Supplementary for: "*Beyond Classification:* Knowledge Distillation using Multi-Object Impressions"

Gaurav Kumar Nayak*, Monish Keswani*
{gauravnayak,monishkumar}@iisc.ac.in
Indian Institute of Science
Bangalore, India
Sharan Seshadri, Anirban Chakraborty
sharan2510@gmail.com,anirban@iisc.ac.in

1 Efficacy of our pseudo-dataset across different Object Detection Methods

We investigate the efficacy of our generated pseudo-dataset ($\hat{D}$) obtained from Pascal trained Faster-RCNN network with Resnet-34 backbone. We train different state-of-the-art detection models on our pseudo-dataset. Particularly, we evaluate on Faster-RCNN with FPN [1], YOLO [4], FCOS [7] and RetinaNet [2].

Object Detection Methods	**Original dataset (D) (Pascal)**	**Our Pseudo-dataset ($\hat{D}$)**
	mAP (Upper Bound)	mAP (training from scratch)
Faster RCNN [5]	70.1	50.9
Feature Pyramid Network [1] (Faster-RCNN)	70.7	50.6
Yolo V3 [4]	69.2	41.1
FCOS [7]	64.0	46.8
RetinaNet [2]	71.3	51.3

Table 1: Performance (in %) with our generated samples using different state-of-the-art approaches for the object detection task in the complete absence of the Pascal dataset. The models are evaluated using the Pascal VOC convention style.

1. **Feature Pyramid Network (FPN)** is often used in addition to the Faster-RCNN model to obtain better feature representations for object detection task. It combines features in bottom-up pathway to yield high-resolution features and top-down pathway to yield low-resolution features. Our pseudo-dataset was not synthesized to deal with such a setup. As shown in Table 1, we obtain 50.6% mAP which is very close to the network trained on Faster-RCNN. This implies that our pseudo-dataset is even applicable to the networks that perform different modifications at the feature level.

2. **YOLO**: We use YOLO-v3 for our experiments. Note that, unlike Faster-RCNN, YOLOv3 is a single stage network. As evident in Table 1, we achieve 41.1% mAP even though the pseudo-dataset was generated using Faster-RCNN. This implies that our pseudo-dataset is generalizable to one-stage methods.

* Equal contribution.

3. **FCOS** is a single stage object detector like Yolo. But it predicts on per pixel basis unlike anchor based methods like Yolo. We obtain 46.8% mAP using FCOS.

4. **RetinaNet** is also a single stage object detector but uses Focal loss i.e. a modified version of the Cross Entropy loss, that helps to overcome the foreground and background class imbalance problem. We obtain decent mAP of 51.3% which shows that our generated data facilitates training of detection model with any additional losses.

Despite methodological and architectural differences, we obtain respectable performances ($\approx 41-51\%$ mAP) in Table 1 while training the network from scratch with our pseudo-dataset ($\hat{D}$) using different detection methods. These observations emphasize that $\hat{D}$ reasonably estimates the training data distribution and is applicable to different object detection methods in the absence of training data.

2 Result Analysis and Discussions

In this section, we briefly analyze the results and discuss our observations that we found based on the experiments across architectures and datasets.

Convergence: Unlike existing data-free works such as [3, 8] takes large iterations and longer time for generation. On the contrary, each batch in proposed Algorithm 2 in the main draft converges within 100 iterations for even large scale datasets like COCO.

Other Losses: In order to further improve the distillation performance, we even tried to use other well-known natural image priors like Total variation [6] and L2 regularizer to generate natural-looking Multi-Object Impressions (MOIs). But we observed that such losses though are helpful in improving visualization of samples but do not yield improvement in mAP. Sometimes, it even leads to lower performance during distillation. We also observed that learning rate (lr) is a crucial hyperparameter whose careful tuning can lead to the generation of a better transfer set. Infact, we found that only tuning of lr works better than using such additional losses. Please refer to section 5 for experimental results and more details.

Beyond transfer set: In order to further study how well our pseudo-dataset has captured the training data distribution, we use our generated data to train the network from scratch. From Table 1 in the main draft, we can observe that we obtain a decent performance of 57.8% mAP on KITTI, even when Resnet-18 is trained from scratch using MOIs. Moreover, we obtained similar respectable performances even in the case of complex and large scale datasets like Pascal and COCO. In the case of Pascal, (as shown in Table 2 in main draft) VGG-16 and Resnet-34 networks obtains 49.3% and 50.9% mAP while training them with our MOIs with no *Teacher* assistance. Similarly, (Table 3 in main draft) with no KD, we observed reasonable performance of 30.9% mAP on evaluation using COCO@0.5. The summarized results across datasets and architectures while training the network from scratch with our generated datset ($\hat{D}$) is put in Table 4 in the main draft. Also, our pseudo-dataset is even suitable to be used as augmentation when arbitrary data or few training samples are present (shown in Table 5 in main draft). Moreover, we obtain reasonable performances across different detection methods (Sec. 1). Thus, our novel pseudo-dataset synthesizing framework can have a lot of practical utility. These observations show that our generated data has even the potential to be used beyond a transfer set for distillation, such as training a

network from scratch. Such insights are important and currently missing in existing data-free methods which requires further investigation.

Performance Analysis: We analyze both the stages of our generation strategy : generation of pseudo-targets and generation of MOIs.

To analyze the first stage, we replace our pseudo-targets with original training data ground truths. In other words, we use label information from the training data like the position, size, and class label of an object rather than actual objects in the images and then generate impressions corresponding to them. We observed only a minor gain in performance, which shows our prepared pseudo-targets reasonably capture the label distribution of training data. More details are in section 4.

Next, we analyze stage two which deals with generation of MOIs. Since neither the training data nor its statistics are assumed to be available, there always exists a domain gap between our generated data and original training data. We tried to reduce this gap by better initialization i.e. initializing MOIs with textures rather than random initialization. The original training data is independent of the architecture. But our generated MOIs depend on the *Teacher* architecture. To decouple the MOIs from the *Teacher*, we use differential batch augmentation which makes them more robust and hence results in improved performance.

