Incremental Few-Shot Instance Segmentation

Dan Andrei Ganea
Utrecht University
dan.andrei.ganea@gmail.com

Bas Boom
Cyclomedia Technology
bboom@cyclomedia.com

Ronald Poppe
Utrecht University
r.w.poppe@uu.nl

Abstract

Few-shot instance segmentation methods are promising when labeled training data for novel classes is scarce. However, current approaches do not facilitate flexible addition of novel classes. They also require that examples of each class are provided at train and test time, which is memory intensive. In this paper, we address these limitations by presenting the first incremental approach to few-shot instance segmentation: iMTFA. We learn discriminative embeddings for object instances that are merged into class representatives. Storing embedding vectors rather than images effectively solves the memory overhead problem. We match these class embeddings at the RoI-level using cosine similarity. This allows us to add new classes without the need for further training or access to previous training data. In a series of experiments, we consistently outperform the current state-of-the-art. Moreover, the reduced memory requirements allow us to evaluate, for the first time, few-shot instance segmentation performance on all classes in COCO jointly[1].

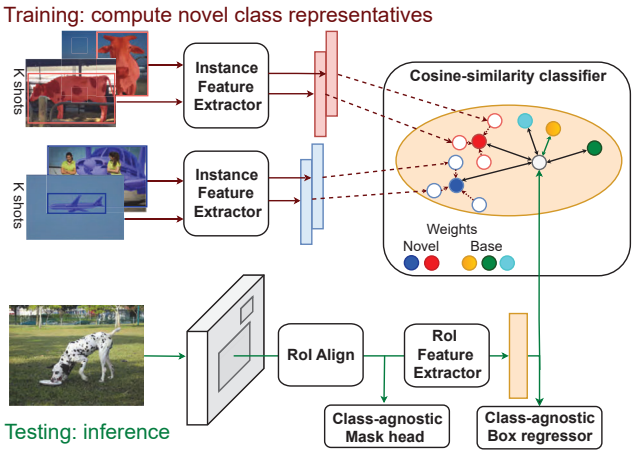

Figure 1. **Incremental few-shot instance segmentation**. For all K instances of each novel class, we produce vector embeddings using an Instance Feature Extractor. The average of these embeddings is stored as a per-class weight vector inside a cosine-similarity classifier. At test time (green), we compare the cosine distance embeddings of object proposal to the per-class weights.

1. Introduction

Convolutional neural networks (CNNs) have led to state-of-the-art results for image classification [16, 32], object detection [28] and instance segmentation [11]. In general, performance increases with network depth and training set size. While we can usually rely on large annotated databases for more general classes, adding a class for which we have little training data available is challenging. For example, we typically have a modest number of labeled training images when adding new classes for state-specific street furniture for self-driving cars, or types of weapons for automated detection in social media videos. Especially for instance segmentation, obtaining pixel-level annotations is costly.

Few-shot learning addresses the problem of learning with limited available data. Typically, one assumes the existence of a set of *base classes*, for which there exist numerous training samples, and a disjoint set of *novel classes*, for which training data is scarce (K examples). The goal is to train a system to correctly classify N classes: only the novel classes, or both novel and base classes jointly.

Compared to few-shot image classification, few-shot object detection (FSOD) and few-shot instance segmentation (FSIS) have received significantly less attention. While the few solutions that have been introduced show great promise, there is room for improvement in terms of practicality and accuracy. Often, long training procedures with both novel and base class samples are required [13, 36, 39]. This is unpractical when we flexibly want to add novel classes to a trained network. In *incremental* few-shot learning, the addition of novel classes is independent from previous data, so computation time is reduced.

In this paper, we introduce the first incremental few-shot instance segmentation method: iMTFA (Figure 1). We employ a two-stage training and fine-tuning approach based on Mask R-CNN [11]. The first stage trains the Mask R-CNN network. In the second stage, the fully-connected layers

[1]Code available at: https://github.com/danganea/iMTFA

at the region of interest (RoI) level are re-purposed. Essentially, we transform a fixed feature extractor into an Instance Feature Extractor (IFE) that produces discriminative embeddings that are aligned with the per-class representatives. These embeddings are subsequently used as weights inside a cosine-similarity classifier.

Our approach has several advantages. First, it eliminates the need for extensive retraining procedures for new classes because these can be added incrementally. The IFE generates embeddings that are used as class representatives without requiring access to base classes. Because we predict localization and segmentation in a class-agnostic manner, these embeddings are all that is needed to add novel classes.

Second, in contrast with related methods [8, 39], our mask predictor is class-agnostic. Similar to [21], no mask labels are needed for the addition of novel classes.

Third, our approach incurs no performance drawbacks at test time. We neither require additional memory for every class example [8, 39] nor require these examples to be passed one-by-one (e.g., [21]).

We make two main contributions:

- We present the first incremental few-shot instance segmentation method: iMTFA. Our method outperforms the current state-of-the-art for FSIS as well as the current state-of-the-art in incremental FSOD.
- To compare between incremental and non-incremental methods, we extend an existing FSOD approach [36] to the instance segmentation task (MTFA), and also demonstrate state-of-the-art results.

The remainder of the paper is structured as follows. We first discuss related work on few-shot learning and instance segmentation. We introduce our novel incremental and non-incremental methods in Section 3, and evaluate both extensively in Section 4. We conclude in Section 5.

2. Related Work

This section provides an overview of instance segmentation and few-shot learning.

Instance segmentation is the task of detecting objects in an image whilst also segmenting all the pixels that belong to them. Approaches generally fall into two categories: *grouping-based* [4, 11, 17, 24] and *proposal-based* [2, 6, 15, 20] detection methods. The former employ a grouping strategy in which a network produces per-pixel information that is post-processed to obtain instance segmentations. In proposal-based methods, a model first identifies potential areas and subsequently classifies and segments these regions. The most widely used two-step detection method is Mask R-CNN [11], which uses a Region Proposal Network (RPN) to propose detection regions which are passed to classification, localization, and mask predictor heads. However, these approaches do not perform well with small amounts of training data [39].

