable interactions with cross-category rankings due to the detections-per-image limit. However, it remains unclear to what extent this issue meaningfully affects prior conclusions drawn on LVIS. To analyze this, we evaluate the importance of different design choices in LVIS detectors with the original evaluation ('AP$^{\text{Old}}$'), with a limit of 300 predictions per image, and our modified evaluation ('AP$^{\text{Fixed}}$'), with a limit of 10,000 detections per class across the whole evaluation set (with no per-image limit).

5.1. Experimental setup

The following experiments use Mask R-CNN [7]. Unless noted differently: the backbone is ResNet-50 [8] with FPN [17], pre-trained on ImageNet [26], and fine-tuned on LVIS v1 [6] with repeat factor sampling for 180k iterations using a minibatch size of 16 images with a 0.1× learning rate decay applied after 120k and 160k iterations. The learning rate starts at 0.02 and a weight decay of 1e−4 is used. Batch norm [11] parameters are frozen (*i.e.* not updated). Results are reported on the LVIS v1 validation set using the mean of three runs with different random seeds.

5.2. Case studies

Loss functions. As discussed in Section 2, a number of new losses have been proposed in the past year. We analyze three in particular: EQL [27], BaGS [16], and a 'Federated' loss [35]. Table 4a (first column) shows that, under the original evaluation, the choice of loss function can robustly improve the AP of a baseline model by up to 2.4 points, from 22.3 using softmax cross-entropy (CE) to 24.7 using the Federated loss. These gains suggests the choice of loss function is important. However, under our 'AP$^{\text{Fixed}}$', the losses are more similar, differing by at most 0.8 points.

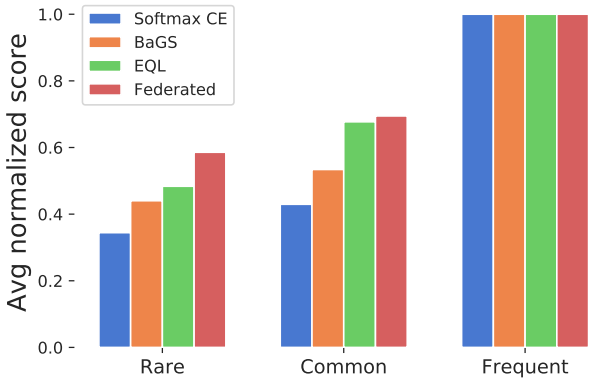


Figure 3. **Score distribution induced by different loss functions when bucketing categories into LVIS rare, common, and frequent groups.** Compared to the baseline softmax CE loss, BaGS, EQL, and Federated losses tilt the distribution, making it more uniform. This indicates an improved calibration across categories.

To gain insight into why the losses improve AP$^{\text{Old}}$, but have less impact on AP$^{\text{Fixed}}$, we plot the score distribution for the LVIS rare, common, and frequent category subsets (normalized so the average score for frequent categories is 1.0). Figure 3 shows that the EQL, BaGS, and Federated losses tilt the distribution relative to softmax CE loss, making it more uniform. This tilt boosts the confidence of rare category detections, making them more likely to appear in the 300 detections-per-image limit. This observation suggests that these losses produce better cross-category rankings compared to softmax CE loss and that AP$^{\text{Old}}$ rewards this calibration. Because AP$^{\text{Fixed}}$ is category independent, it does not reward improved cross-category rankings.

Classifier heads. Next, we evaluate two commonly used modifications to the linear classifier in object detectors in Table 4b. The first modification trains a linear *objectness* binary classifier in parallel to the K-way classifier [16, 25, 29], denoted 'Obj'. The second modification L2-normalizes the input features and weights of the classifier during both training and inference [28, 19, 29], denoted 'Norm.' See the appendix for implementation details.

The first block in Table 4b shows that while adding an objectness predictor modestly improves AP$^{\text{Old}}$ (+0.9), it results in a slightly lower AP$^{\text{Fixed}}$ (−0.3). This discrepancy suggests the objectness predictor optimizes the ranking of predictions across classes, but doesn't meaningfully improve the quality of the detections. On the other hand, using a normalized classifier consistently leads to higher accuracy under both AP$^{\text{Old}}$ and AP$^{\text{Fixed}}$ (+0.9 in both cases). Finally, we find that applying both these modifications to the classifier results in a strong baseline under both AP$^{\text{Old}}$ and AP$^{\text{Fixed}}$. The second block in Table 4b further shows that under AP$^{\text{Fixed}}$, the choice of loss function is largely irrelevant when both of these classifier modifications are used.

Loss	AP^{Old}	AP^{Fixed}
Softmax CE	22.3	25.5
Sigmoid BCE	22.5 (+0.2)	25.6 (+0.1)
EQL [27]	24.0 (+1.7)	26.1 (+0.6)
Federated [35]	**24.7 (+2.4)**	**26.3 (+0.8)**
BaGS [16]	24.5 (+2.2)	25.8 (+0.3)

(a) **Loss functions.** Choosing the right loss is more important under AP^{Old}, providing an improvement of up to +2.4 AP. Under our proposed AP^{Fixed}, the impact of losses is reduced, to at most +0.8 AP. This result indicates that these loss functions may primarily improve cross-category rankings (also see Figure 3).