3 Motivation for Power Law distribution

As discussed in Algorithm 1 (in the main draft), the maximum number of possible objects in any sample is denoted by M. While preparing the target labels, the number of objects for each sample (denoted by N_i for i^{th} sample) is uniformly sampled between 1 to M. For simplicity, we restrict our discussion of objects' size with respect to a particular sample and use N to denote the number of objects belonging to that sample. The minimum and maximum possible object sizes obtained using the information of anchor scales are denoted by A^{min} and A^{max} respectively. This implies that the sizes of each object need to lie in the range $[A^{min}, A^{max}]$.

Uniform Sampling ($\mathcal{U}(A^{min}, A^{max})$): An intuitive strategy for deciding objects' size is to uniformly sample each object size from the range $[A^{min}, A^{max}]$. If one of the sampled object sizes is very large (close to A^{max}), then it would result in high overlaps while placing other objects. In extreme cases when N is large, it may end up with most of the objects contained inside another object. Therefore, with such a strategy it will be difficult to satisfy the $IoU_{threshold}$ constraint. The major reason for high overlaps is that there is no restriction on the object sizes and each object can be as large as A^{max}. In order to overcome this problem, we need to enforce a constraint on A^{max}.

Constrained Uniform Sampling ($\mathcal{U}(A^{min}, min((W \cdot H)/N, A^{max}))$): One way to enforce the constraint on A^{max} is to have maximum object size as $A^{'max} \leftarrow min((W \cdot H/N), A^{max})$ which is fixed for all the objects in a sample. W and H denote the width and height of the sample respectively. Since each object's maximum size is same and reduced to $A^{'max}$, this would help in placing the objects within the $IoU_{threshold}$ constraint. By putting such restrictions on the maximum object size where $A^{'max}$ becomes small with a large value of N, objects of small sizes are more favoured.

Interval based Uniform Sampling: In order to explicitly enforce the favoring of large object sizes and reduce biases towards the small sizes, we can adopt a interval based uniform sampling strategy ($\mathcal{U}(max(A^{min},(W\cdot H)/(N+1)),min(W\cdot H/(N),A^{max}))$. For e.g., when N is 1, the interval can be used to allow objects of large sizes. Similarly, for $N=2$, the object sizes can be sampled in the interval $[max(A^{min},(W\cdot H)/3),min((W\cdot H)/2,A^{max})]$ and so on. We can perform uniform sampling on each of these intervals. The problem with this strategy is that the objects are sampled within a particular interval for a fixed value of N and the probability of objects belonging to another interval for the same value of N is zero. That means the small and large sized objects cannot be sampled together for a particular value of N. The power law overcomes this problem by allowing objects of small and large sizes to occur together with some probability.

The power law is defined as:

$$P(x;a_o) = a_o x^{a_o-1}, 0 \leq x \leq 1, a_o > 0 \tag{1}$$

Let sampling from power law be represented by : $x \sim P(a_o)$. The mean of the power law distribution is given by $\frac{a_o}{a_o+1}$.

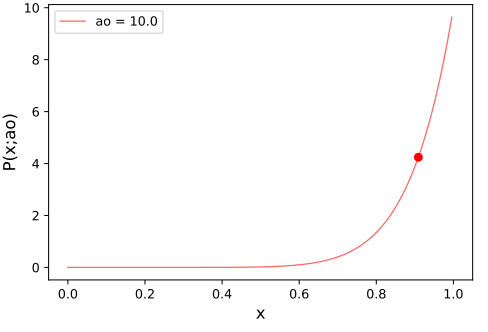


Figure 1: The Power Law distribution with $a_o = 10$. $x = 0$ represents A^{min} and $x = 1$ represents $A^{'max}$. The red dot represents the mean of the distribution which is favouring areas closer to $A^{'max}$.

Power Law with fixed a_o : The power law distribution defined using a_o is used to sample x which lies between 0 to 1. The sampled x is rescaled to interval $[A^{min},A^{'max}]$ where $A^{'max}$ is $min((W\cdot H)/N,A^{max})]$. We assume that a_o is fixed. This would imply that the mean of the distribution is also fixed. Thus, the majority of the sampled objects' sizes lie around the mean (as shown in Figure 1. When $a_o = 1$, it is same as constrained uniform sampling which favours small areas. While on the other hand, for a large value of a_o, mean is very close to 1 which leads to negligence of small areas. This observation motivates to have a variable a_o.

Power Law with variable a_o: Our proposed Algorithm 1 (in the main draft) defines a_o as the ratio $\frac{M}{N}$ for a particular sample. For different values of a_o, the distribution curves is shown in Figure 2. For $N = 1$, $a_o = M$ which allows sampling large objects' sizes with high probability and small objects sizes' with low probability. As the value of N increases,

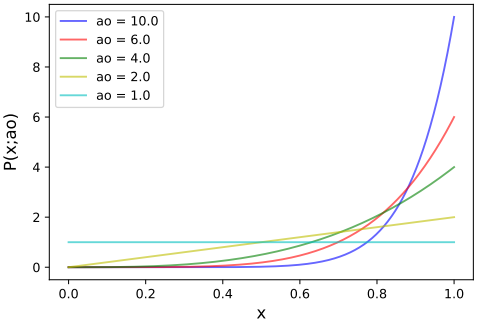


Figure 2: Power Law distribution for different values of a_o. As the value of a_o decreases to 1, the distribution gradually switches to Uniform distribution.

we prefer sampling of small sized objects. When $N = M$, $a_o = 1$, it is same as constrained uniform sampling. Since it overcomes the limitations of previous strategies, therefore, we use our proposed, variable a_o based power law distribution in Algorithm 1 (in the main draft) to define the object sizes for the target labels.