Few-shot learning enables models to accommodate new classes for which little training data is available. Often, an episodic methodology [35] is used by providing *query* items to be classified into N classes and a *support set* containing training examples of the N classes. Approaches for few-shot learning can largely be split up in *optimization-based* [1, 9, 26] and *metric-learning* [5, 10, 14, 33, 34, 35].

Optimization-based methods train a *meta-learner* from a series of tasks such that it is able to generate weights for a *learner* which learns parameters for new tasks that have few training examples. The meta-learner is generally modeled as an optimization procedure [1, 9] or is a separate network enhanced with memory [22, 26] that uses previous tasks as experience and is trained to produce a learner.

Metric-learning methods learn a feature embedding such that objects from the same class are close in the embedding space and objects of different classes are far apart. Koch *et al*. [14] employed a Siamese network [3], where the distance between query and support image embeddings is minimized if they are of the same class, and maximized otherwise. Matching Networks [35] compute the distance between every learned query and support embedding, while Prototypical Networks [33] compute per-class representatives. Relation Networks [34] learn both a distance function and an embedding. In contrast to previous methods that focus solely on the performance on the novel classes, Gidaris and Komodakis [10] focus on classifying both novel and base classes jointly using a softmax cosine-similarity classifier along with a weight generator for novel classes. Recently, Chen *et al*. [5] have shown that fine-tuning on the novel classes, which was largely ignored previously, generally performs better than episodic training. Finally, Qi *et al*. [25] propose *weight-imprinting* by adding novel class embeddings into an existing weight matrix, allowing incremental addition of classes without training.

Few-shot object detection extends few-shot learning to object detection. RepMet [30] trains a metric-learning sub-network to encode the support set, while Kang *et al*. [13] directly train a meta-learner on top of YOLOv2 [27]. Inspired by [5], Wang *et al*. developed TFA [36], which achieves state-of-the-art in object detection with a two-stage approach. Instead of fine-tuning the entire network, TFA first trains Faster R-CNN [28] on the base classes and then only fine-tunes the predictor heads.

Few-shot instance segmentation. Few works have addressed FSIS [8, 21, 39]. Most approaches provide guidance to certain parts of the Mask R-CNN architecture to ensure the network is better informed of the novel classes. Both Meta R-CNN [39] and Siamese Mask R-CNN [21] compute embeddings of the support set and combine these with the feature map produced by the network backbone. The combination is implemented through different operations such as subtraction [21] to focus the network on spe-

cific image areas, or concatenation [39] to provide additional information at a certain stage. FGN [8] guides the RPN, RoI detector and mask upsampling layers with the support set feature embeddings through similar operations.

Incremental few-shot object detection has been considered in ONCE [23], which uses CenterNet [40] as a backbone to learn a class-agnostic feature extractor and a per-class code generator network for novel classes.

Incremental few-shot instance segmentation. To our knowledge, we are the first to target incremental FSIS. FGN and Siamese Mask R-CNN depend on being passed examples of every class at test time, which requires a large amount of memory when considering many classes. Meta R-CNN can pre-compute per-class attention vectors, but requires retraining to handle a different number of classes. In contrast, our method can incrementally add classes without retraining or requiring examples of base classes.

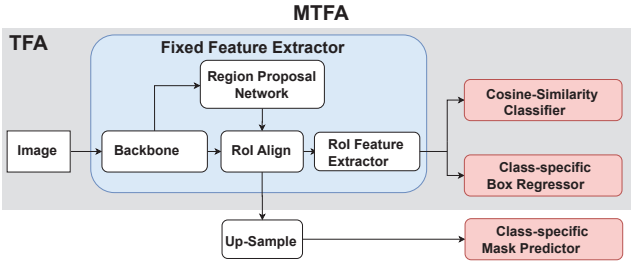

Figure 2. **Architecture of TFA and MTFA.** MTFA extends TFA with a mask prediction branch. In the first training stage, the whole network is trained on the base classes. In the second stage, the feature extractor is frozen (blue) while the classifier and box and mask heads (in red) are fine-tuned on base and novel classes.

3. Methodology

We first introduce common terminology in few-shot learning (Section 3.1). We then introduce our baseline few-shot instance segmentation method MTFA (Section 3.2). In Section 3.3, we introduce our incremental method: iMTFA.

3.1. Formulation of few-shot instance segmentation

In few-shot learning, we have a set of base classes C_{base}, for which a large amount of training data is available, and a disjoint set of novel classes C_{novel}, which has a small amount of training data. The goal is to train a model that does well on the novel classes $C_{test} = C_{novel}$ [33, 35] or on both base and novel classes jointly $C_{test} = C_{base} \cup C_{novel}$ [10]. In few-shot classification, Vinyals *et al.* [35] introduce the *episodic-training* methodology. Episodic-training sets up a series of episodes $E_i = (\mathbf{I}^q, S_i)$ where S_i is a support set containing N classes from $C_{train} = C_{novel} \cup C_{base}$ along with K examples per class (N-*way* K-*shot*). A network is then tasked to classify an image $\mathbf{I}^q$, termed *query*, out of the classes in S_i. The idea is that solving a different classification task each episode leads to better generalization and results on C_{novel}. This approach has also been extended to FSOD (e.g., [13]) and FSIS (e.g., [8, 39]) by considering all objects in an image as queries and having a single support set per-image instead of per-query.

The challenge of FSIS is not only to classify the query objects, but also to determine their localization and segmentation. Given a query image $\mathbf{I}^q$, FSIS produces labels y_i, bounding boxes b_i and segmentation masks $\mathbf{M}_i$ for all objects in $\mathbf{I}^q$ that belong to C_{test}.

3.2. MTFA: A non-incremental baseline approach

Our non-incremental baseline approach extends the Two-Stage Fine-tuning (TFA, [36]) object-detection method introduced by Wan *et al.* We first give an overview of TFA and then describe our extension, Mask-TFA (MTFA), which includes an instance segmentation task. In Section 3.3, we extend MTFA to an incremental approach.

