Revisiting Few-Shot Object Detection with Vision-Language Models

Anish Madan[1], Neehar Peri[1], Shu Kong[2], Deva Ramanan[1]
Carnegie Mellon University[1], Texas A&M University[2]
{amadan, nperi, deva}@cs.cmu.edu, shu@tamu.edu

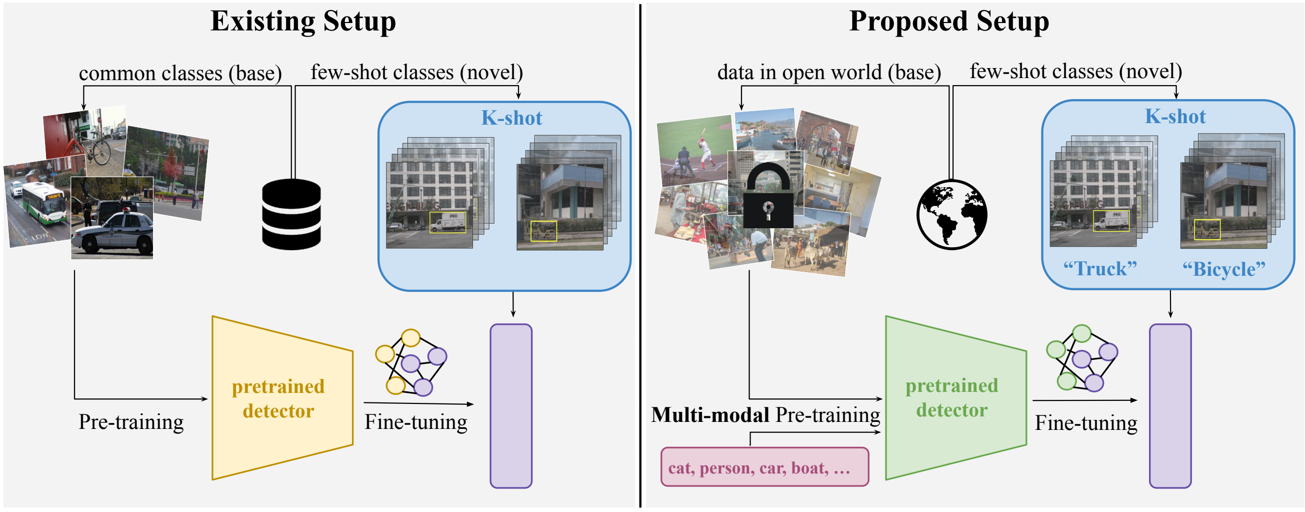


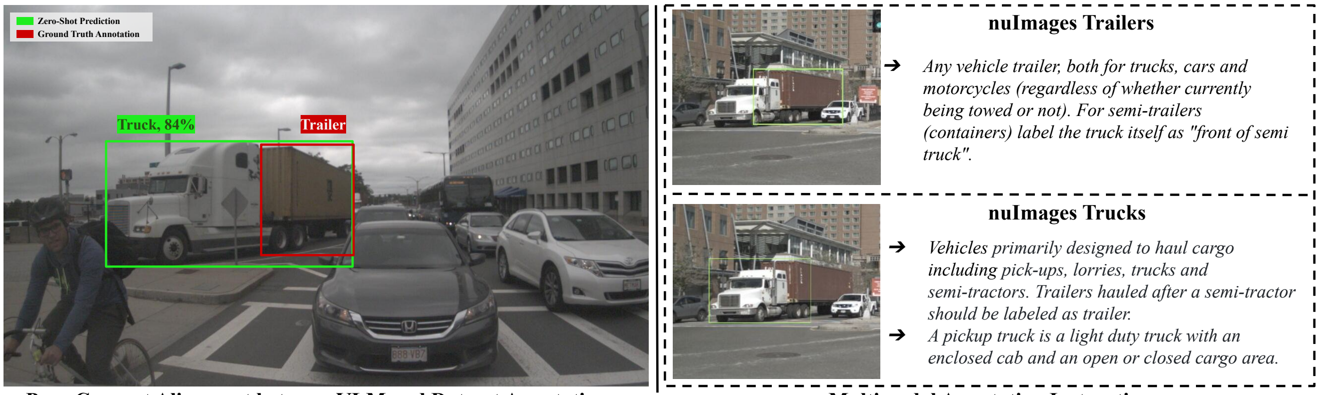
Figure 1. We propose a new few-shot object detection (FSOD) benchmark which embraces the challenge of aligning multimodal foundation models to target concepts with vision and language. On the **left**, we describe the standard setup: methods pre-train on `base` classes (with many examples per class) and then fine-tune on K-shots of `novel` (and optionally `base`) classes. On the **right**, we describe our proposed setup: Given the scale and often private nature of data used to train VLMs, it is impractical to maintain a split of `base` and `novel` classes. Instead, one should directly fine-tune VLMs on K-shots of the target classes (and evaluate only those target classes). Importantly, VLMs allow us to exploit additional language cues such as class names and descriptions for fine-tuning. We show that such "zero-shot" language cues already outperforms state-of-the-art methods without any fine-tuning. However, such foundational VLMs can be significantly improved by aligning them with target concepts (Fig. 2).

Abstract

Few-shot object detection (FSOD) benchmarks have advanced techniques for detecting new categories with limited annotations. Existing benchmarks repurpose well-established datasets like COCO by partitioning categories into `base` *and* `novel` *classes for pre-training and fine-tuning respectively. However, these benchmarks do not reflect how FSOD is deployed in practice. Rather than only pre-training on a small number of* `base` *categories, we argue that it is more practical to fine-tune a foundation model (e.g., a vision-language model (VLM) pre-trained on web-scale data) for a target domain. Surprisingly, we find that zero-shot inference from VLMs like GroundingDINO significantly outperforms the state-of-the-art (48.3 vs. 33.1 AP) on COCO. However, such zero-shot models can still be misaligned to target concepts of interest -* `trailers` *on the web may be different from* `trailers` *as defined for a target application like autonomous vehicles. In this work, we propose Foundational FSOD, a new benchmark protocol that evaluates detectors pre-trained on any external dataset and fine-tuned on K-shots spanning both vision and language modalities. Further, we note that current FSOD benchmarks are actually federated datasets containing exhaustive annotations for each category on only a subset of the data. We leverage this insight to propose simple strategies for fine-tuning VLMs with federated losses. We demonstrate our approach on LVIS and nuImages, improving over prior work by 5.9 AP.*

1. Introduction

Object detection is a fundamental problem in computer vision [8, 21] that has matured in recent years [22, 24, 30, 31]. Given a large-scale annotated dataset, one can train a detector from scratch. However, training object detectors



nuImages Trailers

➔ *Any vehicle trailer, both for trucks, cars and motorcycles (regardless of whether currently being towed or not). For semi-trailers (containers) label the truck itself as "front of semi truck".*

nuImages Trucks

➔ *Vehicles primarily designed to haul cargo including pick-ups, lorries, trucks and semi-tractors. Trailers hauled after a semi-tractor should be labeled as trailer.*

➔ *A pickup truck is a light duty truck with an enclosed cab and an open or closed cargo area.*

Poor Concept Alignment between VLM and Dataset Annotations

Multimodal Annotation Instructions

Figure 2. **Poor Alignment Between VLM and Target Concepts**. While VLMs show impressive zero-shot performance, they struggle when the concept of the target class is different from vision-language concepts encountered in web-scale training. On the **left**, we see that the nuImages dataset defines the cab of the `truck` as a separate object concept from its `trailer` (shown in **red**), while the zero-shot VLM predicts the entire vehicle as a `truck` (shown in **green**). The **right** visualizes the actual *class definitions* given to the nuImages annotators, provided as both textual descriptions and visual examples. Just as human annotators learn concepts from few-shot multi-modal vision-language examples, we argue that VLMs should be fine-tuned with K shots spanning such visual and language modalities.

for domains with limited annotated data remains challenging, motivating the problem of few-shot object detection.

Status Quo. Few-shot object detection (FSOD) benchmarks have made considerable progress on learning to detect new categories from limited training data. Existing benchmarks are constructed by partitioning popular object detection datasets like PASCAL VOC [5] and COCO [21] into `base` categories (with many examples per class) and target `novel` categories (with few examples per class). Detectors are first pre-trained on `base` classes and are fine-tuned on K examples (or K-shots) from `novel` classes.

Conventional FSOD benchmarks enforce `base` and `novel` classes to be disjoint to prevent concept leakage and measure generalization to unseen categories. However, as most detectors are pre-trained on ImageNet, concept leakage already occurs in contemporary benchmarks. For example, `cat` and `person` are considered `novel` in the COCO FSOD benchmark but are already present in ImageNet. Similarly, `car` is considered `novel` even though similar concepts like `sports car` and `race car` are present in ImageNet. Because concept leakage is difficult to avoid, we take the view that it should instead be embraced. Intuitively, pre-training on large-scale diverse `base` categories (which may overlap with `novel` concepts) will ultimately improve generalization to `novel` classes.

Foundational FSOD. Rather than explicitly filtering target classes from pre-training [3, 12, 49], practitioners will likely employ foundational vision-language models (VLMs) pre-trained on (potentially private) web-scale data [14, 20, 32, 34] and fine-tune them for their task. As VLMs' pre-training datasets contain diverse concepts [20, 29][1], it is challenging to prevent concept leakage. Therefore, one may be hesitant to exploit VLMs for FSOD. But the performance of such foundational models is undeniable; state-of-the-art VLMs like GroundingDINO already dominate all leading FSOD methods on COCO (48.3 vs. 33.1 AP) *without fine-tuning* (cf. Table 1).