4 Distribution of objects in prepared Pseudo-Targets

Comparison of prepared pseudo-targets and Pascal labels with respect to the distribution of the object sizes is shown below:

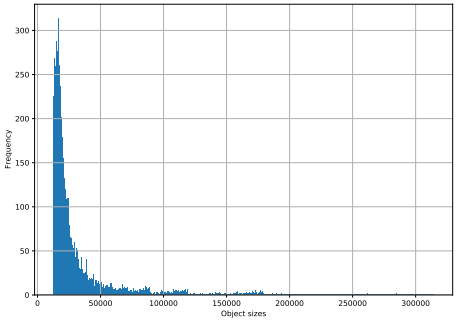


Figure 3: Histogram plot of object sizes and their frequency on our prepared pseudo-targets.

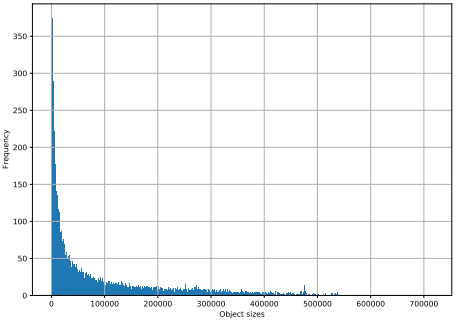


Figure 4: Histogram plot of object sizes and their frequency from the Pascal Dataset

From Figure 3 and 4, we can observe that in the absence of any prior knowledge of the training data, our pseudo-data obtained using Algorithm 1 (in the main draft) is a good approximate for training label distribution. More precisely, the distribution of object sizes of pseudo-targets using the Power Law distribution and anchor information (ratios and scales) of the *Teacher* network reasonably captures the training data distribution of object sizes. However, we do observe that we have low count of large sized objects. This may be due to the fact that our MOIs are of fixed 600×600 dimension. Next, we empirically investigate the effect of prior knowledge of the training label distribution on the distillation performance of the *Student* model.

Zero-Shot Distillation	Without AI (mAP in %)	With AI (mAP in %)
Resnet 34 $\rightarrow$ Resnet 34	54.88	55.11

Table 2: Distillation using our pseudo-dataset where MOIs are generated without and with additional information (AI). AI is the prior knowledge about the class label, size and location of the objects in the samples of training data (Pascal). Note that the number of samples generated is equal to number of samples in the Pascal training dataset for which the prior label information is available. Thus, K is taken as 5011 for both with and without AI experiments.

As shown in Table 2 , we gain only slight improvement in distillation performance even if the class label, object sizes and their locations on training data are known apriori. This shows that our prepared pseudo-targets reasonably estimates the label distribution of the training data (when only the pretrained model and not the training data is available).

5 Experiments with other losses for crafting MOIs

Mask Total Variation Loss (L_{mtv}): If we directly apply the total variation loss [6] on MOIs, it can also blur out the texture background which is not desired. So, we apply it on the mask which is the difference between the current optimized MOI and initialized MOI. This loss makes the generation of MOIs less sensitive to learning rate and optimization easier by retaining the smoothness. But we do not observe any improvement in mAP while performing distillation from Resnet-18 *Teacher* trained on KITTI to Resnet-18-half with 2500 generated samples using this additional loss (shown in Table 3). By only carefully tuning the learning rate (lr) without using this loss, we observe better performance. However, this loss can give improvement in mAP in cases when the lr is set high (shown in Table 4). But we can clearly observe from Table 3 and 4, that MOIs synthesized with only L_{gt} loss and no L_{mtv} loss yields better performance when tuned with proper lr.

Loss	mAP (in %)
L_{gt}	**27.2**
$L_{gt} + L_{mtv}$	20.9
$L_{gt} + 0.1 \cdot L_{mtv}$	24.7

Table 3: Distillation when MOIs generated with lr 0.01

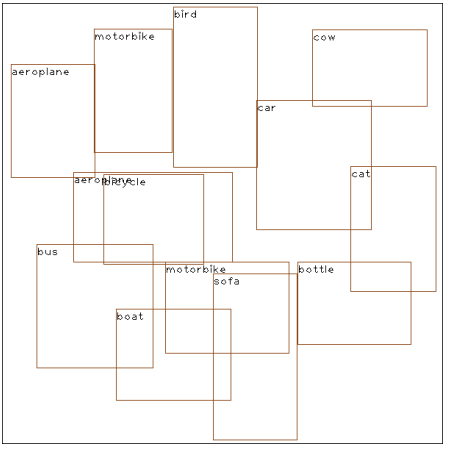
Loss	mAP (in %)
L_{gt}	15.5
$L_{gt} + L_{mtv}$	20.5

Table 4: Distillation when MOIs generated with lr 0.1

6 Visualization of Multi-Object Impressions (MOIs)

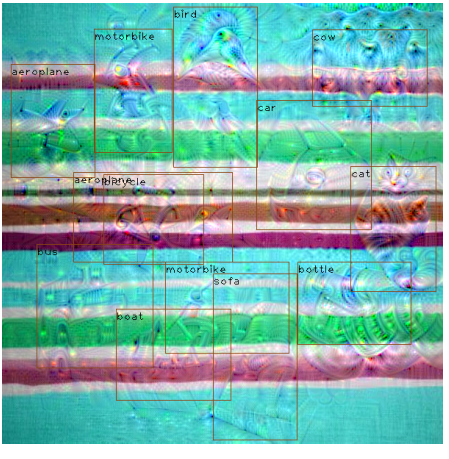
Some of the generated samples obtained using Resnet-34 *Teacher* trained on Pascal dataset are shown below. For each sample we have shown the pseduo-targets, background texture, MOIs with and without background. We use additional regularization to highlight the foreground regions for better visualization. Even though no explicit loss like batch norm is used to enforce the generated samples to look similar to original training data, yet surprisingly many of the patterns in the generated samples can be recognized. The model generates features such as the head of horse, whiskers and face of the cat, fur of the sheep, etc. The model reconstructs the object features on which it paid attention to while training with original training data.