TFA (Figure 2) uses Faster R-CNN [28] with a two-stage training scheme. In the first stage, the network is trained on the base classes C_{base}. In the second stage, feature-extractor $\mathcal{F}$ is frozen and only the prediction heads are trained. $\mathcal{F}$ consists of network backbone $\mathcal{B}$, region proposal network (RPN) and RoI feature extractor $\mathcal{G}$. Thus, only RoI classifier $\mathcal{C}$ and box regressor $\mathcal{R}$ are fine-tuned in the second stage. Fine-tuning is performed on a dataset containing an equal number of examples of C_{base} and C_{novel} classes.

MTFA. We extend TFA similarly to how Mask R-CNN extends Faster R-CNN: by adding a mask prediction branch at the RoI level (Figure 2). Thus, MTFA includes a branch with an up-sampling component and a mask predictor $\mathcal{M}$. We also employ a two-stage fine-tuning approach by first training the network on the base classes and then fine-tuning all predictor heads $\mathcal{C}$, $\mathcal{R}$ and $\mathcal{M}$ on a balanced dataset of K shots for every class.

Cosine-similarity classifier. Similar to TFA and other recent metric-learning methods [5, 10], a cosine-similarity classifier is used for $\mathcal{C}$ to learn more discriminative per-class representatives. $\mathcal{C}$ is a fully-connected layer which, given embeddings computed by the fixed feature extractor $\mathcal{F}$ for a RoI, produces classification scores $\mathbf{S}$. $\mathcal{C}$ is parameterized by weight matrix $\mathbf{W} \in \mathbb{R}^{e \times c}$ where e is the size of an embedding vector produced by $\mathcal{F}$ and c is the number of classes. We denote the columns of $\mathbf{W}$ as $\mathbf{w}_j \in \mathbb{R}^e$ such that $\mathbf{W} = [\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_c]$. Similar to [11, 16, 39], classification scores $\mathbf{S}_{i,j}$ for the i-th object proposal of an image $\mathbf{X}$ and the j-th class are produced as:

$$\mathbf{S}_{i,j} = \mathcal{F}(\mathbf{X})_i^\top \cdot \mathbf{w}_j. \quad (1)$$

Normalizing both the output of the feature extractor $\mathcal{F}$ and the weights $\mathbf{w}_i$ causes $\mathcal{C}$ to compute the cosine similarity between $\mathcal{F}(\mathbf{X})_i$ and class-representative $\mathbf{w}_j$:

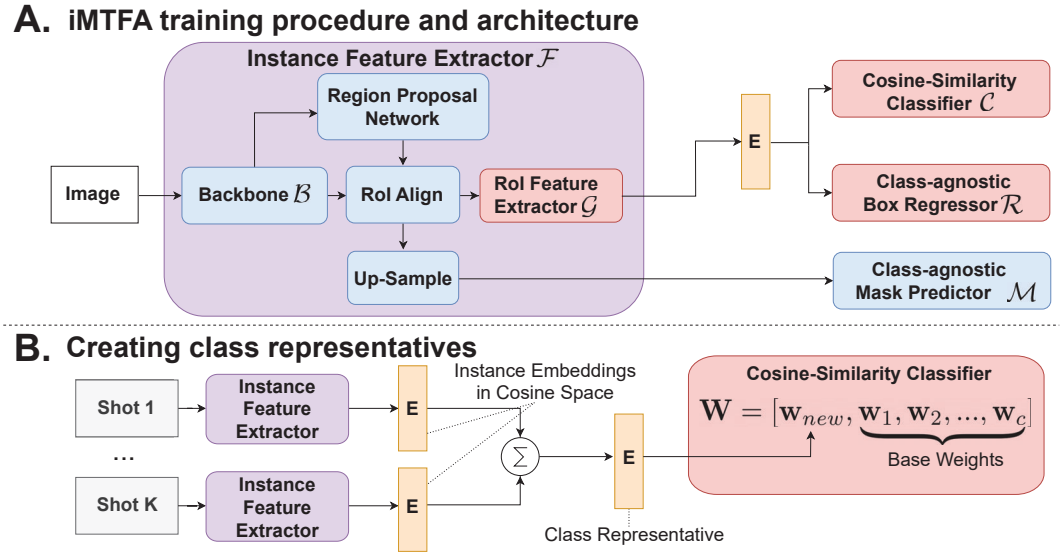

Figure 3. **Architecture of iMTFA.** (A.) In the first stage, the whole network is trained. In the second stage, the blue components are frozen, while the RoI Feature Extractor $\mathcal{G}$ is trained to produce discriminative embeddings aligned with the class-representatives in the cosine-similarity classifier $\mathcal{C}$. Both stages are trained only on the base classes. (B.) Given the K shots for each novel class, the IFE computes class weight vectors that are placed alongside the weights for the base classes.

$$\mathbf{S}_{i,j} = \frac{\alpha \mathcal{F}(\mathbf{X})_i^\top \cdot \mathbf{w}_j}{\|\mathcal{F}(\mathbf{X})_i\| \, \|\mathbf{w}_j\|}, \tag{2}$$

where α is used to scale the scores before they are passed to a softmax layer.

Forcing all embeddings to align to a single class representative $\mathbf{w}_j$ results in class prototypes that are similar to prototypical networks [33]. Normalization bounds the dot product, which simplifies the network's training task by allowing only angular degrees of freedom for optimization.

3.3. iMTFA: Incremental MTFA

The main drawback of MTFA is the procedure of adding new classes. The second fine-tuning stage fixes the number of novel classes that can be recognized. Adding new classes requires this stage to be run again, which is not practical. The class-specific mask and box regressor heads also require adaptation to novel classes in the form of weights learned through fine-tuning. In this section, we extend MTFA to an incremental approach: iMTFA. To this end, we make the model class-agnostic and learn discriminative embeddings at the feature extractor level. These embeddings are used as novel class representatives in the classification head without further need for training. The architecture and procedure for adding new classes are depicted in Figure 3.