Multi-Modal Concept Alignment. Does the strong zero-shot performance of VLMs imply that few-shot detection is no longer an interesting problem? No! We find that the *target class name* is often an insufficient description of the target concept. For example, `trailers` in nuImages are defined differently than `trailer` in web-scale data. Fig. 2 shows the annotator *instructions* used to "align" human annotators to subtle aspects of the target concept [2]. Interestingly, such instructions are naturally multi-modal, often including a few visual examples and textual descriptions. We advocate for a FSOD setup that uses similar visual and language cues for *VLM concept alignment*. We refer to our proposed setup as Foundational FSOD (Fig. 1).

Federated Few-shot Learning. In order to effectively align VLM concepts with K-shot multimodal "instructions", we leverage a simple but evidently underappreciated observation: K-shot object detection datasets are actually federated datasets [11]. A federated dataset is a dataset comprised of smaller subsets, where each subset is exhaustively annotated for only a single category. For example, `cars` may or may not appear in the background of the K images annotated with `motorcycles` (see Fig. 3). However, existing FSOD methods incorrectly assume that no `cars` are present in the background of non-`car` images. Inspired by prior work in learning with federated datasets [47] and weakly-supervised learning [15, 38, 39], we demonstrate that fine-tuning VLMs

[1] The CLIP [29] pre-training dataset contains 500,000 concepts, spanning many categories encountered in the real world.

with federated losses consistently improves over zero-shot inference (cf. Tables 2, 4).

Contributions. We present three major contributions

1. We modernize the FSOD benchmark by embracing vision-language foundation models that are pretrained on internet-scale data. We highlight the practical challenge of using multi-modal few-shot examples to define the target semantic concept (as shown in Fig. 2).
2. We point out that existing FSOD benchmarks are actually federated datasets, and present simple strategies for fine-tuning VLMs to align concepts for solving FSOD.
3. We conduct extensive experiments to ablate our design choices and demonstrate that our simple method achieves state-of-the-art results on the LVIS and nuImages Foundational FSOD benchmarks.

2. Related Works

Few-Shot Object Detection aims to detect new object categories given limited training data [16]. Recent work explores two primary approaches: meta-learning and transfer learning. Meta-learning-based methods focus on acquiring generalizable features from a set of base classes, which can then be applied to identify novel classes. For example, [13] proposes a technique that re-weights features from base classes to predict novel classes. [42] proposes a framework addressing both few-shot object detection and few-shot viewpoint estimation. [6] introduces a general FSOD network that learns a matching metric between image pairs, while [40] enhances object features using a universal prototype. More recently, [44] proposes a generative approach that is robust to noisy object proposals for novel classes. In contrast, transfer learning involves freezing the network weights pretrained on a `base` dataset to improve a model's ability to generalize to `novel` classes with limited data. Transfer learning approaches often follow a two-stage fine-tuning strategy: first train on the `base` classes and then fine-tune the box classifier and regressor with K-shots from `novel` classes. This strategy historically outperformed meta-learning approaches [37]. Recent work has primarily focused on improving classification performance. FSCE [35] utilizes a contrastive proposal encoding loss to encourage instance-level intra-class compactness and inter-class variance. Similarly, [19] applies a class margin loss to balance inter and intra-class margins. Our approach leverages transfer-learning by fine-tuning vision-language models (VLMs) pre-trained on large-scale datasets.

Vision Language Models are trained on a large-scale set of image-text pairs collected from the web. These models embed images and text into a shared space, enabling open-vocabulary detection. Early works adapt VLMs for object detection by either distilling the model's predictions for specific image regions [9, 10] or directly incorporating detection components into frozen [17] or fine-tuned [4, 26, 27] encoders. In contrast, RegionCLIP [46] employs a multi-stage training approach, which involves generating pseudo-labels from captioning data, conducting region-text contrastive pre-training, and fine-tuning on detection data. GLIP [20] uses a single text query for the entire image and frames detection as a phrase grounding problem. Detic [48] addresses long-tail detection performance by leveraging image-level supervision. In the context of open-vocabulary detection, there may be some overlap between the object categories seen during training and those in testing. We use the term "zero-shot inference" to signify that a model has never been trained on the target dataset.

Federated Datasets are constructed by combining smaller datasets, each resembling a conventional object detection dataset for a single category [11]. Each of these smaller datasets ensures exhaustive annotations for a specific category. Images within each smaller dataset may overlap, resulting in some images with exhaustive annotations for multiple categories. Importantly, since exhaustive annotations for a particular category are only guaranteed within each small dataset, most images are sparsely annotated. Consequently, naively training models with federated datasets leads to much sparser gradients [36]. To address this challenge, CenterNet2 [47] introduced FedLoss, a simple modification of cross-entropy loss which randomly samples a subset of negative categories for each image. We adopt FedLoss for FSOD, achieving consistent improvements over zero-shot inference.

Weakly Supervised Learning techniques are especially popular when learning in data-constrained settings, leveraging large-scale noisy annotations to improve model performance. Prior works [15] learn from noisy annotations using negative labels for classification. Further, [18, 43] leverage pseudo-labels for self-training. We modify our fine-tuning approach to leverage negatives derived from pseudo-labels to improve FSOD performance.

3. FSOD with Vision-Language Models

As shown in Fig 1, our proposed Foundational FSOD benchmark uses vision-language models (VLMs) pre-trained on diverse, large-scale datasets prior to fine-tuning on K-shots per C target classes. We contrast our proposed setup with the standard FSOD benchmark, demonstrate that FSOD benchmarks are actually federated datasets, and present simple strategies for fine-tuning VLMs below.

3.1. Foundational FSOD Benchmark

Existing FSOD benchmarks repurpose well-established datasets like PASCAL VOC [5] and COCO [21] by partitioning them into `base` and `novel` classes for pre-training and fine-tuning, respectively. For COCO, the 60 categories

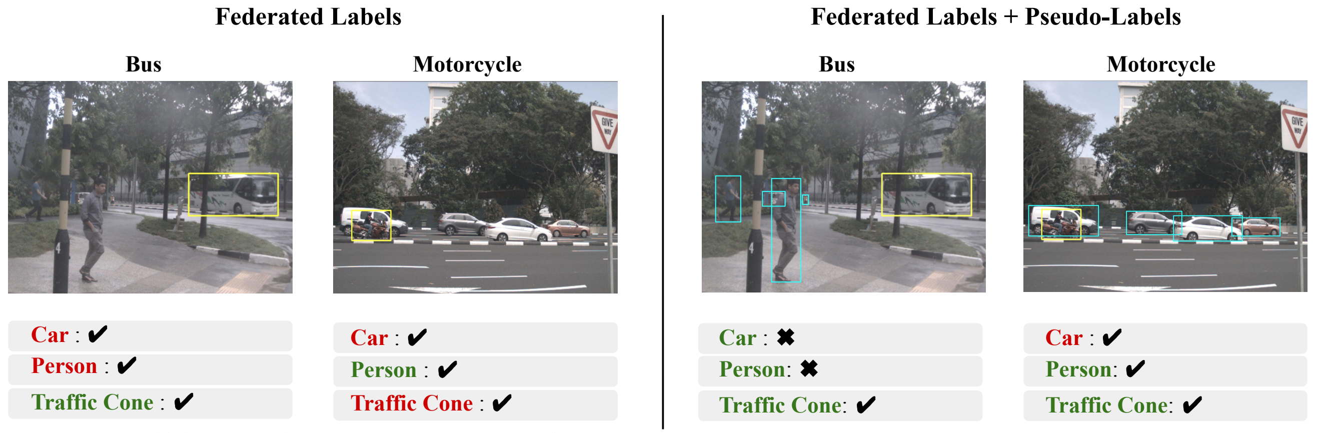


Figure 3. **Identifying Negative Pseudo-Labels**. The **left** visualizes the standard K-shot detection setup, which we argue is actually a *federated* dataset [11] where one is given multiple mini-datasets of K images. In this case, we visualize two $K = 1$ datasets of `bus` and `motorcycle`. Importantly each mini-dataset does *not* provide information about the presence of other objects. Previous FSOD methods apparently ignore this fact, and instead assume the collective set of few-shot images are *fully* annotated across all object classes. This will likely produce many incorrect negative labels – e.g., all unlabeled `cars` in the background of the `motorcycle` mini-dataset will be incorrectly treated as negative cars. We use a ✓ to denote that a given image will be treated as a negative example of a given class by the learner and a ✗ to denote that a given image will be ignored when learning a given class. We color such negative labels as **green** when correct and **red** when incorrect. Naive FSOD approaches learn about *all* classes from *all* images, which results in many incorrect negative labels (shown in **red** on the **left**). Instead, we embrace the partially-labeled nature of the data and exploit tools from weakly-supervised learning, such as the use of psuedo-labels predicted by a teacher. We train (or rather, fine-tune) initial detectors on only the appropriate mini-dataset and use thresholded psuedo-detections (shown as **cyan** boxes on the **right**) to find images that can be confidently treated as (pseudo) negatives, which results in much fewer mistakes (shown in **red** on the **right**). This in turns produces improved performance. We also attempted to learn from psuedo positive labels, but found these to be less reliable. We include pseudo-code in the supplement.

disjoint with PASCAL VOC are used as `base` classes and the remaining 20 are used as `novel` classes [37]. However, this setup is artificial and does not reflect how FSOD is deployed in practice. First, it requires FSOD methods to detect common categories such as `car` and `person` by assuming they have only few examples. Importantly, VLMs like GroundingDINO [23] can already detect common categories with high accuracy *without fine-tuning* on COCO (cf. Table 1). Therefore, we focus on benchmarking Foundational FSOD on more realistic and challenging datasets like LVIS and nuImages. In addition, existing FSOD benchmarks require that datasets are partitioned into `base` and `novel` classes, which is not feasible for large-scale (often private) foundational datasets. For example, although CLIP's [29] model weights are publicly available, its pretraining dataset is not. Instead, FSOD methods should only fine-tune VLMs on K-shot annotations for C target classes, and also evaluate performance on these C classes.