Loss	Obj	Norm	AP^{Old}	AP^{Fixed}
Softmax CE	✗	✗	22.3	25.5
	✓	✗	23.2 (+0.9)	25.3 (−0.2)
	✗	✓	23.2 (+0.9)	**26.3 (+0.8)**
	✓	✓	24.4 (+2.1)	26.3 (+0.8)
Sigmoid BCE	✓	✓	24.2 (−0.2)	26.3 (+0.0)
EQL [27]	✓	✓	24.7 (+0.3)	26.1 (−0.2)
Federated [35]	✓	✓	**25.1 (+0.7)**	26.3 (+0.0)
BaGS [16]	✓	✓	**25.1 (+0.7)**	26.2 (−0.1)

(b) **Classifier modifications.** We evaluate two ideas commonly used for improving long-tail detection: an objectness predictor ('Obj') [16], and L2-normalizing both the linear classifier weights and input features ('Norm'). Once again, we find that these components improve the baseline significantly under the AP^{Old}, but provide minor improvements under our AP^{Fixed}. Nevertheless, our results indicate these components provide a strong, simple baseline that erases the impact of the training loss choice.

Sampler	AP^{Old}	AP^{Fixed}
Uniform	18.4	22.8
CAS	19.2 (+0.8)	21.5 (−1.3)
RFS	**22.3 (+3.9)**	**25.5 (+2.7)**

(c) **Samplers.** Category Aware Sampling (CAS) and Repeat Factor Sampling (RFS) are common sampling strategies for addressing class imbalance. While both strategies outperform the uniform sampling baseline under AP^{Old}, only RFS provides significant improvements under AP^{Fixed}.

Phase 1	Phase 2	AP^{Old}	AP^{Fixed}
RFS	-	22.3	**25.5**
Uniform	RFS	21.6 (−0.7)	24.9 (−0.6)
Uniform	CAS	23.1 (+0.8)	24.9 (−0.6)
RFS	CAS	**23.6 (+1.3)**	**25.6 (+0.1)**

(d) **Classifier retraining.** We evaluate the efficacy of training detectors in two phases, a common technique [12, 29]. Phase 1: the model is trained end-to-end with one sampler. Phase 2: only the final classification layer is trained, using a different sampler. This strategy improves AP^{Old}, but not AP^{Fixed}, suggesting that classifier retraining may primarily improve cross-category rankings.

Backbone	AP^{Old}	AP^{Fixed}
ResNet-50	22.3	25.5
ResNet-101	24.6 (+2.3)	27.7 (+2.2)
ResNeXt-101	**26.2 (+3.9)**	**28.7 (+3.2)**

(e) **Stronger backbones.** Using larger backbones consistently improves the detector under both AP^{Old} and AP^{Fixed}, indicating, as one might expect, that larger backbones improve overall detection quality and not just cross-category rankings. ResNeXt-101 uses the 32x8d configuration.

Table 4. **Impact of various design choices on the LVIS v1 validation dataset, comparing AP^{Old} to AP^{Fixed}.** Unless specified otherwise, each experiment uses a ResNet-50 FPN Mask R-CNN model trained with Repeat Factor Sampling (RFS) for 180k iterations with 16 images per batch. All numbers are the average of three runs with different random seeds and initializations.

Sampling strategies. Modifying the image sampling strategy is a common approach for addressing class imbalance in LVIS. Table 4c analyzes three standard sampling strategies: Uniform, which samples images uniformly at random; Class Aware Sampling (CAS), which first samples a category and then an image containing that category; and Repeat Factor Sampling (RFS) [6], which oversamples images containing rare classes. RFS consistently performs the others by a significant margin under both AP^{Old} and AP^{Fixed}. Surprisingly, while CAS outperforms uniform sampling under AP^{Old}, it hurts accuracy under AP^{Fixed}, suggesting that CAS improves primarily due to how it ranks predictions across classes.

Classifier retraining. A common alternative to training with a single sampler is to train the model end-to-end using one sampler, and fine-tune the linear classifier with a different sampler [12, 33]. Under AP^{Old}, carefully choosing the samplers for these phases appears important, improving by +1.3 AP. However, under AP^{Fixed}, this improvement disappears, indicating that on LVIS, classifier retraining primarily improves by aligning scores across classes.

Stronger backbones. Finally, we evaluate the improvements due to stronger backbone architectures. We evaluate four progressively stronger models: ResNet-50, ResNet-101 [8], and ResNeXt-101 32x8d [30]. Unlike many other LVIS-specific design choices, we find that the choice of a larger backbone consistently improves accuracy for both AP^{Old} and AP^{Fixed}.

5.3. Discussion: something gained, something lost

AP^{Fixed} makes AP evaluation category independent by design. As a result, it is no longer vulnerable to gaming-by-re-ranking, as we demonstrate is possible with AP^{Old} in Section 3. However, by benchmarking several recent advances in long-tailed object detection we observe evidence that several of the improvements may be due to better cross-category rankings, because the improvements that were observed with AP^{Old} largely disappear when evaluated with AP^{Fixed}. Have we lost something by making the new evaluation invariant to calibration (*i.e.* per-category, monotonic score transformations do not change AP^{Fixed})?