Instance Feature Extractor (IFE). The fixed feature extractor $\mathcal{F}$ employed by TFA and MTFA is not trained to produce discriminative vector embeddings. Instead, the classification head $\mathcal{C}$ is fine-tuned in order to align the learned per-class weights $\mathbf{w}_i$ to fixed features computed by $\mathcal{F}$ for every RoI. We replace the fixed feature extractor by an instance feature extractor (IFE).

The key idea of our approach is to generate discriminative embeddings for each instance. To add new classes, the average of the generated embeddings is used as a per-class representative $\mathbf{w}_i$ in $\mathbf{W}$. This allows us to directly use instance embeddings as class representatives, without the need for fine-tuning.

The backbone $\mathcal{B}$ of Mask R-CNN produces feature maps for every RoI $\mathbf{R}_i = \mathcal{B}(\mathbf{X})_i$ with i indicating one RoI. The RoI-level feature extractor $\mathcal{G}$, typically consisting of two fully-connected layers, then takes these feature maps and computes embeddings $\mathbf{z}_i$ that are compared to the per-class representatives $\mathbf{w}_i$ in the classifier head $\mathcal{C}$. The vector embedding for each RoI is thus:

$$\mathbf{z}_i = \mathcal{F}(\mathbf{X})_i = \mathcal{G}(\mathbf{R}_i) = \mathcal{G}(\mathcal{B}(\mathbf{X})_i) \tag{3}$$

We propose to train $\mathcal{G}$ such that it produces discriminative embeddings per instance. This is achieved in two stages. First, we employ the same first training stage as MTFA – fully training Mask R-CNN on the C_{base} base classes. Second, we fine-tune $\mathcal{G}$ alongside the classifier $\mathcal{C}$ and box regressor $\mathcal{R}$, whilst keeping the backbone $\mathcal{B}$ and the RPN frozen. The fine-tuning is only performed on the set of base classes C_{base}, with the goal of generalizing to unseen classes in C_{novel}. The architecture of MTFA remains unchanged, only the training procedure is different. By training $\mathcal{G}$ as a sub-network with a cosine-similarity classifier, it produces embeddings which act as class representatives.

Creating class representatives. The final goal is to create novel class weight vectors that can be placed alongside

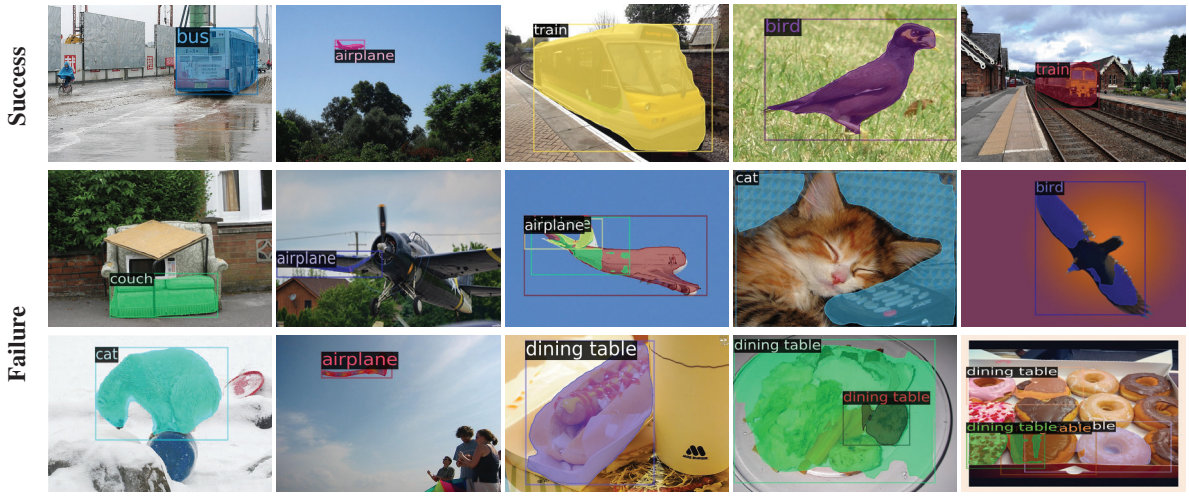

Figure 4. **Inference examples.** Successful (top row) and failure cases (bottom two rows), obtained on the 5-shot setting for the COCO novel classes. Failures include wrong classifications, wrong detections and inaccurate instance segmentations. See Section 4.3 for details.

the weights for the base classes, held in $\mathcal{C}$'s weight matrix $\mathbf{W}$ after the second fine-tuning stage. To accomplish this, every image $\mathbf{X}$ containing one of the K available novel shots is passed to the IFE, producing a feature embedding for each of the shots, $\mathbf{z}_i = \mathcal{F}(\mathbf{X})_i$. This is done for all K shots, with novel class representatives $\mathbf{w}_{new}$ computed as:

$$\mathbf{w}_{new} = \frac{1}{K} \sum_{i=0}^{K} \frac{\mathbf{z}_i}{\|\mathbf{z}_i\|}. \tag{4}$$

Because the normalized feature embeddings $\mathbf{z}_i$ are points on a hypersphere, their normalized average $\mathbf{w}_{new}$ is meaningful in this space and can be used as a class representative. We can pre-compute the class representatives and do not require all shots to be passed in at once. This greatly reduces the memory bottleneck in [8, 39].

Class-agnostic box and mask predictors. iMTFA does not need class-specific weights for the box regressor and mask predictor $\mathcal{R}$ and $\mathcal{M}$. Instead, we use class-agnostic variants of these components and can add new classes by simply averaging their computed embeddings and placing them in the classification head's weight matrix $\mathbf{W}$. This also implies that we can train on novel classes without providing instance masks.

Inference. Because we predict the localization and segmentation components in a class-agnostic manner, class representative are all we need at test time. The lowest cosine distance between an RoI's embedding and the class representatives gives us the class predictions.

Relation to other methods. Related approaches [8, 21, 39] rely on being passed examples of every class during training and at test time. This causes a memory bottleneck and forces some methods to only report evaluation results on the ground truth classes in an image [8, 21], train for significant amounts of time to match classes in a pairwise manner [21], or to use greatly reduced image sizes [39]. In contrast, iMTFA uses weight-imprinting [25, 31] to keep an internal memory for the class representatives, and thus does not need this memory consumption at test or train time.