Pseudo-Targets

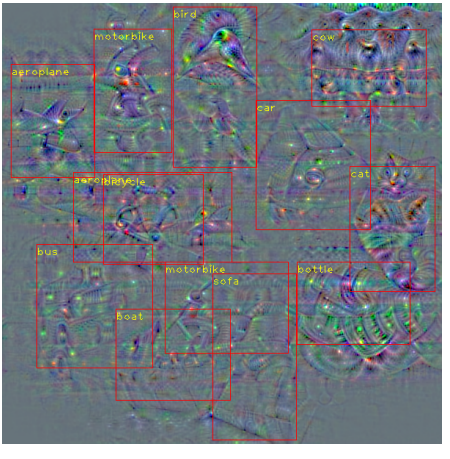


Background Initialization

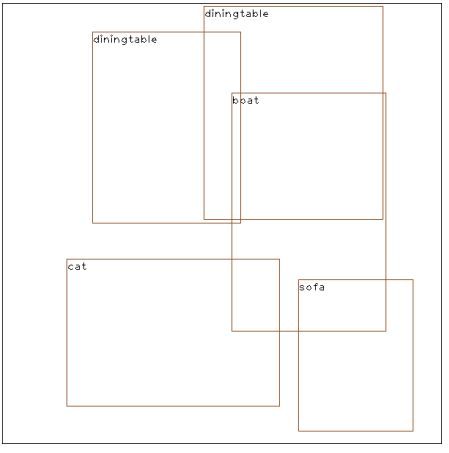
MOI



MOI without background

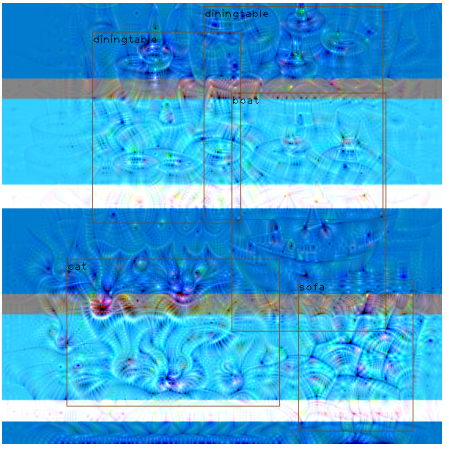


Pseudo-Targets

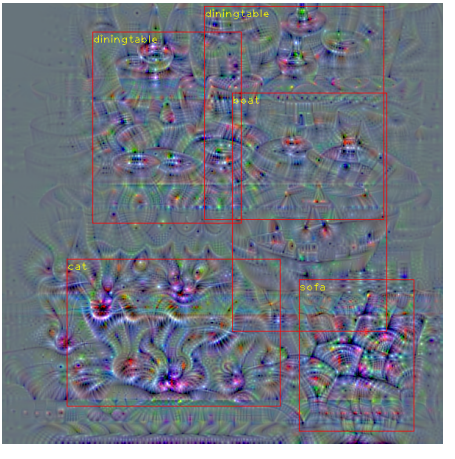


Background Initialization

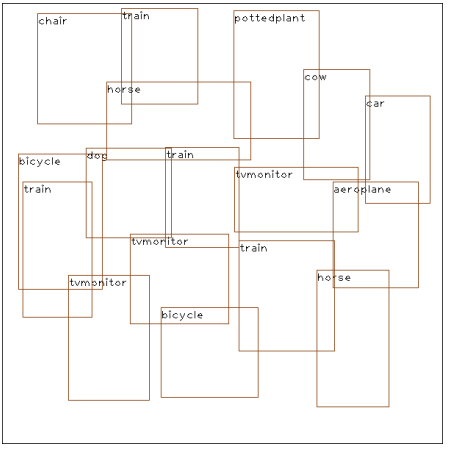
MOI



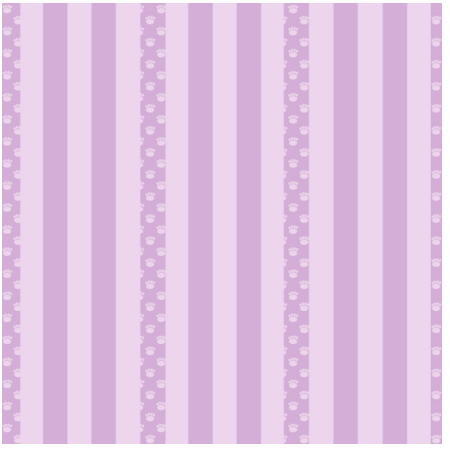
MOI without background



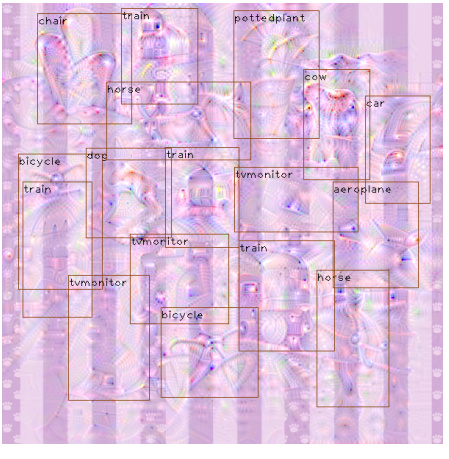
Pseudo-Targets



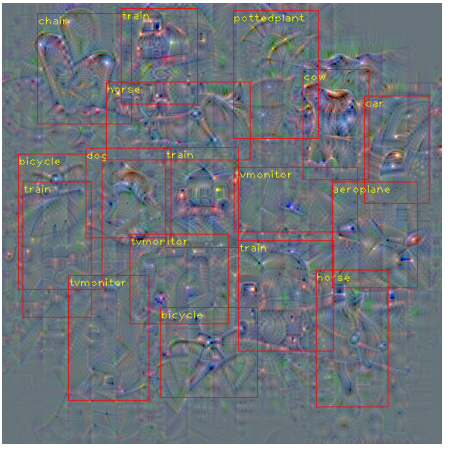
Background Initialization



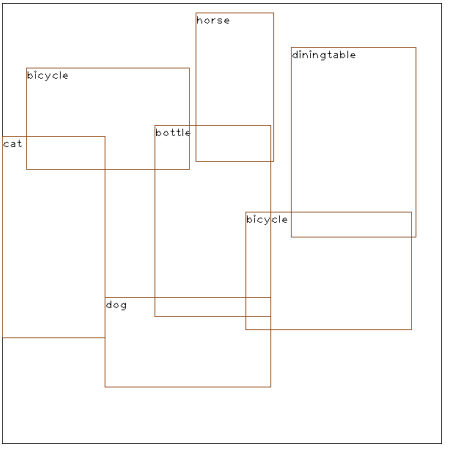
MOI



MOI without background



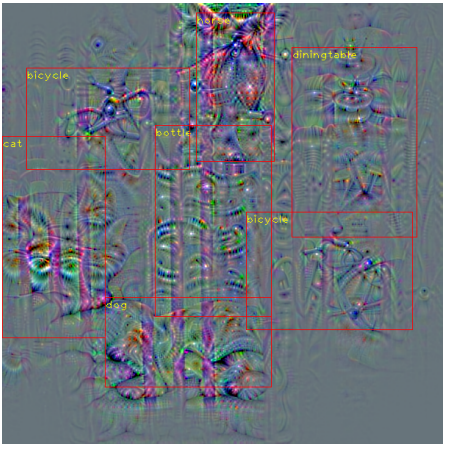
Pseudo-Targets



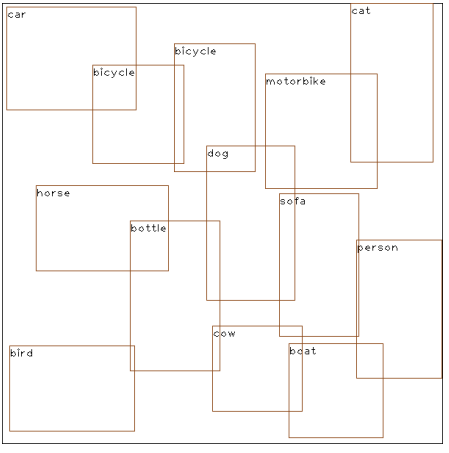
Background Initialization

MOI

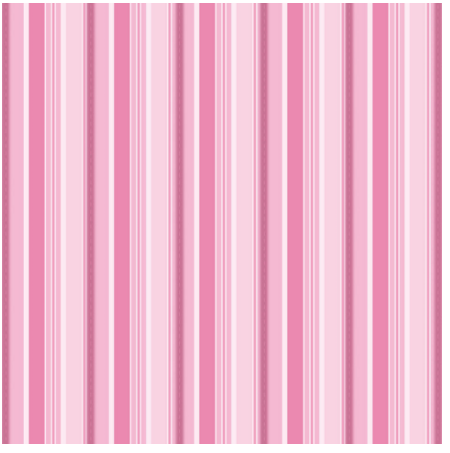
MOI without background



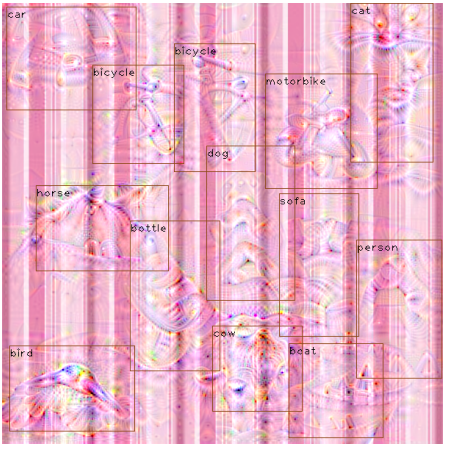
Pseudo-Targets



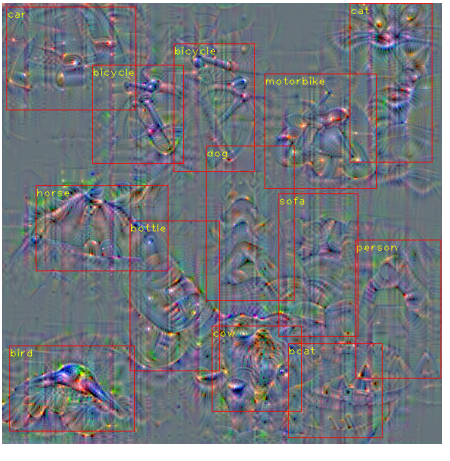
Background Initialization



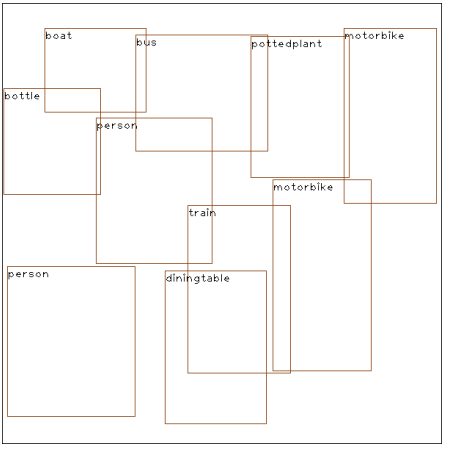
MOI