Arguably, yes. Albeit in a flawed manner, AP^{Old} was

dets/im	AP^{Pool}	AP^{Pool}_r	AP^{Pool}_c	AP^{Pool}_f
300	26.2	8.0	16.7	27.0
1,000	26.8 (+0.6)	10.6 (+2.6)	19.8	27.6
2,000	27.0 **(+0.8)**	11.0 (+3.0)	20.5	27.7
5,000	27.0 **(+0.8)**	11.3 **(+3.3)**	20.8	27.7
10,000	27.0 **(+0.8)**	11.3 **(+3.3)**	20.8	27.7

Table 5. **Impact of limiting detections-per-image on AP^{Pool}.** As expected, AP^{Pool} is less sensitive to this limit than AP^{Old} because each instance is weighted equally.

Loss	AP^{Fixed}	AP^{Fixed}_r	AP^{Fixed}_c	AP^{Fixed}_f	AP^{Pool}	AP^{Pool}_r	AP^{Pool}_c	AP^{Pool}_f
Softmax CE	25.5	18.9	24.9	29.1	25.6	11.5	20.5	26.2
Sigmoid BCE	25.6 (+0.1)	19.4	24.9	28.9	25.6 (+0.0)	10.8	20.1	26.1
EQL [27]	26.1 (+0.6)	19.9	26.1	28.9	25.9 (+0.3)	11.3	22.9	26.3
Federated [35]	**26.3 (+0.8)**	20.7	24.9	30.2	**27.8 (+2.2)**	16.1	22.0	28.2
BaGS [16]	25.8 (+0.3)	17.9	25.6	29.5	26.0 (+0.4)	9.1	20.8	26.4

Table 6. **AP^{Fixed} and AP^{Pool} for models trained with varying losses.** Federated significantly outperforms others under AP^{Pool}.

sensitive to an important property of object detectors which is essential in real-world applications: score calibration. In the simplest example, one may want to produce a demo that visualizes all detections above a global score threshold (*e.g.* 0.5) and expect to see consistent results across all categories. Given that such practical demand exists, we consider in the next section a variant of AP, called AP^{Pool}, that directly rewards cross-category rankings, without the vulnerability to gaming displayed by AP^{Old}. Furthermore, we develop a simple detector score calibration method and show that it improves AP^{Pool}.

6. Evaluating cross-category rankings

An independent, per-class evaluation is appealing in its simplicity. Most practical applications, however, require comparing the confidence of predictions across classes to form a unified understanding of the objects in an image. As an extreme example, note that a detector can output arbitrary range of scores for each class for a truly independent evaluation: that is, all detections for one class (say, 'banana') may have confidences above 0.5, while all detections for another class (say, 'person') has confidences below 0.5. Using such a detector in practice requires carefully calibrating scores across classes – an open challenge that is not evaluated by current detection evaluations.

6.1. AP^{Pool}: A cross-category rank sensitive AP

To address this, we consider a complementary metric, AP^{Pool}, which explicitly evaluates detections across all classes together [3]. To do this, we first match predictions to groundtruth per-class, following the standard evaluation. Next, instead of computing a precision-recall (PR) curve for each class, we pool detections across all classes to generate a single PR curve across all classes, and compute the Average Precision on this curve to get AP^{Pool}.

This evaluation has two key properties. First, it ranks detections across all classes to generate a single precision-recall curve, incentivizing detectors to rank more confident predictions above lower confidence ones. Second, it weights all groundtruth instances, rather than classes, equally. This property removes a counter-intuitive effect, illustrated in Figure 2, that can occur with class averaging. Further, it reduces the impact of the detections-per-image limit, as low-confidence predictions for some rare classes do not significantly impact the evaluation. Because of this, however, the evaluation is influenced more heavily by frequent classes than rare classes. To analyze performance for rare classes, we further report three diagnostic evaluations which evaluate predictions only for classes within a specified frequency: AP^{Pool}_r (for rare classes), AP^{Pool}_c (common), and AP^{Pool}_f (frequent).

6.2. Analysis

How does the dets/im limit affect AP^{Pool}? Table 5 analyzes how the detections-per-image limit impacts AP^{Pool}. As expected, increasing this limit does not significantly affect AP^{Pool}: while AP can change drastically due to a few additional true positives for rare classes, AP^{Pool} treats true positives for all classes equally. Increasing the limit beyond 300 detections improves the diagnostic AP^{Pool}_r metric, but only mildly improves AP^{Pool} by 0.8 points. Nonetheless, for consistency, we evaluate all models with the same detections as AP^{Fixed}: the top 10,000 per class, with no limit on detections per image.

Do losses impact AP^{Pool}? Next, we analyze whether recent advances in LVIS detection lead to better models under AP^{Pool}. Table 6 compares various losses under AP^{Fixed} and AP^{Pool}. Perhaps surprisingly, while EQL and BaGS do not meaningfully impact AP^{Pool}, the Federated loss provides a 2.0 point improvement over the baseline Softmax CE loss. This improvement provides a new perspective for the Federated loss: Although the loss does not explicitly aim to calibrate models, it meaningfully improves *cross-category* ranking of predictions compared to other losses.