3.2. Few-Shot Multi-Modal Concept Alignment

Although VLMs achieve strong zero-shot performance on common classes, they struggle when the concept of the target class is different from vision-language concepts encountered in web-scale training (cf. Fig. 2). For example, nuImages [1] defines `trailer` as independent from the truck cab. However, Detic jointly detects the truck cab and its trailer together. This fine-grained distinction is provided to human annotators with visual examples and textual descriptions. Similarly, we provide a few multi-modal examples for each class to align VLMs with dataset annotations.

We start by fine-tuning Detic [48] on the provided K-shot examples. We ablate the impact of freezing different parts of Detic in Table 5 and find that freezing the backbone, RPN, and classifier head with CLIP embeddings yields the best performance (cf. Figure 4). Due to the sparsely annotated nature of the FSOD task, we posit that the model will receive sparser gradients which degrades the object detector's performance because all unannotated objects in the image would be treated as negatives [36]. Therefore, we explore three strategies for handling negatives as described below.

3.3. FSOD Benchmarks are Federated Datasets

Prior works follow the K-shot dataset creation process established by [37]. To construct a K-shot dataset, we select a target class C and an image at random. If the total annotations for class C in the image are less than or equal to K, we add the image to our dataset. We repeat this process for all classes until we have exactly K annotations per class. Importantly, each image in the dataset is exhaustively annotated for a subset of all classes. Recall, a federated dataset is also comprised of images that are exhaustively annotated for a specific category. This suggests that we can leverage insights about federated datasets [11, 47] and train better few-shot object detectors.

Fine-Tuning with FedLoss. We fine-tune Detic with Federated Loss (FedLoss) [47] using a subset S of classes C for each training image. Specifically, we use a binary cross-entropy loss on all classes in S and ignore classes outside of S during training. S is comprised of the ground-truth annotation class along with randomly sampled negative classes for each image. We sample these negative classes in proportion to their square-root frequency in the training set. We find that probablistically sampling negatives rather than labeling all unannotated classes as negatives improves fine-tuning results, reliably beating zero-shot performance. Importantly, although FedLoss has been explored in the context of long-tailed detection, applying it to FSOD provides considerable performance improvements, reaffirming that FSOD benchmarks are actually federated datasets.

Fine-Tuning with Inverse FedLoss. However, we note that FedLoss samples common classes like `car` more frequently as negatives, hurting detection accuracy for long-tailed datasets like LVIS and nuImages. Instead, we propose Inverse FedLoss (InvFedLoss), a minor modification of FedLoss that samples negative categories in proportion to the *inverse* of their square root frequency. This ensures that we sample rare categories as negatives more frequently to better match the true data distribution. Leveraging this insight improves over just using FedLoss and naive fine-tuning.

Fine-Tuning with Pseudo-Negatives. Despite the effectiveness of InvFedLoss, probablistically sampling negatives using dataset-wide statistics is sub-optimal because it does not consider the content of each image. We can improve the accuracy of sampled negatives with pseudo-labels to determine which classes are likely not in a particular image. If the maximal score for any class prediction is less than a threshold, we consider this class to be a negative. Using image predictions to identify pseudo-negatives yields better results than simply using dataset-wide statistics. We present pseudo-code in the supplement.

4. Experiments

We conduct extensive experiments to validate that zero-shot inference from VLMs significantly improves over state-of-the-art FSOD approaches, suggesting that existing benchmarks should be re-framed to include foundation models. Moreover, we demonstrate that using federated losses consistently improve fine-tuning performance on LVIS and nuImages. Lastly, we analyze the upper bound performance and ablate the impact of freezing different detector components on fine-tuning performance. We will release our code and data splits to foster future Foundational FSOD research.

Table 1. **VLM Zero-Shot Inference Is a Strong FSOD Baseline.** Zero-shot inference with VLMs like GroundingDINO resoundingly outperform state-of-the-art FSOD methods on the COCO FSOD benchmark, motivating the need to re-frame FSOD to embrace foundation models.

Approach	30-shots		
	AP	bAP	nAP
FRCN-ft-full [45]	18.6	20.6	12.5
FRCN-BCE [45]	30.2	36.8	10.3
TFA w/ fc [37]	29.3	34.5	13.5
TFA w/cos [37]	29.9	35.3	13.6
MPSR [41]	17.1	18.1	14.1
Meta-RCNN [45]	7.8	7.1	9.1
FsDetView [42]	10.0	9.3	12.0
Retentive R-CNN [7]	32.9	39.3	13.8
DiGeo [25]	33.1	39.4	14.2
GroundingDINO (Zero-Shot) [23]	**48.3**	**46.3**	**54.3**

Datasets and Metrics. We repurpose two established datasets for Foundational FSOD:

- **LVIS** [11] re-annotates COCO images using 1,230 fine-grained classes, which are divided into frequent, common and rare based on the cardinality of each class. Frequent and common classes are combined to form `LVIS-base` and is used for pre-training. Rare classes are used for `LVIS-novel`. Following [25, 37], we benchmark with LVIS v0.5 on publicly released data splits and report performance across the frequent, common, and rare groups (AP_f, AP_c, AP_r) on the LVIS val-set.
- **nuImages** [1] annotates 18 classes, which are divided into groups with `many`, `medium`, and `few` examples [28] and report AP for each cohort. Although this dataset is not traditionally used for FSOD, nuImages' open-world categories like `debris` and `pushable-pullable` make it particularly challenging (even for VLMs), and is a realistic benchmark for Foundational FSOD. Unlike LVIS, nuImages is fully annotated, so we construct our Foundational FSOD benchmark using the procedure described in subsection 3.3.

4.1. Zero-Shot Inference Is A Strong FSOD Baseline

We compare state-of-the-art FSOD methods with zero-shot inference from GroundingDINO [23] on COCO in Table 1. Surprisingly, GroundingDINO outperforms DiGeo [25] by 16.2% AP averaged across both `base` and `novel` categories despite never being trained on COCO images. GroundingDINO's impressive performance is due to its large-scale multi-modal pre-training on Objects365 [33], GoldG [14] and Cap4M [20]. It is worth noting that GroundingDINO achieves higher AP on `novel` classes than `base`, suggesting that `novel` classes in existing

Table 2. **LVIS Foundational FSOD Performance**. Detic pre-trained only on LVIS-base outperforms specialized methods such as TFA and DiGeo by ∼**6 AP**, without even seeing the rare-class data. Importantly these performance improvements can be attributed to Detic's CLIP-based classifier and demonstrates how concept leakage through language can be advantageous in data constrained settings. Secondly, leveraging our insight that FSOD benchmarks are actually federated datasets, we show that fine-tuning with pseudo-negatives improves over standard fine-tuning by $\mathbf{4.3AP_r}$ (15.5 vs. 19.8, ResNet-50 backbone). Training with pseudo-negatives improves fine-tuning performance because we do not naively assume all classes not labeled in an image are negatives. Lastly, we note that simply swapping the ResNet-50 backbone to Swin-B significantly improves performance (without modifying training data). Notably, we find Detic's rare class performance improves over fine-tuning by $\mathbf{5.9AP_r}$ (from 26.7 AP to 32.6 AP). All the methods in this table are pre-trained on LVIS-base.

Approach	10-shots			
	AP	AP_f	AP_c	AP_r
ResNet-50 Backbone				
TFA w/ fc [37]	24.1	27.9	23.9	14.9
TFA w/ cos [37]	24.4	27.7	24.3	16.9
DiGeo [25]	24.9	28.5	24.6	17.3
Detic (Base Only) [48]	30.0	34.4	30.8	16.3
+ Fine-Tuning (Base + Novel)	30.0	33.2	31.9	15.5
w/ FedLoss	30.8	33.9	32.7	17.4
w/ InvFedloss	31.1	34.3	32.5	18.7
w/ Pseudo-Negatives	**31.6**	**34.8**	**32.8**	**19.8**
Swin Backbone				
GroundingDINO (Zero-Shot) [23] [2]	27.4	32.7	23.3	18.1
Detic (Base Only) [48]	35.2	**38.7**	36.8	21.4
+ Fine-Tuning (Base + Novel)	35.9	37.1	37.8	26.7
w/ FedLoss	36.5	36.7	38.3	30.4
w/ InvFedloss	37.1	37.8	**38.5**	31.1
w/ Pseudo-Negatives	**37.2**	37.7	38.2	**32.6**

benchmarks are actually not rare in the real world. Therefore, FSOD benchmarks should be re-framed to reflect real-world applications.

4.2. Foundational FSOD with LVIS

In this section, we evaluate GroundingDINO [23] and Detic's [48] performance on the LVIS Foundational FSOD benchmark. Note that we only report GroundingDINO's zero-shot inference results because the authors have not released training code at the time of this submission. We train Detic from scratch on LVIS-base with a ResNet-50 backbone for fair comparison with prior work [25, 37].

As shown in Table 2, Detic outperforms all recent FSOD baselines including DiGeo [25] by about ∼6 points on AP_c and AP_f and achieves 16.3 AP_r without ever seeing any rare class data (e.g by prompting Detic (Base only) with the rare class names). Importantly, these performance improvements can be attributed to Detic's CLIP-based classifier, which has been trained with significantly more than K examples of each class. This highlights the role of language in data-constrained settings.

Table 3. **nuImages Foundational FSOD Performance.** We repurpose nuImages for FSOD following the dataset creation process established by [37]. We group categories by frequency into cohorts with many, medium and few examples per class (according to the fully-annotated dataset) [28]. We fine-tune Detic pre-trained on LVIS, COCO and ImageNet-21K on K examples for each of the 18 nuImages classes. We observe that prior FSOD methods like TFA perform poorly on nuImages($<$ 3AP). But we drastically improve performance if we upgrade TFA according to our proposed setup: by increasing pre-training data and leveraging language cues. Interestingly, accuracy across cardinalities decreases despite all classes being trained with K examples. This suggests that despite pre-training on web-scale datasets, VLMs still struggle to detect rare categories like strollers, pushable-pullable, and debris, highlighting the challenge of working with nuImages.