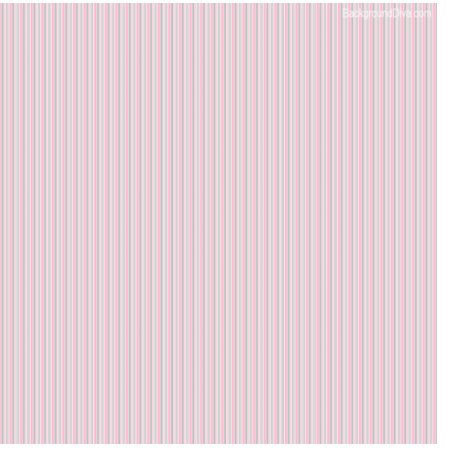
MOI without background



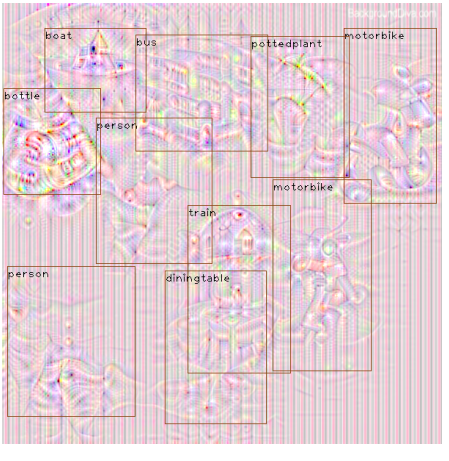
Pseudo-Targets



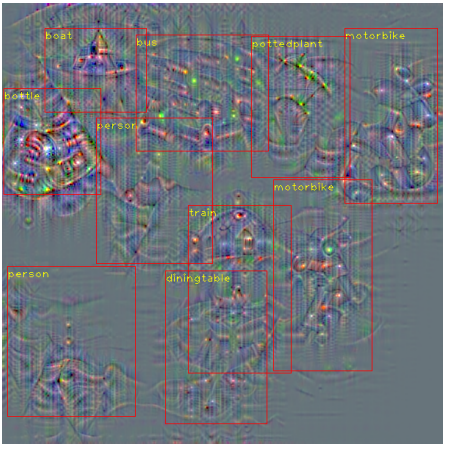
Background Initialization



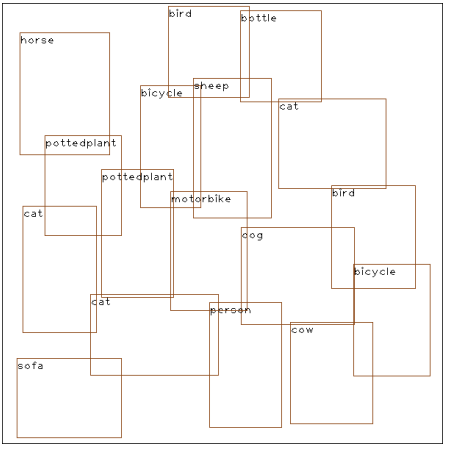
MOI



MOI without background

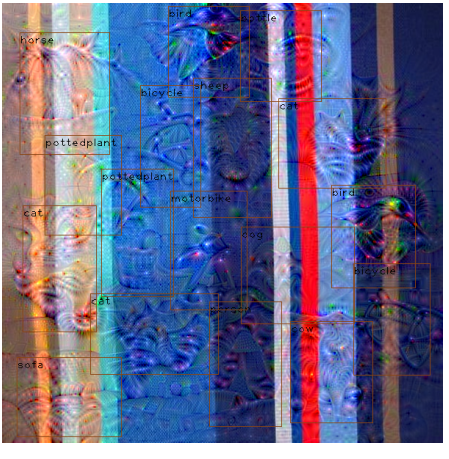


Pseudo-Targets

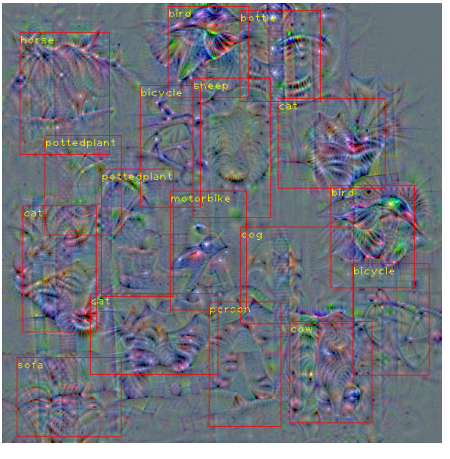


Background Initialization

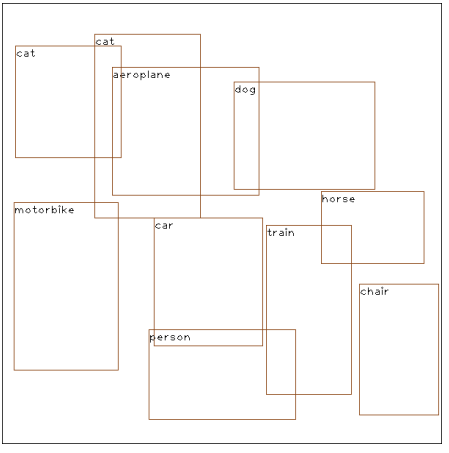
MOI



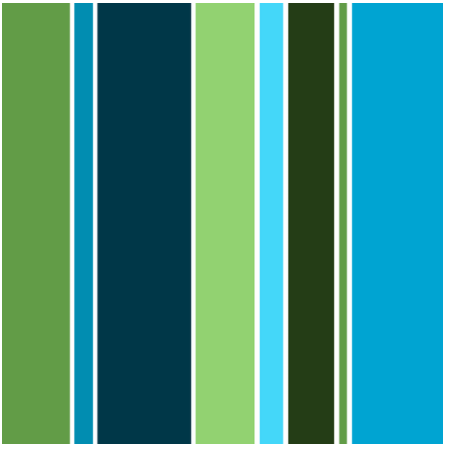
MOI without background



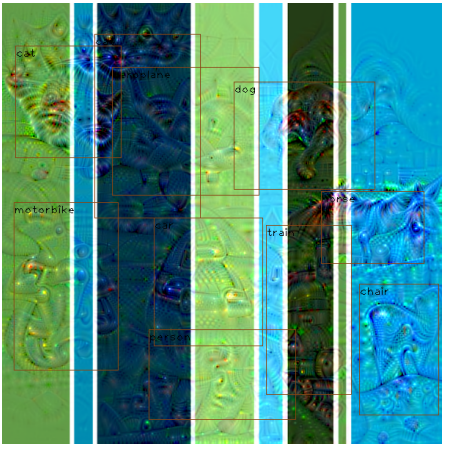
Pseudo-Targets



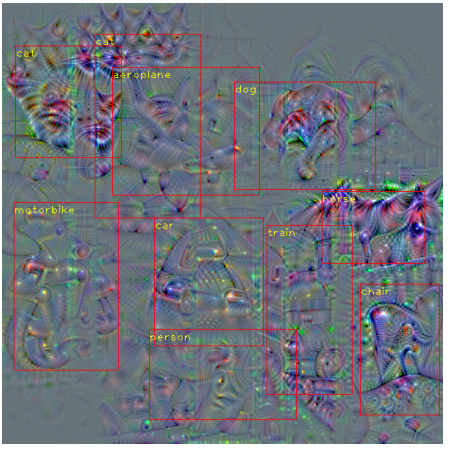
Background Initialization



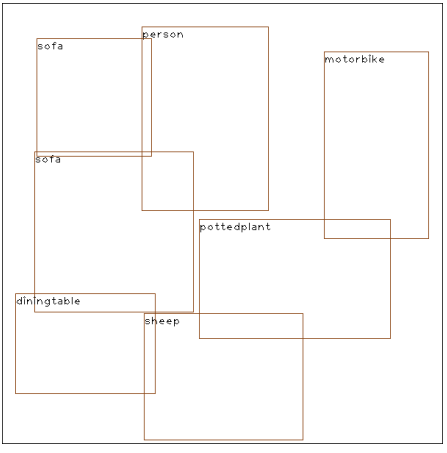
MOI



MOI without background

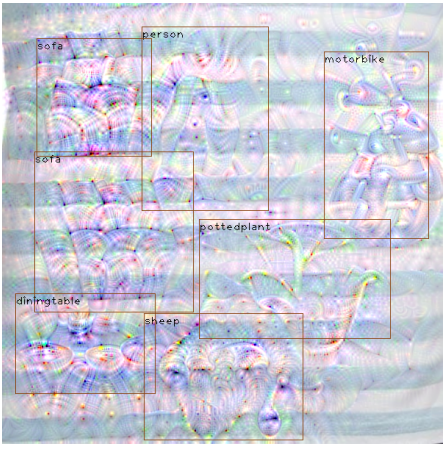


Pseudo-Targets

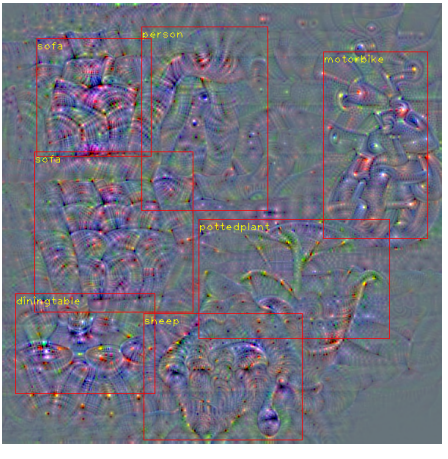


Background Initialization

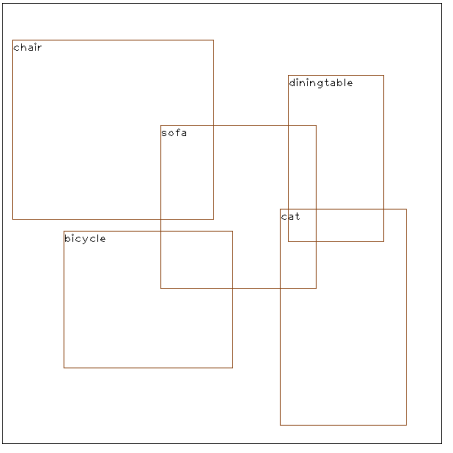
MOI



MOI without background

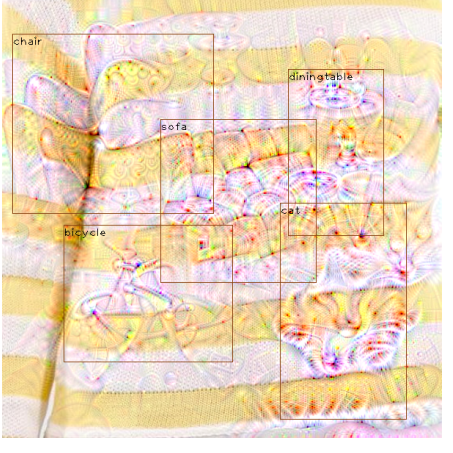


Pseudo-Targets

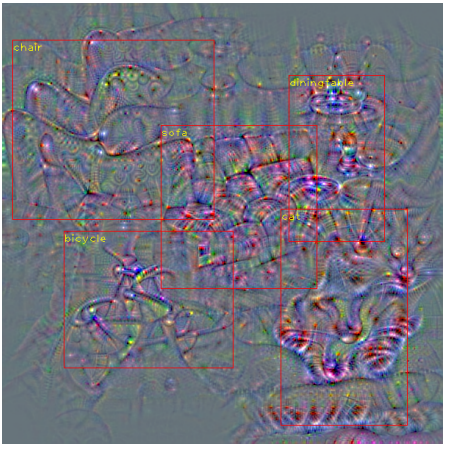