6.3. Calibration

We now propose a simple and effective strategy for improving AP^{Pool}. We re-purpose classic techniques for calibrating model uncertainty for the task of large-vocabulary object detection. Calibration aims to ensure that the model's confidence for a prediction corresponds to the probability that the prediction is correct. In the detection setting, if a model detects a box with confidence s, it should correctly localize a groundtruth box of the same category $s\%$ of the

Figure 4. Examples illustrating the effect of calibration. Each row shows the 20 highest-scoring predictions from the baseline, uncalibrated model (left) and its calibrated version (right). True-positives and false-positives (at IoU 0.5) are indicated with a green and red label, respectively. The calibrated model increases the rank of low-confidence but accurate predictions, such as the 'bird's (top row), 'cowboy hat's (middle), and 'person's (bottom), over incorrect predictions with artificially high scores, such as some 'boat's (top), 'horse's (middle), and 'ski's (bottom).

time [14]. While this property is not necessary for AP^{Pool}, it provides a sufficient condition for improving cross-category rankings (AP^{Pool} only requires that true positives are ranked higher than false positives across all classes, without requiring the scores to be *probabilistically* calibrated).

Following [14], we analyze various calibration strategies: histogram binning [31], Bayesian Binning into Quantiles (BBQ) [21], beta calibration [13], isotonic regression [32], and Platt scaling [23]. Prior work on calibrating detectors applies calibration strategies to predictions across all classes [14]. However, this approach does not take into account class frequency: rare classes may, for example, tend to have low-scoring predictions, while frequent class predictions are over-confident. Instead, we propose to calibrate each class individually, allowing the method to boost scores of under-confident classes and diminish scores of over-confident classes.

Standard calibration strategies require a held-out dataset to be used for training the calibration model. However, in the large-vocabulary setting, many classes have only a handful of examples in the entire dataset. We instead calibrate directly on the detector's *training* set. To understand the impact of this choice, we also report an upper-bound by calibrating on the *validation* set.

Calibration	AP^{Pool}	AP^{Pool}_r	AP^{Pool}_c	AP^{Pool}_f
Uncalibrated	27.8	**16.1**	22.0	28.2
Histogram Bin	28.6 (+0.8)	12.4	20.6	29.2
BBQ (AIC)	28.8 (+1.0)	13.6	21.6	29.3
Beta calibration	**29.5** (+1.7)	12.8	22.7	**30.0**
Isotonic reg.	28.3 (+0.5)	14.4	22.2	28.7
Platt scaling	**29.5** (+1.7)	13.1	**22.8**	**30.0**
Calibrate on validation (upper-bound oracle)				
HistBin	30.1 (+2.3)	24.4	**27.8**	30.2
BBQ (AIC)	30.0 (+2.2)	22.9	26.9	30.2
Beta calibration	29.8 (+2.0)	22.4	25.2	30.1
Isotonic reg.	**30.3** (+2.5)	**24.6**	27.2	**30.4**
Platt scaling	29.8 (+2.0)	22.2	24.9	30.1

Table 7. **Calibrating detection outputs on the train set significantly improves AP pooled.** The gains are due to improved rankings across categories. Calibrating on validation significantly improves AP^{Pool}_r, indicating calibration remains challenging in the tail. All models trained with the Federated loss.

Table 7 reports AP^{Pool} using various calibration approaches applied to a model trained with the Federated loss. The results show that calibrating per class improves AP^{Pool} by 1.7 points, from 27.8 to 29.5, and the precise choice of calibration strategy does not seem critical. Surprisingly, calibrating on the *validation* set, as in the last row, outperforms calibrating on the training set by only 0.8 points, suggesting that calibrating on the training set is a viable strategy. However, calibrating on the validation set significantly improves AP^{Pool}_r while calibrating on the training set *harms* AP^{Pool}_r, indicating that calibration for rare classes remains an open challenge. Figure 4 presents qualitative examples of this improvement: calibration increases the scores of underconfident, accurate predictions from some classes (*e.g.* 'bird' or 'cowboy hat') and suppresses overconfident predictions from other classes (*e.g.* 'horse').

7. Discussion

Robust, reliable evaluations are critical for advances in large-vocabulary detection. Our analysis reveals that current evaluations fail to properly handle cross-category interactions by neither eliminating them (as intended) nor evaluating them in a principled fashion (as potentially desired). We show that, as a result, the current AP implementation (AP^{Old}) is vulnerable to gaming. We propose AP^{Fixed}, which addresses this gameability by removing the effect of cross-category score calibration, and recommend it as a *replacement* for AP^{Old} moving forward. AP^{Fixed} provides new conclusions about the importance of different LVIS advances. Finally, we recommend a complementary *diagnostic* metric, AP^{Pool}, for applications requiring cross-category score calibration, and show that a simple calibration strategy offers off-the-self detectors solid improvements to AP^{Pool}.

Appendix

We first present additional analysis of our experiments in Section A. Section A.1 reports AP^{Pool} for all experiments in Table 4, and discusses key results. Section A.2 analyzes all variants of classifier retraining. Section A.3 analyzes losses and classifier modifications using a stronger baseline detector. Finally, Section B presents implementation details, including the experimental setup for tables in the main paper, classifier modifications, and the RegNetY-4GF model used in this appendix.

