Beyond Classification: Knowledge Distillation using Multi-Object Impressions

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

Indian Institute of Science
Bangalore, India

Abstract

Knowledge Distillation (KD) utilizes training data as a transfer set to transfer knowledge from a complex network (Teacher) to a smaller network (Student). Several works have recently identified many scenarios where the training data may not be available due to data privacy or sensitivity concerns and have proposed solutions under this restrictive constraint for the classification task. Unlike existing works, we, for the first time, solve a much more challenging problem, i.e., "KD for object detection with zero knowledge about the training data and its statistics". Our proposed approach prepares pseudo-targets and synthesizes corresponding samples (termed as "*Multi-Object Impressions*"), using only the pretrained Faster RCNN Teacher network. We use this pseudo-dataset as a transfer set to conduct zero-shot KD for object detection. We demonstrate the efficacy of our proposed method through several ablations and extensive experiments on benchmark datasets like KITTI, Pascal and COCO. Our approach with no training samples, achieves a respectable mAP of 64.2% and 55.5% on the student with same and half capacity while performing distillation from a Resnet-18 Teacher of 73.3% mAP on KITTI.

1 Introduction

Object detection has been an important and active area of research in the computer vision community. It deals with assigning a class label and a bounding box to each object in a given image. It is widely used across several applications. For example, in autonomous cars [6, 42] where correctly localizing the traffic signals, signboards, pedestrians, etc., is crucial to avoid accidents. They have also been used to analyze aerial images [36, 44] and perform multi-object tracking [5, 43]. As detection is a crucial component in several vision-based applications, most of the research works primarily focus on making the object detection models as accurate as possible by leveraging deep networks and large amounts of training data. Such models are not suitable for deployment on portable devices that have limited memory and computational power. Therefore, there is an essential requirement to make such models compact and fast while retaining high accuracy.

Several compression techniques exist in the literature for obtaining a lightweight model from a complex deep model like pruning [13, 26, 37], quantization [22] and low-rank factorization [35]. Their limitations are: i) architecture-dependence, ii) heuristics-based, iii) drop in accuracy. On the other hand, Knowledge Distillation [17] (KD) transfers the knowledge from a trained large network (*Teacher*) to a smaller network (*Student*) by matching the temperature raised soft labels along with cross-entropy loss on the ground truth. This

* Equal contribution.

"dark knowledge" provided by the teacher helps the student models to generalize well, without much drop in accuracy. Also, it is architecture-independent and does not require any heuristics. So, we restrict ourselves to KD as a means to learn compact models.

In general, KD uses training dataset on which *Teacher* was trained, as a transfer set for knowledge transfer from *Teacher* to *Student*. However, we may only have access to the trained models and not their training data as several companies may have proprietary rights over them (*e.g*. pretrained models [19] on Google's JFT-300M [21] proprietary dataset). Also, if the data is sensitive (*e.g*. medical and biometric data), it may not be shared due to privacy concerns. Recently several works have identified such issues [4, 28, 29] for classification setting. The solutions proposed for classification problems either synthesize transfer set directly using the trained Teacher model [29, 46] or learn the target data distribution through generative models [4, 28]. However, both of these approaches cannot be readily applied for object detection which is a more challenging and difficult problem. Object detection has dual priorities of object classification along with localization, and at the same time each image may contain variable number of objects. Moreover, each object can be of different spatial scales and aspect ratios, which can be present at various locations in an image.

Several works[3, 38, 41] in object detection, assume the availability of the training data on which *Teacher* is trained and uses them to distill the knowledge to *Student*. However, to perform KD for object detection in the absence of training data is a non-trivial and challenging problem. In our current work, we addressed this problem by synthesizing pseudo-dataset using only the trained Faster RCNN based *Teacher* detection network and use them as a transfer set to conduct knowledge transfer in the data-free setting. Our proposed approach is broadly divided into two phases: Generation and Distillation as shown in Fig. 1.

In the generation phase, we first prepare pseudo-targets using proposed Algo. 1 in absence of the training data and its statistics. Every object has three attributes associated with it : size, location and class label. We use the anchor scales and ratios obtained from *Teacher* model to estimate the range of the object sizes. Based on the number of objects required to be placed in a given image dimension with a given IOU overlap constraint, we restrict the minimum and maximum possible areas for each object. We use Power Law distribution to sample the object area from this range. As the class distribution of objects is not known, we uniformly sample the class labels for each object.