Background Initialization

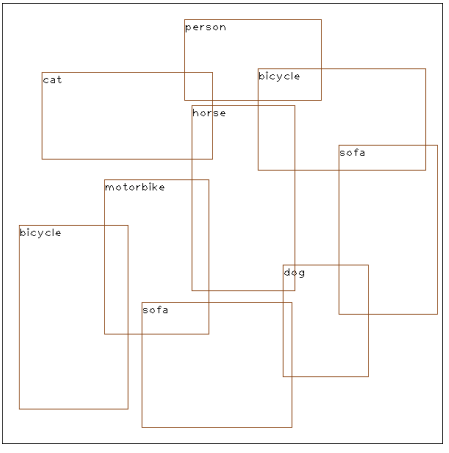
MOI



MOI without background

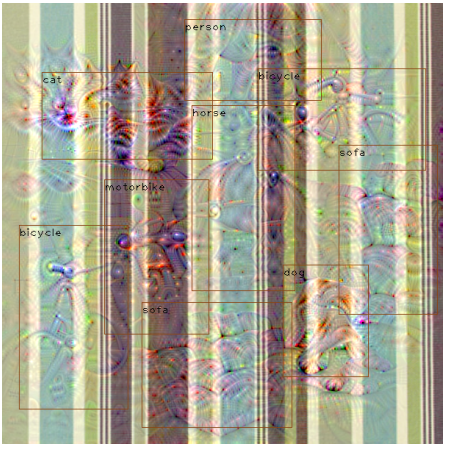


Pseudo-Targets

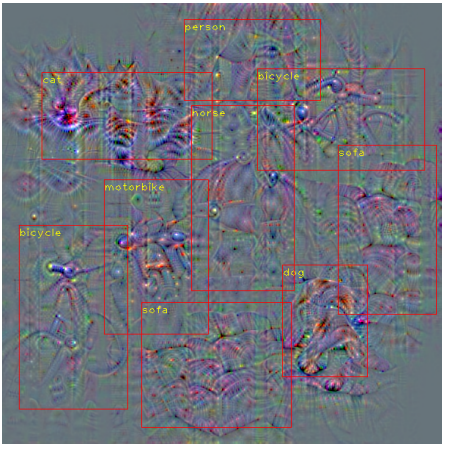


Background Initialization

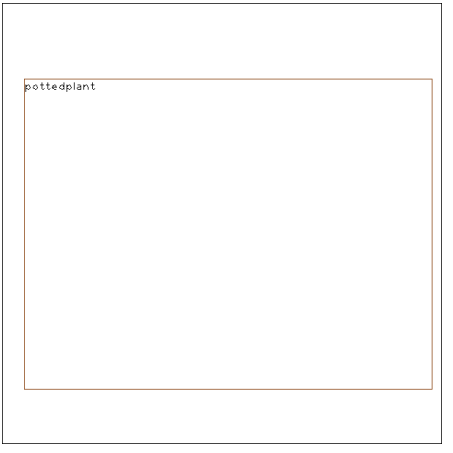
MOI



MOI without background

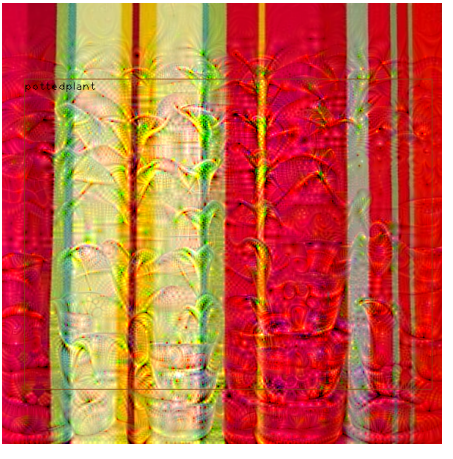


Pseudo-Targets

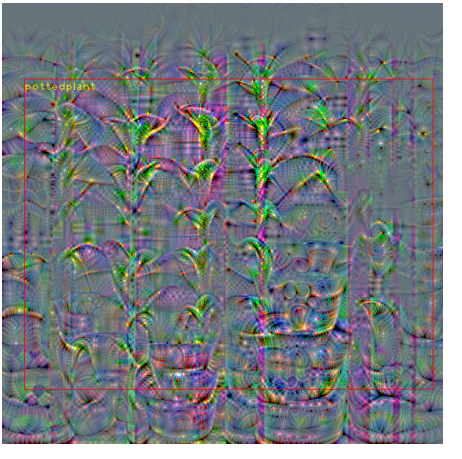


Background Initialization

MOI



MOI without background



7 Hyperparameter Details

The hyperparameters used in experiments can be divided into three categories :

- Generation of MOIs

Hyperparameters	Generation of MOIs from the *Teacher* network		
	Resnet-18 (KITTI)	VGG-16 (Pascal)	Resnet-34 (Pascal)
Number of distinct initializations	5000	5000	5000
Number of samples (K)	15000	15000	15000
Augmentation	Flip, Cutout	Flip, Cutout	Flip, Cutout
Maximum number of objects (M)	20	20	20
Imagenet Pretrained Model	Pytorch	Pytorch	Pytorch
Learning Rate	0.01	0.01	0.01