A. Additional analyses

A.1. AP^{Pool}: Long-tail detector advances

Table 8 reports AP^{Pool} for all experiments in Table 4. We highlight a few results here. Table 8a reports the same results as Table 6: while most losses do not improve AP^{Pool}, the Federated loss provides significant improvements of +2.2 points under AP^{Pool}, indicating it helps to calibrate cross-category scores. Table 8b shows that the classifier modifications described in Section 5.2 provide small improvements in AP^{Pool} (+0.7 using both modifications). Surprisingly, the Federated loss performs slightly better without these modifications under AP^{Pool} (27.8 without modifications in Table 8a *vs*. 27.6 with modifications). Even with both classifier modifications, the choice of loss is important under AP^{Pool}, as the softmax CE loss significantly underperforms the Federated loss. Table 8c shows the uniform sampler performs on par with Repeat Factor Sampling, likely because AP^{Pool} weights all instances equally, while AP^{Fixed} and AP^{Old} weight each *class* equally. Table 8d shows that two-stage training does not improve AP^{Fixed} or AP^{Pool} significantly. Finally, larger backbones consistently improve AP^{Old}, AP^{Fixed}, and AP^{Pool} (Table 8e).

A.2. Classifier retraining

Table 4d evaluates the efficacy of training detectors in two phases, using a few different data sampling configurations. For completeness, we report results using all sampler configurations in Table 9. These results further support the conclusions from the main paper: while certain configurations improves over the baseline under AP^{Old} (up to +1.3AP), they provide little to no improvements under AP^{Fixed}, suggesting they modify cross-category rankings. However, they also do not impact AP^{Pool}, indicating they do not meaningfully improve the calibration of scores across categories — *i.e*., they may take advantage of the vulnerability in AP^{Old} discussed in Section 3.

A.3. Evaluating losses for stronger models

Since certain detector modifications may behave differently as model capacity varies, we re-evaluate the importance of losses and classifier modifications using a stronger model in Table 10. We use a Cascade R-CNN model with a RegNetY-4GF backbone (see Section B for details), which we refer to as the strong model. We report results using this model in Table 10, and compare with the results using a ResNet-50 model (the weak model) reported in Table 8. We highlight two key differences. First, the Federated loss provides clear, significant improvements under both AP^{Fixed} and AP^{Pool} using the strong model (Table 10a), while it provides only a minor improvement over the weak model under AP^{Fixed} (Table 8a). This suggests the Federated loss may be more helpful for models with higher capacity. Second, while the normalized linear layer improves both the weak model and the strong model under all metrics, adding an objectness predictor in addition to the normalized classifier hurts the strong model considerably (dropping from 33.3 to 32.2 under AP^{Fixed}). These results suggest that the normalized classifier is helpful even for high-capacity models, but the objectness predictor is not.

B. Implementation details

Experimental Setup for Tables 1, 2, 3. For experiments in Tables 1 and 3, and the LVIS results in Table 2, we closely follow the setup in Section 5.1, but all results are from a single random seed and init. We follow the same setup for COCO results in Table 2 with two modifications: (1) We train for 270k iterations with a $0.1\times$ learning rate decay at 210k and 250k iterations, and (2) we use a uniform sampler.

Classifier heads: Objectness. The objectness predictor in Table 4b is implemented as an additional linear layer parallel with the classifier. This predictor is trained with sigmoid BCE loss, with a target of 1 for proposals matched to any object, and 0 otherwise. Let s_{obj} be the output of the objectness predictor after sigmoid, and s_c be the score for class c from the classifier after softmax. At test time, we update scores for each class as $s'_c = s_{\text{obj}} \cdot s_c$. BaGS uses this same objectness predictor by default, so we do not add a separate objectness predictor.

Classifier heads: Normalized layer. We replace the standard classifier with the following, as in [29]:

$$f(x; w, b, \tau) = \frac{\tau}{\|w\|_2 \|x\|_2} w^T x + b,$$

where τ is a temperature parameter tuned separately for each loss. For softmax CE and BaGS, $\tau = 20.0$; for sigmoid BCE, Federated, and EQL losses, $\tau = 50.0$. When using an objectness predictor with the normalized classifier, we replace the objectness predictor with a normalized layer as well. Figure 5 shows a concise PyTorch implementation.

Classifier retraining. For classifer retraining experiments in Table 4d, we train models in two phases. In Phase 1, we train the baseline model end-to-end following the setup in Section 5.1, using the Phase 1 sampler. In Phase 2, we

Loss	AP^{Old}	AP^{Fixed}	AP^{Pool}
Softmax CE	22.3	25.5	25.6
Sigmoid BCE	22.5 (+0.2)	25.6 (+0.1)	25.6 (+0.0)
EQL [27]	24.0 (+1.7)	26.1 (+0.6)	25.9 (+0.3)
Federated [35]	**24.7 (+2.4)**	**26.3 (+0.8)**	**27.8 (+2.2)**
BaGS [16]	24.5 (+2.2)	25.8 (+0.3)	26.0 (+0.4)

(a) **Loss functions.** Most losses perform equally under AP^{Pool}, with the exception of Federated loss, which significantly improves AP^{Pool} by 2.2 points.