The second step in the generation phase is to synthesize inputs corresponding to the prepared pseudo-targets (details in proposed Algo. 2). We use random texture images as background initialization and optimize them by backpropagating the gradients of Faster RCNN loss (L_{gt}) and our proposed diversity loss (L_{div}) via the frozen *Teacher* model. Our generated samples named as *Multi-Object Impressions* (MOIs) are impressions of multiple objects of same/different classes at different locations with different scales. We further make our generated MOIs invariant to augmentation operations like flipping, cutout via differential batch augmentation, and the proposed diversity loss further improves the intraclass variation on objects (shown in Fig. 4). In the distillation phase, we use our pseudo-dataset as transfer set and perform zero-shot KD. Our generated data is even suitable to be used beyond transfer set as we obtain reasonable mAP even while training the network from scratch with our data.

We thus, summarize our contributions as follows:

- We are the first to attempt knowledge distillation on object detection, assuming zero knowledge about the training data and their statistics.
- We propose an algorithm to synthesize pseudo-dataset comprising of target labels and corresponding input data, i.e. Multi-Object Impressions using a trained deep Faster RCNN detection model. Our pseudo-dataset is robust to augmentation operations.

- We show the utility of such pseudo-dataset as a transfer set for distillation and are even suitable to train a detection network without any *Teacher* assistance. Moreover, they can be used as augmentation in the presence of proxy data or few training samples.
- We propose a diversity loss that enhances the intraclass variation on the foreground objects of MOIs, leading to an improved distillation performance.
- The effectiveness of our proposed approach is demonstrated across several architectures and benchmark datasets like KITTI, Pascal and COCO.

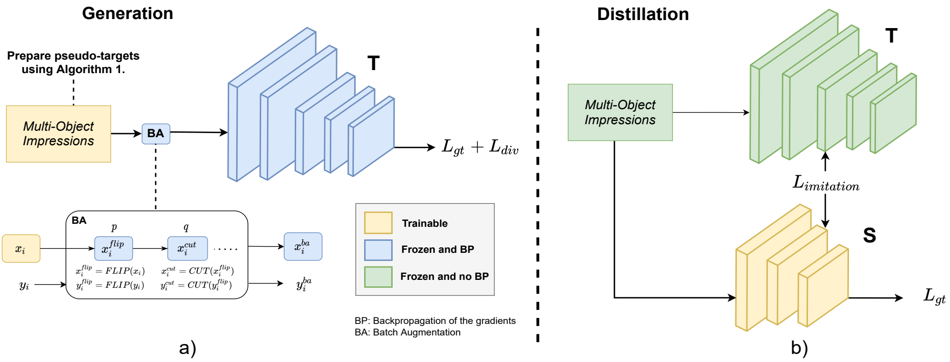


Figure 1: Our proposed approach for zero-shot kd on object detection: a. Generation (left) where pseudo-dataset is synthesized using Algo. 2.; b. Distillation (right) is done using our pseudo-dataset as a transfer set. Note: we use our prepared pseudo-targets as ground truth to calculate the loss values.

2 Related Works

KD for classification: The task of KD for the image classification, involves training a *Student* network on a temperature-raised softmax output of the teacher network. In the past years, the trend in this domain has been towards using as less data as possible. Kimura *et al.* [20] used few training samples and generated pseudo-examples for KD , whereas Lopes *et al.* [25] stored feature activations of all layers and used them in the form of meta-data. Nayak *et al.* [29] proposed a data-free approach to KD, where the training set was composed of data-impressions obtained via modeling the softmax output space through Dirichlet distribution. Micaelli *et al.* [28] proposed an adversarial method that trains a generator iteratively to craft images that cause the student to poorly match the teacher and subsequently used them to perform distillation. Chen *et al.* [4] used an adversarial generator to synthesize images such that the teacher network gives high feature response and classifies them with high probability. Few works [14, 47] also uses batchnorm statistics of the trained classifier to regularize the feature distribution. Li *et al.* [23] uses features from several pretrained models and data mixup strategy to avoid model bias while matching such feature statistics. Thus, the problem of implementing a zero-shot/data-free approach to KD for image classification has produced a diverse set of solutions, each with a distinctive approach.

KD for object detection: Chen *et al.* [3] proposed a distillation framework for object detection that used the Faster-RCNN model, which acts as a baseline model in a data-driven distillation framework. This approach used hint learning [32] to improve the feature representation capability of the student, after which the classification and regression outputs of both the proposal network and the region CNN were distilled to the student. Wang *et al.* [41] avoid full imitation to overcome the noise introduced from various background instances and use fine-grained feature imitation exploiting important information in the near object anchor locations, which helps the Teacher to generalize better. All of these approaches in detection use large training datasets, often with several classes to perform distillation.

One way to handle the absence of training data is to generate synthetic data and use them for the downstream task. Few works such as [12, 16, 40] generate synthetic data using external tools like CAD or depend on the availability of cropped images of objects. Therefore, there is an implicit assumption of familiarity with the target categories. In absence of such priors, synthetic samples are generated from a pretrained Yolo network in [2] but their method depends on training data statistics in the form of batch norm. Also, their focus is on one-stage detectors with the goal of only knowledge transfer.