Optimizer	Adam	Adam	Adam
Batch size	16	4	8
Maximum Iterations	100	100	100
Weight to diversity loss (λ)	2	1.5	1.5
$IOU_{threshold}$	0.1	0.1	0.1

Table 6: Details of the hyperparameters used in the generation of MOIs.

- Distillation with MOIs

Hyperparameters	Distillation using MOIs as transfer set in the absence of original training data		
	Resnet-18 to Resnet-18-half (KITTI)	VGG-16 to VGG-16 (Pascal)	Resnet-34 to Resnet-18 (Pascal)
Number of samples	15000	15000	15000
Total samples with augmentation	30000	30000	30000
Weight to feature imitation loss	5	0.01	1

Table 7: Details of hyperparameters used in distillation with MOIs as a transfer set.

- Teacher training with original data

Hyperparameters	*Teacher* training on original data		
	Resnet-18 (KITTI)	VGG-16 (Pascal)	Resnet-34 (Pascal)
Learning Rate (LR)	0.001	0.001	0.001
Step decay	5	6	6
LR decay	0.1	0.1	0.1

Table 8: Hyperparameters involved in *Teacher* training

References

[1] 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.

[2] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017.

[3] Gaurav Kumar Nayak, Konda Reddy Mopuri, Vaisakh Shaj, Venkatesh Babu Radhakrishnan, and Anirban Chakraborty. Zero-shot knowledge distillation in deep networks. In *International Conference on Machine Learning*, pages 4743–4751, 2019.

[4] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018.

[5] 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.

[6] Leonid I Rudin, Stanley Osher, and Emad Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D: nonlinear phenomena*, 60(1-4):259–268, 1992.

[7] Zhi Tian, Chunhua Shen, Hao Chen, and Tong He. Fcos: Fully convolutional one-stage object detection. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9627–9636, 2019.

[8] Hongxu Yin, Pavlo Molchanov, Jose M. Alvarez, Zhizhong Li, Arun Mallya, Derek Hoiem, Niraj K. Jha, and Jan Kautz. Dreaming to distill: Data-free knowledge transfer via deepinversion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.