Loss	Obj	Norm	AP^{Old}	AP^{Fixed}	AP^{Pool}
Softmax CE	✗	✗	22.3	25.5	25.6
	✓	✗	23.2 (+0.8)	25.3 (–0.2)	26.2 (+0.6)
	✗	✓	23.2 (+0.8)	**26.3 (+0.8)**	25.7 (+0.1)
	✓	✓	**24.4 (+2.0)**	**26.3 (+0.8)**	**26.3 (+0.7)**
Sigmoid BCE	✓	✓	24.2 (–0.2)	**26.3 (+0.0)**	26.6 (+0.3)
EQL [27]	✓	✓	24.7 (+0.3)	26.1 (–0.2)	26.6 (+0.3)
Federated [35]	✓	✓	**25.1 (+0.9)**	**26.3 (+0.0)**	**27.6 (+1.3)**
BaGS [16]	✓	✓	**25.1 (+0.9)**	26.2 (–0.1)	25.9 (–0.4)

(b) **Classifier modifications.** We evaluate two ideas commonly used for improving long-tail detection: an objectness predictor ('Obj') [16], and L2-normalizing both the linear classifier weights and input features ('Norm'). These components mildly improve AP^{Pool}, and the Federated loss still outperforms all other losses. Perhaps surprisingly, the modifications hurt the Federated loss (27.6 with *vs*. 27.8 without in Table 8a).

Sampler	AP^{Old}	AP^{Fixed}	AP^{Pool}
Uniform	18.4	22.8	**25.7**
CAS	19.2 (+0.8)	21.5 (–1.3)	21.7 (–4.0)
RFS	**22.3 (+3.9)**	**25.5 (+2.7)**	25.6 (–0.1)

(c) **Samplers.** Category Aware Sampling (CAS) and Repeat Factor Sampling (RFS) are common sampling strategies for addressing class imbalance. As AP^{Pool} weights all instances equally, unlike AP^{Fixed} and AP^{Old} (which weight all classes equally), uniform sampling performs on par with RFS.

Phase 1	Phase 2	AP^{Old}	AP^{Fixed}	AP^{Pool}
RFS	-	22.3	25.5	25.6
Uniform	RFS	21.6 (–0.7)	24.9 (–0.6)	**25.8 (+0.2)**
Uniform	CAS	23.1 (+0.8)	24.9 (–0.6)	25.6 (+0.0)
RFS	CAS	**23.6 (+1.3)**	**25.6 (+0.1)**	25.5 (–0.1)

(d) **Classifier retraining.** We evaluate the efficacy of training detectors in two phases, a commonly used technique [12, 29]. In Phase 1, the model is trained end-to-end with one sampler. In Phase 2, only the final classification layer is trained, using a different sampler. This strategy provides only minor improvements on AP^{Pool}.

Backbone	AP^{Old}	AP^{Fixed}	AP^{Pool}
ResNet-50	22.3	25.5	25.6
ResNet-101	24.6 (+2.3)	27.7 (+2.2)	27.2 (+1.6)
ResNeXt-101	**26.2 (+3.9)**	**28.7 (+3.2)**	**29.0 (+3.4)**

(e) **Stronger backbones.** Using larger backbones consistently improves the detector under AP^{Old}, AP^{Fixed}, and AP^{Pool}, indicating, as one might expect, that larger backbones improve overall detection quality and not just cross-category rankings. ResNeXt-101 uses the 32×8d configuration.

Table 8. **Impact of various design choices on the LVIS v1 validation dataset, comparing AP^{Old}, AP^{Fixed} and AP^{Pool}.** We report all experiments from Table 4 of our main paper using AP^{Pool} in addition to AP^{Fixed} and AP^{Old}. As in the main paper, unless specified otherwise, each experiment uses a ResNet-50 FPN Mask R-CNN model trained with Repeat Factor Sampling (RFS) for 180k iterations with 16 images per batch. All numbers are the average of three runs with different random seeds and initializations.

Phase 1	Phase 2	AP^{Old}	AP^{Fixed}	AP^{Pool}
RFS	-	22.3	25.5	25.6
Uniform	Uniform	19.3 (–3.0)	24.0 (–1.5)	**25.8 (+0.2)**
Uniform	RFS	21.6 (–0.7)	24.9 (–0.6)	**25.8 (+0.2)**
Uniform	CAS	23.1 (+0.8)	24.9 (–0.6)	25.6 (+0.0)
RFS	Uniform	20.8 (–1.5)	25.4 (–0.1)	25.8 (+0.2)
RFS	RFS	22.6 (+0.3)	**25.8 (+0.3)**	25.7 (+0.1)
RFS	CAS	**23.6 (+1.3)**	25.6 (+0.1)	25.5 (–0.1)
CAS	Uniform	17.4 (–4.9)	21.1 (–4.4)	22.1 (–3.5)
CAS	RFS	18.2 (–4.1)	21.3 (–4.2)	22.1 (–3.5)
CAS	CAS	19.1 (–3.2)	21.2 (–4.3)	21.7 (–3.9)

Table 9. **Classifier retraining.** We report all variants of classifier retraining, a subset of which are reported in Table 4d. This strategy improves AP^{Old}, but only mildly affects AP^{Fixed}, suggesting that classifier retraining may primarily improve cross-category rankings. All results are the average of 3 runs with different random seeds and initializations.

randomly re-initialize the classifier weights and biases of the model. We fine-tune only the classifer weights and biases with the specified Phase 2 sampler for 90k iters using a minibatch size of 16 images with a 0.1× learning rate decay applied after 60k and 80k iterations. The learning rate starts at 0.02 and a weight decay of 1e–4 is used.

```python
from torch.nn import functional as F

def forward_normalized(linear, x, temperature):
    """
    Args:
        x (torch.Tensor): Input features, shape (n, d).
        temperature (float): Temperature hyperparam.
        linear (nn.Linear): Standard linear layer.
    """
    x_normed = F.normalize(x, p=2, dim=1)
    w_normed = F.normalize(linear.weight, p=2, dim=1)
    return F.linear(
        temperature * x_normed, w_normed, linear.bias
    )
```

Figure 5. Pytorch code for implementing the forward pass of a normalized linear layer using a standard linear layer.