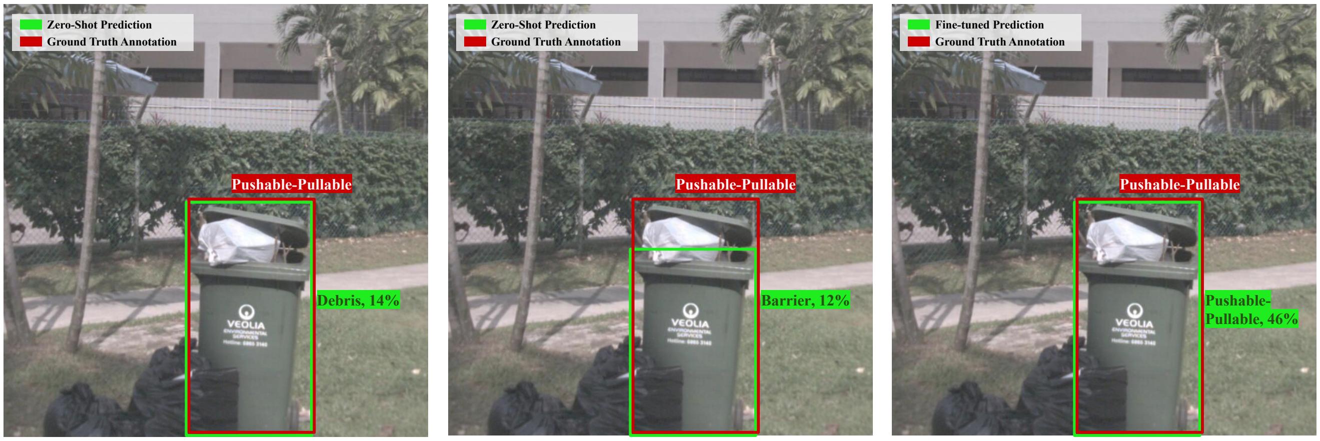
Approach	Average Precision (AP)			
	All	Many	Medium	Few
GroundingDINO (Zero-Shot) [23]	11.44	17.42	16.13	3.38
Detic (Zero-Shot) [48]	14.26	27.28	16.88	2.36
5-shots				
TFA [37] w/ COCO-base	1.33	2.78	1.43	0.23
TFA [37] w/ LVIS-base	2.02	1.69	4.08	0.58
TFA [37] w/ LVIS, IN-21K, COCO + CLIP Classifier	14.77	25.16	18.65	**3.63**
Ours	**15.94**	**28.47**	**19.53**	3.50
10-shots				
TFA [37] w/ COCO-base	1.21	2.55	1.19	0.31
TFA [37] w/ LVIS-base	2.27	2.05	4.51	0.58
TFA [37] w/ LVIS, IN-21K, COCO + CLIP Classifier	15.53	26.01	19.93	3.88
Ours	**16.67**	**29.15**	**20.66**	**3.90**
30-shots				
TFA [37] w/ COCO-base	1.14	2.81	0.84	0.23
TFA [37] w/ LVIS-base	2.23	1.48	4.98	0.45
TFA [37] w/ LVIS, IN-21K, COCO + CLIP Classifier	16.83	27.90	21.59	4.45
Ours	**17.87**	**30.32**	**22.35**	**4.70**

Further, fine-tuning Detic with pseudo-negatives improves rare class performance by 1.6% (30.0 vs 31.6) over naive fine-tuning. Finally, we note that simply replacing the ResNet-50 backbone with a Swin-B model yields a massive 12.8 AP improvement for rare classes (19.8 vs. 32.6).

4.3. Foundational FSOD with nuImages

In the context of foundational models, we argue that partitioning datasets into base and novel classes no longer makes sense. Instead, FSOD methods should only train on K-shot annotations for C target classes, and also evaluate performance on these C classes. We fine-tune Detic, pre-trained on LVIS, COCO and ImageNet-21K, on K exam-

[2] GroundingDINO is only pre-trained on Objects365, GoldG and Cap4M



GroundingDINO Zero-Shot **Detic Zero-Shot** **After Concept Alignment**

Figure 4. **Multi-Modal Fine-Tuning Improves Concept Alignment**. While VLMs demonstrate strong zero-shot capabilities, they struggle when the concept of the target class is different from VLM concepts encountered in web-scale training. Specifically, we find that GroundingDINO [23] (**left**) and Detic [48] (**center**) struggle to detect open-world categories like `pushable-pullable`. Fine-tuning Detic (**right**) with federated losses improves VLM concept alignment. Note that **red** boxes represent ground-truth annotations and **green** boxes are predictions by their respective models.

ples per class and highlight model performance in Table 3. Since the specific K examples can have a significant impact on the overall performance, we run each experiment over 10 random data splits and report the average. As expected, detection accuracy improves as we add more training examples. Despite large-scale pre-training, we see low accuracy for classes with `few` examples, highlighting the difficulty of the nuImages dataset.

High intra-class variance for categories such as `debris` makes it difficult to generalize given few examples. According to nuImages' annotation instructions, `debris` can include anything that is too big to be safely driven over, including *fallen tree branches* and *trash bags*. Similarly `pushable-pullable` includes *trash cans, luggage, dollies, wheel barrows*, and *shopping carts*.

To contextualize our results, we evaluate TFA [37] on the nuImages Foundational FSOD benchmark. We train two variants of TFA trained on `COCO-base` and `LVIS-base` and fine-tune both models on K examples of the nuImages classes. Surprisingly, both variants of TFA achieve less than 3 AP (Table 3). We posit that this is largely due to poor classification performance. Since both LVIS and COCO classes do not significantly overlap with nuImages classes, learning a classifier from few examples is extremely difficult. However, we find that simply re-training TFA with a frozen CLIP-based classifier (similar to Detic) dramatically increases performance, reiterating the utility of language and web-scale pre-training in data-constrained settings.

4.4. Oracle Performance Analysis

To further contextualize our results, we compute upper bounds when given access to ground-truth negatives and exhaustive annotations for the few-shot data split. Recall, nuImages is exhaustively annotated, but is repurposed for Foundational FSOD in our work.

To compute the set of ground-truth negatives, we use exhaustive ground-truth annotations to determine which categories are not present for each image. Note that this information doesn't exist in LVIS because its ground-truth is sparsely annotated. Training with ground-truth negatives provides an upper bound on our pseudo-negatives experiment. Next, we train using exhaustive ground-truth annotations to provide an upper bound for the specific set of images used during training. In addition, this experiment highlights the performance gap between having exhaustive negatives and exhaustive annotations.

Table 4 shows that using pseudo-negatives nearly matches the true negative upper bound (16.67 AP vs 16.99 AP). This demonstrates that we are able to reliably estimate negatives in an image, alleviating the problem of learning with sparse annotations. Training with exhaustive annotations yields significantly better results for `many` and `medium` classes. This is unsurprising because 10-shot FSOD includes 10 car annotations and exhaustively annotating the same images include over 550 car annotations!

Despite strong performance on classes with `many` and `medium`, the upper bound for classes with `few` examples remains low (4.21 AP and 3.93 AP). We posit that it is very hard to capture the correct semantics of nuImages' rare cat-

Table 4. **Analysis of nuImages Upper Bound Performance**. We compare the accuracy of our proposed approach against upper bounds computed for the FSOD task. Our pseudo-negatives strategy approaches the performance of using ground-truth negatives, showing that pesudo-labels can provide a reliable signal about negatives, especially across classes with `many` and `medium` examples. The performance gap between our best method and exhaustive annotations can be attributed to the large number of extra annotations, particularly for classes with `many` and `medium` examples. Compared to the baseline (14.3 AP), our approach (16.7 AP) closes the gap to the (18.5 AP) upperbound by over **50%**.

Approach	**10 Shots**: Average Precision (AP)			
	All	Many	Medium	Few
Detic (Zero-Shot) [48]	14.26	27.28	16.88	2.36
+ Fine-Tuning	15.53	26.01	19.93	3.88
w/ FedLoss	15.57	27.20	20.09	2.89
w/ Inverse FedLoss	15.89	27.56	20.19	3.42
w/ Pseudo-Negatives	**16.67**	**29.15**	**20.66**	**3.90**
w/ True Negatives (*Oracle*)	16.99	29.60	20.93	4.21
w/ Exhaustive Annotations (*Oracle*)	18.51	33.51	22.35	3.93

egories only using K-shots. We observe similar trends for the 5 and 30-shot cases and present further analysis in the supplement.

Given the success of training with pseudo-negatives, a natural next-step is to train with pseudo-positives. Our preliminary results suggest that incorporating pseudo-positives does not provide significant improvement over simply training with pseudo-negatives. We posit that training with incorrect pseudo-positives may incur a higher penalty than training with incorrect pseudo-negatives. This is a promising direction for future work.

4.5. Fine-Tuning Ablation

We explore different fine-tuning strategies for training Detic with few-shot annotations. We broadly divide Detic's architecture into four components: Backbone, Region Proposal Network (RPN), Box Regressor, and Classifier. We ablate the impact of freezing different components and present results in Table 5.

Intuitively, as we have limited training data, we attempt to fine-tune a minimal number of parameters. Initializing and freezing the classifier head with CLIP embeddings corresponding to class names provides the most significant improvement. Prior works that fine-tune vision-only models have no notion of language embeddings and therefore must train classifiers from scratch. In contrast, Detic can represent concept names using CLIP embeddings and can more easily adapt to `novel` categories with few examples. We find that freezing the backbone, RPN and classifier head with CLIP embeddings, and training the classifier projection layer and box regressor performs the best. Importantly, as Detic has been trained on large-scale datasets, its RPN can easily localize novel objects without fine-tuning.