4. Experiments

We first introduce our experiment setting (Section 4.1) and the implementation details (Section 4.2). Then we evaluate iMTFA and MTFA and compare them to related approaches (Section 4.3), followed by an ablation study.

4.1. Experiment setup

Our main evaluation procedure follows conventions established in FSOD [13, 36, 39]. We evaluate on the COCO [19], VOC2007 [7] and VOC2012 [7] datasets. We split the 80 COCO classes as proposed in [13]. The 20 classes that intersect with VOC are set as novel classes and the remaining 60 classes as base classes. The union of COCO's 80k train and 35k validation images are used for training and the remaining ∼5k images are the test set. The VOC dataset combines VOC2007 and VOC2012 and the resulting validation set is used for testing. We evaluate the performance of having $K = 1, 5, 10$ shots per novel class. To reduce the effect of outliers as a result of the random selection of the K shots, we run all tests 10 times with K random examples per class, and report the mean result. Our few-shot evaluation procedure is the same as in [36].

Comparison with other methods. We compare the instance segmentation performance of iMTFA and MTFA with the three other known FSIS methods: Meta R-CNN [39], Siamese Mask R-CNN [21], and FGN [8]. Additionally, we compare the object detection performance to the only known incremental FSOD method, ONCE [23].

For Meta R-CNN and ONCE, we use the method described above. However, Siamese Mask R-CNN and FGN

Shots	Inc.	Method	Detection						Segmentation					
			Overall		Base		Novel		Overall		Base		Novel	
			AP	AP50	AP	AP50	AP	AP50	AP	AP50	AP	AP50	AP	AP50
1		Base-Only	**28.67**	**43.53**	**38.22**	**58.04**	—	—	**26.34**	**41.55**	**35.12**	**55.40**	—	—
		MTFA	24.32	39.64	31.73	51.49	2.10	4.07	22.98	37.48	29.85	48.64	2.34	3.99
	✓	ONCE	13.6	N/A	17.9	N/A	0.7	N/A	—	—	—	—	—	—
		iMTFA	21.67	31.55	27.81	40.11	**3.23**	**5.89**	20.13	30.64	25.9	39.28	**2.81**	**4.72**
5		Base-Only	**28.67**	**43.53**	**38.22**	**58.04**	—	—	**26.34**	**41.55**	**35.12**	**55.40**	—	—
		MTFA	26.39	41.52	33.11	51.49	**6.22**	**11.63**	25.07	39.95	31.29	49.55	**6.38**	**11.14**
	✓	ONCE	13.7	N/A	17.9	N/A	1.0	N/A	—	—	—	—	—	—
		iMTFA	19.62	28.06	24.13	33.69	6.07	11.15	18.22	27.10	22.56	33.25	5.19	8.65
10		Base-Only	**28.67**	**43.53**	**38.22**	**58.04**	—	—	**26.34**	**41.55**	**35.12**	**55.40**	—	—
		MTFA	27.44	42.84	33.83	52.04	**8.28**	**15.25**	25.97	41.28	31.84	50.17	**8.36**	**14.58**
	✓	ONCE	13.7	N/A	17.9	N/A	1.2	N/A	—	—	—	—	—	—
		iMTFA	19.26	27.49	23.36	32.41	6.97	12.72	17.87	26.46	21.87	32.01	5.88	9.81

Table 1: **FSOD and FSIS performance on COCO for both base and novel classes.** ONCE cannot perform instance segmentation and Base-Only does not consider novel classes. For ONCE, no AP50 is reported. Both MTFA and iMTFA outperform ONCE for object detection while also being able to perform instance segmentation. Inc. stands for incremental.

use an evaluation scheme in which the classes that belong to each query image are known during testing. Only the classes that appear in the ground truth are included in the support set for each image. This eliminates many potential false positives, since similar classes that do not appear in an image together will not be confused. In contrast, we perform a N-*way* K-*shot* evaluation for every image, where N is the number of test classes. This ensures that all classes in a dataset can be detected in an image. For comparison, we emulate an evaluation method with a similar property by zeroing the probabilities computed by our softmax classifier for classes that do not appear in the query image. This discredits our methods, since it leaves non-occurring classes inside our metric space. Nevertheless, it serves as a lower-bound for the performance of non-episodic testing. We name this procedure *ground-truth only evaluation* (GTOE).

To compare against Siamese Mask R-CNN, we use one of their evaluation setups, which we term COCO-Split-2. This split consists of COCO classes with indices $4k, 1 \leq k \leq 20$ as novel classes, leaving the rest as base classes. The ResNet-50 backbone of Siamese Mask R-CNN is trained on 687 classes from ImageNet1K [29] that do not overlap with the classes in COCO. Unfortunately, it is unknown which classes have been used, so we opt to train on all 1,000 classes in ImageNet1K. To compare against FGN, we use the COCO2VOC setup, where we train on COCO but test on the VOC test set.

Following COCO evaluation practices, we report the performance using AP and AP50, using an intersection-over-union (IoU) overlap of bounding boxes and masks, for object detection and instance segmentation, respectively.

4.2. Implementation details

Our Mask R-CNN [11] is implemented using Detectron2 [38]. Our backbone is a ResNet-50 [12] with a Feature Pyramid Network [18]. All models are trained using SGD and a batch size of 8 on two NVIDIA V100s, with four images per GPU. The second fine-tuning stage has a learning rate of 0.0007 for iMTFA and a learning rate of 0.0005 for MTFA. We set the cosine-similarity scaling factor α to 1.0 for the iMTFA COCO-Novel, 10.0 for iMTFA COCO-All and 20.0 for MTFA (see also Section 4.4). Mask R-CNN has many parameters, hence we encourage the reader to visit the public repository for more details.