RegNetY-4GF model. For the 'RegNetY-4GF' model in Table 8e and Table 10, we use a Cascade R-CNN model [1]

Loss	AP^{Old}	AP^{Fixed}	AP^{Pool}
Softmax CE	28.6	31.9	32.2
Sigmoid BCE	28.5 (–0.1)	31.8 (–0.1)	32.0 (–0.2)
EQL [27]	29.4 (+0.8)	31.8 (–0.1)	32.2 (+0.0)
Federated [35]	**31.8** (+3.2)	**33.6** (+1.7)	**34.7** (+2.5)
BaGS [16]	30.2 (+1.6)	31.9 (+0.0)	32.5 (+0.3)

(a) **Loss functions.** With the stronger model, most losses perform roughly equally well under AP^{Fixed} and AP^{Pool}, but the Federated loss shows significant improvements for all metrics.

Loss	Obj	Norm	AP^{Old}	AP^{Fixed}	AP^{Pool}
Softmax CE	✗	✗	28.6	31.9	32.2
	✓	✗	29.3 (+0.7)	31.5 (–0.4)	32.4 (+0.2)
	✗	✓	29.7 (+1.1)	**33.3** (+1.4)	32.4 (+0.2)
	✓	✓	**30.2** (+1.6)	32.2 (+0.3)	**32.5** (+0.3)

(b) **Classifier modifications.** While the normalized linear layer ('norm') improves both AP^{Fixed} and AP^{Pool}, the objectness predictor ('obj') significantly hurts AP^{Fixed}.

Table 10. Analyzing losses and components with a higher-capacity detector using a RegNetY-4GF backbone. See Section A.3 for details.

Loss	Param	Description	Default	Search	Final
EQL	λ	Frequency threshold	1.76e–3	{1e-4, 5e-4, 1e-3, 1.76e-3, 5e-3}	1e–3
BaGS	β	BG sample ratio	8.0	{4.0, 8.0, 16.0, 32.0, 64.0}	16.0
Federated	$\|S\|$	Neg. classes sampled	50	{10, 50, 100}	50

Table 11. Parameters tuned for each loss. See Section B for details.

with a RegNetY-4GF [24] backbone using FPN [17]. This model is trained following the Section 5.1 setup, with important modifications to achieve high accuracy using the stronger capacity backbone. The model is trained for 270k iterations, with a $0.1\times$ learning rate decay applied after 210k and 250k iterations. The weight decay is set to 5e–5 to match the weight decay used for ImageNet pre-training (a standard weight decay of 1e–4 decreases AP^{Old} by more than 3 points). Stronger data augmentation was needed to prevent overfitting for rare categories; we resize the larger size of training images randomly between 400px and 1000px, instead of the default strategy (used for all other experiments) of picking a random scale from 640px to 800px with a step of 32.

Losses. As EQL [27] and BaGS [16] were developed for LVIS v0.5, and the Federated loss [35] was tuned for a CenterNet-based detector [36], we tune key parameters for each loss in our setting. We tune parameters using a Mask R-CNN model with ResNet-50, trained on LVIS v1 closely following the setup in Section 5.1, except with a Uniform sampler instead of RFS, and using a single random initialization and seed instead of three. We choose hyperparameters which optimize AP^{Old} on the LVIS v1 validation set. We detail these hyperparameters and their optimal choices in Section B. In addition to these recently proposed losses, we found it necessary to lengthen the warmup schedule for the 'Sigmoid BCE' loss for training stability. The default for most models is a linear warmup schedule starting with a learning rate of 1e–3, ramping up for the first $1,000$ iterations. For Sigmoid BCE, we found it necessary to start with a lower learning rate of 1e–4 and ramp up for $10,000$ iterations. Our Sigmoid BCE implementation first *sums* the BCE loss over all $K \cdot N$ loss evaluations for the K classes (*e.g.*, 1203 in LVIS v1) and N object proposals in a mini-batch and then divides this sum by N.

References

[1] Zhaowei Cai and Nuno Vasconcelos. Cascade r-cnn: Delving into high quality object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6154–6162, 2018. 10

[2] Thomas Dean, Mark A Ruzon, Mark Segal, Jonathon Shlens, Sudheendra Vijayanarasimhan, and Jay Yagnik. Fast, accurate detection of 100,000 object classes on a single machine. In *CVPR*, 2013. 2

[3] Chaitanya Desai, Deva Ramanan, and Charless C Fowlkes. Discriminative models for multi-class object layout. *IJCV*, 95(1):1–12, 2011. 2, 7

[4] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The PASCAL visual object classes (VOC) challenge. *IJCV*, 2010. 1, 2

[5] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. In *ICML*, 2017. 2

[6] Agrim Gupta, Piotr Dollar, and Ross Girshick. LVIS: A dataset for large vocabulary instance segmentation. In *CVPR*, 2019. 1, 2, 3, 5, 6