Table 5. **Detic Fine-Tuning on nuImages**. ❄ denotes freezing parameters and **-** implies fine-tuning all parameters. The Detic classifier consists of a fully connected projection layer followed by a classifier head. **CLIP** signifies using CLIP embeddings for the classifier head and only training the classifier projection layer. We find that freezing the backbone and RPN and initializing the classifier head with CLIP embeddings performs the best.

Detic Components				**10 Shots**: Average Precision (AP)			
Backbone	RPN	Box Regressor	Classifier	All	Many	Medium	Few
-	-	-	-	12.11	19.41	18.44	0.87
❄	-	-	-	12.37	21.20	17.66	0.91
❄	-	-	CLIP	15.08	22.88	**20.99**	**3.78**
❄	❄	-	-	11.63	21.65	15.42	0.83
❄	❄	-	CLIP	**15.37**	**26.93**	19.73	2.83
❄	❄	❄	-	10.66	18.54	15.53	0.56
❄	❄	❄	CLIP	15.31	26.83	19.58	2.89

4.6. Limitations and Future Work

Despite using VLMs pre-trained on large-scale datasets, we find that performance for rare categories (as defined by the cardinality of each class in the original dataset) is considerably lower than for common classes. We posit that VLMs are pre-trained with imbalanced data which includes many examples of common categories like `truck` but few examples of rare categories like `stroller`. Our work does not explicitly improve detection performance on the rare classes. Interestingly, since VLMs like Detic [48], GLIP [20], and GroundingDINO [23] are trained with different data sources, each model has dramatically different zero-shot performance on novel categories like `stroller`. Ensembling predictions from different VLMs may yield better detection accuracy for rare categories. In addition, although our work motivates the use of rich textual descriptions for multi-modal alignment, our approach only uses class names as text features. We hope future work can address the above limitations.

5. Conclusion

We revisit few-shot object detection (FSOD) with vision-language models (VLMs) and find that zero-shot inference from web-scale VLMs significantly outperforms leading FSOD methods. But importantly, such foundational models do not fully address few shot recognition because of the *concept alignment* problem; particular concepts in target applications may be different than their use on web-scale datasets. Just as human annotators require concept alignment via multimodal text and visual examples, we argue that VLMs should be aligned with such few-shot data, formalizing the problem of Foundational FSOD. We also point out that existing FSOD benchmarks are actually federated datasets, and demonstrate that federated losses improve Foundational FSOD performance, approaching the oracle upper bound where few-shot images are fully annotated. Our analysis suggests that future few-shot (or "foundational concept alignment") benchmarks may benefit from assuming images are fully annotated, since the cost of annotating a small set of K images is manageable in practice.

References

[1] Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020. 4, 5

[2] Nadine Chang, Francesco Ferroni, Michael J Tarr, Martial Hebert, and Deva Ramanan. Thinking like an annotator: Generation of dataset labeling instructions. *arXiv preprint arXiv:2306.14035*, 2023. 2, 11

[3] Ding-Jie Chen, He-Yen Hsieh, and Tyng-Luh Liu. Adaptive image transformer for one-shot object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12247–12256, 2021. 2

[4] Yu Du, Fangyun Wei, Zihe Zhang, Miaojing Shi, Yue Gao, and Guoqi Li. Learning to prompt for open-vocabulary object detection with vision-language model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14084–14093, 2022. 3

[5] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *International Journal of Computer Vision*, 88(2): 303–338, 2010. 2, 3

[6] Qi Fan, Wei Zhuo, Chi-Keung Tang, and Yu-Wing Tai. Few-shot object detection with attention-rpn and multi-relation detector. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4013–4022, 2020. 3

[7] Zhibo Fan, Yuchen Ma, Zeming Li, and Jian Sun. Generalized few-shot object detection without forgetting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4527–4536, 2021. 5

[8] Pedro F Felzenszwalb, Ross B Girshick, David McAllester, and Deva Ramanan. Object detection with discriminatively trained part-based models. *IEEE transactions on pattern analysis and machine intelligence*, 32(9):1627–1645, 2009. 1

[9] Xiuye Gu, Tsung-Yi Lin, Weicheng Kuo, and Yin Cui. Open-vocabulary object detection via vision and language knowledge distillation. *arXiv preprint arXiv:2104.13921*, 2021. 3

[10] Xiuye Gu, Tsung-Yi Lin, Weicheng Kuo, and Yin Cui. Open-vocabulary object detection via vision and language knowledge distillation. *arXiv preprint arXiv:2104.13921*, 2021. 3

[11] Agrim Gupta, Piotr Dollar, and Ross Girshick. Lvis: A dataset for large vocabulary instance segmentation. In *CVPR*, 2019. 2, 3, 4, 5, 11

[12] Ting-I Hsieh, Yi-Chen Lo, Hwann-Tzong Chen, and Tyng-Luh Liu. One-shot object detection with co-attention and co-excitation. *Advances in neural information processing systems*, 32, 2019. 2

[13] Bingyi Kang, Zhuang Liu, Xin Wang, Fisher Yu, Jiashi Feng, and Trevor Darrell. Few-shot object detection via feature reweighting. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8420–8429, 2019. 3

[14] Gaoussou Youssouf Kebe, Padraig Higgins, Patrick Jenkins, Kasra Darvish, Rishabh Sachdeva, Ryan Barron, John Winder, Donald Engel, Edward Raff, Francis Ferraro, and Cynthia Matuszek. A spoken language dataset of descriptions for speech-based grounded language learning. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1)*, 2021. 2, 5

[15] Youngdong Kim, Junho Yim, Juseung Yun, and Junmo Kim. Nlnl: Negative learning for noisy labels. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 101–110, 2019. 2, 3

[16] Mona Köhler, Markus Eisenbach, and Horst-Michael Gross. Few-shot object detection: A comprehensive survey. *arXiv preprint arXiv:2112.11699*, 2021. 3

[17] Weicheng Kuo, Yin Cui, Xiuye Gu, AJ Piergiovanni, and Anelia Angelova. F-vlm: Open-vocabulary object detection upon frozen vision and language models. *arXiv preprint arXiv:2209.15639*, 2022. 3

[18] Dong-Hyun Lee et al. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. 3

[19] Bohao Li, Boyu Yang, Chang Liu, Feng Liu, Rongrong Ji, and Qixiang Ye. Beyond max-margin: Class margin equilibrium for few-shot object detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 7363–7372, 2021. 3

[20] Liunian Harold Li, Pengchuan Zhang, Haotian Zhang, Jianwei Yang, Chunyuan Li, Yiwu Zhong, Lijuan Wang, Lu Yuan, Lei Zhang, Jenq-Neng Hwang, et al. Grounded language-image pre-training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10965–10975, 2022. 2, 3, 5, 8

[21] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft coco: Common objects in context. In *European Conference on Computer Vision*, 2014. 1, 2, 3

[22] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *ICCV*, 2017. 1

[23] Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Chunyuan Li, Jianwei Yang, Hang Su, Jun Zhu, et al. Grounding dino: Marrying dino with grounded pre-training for open-set object detection. *arXiv preprint arXiv:2303.05499*, 2023. 4, 5, 6, 7, 8

[24] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *ECCV*, 2016. 1

[25] Jiawei Ma, Yulei Niu, Jincheng Xu, Shiyuan Huang, Guangxing Han, and Shih-Fu Chang. Digeo: Discriminative geometry-aware learning for generalized few-shot object detection. In *Proceedings of the IEEE/CVF Conference*

on Computer Vision and Pattern Recognition, pages 3208–3218, 2023. 5, 6

[26] Matthias Minderer, Alexey Gritsenko, Austin Stone, Maxim Neumann, Dirk Weissenborn, Alexey Dosovitskiy, Aravindh Mahendran, Anurag Arnab, Mostafa Dehghani, Zhuoran Shen, et al. Simple open-vocabulary object detection. In *European Conference on Computer Vision*, pages 728–755. Springer, 2022. 3

[27] Matthias Minderer, Alexey Gritsenko, and Neil Houlsby. Scaling open-vocabulary object detection. *arXiv preprint arXiv:2306.09683*, 2023. 3

[28] Neehar Peri, Achal Dave, Deva Ramanan, and Shu Kong. Towards long-tailed 3d detection. In *Conference on Robot Learning*, pages 1904–1915. PMLR, 2023. 5, 6

[29] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021. 2, 4

[30] Joseph Redmon and Ali Farhadi. Yolo9000: better, faster, stronger. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7263–7271, 2017. 1

[31] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems*, 2015. 1

[32] Shuai Shao, Zeming Li, Tianyuan Zhang, Chao Peng, Gang Yu, Xiangyu Zhang, Jing Li, and Jian Sun. Objects365: A large-scale, high-quality dataset for object detection. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 8429–8438, 2019. 2

[33] Shuai Shao, Zeming Li, Tianyuan Zhang, Chao Peng, Gang Yu, Xiangyu Zhang, Jing Li, and Jian Sun. Objects365: A large-scale, high-quality dataset for object detection. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 8430–8439, 2019. 5

[34] Piyush Sharma, Nan Ding, Sebastian Goodman, and Radu Soricut. Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2556–2565, Melbourne, Australia, 2018. Association for Computational Linguistics. 2, 13

[35] Bo Sun, Banghuai Li, Shengcai Cai, Ye Yuan, and Chi Zhang. Fsce: Few-shot object detection via contrastive proposal encoding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7352–7362, 2021. 3

[36] Jingru Tan, Changbao Wang, Buyu Li, Quanquan Li, Wanli Ouyang, Changqing Yin, and Junjie Yan. Equalization loss for long-tailed object recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11662–11671, 2020. 3, 4