#	Inc.	Method	Detection		Segmentation	
			AP	AP50	AP	AP50
1		MTFA	2.47	4.85	2.66	4.56
	✓	iMTFA	**3.28**	**6.01**	**2.83**	**4.75**
5		MRCN+ft-full	1.3	3.0	1.3	2.7
		Meta R-CNN	3.5	9.9	2.8	6.9
		MTFA	**6.61**	**12.32**	**6.62**	**11.58**
	✓	iMTFA	6.22	11.28	5.24	8.73
10		MRCN+FT-full	2.5	5.7	1.9	4.7
		Meta R-CNN	5.6	14.2	4.4	10.6
		MTFA	**8.52**	**15.53**	**8.39**	**14.64**
	✓	iMTFA	7.14	12.91	5.94	9.96

Table 2: **FSOD and FSIS performance on the COCO novel classes.** MTFA and iMTFA outperform the current state-of-the-art in terms of AP. Inc. stands for incremental.

4.3. Results

Results on the COCO novel classes. We compare against Meta R-CNN [39] and a fully-converged Mask R-CNN model fine-tuned on the novel classes (MRCN+ft-full, [39]). We report object detection and instance segmentation performance on the 20 COCO novel classes (COCO-Novel) in Table 2. For all methods, detection and segmentation performance increases with the number of shots. Both iMTFA and MTFA outperform Meta R-CNN and MRCN+ft-full by a large margin in terms of AP, for every number of tested shots. In terms of AP50, MTFA surpasses Meta R-CNN but iMTFA is slightly behind. This suggests we may have difficulties finding the coarse location of an object but perform better at higher IoU thresholds.

Meta R-CNN directly uses image crops of the K available shots per class to infer class-attentive vectors. In contrast, iMTFA and MTFA re-use the largest part of the network $\mathcal{F}$ by directly working at a feature-map level. We argue this generates more representative vector embeddings for the novel shots, which would explain the large performance gap between Meta R-CNN and our methods.

Examples of inference results for iMTFA on the COCO novel classes with $K = 5$ appear in Figure 4. Successful segmentations are generally accurate. Failure cases include correctly classifying but incorrectly localizing an object (row 2, columns 1–3), correctly classifying and localizing but incorrectly segmenting (row 2, columns 4–5), and incorrectly classifying but correctly localizing and segmenting (row 3, columns 1–2). Classes that are diverse in appearance have more false positives. This is noticeable especially for the `dining table` class. Many objects that resemble food will be incorrectly classified as a dining table. For the `person` class, a similar trend is observed.

Results on both base and novel COCO classes. In this experiment, we strive to detect all 80 COCO classes (COCO-All). We report the standard evaluation of AP and AP50 over the 80 COCO classes. Additionally, we report the performance of the base and novel classes individually. We are the first to report performance for $C_{test} = C_{base} \cup C_{novel}$ in FSIS. To understand the merits of our approaches, we compare the object detection performance of iMTFA and MTFA with the state-of-the-art incremental FSOD method ONCE [23]. We also report on a model trained only on the base classes (Base-Only). While this model cannot be used for the novel classes, it demonstrates how much the performance on base classes is affected.

Results are summarized in Table 1. As expected, Base-Only performs best on the base classes. iMTFA surpasses the object detection performance of ONCE in terms of base and novel class performance. MTFA consistently outperforms iMTFA on the base classes, which may be caused by iMTFA's inability to adapt to existing per-class representatives when generating new ones. See also Section 4.4. Apart from $K = 1$, MTFA also performs better than iMTFA on the novel classes, in line with the results on COCO-All.

#	Inc.	Method	Detection		Segmentation	
			AP	AP50	AP	AP50
1		Siamese Mask R-CNN	8.6	15.3	6.7	13.5
		MTFA	8.26	15.24	8.25	14.31
	✓	iMTFA	**10.06**	**17.85**	**8.67**	**15.47**
5		Siamese Mask R-CNN	9.4	16.8	7.4	14.8
		MTFA	**15.80**	**28.12**	**15.14**	**25.83**
	✓	iMTFA	14.55	25.73	12.33	21.95

Table 3: **FSIS performance on COCO-Split-2.** iMTFA outperforms Siamese Mask R-CNN for $K = 1$ and $K = 5$, while MTFA performs best on $K = 5$.

Comparison with Siamese Mask R-CNN We follow the GTOE evaluation procedure described in Section 4.1 and report AP and AP50 for COCO-Split-2 in Table 3.

For both object detection and instance segmentation, MTFA and iMTFA outperform Siamese Mask R-CNN. Siamese Mask R-CNN uses image crops for the K shots to guide the network. This may prove to be detrimental in terms of performance, similar to Meta R-CNN. Additionally, the learned embeddings may not have strong discriminative power since they are not directly optimized through the loss function. In contrast, iMTFA's IFE is trained to produce discriminative embeddings for the K shots.

Our higher performance might also be due to our use of a cosine-similarity classifier, which has been shown to produce more meaningful embeddings than the binary cross-entropy loss employed by Siamese Mask R-CNN [37]. Finally, our models are trained on all 1,000 ImageNet1K classes, whereas Siamese Mask R-CNN only uses 687.

#	Inc.	Method	Detection		Segmentation	
			AP	AP50	AP	AP50
1		FGN	N/A	**30.8**	N/A	16.2
		MTFA	9.99	21.68	**9.51**	**19.28**
	✓	iMTFA	**11.47**	22.41	8.57	16.32

Table 4: **FSIS performance on COCO2VOC.** For FGN, no AP results are reported by the authors.

Comparison with FGN. We compare iMTFA and MTFA to FGN using the cross-dataset COCO2VOC evaluation setting and the GTOE evaluation procedure. The FGN paper evaluates 1-way 1-shot, 3-way 1-shot and 3-way 3-shot performance. Since FGN's source code is not released and the used evaluation scheme is not common, we are only able to compare against 1-way 1-shot results. From Table 4, it shows that MTFA has superior performance in terms of instance segmentation while iMTFA's incremental approach is on par with FGN.