[7] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask R-CNN. In *CVPR*, 2017. 3, 5

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 5, 6

[9] Derek Hoiem, Yodsawalai Chodpathumwan, and Qieyun Dai. Diagnosing error in object detectors. In *ECCV*, 2012. 2

[10] Ronghang Hu, Piotr Dollár, Kaiming He, Trevor Darrell, and Ross Girshick. Learning to segment every thing. In *CVPR*, 2018. 2

[11] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *ICML*, 2015. 5

[12] Bingyi Kang, Saining Xie, Marcus Rohrbach, Zhicheng Yan, Albert Gordo, Jiashi Feng, and Yannis Kalantidis. Decou-

pling representation and classifier for long-tailed recognition. *ICLR*, 2020. 2, 6, 10

[13] Meelis Kull, Telmo Silva Filho, and Peter Flach. Beta calibration: a well-founded and easily implemented improvement on logistic calibration for binary classifiers. In *Artificial Intelligence and Statistics*, 2017. 2, 8

[14] Fabian Kuppers, Jan Kronenberger, Amirhossein Shantia, and Anselm Haselhoff. Multivariate confidence calibration for object detection. In *CVPR Workshops*, 2020. 2, 8

[15] Alina Kuznetsova, Hassan Rom, Neil Alldrin, Jasper Uijlings, Ivan Krasin, Jordi Pont-Tuset, Shahab Kamali, Stefan Popov, Matteo Malloci, Tom Duerig, et al. The open images dataset v4: Unified image classification, object detection, and visual relationship detection at scale. *arXiv preprint arXiv:1811.00982*, 2018. 2

[16] Yu Li, Tao Wang, Bingyi Kang, Sheng Tang, Chunfeng Wang, Jintao Li, and Jiashi Feng. Overcoming classifier imbalance for long-tail object detection with balanced group softmax. In *CVPR*, 2020. 2, 4, 5, 6, 7, 10, 11

[17] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *CVPR*, 2017. 5, 11

[18] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft COCO: Common objects in context. In *ECCV*, 2014. 1, 2

[19] Ziwei Liu, Zhongqi Miao, Xiaohang Zhan, Jiayun Wang, Boqing Gong, and Stella X Yu. Large-scale long-tailed recognition in an open world. In *CVPR*, 2019. 2, 5

[20] Christopher D Manning, Hinrich Schütze, and Prabhakar Raghavan. *Introduction to information retrieval*. Cambridge university press, 2008. 2

[21] Mahdi Pakdaman Naeini, Gregory F Cooper, and Milos Hauskrecht. Obtaining well calibrated probabilities using bayesian binning. In *AAAI*, 2015. 2, 8

[22] Alexandru Niculescu-Mizil and Rich Caruana. Predicting good probabilities with supervised learning. In *ICML*, 2005. 2

[23] John Platt. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advances in large margin classifiers*, pages 61–74, 1999. 2, 8

[24] Ilija Radosavovic, Raj Prateek Kosaraju, Ross Girshick, Kaiming He, and Piotr Dollár. Designing network design spaces. In *CVPR*, 2020. 11

[25] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *CVPR*, 2016. 5

[26] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. ImageNet large scale visual recognition challenge. *IJCV*, pages 211–252, 2015. 5

[27] Jingru Tan, Changbao Wang, Buyu Li, Quanquan Li, Wanli Ouyang, Changqing Yin, and Junjie Yan. Equalization loss for long-tailed object recognition. In *CVPR*, 2020. 4, 5, 6, 7, 10, 11

[28] Hao Wang, Yitong Wang, Zheng Zhou, Xing Ji, Dihong Gong, Jingchao Zhou, Zhifeng Li, and Wei Liu. Cosface: Large margin cosine loss for deep face recognition. In *CVPR*, 2018. 5

[29] Jiaqi Wang, Wenwei Zhang, Yuhang Zang, Yuhang Cao, Jiangmiao Pang, Tao Gong, Kai Chen, Ziwei Liu, Chen Change Loy, and Dahua Lin. Seesaw loss for long-tailed instance segmentation. *arXiv preprint arXiv:2008.10032*, 2020. 2, 5, 6, 9, 10

[30] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *CVPR*, 2017. 6

[31] Bianca Zadrozny and Charles Elkan. Obtaining calibrated probability estimates from decision trees and naive bayesian classifiers. In *ICML*, 2001. 2, 8

[32] Bianca Zadrozny and Charles Elkan. Transforming classifier scores into accurate multiclass probability estimates. In *SIGKDD*, 2002. 2, 8

[33] Yubo Zhang, Pavel Tokmakov, Martial Hebert, and Cordelia Schmid. A study on action detection in the wild. *arXiv preprint arXiv:1904.12993*, 2019. 2, 6

[34] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Semantic understanding of scenes through the ADE20K dataset. *IJCV*, 2019. 2

[35] Xingyi Zhou, Vladlen Koltun, and Philipp Krähenbühl. Joint COCO and LVIS workshop at ECCV 2020: LVIS challenge track technical report: CenterNet2. 2020. 2, 5, 6, 7, 10, 11

[36] Xingyi Zhou, Dequan Wang, and Philipp Krähenbühl. Objects as points. In *arXiv preprint arXiv:1904.07850*, 2019. 11