[37] Xin Wang, Thomas E Huang, Trevor Darrell, Joseph E Gonzalez, and Fisher Yu. Frustratingly simple few-shot object detection. *arXiv preprint arXiv:2003.06957*, 2020. 3, 4, 5, 6, 7, 11

[38] Yuchao Wang, Haochen Wang, Yujun Shen, Jingjing Fei, Wei Li, Guoqiang Jin, Liwei Wu, Rui Zhao, and Xinyi Le. Semi-supervised semantic segmentation using unreliable pseudo-labels. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4248–4257, 2022. 2

[39] Xiu-Shen Wei, H-Y Xu, Faen Zhang, Yuxin Peng, and Wei Zhou. An embarrassingly simple approach to semi-supervised few-shot learning. *Advances in Neural Information Processing Systems*, 35:14489–14500, 2022. 2

[40] Aming Wu, Yahong Han, Linchao Zhu, and Yi Yang. Universal-prototype enhancing for few-shot object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9567–9576, 2021. 3

[41] Jiaxi Wu, Songtao Liu, Di Huang, and Yunhong Wang. Multi-scale positive sample refinement for few-shot object detection. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XVI 16*, pages 456–472. Springer, 2020. 5

[42] Yang Xiao, Vincent Lepetit, and Renaud Marlet. Few-shot object detection and viewpoint estimation for objects in the wild. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(3):3090–3106, 2022. 3, 5

[43] Qizhe Xie, Minh-Thang Luong, Eduard Hovy, and Quoc V Le. Self-training with noisy student improves imagenet classification. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10687–10698, 2020. 3

[44] Jingyi Xu, Hieu Le, and Dimitris Samaras. Generating features with increased crop-related diversity for few-shot object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19713–19722, 2023. 3

[45] Xiaopeng Yan, Ziliang Chen, Anni Xu, Xiaoxi Wang, Xiaodan Liang, and Liang Lin. Meta r-cnn: Towards general solver for instance-level low-shot learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9577–9586, 2019. 5

[46] Yiwu Zhong, Jianwei Yang, Pengchuan Zhang, Chunyuan Li, Noel Codella, Liunian Harold Li, Luowei Zhou, Xiyang Dai, Lu Yuan, Yin Li, et al. Regionclip: Region-based language-image pretraining. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16793–16803, 2022. 3, 13

[47] Xingyi Zhou, Vladlen Koltun, and Philipp Krähenbühl. Probabilistic two-stage detection. In *arXiv preprint arXiv:2103.07461*, 2021. 2, 3, 5, 11

[48] Xingyi Zhou, Rohit Girdhar, Armand Joulin, Philipp Krähenbühl, and Ishan Misra. Detecting twenty-thousand classes using image-level supervision. In *European Conference on Computer Vision*, pages 350–368. Springer, 2022. 3, 4, 6, 7, 8, 12, 13

[49] Chenchen Zhu, Fangyi Chen, Uzair Ahmed, Zhiqiang Shen, and Marios Savvides. Semantic relation reasoning for shot-stable few-shot object detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8782–8791, 2021. 2

A. Implementation Details

Pseudo-Negative Federated Loss. First, we sample an image at random. We start by running a pre-trained detector on the image to generate predictions. We partition the predictions into two groups based on a confidence threshold; all predictions above the threshold are pseudo-positives (`pseudo_pos`), all predictions below the threshold are ignored. Next, we compute negative classes (`neg_classes`) based on the pseudo-positives. Specifically, all classes not included in the psuedo-positive predictions are considered negative classes. Following the federated loss described in [47] (modified to use `neg_classes` instead of randomly sampled negative classes), we compute the binary cross entropy loss for the negative (`neg_classes`) and ground truth classes. See Algorithm 1 for pseudo-code.

Algorithm 1: Psuedo-Negative Federated Loss

```
# Inputs
# img: Randomly Sampled Image
# all_classes: All Classes in Dataset
# gt: Ground Truth Annotations for img
# gt_classes: List of Classes in gt
#
# Outputs
# loss: Psuedo-Negative Federated Loss
#
# Functions
# filter: Returns All Predictions w/
#        Confidence > Threshold
# get_neg: Returns List of Classes Not
#          In Pseudo-Positives
# or: Set Union Operation
# BCE: Binary Cross Entropy Loss

#Step 1: Compute Predictions
#        and Filter by Confidence
pred = Detector(img) # predictions
pseudo_pos = filter(pred, thresh=0.2)

#Step 2: Get Pseudo-Negatives for Image
neg_classes = get_neg(pseudo_pos, all_classes)
select_classes = or(neg_classes gt_classes)

#Step 3: Compute Deterministic Federated Loss
#         w/ Pseudo-Negatives
loss = 0
for cls in select_classes:
    pred_cls = pred[cls] #predictions for cls
    gt_cls = gt[cls] #ground-truth for cls

    loss += BCE(pred_cls, gt_cls)

return loss
```

LVIS v0.5 Experiment Details. We select Detic with a Resnet-50 backbone for fair comparison with prior work. We pre-train Detic on `LVIS-base` for $90k$ iterations with a batch size of 32 using an AdamW optimizer and a learning rate of $2e-3$. All images are resized to 640×640 and we also enable Repeat Factor Sampling [11].

Following [37], we sample *up to* 10 shots for each class in LVIS (since all classes may not have 10 examples). We use a batch size of 32, learning rate of $2.5e-5$ for $46k$ iterations. We do not use Repeat Factor Sampling for fine-tuning. We sample 50 categories for each training image, i.e $|S| = 50$ for the FedLoss and InvFedLoss experiments. We derive negatives from pseudolabels with atleast 20% confidence for the Psuedo-Negative experiment.

nuImages Experiment Details. We select Detic with a Swin-T backbone pre-trained on `LVIS+COCO` and ImageNet-21k data. We use an image size of 1600×900, batch size of 8 and an AdamW optimizer with learning rate of $3.75e-6$. We fine-tune this model for 8000 iterations on nuImages. We sample 6 categories for each training image, i.e $|S| = 6$ for the FedLoss and InvFedLoss experiments. We derive negatives from pseudolabels with atleast 20% confidence for the Psuedo-Negative experiment.

The original Detic implementation samples $|S|$ categories per batch, rather than per image. This works for LVIS, which has 1200 categories. However, nuImages only has 18 classes, which required re-implementing the negative category sampling step for each image.

B. Analysis of Iconic Few-Shot Images

The specific examples used during few-shot fine-tuning significantly impacts target class performance [37]. However, prior work constructs few-shot splits by randomly sampling K examples per class. In contrast, when creating annotator *instructions*, selecting the right examples to "align" human annotators [2] to subtle aspects of the target concept is carefully considered. To more closely match VLM *concept alignment* with human annotator alignment, we design a simple algorithm to construct the best K-shot split for fine-tuning. This allows us to understand which examples are most informative and measure an upper bound in performance.

We construct our *best split* by picking examples corresponding to the best class-wise performance, based on the evaluation of each split on a held-out validation set. For instance, out of 10 random splits for the 5-shot task, one may pick `car` examples from split 1, `bicycle` from split 4 and `debris` from split 8. In Table 6, we observe that the *best-split* performance is always better than its random counterpart. As expected, the choice of examples in 5-shot case is more important than the 30-shot case (1.63 AP difference for 5-shot vs 0.08 AP for 30-shots). We visualize the difference in the splits for the stroller class in nuImages (See Figure 5). Unsurprisingly, iconic examples are large and unoccluded.

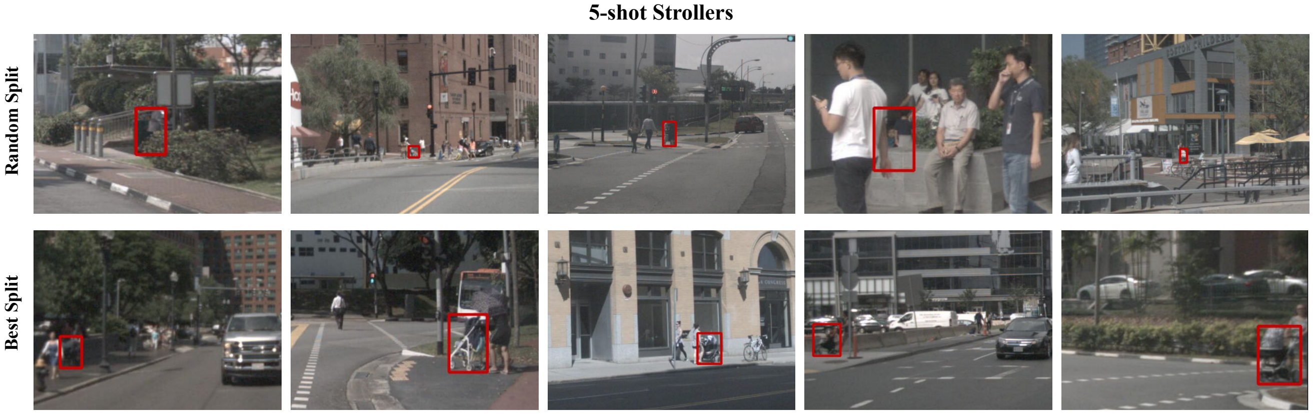


Figure 5. **Visualizing Random and Best Split**. In the top row, we visualize the 5-shot training examples of strollers from a *random split*. Similarly, we visualize the 5-shot training examples for the *best split* in the bottom row. We observe that strollers in the *random split* are often occluded, small in size and are blurry, making few-shot learning harder. On the other hand, the *best split* examples are larger, have better visual quality and are relatively un-occluded. This visual difference directly translates into better few-shot performance. We achieve **13.09 Stroller AP** for the *random split* and **18.54 Stroller AP** for the *best split*. We perform a more comprehensive evaluation in Table 6.