FGN's higher object detection performance suggests that guidance at the RPN and classifier stages is effective, although the better performance could partly be due to the use of a deeper backbone (ResNet-101 vs. ResNet-50). A combined approach appears promising for future work. Although the instance segmentation performance between iMTFA and FGN is similar, iMTFA maintains the key advantage of being incremental.

#	Inc.	Method	Detection		Segmentation	
			AP	AP50	AP	AP50
5		MTFA	6.61	12.32	**6.62**	**11.58**
		CA MTFA	**7.00**	**12.58**	6.11	10.16
		CA MTFA w/o FT $\mathcal{M}$	7.00	12.64	5.83	9.48
	✓	iMTFA	6.22	11.28	5.24	8.73

Table 5: **Ablation MTFA/iMTFA.** Comparison between different variants of MTFA and iMTFA on COCO-Novel.

4.4. Ablation study

We perform several ablations on the COCO 5-shot setting for novel classes.

Comparison between iMTFA and MTFA. We identify two main reasons that can account for the performance difference between MTFA and iMTFA: using class-specific components and adjusting parts of the network through fine-tuning. To measure their effect, we compare MTFA and iMTFA along with (1) MTFA with a class-agnostic mask predictor $\mathcal{M}$ and box regressor $\mathcal{R}$ (CA MTFA) and (2) a class-agnostic MTFA without fine-tuning the mask predictor $\mathcal{M}$ (CA MTFA w/o FT $\mathcal{M}$). Results appear in Table 5.

Class-specific components and fine-tuning both help MTFA to achieve better segmentation performance. iMTFA is unable to adjust the generated novel weights based on existing weights in the metric space, which fine-tuning can do. We also find that fine-tuning the class-agnostic mask predictor is advantageous. This may be because iMTFA does not explicitly use the segmentation information for the K shots to inform the mask predictor, whereas MTFA and class-agnostic MTFA achieve this by optimizing the segmentation loss directly. The performance loss from class-specific to class-agnostic is in line with [11] and may be attributed to the additional number of trainable parameters.

#	Inc.	Method	Detection		Segmentation	
			AP	AP50	AP	AP50
5	✓	One-Stage-Cosine	5.37	9.91	4.3	7.32
		One-Stage-Linear	5.32	9.87	4.49	7.54
		iMTFA	**6.19**	**11.24**	**5.22**	**8.71**

Table 6: **Ablation second fine-tuning stage.** Results on COCO-Novel for different classification heads in iMTFA.

Effectiveness of the second fine-tuning stage. To judge the merits of the second fine-tuning stage for feature extractor $\mathcal{G}$, we compare iMTFA to a variant that directly trains Mask R-CNN using a cosine-similarity head (One-Stage-Cosine) and one that directly uses the linear classification head in Mask R-CNN (One-Stage-Linear). iMTFA outperforms both, see Table 6. This demonstrates the effectiveness of the second fine-tuning stage. Instead of focusing on the cosine-similarity sub-network during training, One-Stage-Cosine seems to focus on the backbone and the RPN. One-Stage-Linear can produce embeddings with similar angles but dissimilar scales, which cannot be easily distinguished using cosine similarity.

Cosine-similarity scaling factor. Parameter α in Eq. 2 scales the classification scores before applying softmax. In Table 7, we experiment with various α values and find that $\alpha = 1.0$ produces the best performance for COCO-Novel. For COCO-All, $\alpha = 10.0$ provides a good balance between high Overall and Novel AP. These values are subsequently used in all experiments on these datasets. For optimal performance, α needs to be tweaked based on the number of classes, which is in line with previous insights [37].

	COCO-Novel		COCO-All			
	Detection	Segmentation	Detection		Segmentation	
α	Overall		Overall	Novel	Overall	Novel
1.0	**6.22**	**5.24**	—	—	—	—
2.0	6.19	5.22	0.36	0.36	1.42	1.46
3.0	6.17	5.21	10.31	6.17	9.55	5.21
5.0	6.09	5.17	16.99	**6.62**	15.72	**5.53**
10.0	5.76	4.94	19.62	6.07	18.22	5.19
15.0	5.34	4.60	**19.67**	5.62	**18.28**	4.82
20.0	4.94	4.26	19.45	5.19	18.09	4.47
25.0	4.62	3.99	19.20	4.86	17.87	4.19

Table 7: **Ablation alpha value.** Comparison of cosine scaling factors for iMTFA in COCO-All and COCO-Novel.

5. Conclusions

We have presented the first incremental approach to few-shot instance segmentation: iMTFA. iMTFA repurposes Mask R-CNN's feature extractor to generate discriminative per-instance embeddings. The mean of these embeddings is used as a class-representative in a cosine-similarity classifier. Because the localization and segmentation components are class-agnostic, the embeddings are all that is needed to add new classes. To compare iMTFA with a stronger non-incremental and class-specific baseline, we also introduced MTFA. It extends the few-shot object detection approach TFA [36] by adding a mask prediction branch. Both iMTFA and MTFA outperform the current state-of-the-art on a variety evaluation scenarios using the COCO and VOC datasets.

There are several ways in which iMTFA can be improved. First, iMTFA cannot adapt to existing embeddings when generating new ones. Attention mechanisms such as those employed by [10, 35] are a promising future direction.

Second, iMTFA's class-agnostic localization and segmentation components are suboptimal compared MTFA's class-specific counterparts. An obvious improvement is to learn a transfer function from the generated embeddings to class-specific box regressor and mask predictor. We believe combining our approach with a guidance mechanism (e.g., [8, 21]) would further improve the performance of iMTFA.

Third, iMTFA's frozen box regressor and mask predictor introduce base class bias compared to MTFA. Employing guidance mechanisms would also alleviate this issue.

With these improvements in mind, the advances made with iMTFA present a promising outlook to narrow the gap between non-incremental and incremental few-shot instance segmentation, and to allow for a flexible addition of novel classes to already powerful networks.

6. Acknowledgements

This work is supported by the Dutch Organization for Scientific Research (NWO) with TOP-C2 grant ARBITER. We also thank Cyclomedia for sponsoring this research.