Table 6. **Random Split vs 'Best" Split**. We construct the "best" split by selecting per-class few-shot examples that lead to the highest performance on a held out set. Unsurprisingly, the best split performs better than any random split, especially for very limited data settings (e.g. 5-shot detection). This evaluation setting closely mimics how human annotators "align" to target concepts, since annotator guides are constructed using hand-picked iconic visual examples.

Approach	Average Precision (AP)			
	All	Many	Medium	Few
Detic (Zero-Shot) [48]	14.26	27.28	16.88	2.36
5-shots				
Ours *(Random Split)*	15.94	28.47	19.53	3.50
Ours *(Best Split)*	**17.57**	**29.99**	**20.70**	**4.92**
10-shots				
Ours *(Random Split)*	16.67	29.15	20.66	3.90
Ours *(Best Split)*	**17.51**	**29.98**	**21.60**	**4.58**
30-shots				
Ours *(Random Split)*	17.87	**30.32**	22.35	4.70
Ours *(Best Split)*	**17.95**	29.50	**22.69**	**4.85**

C. nuImages Foundational FSOD Performance for K={5, 30} Shots

We observe similar trends for $K = \{5, 30\}$ to the 10-shot performance reported in the main paper. Note that the reported results are averaged over 10 seeds to reduce variance.

For $K = 5$ shots, we note that naive fine-tuning barely improves the overall AP (cf. Table 7). Incorporating pseudo-negatives in this data-scarce setting closes the gap to the true negatives upper bound (15.94 vs 16.15 AP). Even with exhaustive annotations, we only improve by ~ 3.3 AP (14.26 vs 17.59 AP) points over the zero-shot results.

Table 7. **Analysis of 5-shot Performance on nuImages**. Fine-tuning with pseudo-negatives improves over naive fine-tuning. Importantly, our pseudo-negative approach nearly matches the upper bound with true negatives (15.94 AP vs 16.15 AP). This suggests that our proposed approach is particularly effective in data-constrained settings.

Approach	**5 shots**: Average Precision (AP)			
	All	Many	Medium	Few
Detic [48] Zero-Shot	14.26	27.28	16.88	2.36
+ Fine-tuning	14.77	25.16	18.65	**3.63**
w/ FedLoss	15.03	26.41	19.07	3.03
w/ Inverse FedLoss	15.21	26.77	19.01	3.29
w/ Pseudo-Negatives	**15.94**	**28.47**	**19.53**	3.50
w/ True Negatives *(Oracle)*	16.15	29.00	19.52	3.70
w/ Exhaustive Annotations *(Oracle)*	17.59	32.66	20.93	3.43

Naive fine-tuning improves over zero shot results by ~ 2.5 AP (16.83 vs 14.26 AP) in the $K = 30$ shot setting. In addition, we see diminishing benefits for using FedLoss variants (cf. Table 8). Nevertheless, pseudo-negatives improve over naive fine-tuning by 1 AP (16.83 vs 17.87 AP). Unsurprisingly, performance for classes with many examples increases the most (19.40 AP) when training with exhaustive annotations. Importantly, training with exhaustive annotations uses orders of magnitude more examples (1300 examples of carcar) than with few-shot annotations (30 examples of car).

D. RegionCLIP Experiments

In this section, we evaluate the importance of using box supervised data in pre-training. Unlike Detic, which trains on box-supervised data from LVIS, COCO and image-text

Table 8. **Analysis of 30-shot performance on nuImages** Naive fine-tuning provides a considerable improvement over zero-shot inference due to a greater number of examples per-class. In constrast to 5-shot results, FedLoss variants provide limited improvement over naive fine-tuning. We find that fine-tuning with pseudo-negatives provides a 1% improvement overall.

Approach	**30 shots**: Average Precision (AP)			
	`All`	`Many`	`Medium`	`Few`
Detic [48] Zero-Shot	14.26	27.28	16.88	2.36
+ Fine-tuning	16.83	27.90	21.59	4.45
w/ FedLoss	16.45	28.88	21.14	3.02
w/ Inverse FedLoss	16.88	29.19	21.41	3.77
w/ Pseudo-Negatives	**17.87**	**30.32**	**22.35**	**4.70**
w/ True Negatives *(Oracle)*	18.19	30.61	22.66	5.11
w/ Exhaustive Annotations *(Oracle)*	19.40	34.24	23.75	4.48

data from `ImageNet21-k`, RegionCLIP[46] only pre-trains on image-text pairs from the Conceptual Caption (CC3M) dataset [34].

We report RegionCLIP's zero-shot and fine-tuning performance on nuImages averaged over 3 random splits in Table 9. Detic zero-shot outperforms RegionCLIP zero-shot by $\sim$ 12 AP (14.26 vs 2.34). While fine-tuning RegionCLIP improves overall performance, Detic achieves higher accuracy for $K = \{5, 10, 30\}$ shots. This highlights the importance supervision type (i.e box-supervised data) and data scale used for pre-training. We find that box-supervised pre-training yields better down-stream performance on FSOD.

Next, we conduct further analysis to diagnose why RegionCLIP zero-shot inference performs so poorly on nuImages (Table 10). RegionCLIP relies on an RPN pre-trained on box-supervised data like `LVIS-base` to extract regions for pre-training. Notably, RegionCLIP (w/ `LVIS-RPN`: 2.34 AP) suffers from poor localization and foreground-vs-background classification compared to Detic. We validate this hypothesis by evaluating RegionCLIP (w/ `GT-RPN)` to measure classification performance. Surprisingly, RegionCLIP achieves significantly higher accuracy (26.44 AP), confirming that RegionCLIP struggles to localize objects in nuImages. This observation highlights the challenge of working with nuImages categories, further motivating our Foundational FSOD benchmark.

Table 9. **RegionCLIP Experiments**. RegionCLIP zero-shot inference performs much worse than Detic. While fine-tuning improves RegionCLIP's performance, it still lags far behind Detic. We posit that this performance difference can be attributed to Detic's box-supervised pre-training and use of language cues from CLIP embeddings.

Approach	Average Precision (AP)			
	`All`	`Many`	`Medium`	`Few`
RegionCLIP (Zero-Shot) [46]	2.34	3.33	3.87	0.22
Detic (Zero-Shot) [48]	**14.26**	**27.28**	**16.88**	**2.36**
5-shots				
RegionCLIP (Fine-Tuning) [46]	3.61	6.20	5.14	0.26
Detic (Fine-Tuning) [48]	**14.50**	**24.09**	**18.53**	**3.70**
10-shots				
RegionCLIP (Fine-Tuning) [46]	3.58	6.10	5.16	0.24
Detic (Fine-Tuning) [48]	**15.28**	**26.93**	**19.89**	**3.27**
30-shots				
RegionCLIP (Fine-Tuning) [46]	3.57	6.13	5.10	0.22
Detic (Fine-Tuning) [48]	**16.65**	**27.45**	**21.51**	**4.02**

Table 10. **Diagnosing RegionCLIP's Poor Zero-Shot Performance**. RegionCLIP's zero-shot performance lags far behind Detic. Using RegionCLIP's classifier on ground-truth region proposals yields high performance, suggesting that RegionCLIP struggles to accuratly localize objects and distinguish between foreground-vs-background.

Approach	Average Precision (AP)			
	`All`	`Many`	`Medium`	`Few`
Detic *(Zero-Shot)* [48]	14.26	27.28	16.88	2.36
RegionCLIP *(Zero-Shot)* w/ `LVIS-RPN` [46]	2.34	3.33	3.87	0.22
RegionCLIP *(Zero-Shot)* w/ `Detic-RPN` [46]	3.79	6.68	3.91	1.12
RegionCLIP *(Zero-Shot)* w/ `Detic-RPN, 0.5` [46]	7.64	12.81	9.57	1.88
RegionCLIP *(Zero-Shot)* w/ `GT-RPN` [46]	26.44	45.33	31.83	3.92

Lastly, we evaluate RegionCLIP's performance with `Detic-RPN`. Notably, we observe that the performance improves over RegionCLIP w/ `LVIS-RPN` demonstrating that improving localization quality yields better performance. Furthermore, we filter out low confidence Detic proposals , i.e $<$ 0.5 objectness score (w/ `Detic-RPN, 0.5`) and find that this doubles RegionCLIP's zero-shot performance to 7.64 AP.

E. NuImages Per-Split Breakdown

We provide per-split breakdowns for the nuImage experiments in Tables 11, 12, and 13.

Table 11. **5-shot nuImages Per-Split Performance**

Approach	Split 1				Split 2				Split 3				Split 4				Split 5			
	All	Many	Med	Few	All	Many	Med	Few	All	Many	Med	Few	All	Many	Med	Few	All	Many	Med	Few
Zero-Shot	14.26	27.28	16.88	2.36	14.26	27.28	16.88	2.36	14.26	27.28	16.88	2.36	14.26	27.28	16.88	2.36	14.26	27.28	16.88	2.36
Fine-tuning	15.07	24.89	18.82	**4.58**	13.27	20.83	17.54	**3.58**	15.15	26.55	19.23	2.93	14.82	25.47	19.10	3.20	14.76	23.96	19.45	3.69
FedLoss	15.27	26.59	**19.25**	3.41	15.08	27.23	18.59	2.80	15.22	26.69	19.34	2.97	14.72	25.42	19.44	2.65	15.03	25.87	19.39	3.01
InvFedLoss	15.36	28.54	18.59	2.77	15.07	28.17	18.04	2.71	15.40	27.91	18.96	2.94	15.04	26.97	18.95	2.80	15.55	28.28	18.93	3.12
Pseudo-Negatives	**15.87**	**28.59**	19.06	3.67	**15.88**	**28.70**	**19.28**	3.30	**16.12**	**27.94**	**20.01**	**3.84**	**15.76**	**28.00**	**19.65**	**3.37**	**16.21**	**28.43**	**19.90**	**3.80**
True Negatives (Oracle)	16.27	28.55	19.35	4.56	15.92	29.03	19.24	3.22	15.97	28.81	19.42	3.29	15.92	28.61	19.57	3.41	16.44	28.91	20.1	3.92
Exh. Annotations (Oracle)	17.59	32.63	20.3	3.97	17.71	32.91	21.23	3.31	17.13	32.03	20.42	3.02	17.81	32.86	21.51	3.41	17.72	32.88	20.99	3.48

Approach	Split 6				Split 7				Split 8				Split 9				Split 10			
	All	Many	Med	Few	All	Many	Med	Few	All	Many	Med	Few	All	Many	Med	Few	All	Many	Med	Few
Zero-Shot	14.26	27.28	16.88	2.36	14.26	27.28	16.88	2.36	14.26	27.28	16.88	2.36	14.26	27.28	16.88	2.36	14.26	27.28	16.88	2.36
Fine-Tuning	14.94	24.77	19.37	**3.84**	14.58	25.02	18.26	3.54	15.15	26.56	19.14	3.38	15.39	26.42	18.90	**4.20**	14.58	27.13	16.73	**3.38**
FedLoss	15.27	26.30	19.82	3.06	14.88	27.23	18.41	2.59	14.85	25.27	19.68	3.04	15.25	26.40	**19.15**	3.58	14.79	27.09	17.61	3.16
InvFedLoss	14.65	23.63	19.56	3.67	14.21	23.34	18.73	**3.55**	14.86	24.71	19.82	**3.44**	**15.74**	27.73	19.13	3.97	15.29	28.43	18.19	3.02
Pseudo-Negatives	**16.30**	**27.66**	**20.89**	3.82	**15.82**	**28.69**	**19.00**	3.47	**15.82**	**27.93**	**20.11**	3.28	15.73	**29.08**	18.95	3.04	**15.87**	**29.67**	**18.46**	3.37
True Negatives (Oracle)	16.46	29.5	20.38	3.43	15.95	29.54	18.94	3.26	16.23	28.4	20.14	4.18	16.52	29.4	19.77	4.12	15.8	29.3	18.31	3.6
Exh. Annotations (Oracle)	18.07	32.93	21.84	3.7	17.41	32.67	20.71	3.17	17.54	32.81	20.77	3.39	17.4	32.45	20.26	3.69	17.54	32.46	21.27	3.14

Table 12. **10-shot nuImages Per-Split Performance**

Approach	Split 1				Split 2				Split 3				Split 4				Split 5			
	All	Many	Med	Few	All	Many	Med	Few	All	Many	Med	Few	All	Many	Med	Few	All	Many	Med	Few
Zero-Shot	14.26	27.28	16.88	2.36	14.26	27.28	16.88	2.36	14.26	27.28	16.88	2.36	14.26	27.28	16.88	2.36	14.26	27.28	16.88	2.36
Fine-Tuning	15.77	26.63	20.13	3.87	15.31	26.85	19.46	3.03	15.64	27.32	20.08	2.91	15.78	25.16	20.79	**4.44**	15.93	27.34	20.28	3.60
FedLoss	15.66	27.74	20.05	2.83	15.53	25.71	20.84	3.07	15.62	28.49	19.69	2.43	15.63	27.14	20.31	2.91	15.72	25.22	**21.37**	3.54
InvFedLoss	15.96	27.22	20.51	3.66	15.80	26.05	**20.88**	**3.54**	15.85	28.71	19.97	2.62	16.04	27.43	20.49	3.68	16.26	27.05	21.22	3.80
Pseudo-Negatives	**16.78**	**29.33**	**20.64**	**4.14**	**16.58**	**29.73**	20.53	3.36	**16.51**	**28.95**	**20.72**	**3.44**	**16.95**	**28.73**	**21.49**	4.12	**17.08**	**29.84**	21.22	**3.90**
True Negatives (Oracle)	16.91	29.67	20.67	4.19	16.92	29.75	21.08	3.73	16.57	29.69	20.38	3.4	17.36	29.96	21.24	4.67	17.52	29.46	22.13	4.61
Exh. Annotations (Oracle)	18.03	33.17	21.33	3.77	18.68	33.98	22.52	3.88	18.3	33.2	22.37	3.45	18.44	33.47	22	4	19.08	33.71	23.36	4.42

Approach	Split 6				Split 7				Split 8				Split 9				Split 10			
	All	Many	Med	Few	All	Many	Med	Few	All	Many	Med	Few	All	Many	Med	Few	All	Many	Med	Few
Zero-Shot	14.26	27.28	16.88	2.36	14.26	27.28	16.88	2.36	14.26	27.28	16.88	2.36	14.26	27.28	16.88	2.36	14.26	27.28	16.88	2.36
Fine-Tuning	15.22	23.86	**21.10**	**3.65**	15.00	22.86	19.67	**5.07**	15.67	27.42	19.67	**3.55**	15.25	25.67	19.61	**3.60**	15.73	27.02	18.52	5.10
FedLoss	15.79	26.93	21.08	2.82	15.57	27.22	19.76	3.17	15.47	27.98	19.46	2.73	15.32	27.02	19.92	2.44	15.40	28.60	18.45	2.92
InvFedLoss	16.07	27.49	21.09	3.15	15.93	**28.80**	19.43	3.31	15.69	28.38	19.45	3.04	15.29	25.94	20.12	2.94	15.99	28.50	18.72	4.43
Pseudo-Negatives	**16.60**	**29.51**	20.83	3.33	**16.50**	27.91	**20.24**	4.82	**16.51**	**29.26**	**20.61**	3.50	**16.37**	**28.86**	**20.75**	3.09	**16.82**	**29.37**	**19.56**	**5.27**
True Negatives (Oracle)	17.06	29.29	22.12	3.59	16.43	29.9	19.55	3.76	17.04	29.85	20.77	4.46	17.11	29.38	21.48	4.02	16.96	29.04	19.86	5.63
Exh. Annotations (Oracle)	18.62	33.65	23.09	3.45	18.75	33.83	22.2	4.51	17.99	33.11	21.45	3.78	18.33	33.18	22.47	3.51	18.94	33.77	22.67	4.55

Table 13. **30-shot nuImages Per-Split Performance**

Approach	Split 1				Split 2				Split 3				Split 4				Split 5			
	All	Many	Med	Few	All	Many	Med	Few	All	Many	Med	Few	All	Many	Med	Few	All	Many	Med	Few
Zero-Shot	14.26	27.28	16.88	2.36	14.26	27.28	16.88	2.36	14.26	27.28	16.88	2.36	14.26	27.28	16.88	2.36	14.26	27.28	16.88	2.36
Fine-Tuning	16.71	28.14	21.37	4.14	15.94	26.49	21.04	3.67	16.95	27.73	**22.11**	4.24	16.33	28.71	20.41	3.52	17.46	28.50	22.64	4.68
FedLoss	16.20	28.86	20.76	2.75	15.48	27.30	20.41	2.31	16.59	28.80	21.71	2.82	16.62	28.85	21.45	3.09	17.15	29.59	21.94	3.53
InvFedLoss	16.47	29.26	20.70	3.28	16.09	27.10	21.52	3.09	16.78	29.52	21.18	3.40	16.97	29.25	21.39	3.90	17.57	29.84	22.14	4.35
Pseudo-Negatives	**18.14**	**30.45**	**22.33**	**5.38**	**17.34**	**29.24**	**22.53**	**3.91**	**17.90**	**31.09**	22.03	**4.42**	**17.83**	**30.44**	**22.34**	**4.35**	**18.33**	**30.78**	**23.04**	**4.87**
True Negatives (Oracle)	18.34	30.55	22.52	5.68	17.82	29.46	22.89	4.83	17.76	31.09	21.93	4.15	18.36	30.99	22.61	5.18	18.67	31.03	23.59	5.18
Exh. Annotations (Oracle)	19.48	34.22	23.24	5.12	19.43	34.39	23.81	4.47	19.33	34.3	23.97	3.98	19.21	34.18	23.44	4.15	19.6	34.2	24.22	4.62

Approach	Split 6				Split 7				Split 8				Split 9				Split 10			
	All	Many	Med	Few	All	Many	Med	Few	All	Many	Med	Few	All	Many	Med	Few	All	Many	Med	Few
Zero-Shot	14.26	27.28	16.88	2.36	14.26	27.28	16.88	2.36	14.26	27.28	16.88	2.36	14.26	27.28	16.88	2.36	14.26	27.28	16.88	2.36
Fine-Tuning	16.66	28.51	21.51	3.59	17.30	28.02	21.89	**5.53**	17.19	27.68	**22.42**	**5.01**	17.05	28.11	21.84	4.56	16.68	27.12	20.67	**5.57**
FedLoss	17.01	28.95	22.57	3.08	16.22	29.48	19.98	3.14	16.46	29.15	21.14	3.03	16.47	29.46	20.89	2.84	16.30	28.33	20.56	3.59
InvFedLoss	17.03	29.64	22.03	3.16	17.36	29.69	21.66	4.60	17.05	29.18	21.90	4.02	16.65	29.73	20.88	3.18	16.82	28.65	20.66	4.77
Pseudo-Negatives	**18.08**	**30.58**	**23.00**	**4.39**	**18.05**	**30.20**	**22.37**	5.43	**17.61**	**30.47**	22.01	4.33	**17.87**	**30.13**	**22.50**	**4.64**	**17.56**	**29.77**	**21.34**	5.29
True Negatives (Oracle)	17.79	30.57	22.92	3.71	18.44	30.47	22.66	6.07	18.3	30.83	22.72	5.34	18.39	30.74	23.02	5.08	18.07	30.35	21.8	5.88
Exh. Annotations (Oracle)	19.32	34.21	24.33	3.71	19.47	34.47	23.38	4.91	19.4	34.17	23.7	4.66	19.47	34.12	23.94	4.57	19.34	34.17	23.4	4.65