References

[1] Marcin Andrychowicz, Misha Denil, Sergio Gomez, Matthew W Hoffman, David Pfau, Tom Schaul, Brendan Shillingford, and Nando De Freitas. Learning to learn by gradient descent by gradient descent. In *Advances in neural information processing systems*, pages 3981–3989, 2016. 2

[2] Min Bai and Raquel Urtasun. Deep watershed transform for instance segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5221–5229, 2017. 2

[3] Jane Bromley, Isabelle Guyon, Yann LeCun, Eduard Säckinger, and Roopak Shah. Signature verification using a "Siamese" time delay neural network. In *Advances in neural information processing systems*, pages 737–744, 1994. 2

[4] Liang-Chieh Chen, Alexander Hermans, George Papandreou, Florian Schroff, Peng Wang, and Hartwig Adam. MaskLab: Instance segmentation by refining object detection with semantic and direction features. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4013–4022, 2018. 2

[5] Wei-Yu Chen, Yen-Cheng Liu, Zsolt Kira, Yu-Chiang Frank Wang, and Jia-Bin Huang. A closer look at few-shot classification. *arXiv preprint arXiv:1904.04232*, 2019. 2, 3

[6] Bert De Brabandere, Davy Neven, and Luc Van Gool. Semantic instance segmentation with a discriminative loss function. *arXiv preprint arXiv:1708.02551*, 2017. 2

[7] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The Pascal Visual Object Classes (VOC) challenge. *International journal of computer vision*, 88(2):303–338, 2010. 5

[8] Zhibo Fan, Jin-Gang Yu, Zhihao Liang, Jiarong Ou, Changxin Gao, Gui-Song Xia, and Yuanqing Li. FGN: Fully guided network for few-shot instance segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9172–9181, 2020. 2, 3, 5, 8

[9] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1126–1135, 2017. 2

[10] Spyros Gidaris and Nikos Komodakis. Dynamic few-shot visual learning without forgetting. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4367–4375, 2018. 2, 3, 8

[11] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask R-CNN. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017. 1, 2, 3, 6, 8

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 6

[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 International Conference on Computer Vision*, pages 8420–8429, 2019. 1, 2, 3, 5

[14] Gregory Koch, Richard Zemel, and Ruslan Salakhutdinov. Siamese neural networks for one-shot image recognition. In *ICML deep learning workshop*, volume 2. Lille, 2015. 2

[15] Shu Kong and Charless C Fowlkes. Recurrent pixel embedding for instance grouping. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9018–9028, 2018. 2

[16] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012. 1, 3

[17] Weicheng Kuo, Anelia Angelova, Jitendra Malik, and Tsung-Yi Lin. ShapeMask: Learning to segment novel objects by refining shape priors. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 9207–9216, 2019. 2

[18] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2117–2125, 2017. 6

[19] 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 *European conference on computer vision*, pages 740–755. Springer, 2014. 5

[20] Shu Liu, Jiaya Jia, Sanja Fidler, and Raquel Urtasun. SGN: Sequential grouping networks for instance segmentation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3496–3504, 2017. 2

[21] Claudio Michaelis, Ivan Ustyuzhaninov, Matthias Bethge, and Alexander S Ecker. One-shot instance segmentation. *arXiv preprint arXiv:1811.11507*, 2018. 2, 5, 8

[22] Nikhil Mishra, Mostafa Rohaninejad, Xi Chen, and Pieter Abbeel. A simple neural attentive meta-learner. *arXiv preprint arXiv:1707.03141*, 2017. 2

[23] Juan-Manuel Perez-Rua, Xiatian Zhu, Timothy M Hospedales, and Tao Xiang. Incremental few-shot object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13846–13855, 2020. 3, 5, 7

[24] Pedro O. Pinheiro, Ronan Collobert, and Piotr Dollár. Learning to segment object candidates. In *Advances in Neural Information Processing Systems*, pages 1990–1998, 2015. 2

[25] Hang Qi, Matthew Brown, and David G Lowe. Low-shot learning with imprinted weights. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5822–5830, 2018. 2, 5

[26] Sachin Ravi and Hugo Larochelle. Optimization as a model for few-shot learning. *International Conference on Learning Representations (ICLR)*, 2017. 2

[27] 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. 2

[28] 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*, pages 91–99, 2015. 1, 2, 3

[29] 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. *International journal of computer vision*, 115(3):211–252, 2015. 6

[30] Eli Schwartz, Leonid Karlinsky, Joseph Shtok, Sivan Harary, Mattias Marder, Sharathchandra Pankanti, Rogerio Feris, Abhishek Kumar, Raja Giries, and Alex M Bronstein. RepMet: Representative-based metric learning for classification and one-shot object detection. *arXiv preprint arXiv:1806.04728*, 2018. 2

[31] Mennatullah Siam, Boris N Oreshkin, and Martin Jagersand. Amp: Adaptive masked proxies for few-shot segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5249–5258, 2019. 5

[32] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 1

[33] Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. In *Advances in neural information processing systems*, pages 4077–4087, 2017. 2, 3, 4

[34] Flood Sung, Yongxin Yang, Li Zhang, Tao Xiang, Philip H.S. Torr, and Timothy M Hospedales. Learning to compare: Relation network for few-shot learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1199–1208, 2018. 2

[35] Oriol Vinyals, Charles Blundell, Timothy P. Lillicrap, Koray Kavukcuoglu, and Daan Wierstra. Matching networks for one shot learning. *CoRR*, abs/1606.04080, 2016. 2, 3, 8

[36] 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. 1, 2, 3, 5, 8

[37] Nicolai Wojke and Alex Bewley. Deep cosine metric learning for person re-identification. In *2018 IEEE winter conference on applications of computer vision (WACV)*, pages 748–756. IEEE, 2018. 7, 8

[38] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2. https://github.com/facebookresearch/detectron2, 2019. 6

[39] 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 International Conference on Computer Vision*, pages 9577–9586, 2019. 1, 2, 3, 5, 6

[40] Xingyi Zhou, Dequan Wang, and Philipp Krähenbühl. Objects as points. *arXiv preprint arXiv:1904.07850*, 2019. 3