We, in this paper, focus on two stage-detectors like Faster RCNN with an objective of model compression along with knowledge transfer where zero knowledge about training samples and training data statistics are assumed.

3 Proposed Approach

Our approach for zero-shot KD for object detection is broadly divided into two phases (also shown in Figure 1):

- **Generation**: In order to synthesize samples in absence of training data and their statistics, we first need to prepare 'pseudo-targets' (in Sec. 3.1). We generate Multi-Object Impressions (MOIs) corresponding to them, using the pretrained weights of *Teacher* network via backpropagation (in Sec. 3.2).
- **Distillation**: The generated dataset ($\hat{D}$) can then be used as a transfer set to distill knowledge into the *Student* network (in Sec. 3.3).

We define our pseudo-dataset as $\hat{D} = \{(x_i, y_i)\}_{i=1}^{K}$, where K is the number of samples to be generated, x_i is the i^{th} input with corresponding pseudo-target y_i. The pseudo-target is defined as $y_i = \{(c_{io}, bbox_{io})\}_{o=1}^{N_i}$, where N_i is the number of objects in input x_i, c_{io} denotes target class and $bbox_{io}$ denotes target bounding box coordinates, for the o^{th} object in x_i.

3.1 Preparation of Pseudo-Targets

In detection, multiple objects can be present in an image. These objects can be of variable sizes, at different locations and belong to same or different classes which makes the problem complex. We aim to prepare pseudo-targets that are close to the annotations of training data in terms of sizes, locations and class labels of objects.

The **size** of an object is defined in terms of scales and aspect ratios [31]. We leverage on the anchor ratios and scales obtained from the pretrained detection model to get some insight regarding the range of object sizes in the training dataset. Please note that it only provides a clue about the estimated range of the object sizes in the training dataset but in no way reveals their actual sizes or the class labels.

The **location** of an object is dependent on the other object's locations. We need to take care of their overlap so that one object does not get placed over other objects or have very high overlap. It is desirable to have the overlap within some predefined threshold. We use IoU measure to enforce such a constraint.

The **class label** distribution of the training data is unknown and no prior information or metadata is available, so we use uniform sampling to assign a class label to an object. Algorithm 1 contains the detailed steps to obtain pseudo-targets for pseudo-dataset ($\hat{D}$).

We define the minimum and maximum object area as A^{min} and A^{max} based on the anchor scales defined by the teacher model (Line 1). The maximum number of objects that are allowed in any sample is denoted by M. For an i^{th} image (x_i), the important factors on which its pseudo-target depends are: the number of objects, class label, size (aspect ratio and

Algorithm 1: Preparation of B pseudo-targets for pseudo-dataset $\hat{D}$ using *Teacher* T

```
   Input: $W_i, H_i$: width and height of $i^{th}$ input,  $IoU_{thresh}$: max IoU threshold,  $C$: # foreground classes
          $t_1, t_2$: control $A^{max}$ and $A^{min}$ respectively,  $R_{min}, R_{max}$: min and max aspect ratio,
   Output: $B$ Pseudo-Targets: $\{y_i\}_{i=1}^{B}$
1  $A^{min}$, $A^{max}$ $\leftarrow$ Minimum and Maximum object area based on range of anchor scales and $t_1, t_2$
2  Choose $M$ such that $M \leq (W_i \cdot H_i)/A^{min}, \forall i \in [1..B]$
3  for $i = 1 : B$ do
4      $y_i \leftarrow \varnothing$;  $cur\_iter \leftarrow 0$;  $max\_iters \leftarrow 50$
5      $N_i \sim \mathcal{U}\{1, M\}$;  $a_i \leftarrow M/N_i$
6      $A_i^{max} \leftarrow \min(A^{max}, (W_i \cdot H_i)/N_i)$
7      for $o = 1 : N_i$ do
8          $c_{io} \sim \mathcal{U}\{1, C\}$;  $r_{io} \sim \mathcal{U}(R_{min}, R_{max})$
9          while $cur\_iter \leq max\_iters$ do
10             $x_{io} \sim P(a_i)$, where $P(x;a) = ax^{a-1}, 0 \leq x \leq 1, a > 0$
11             $A_{io} \leftarrow (A_i^{max} - A^{min}) \cdot x_{io} + A^{min}$
12             Obtain width $w_{io}$ and height $h_{io}$ using area $A_{io}$ and aspect ratio $r_{io}$
13             $[\,ctr_{io}^{x},\ ctr_{io}^{y}] \sim [\,\mathcal{U}\{w_{io}/2, W_i - w_{io}/2 + 1\},\ \mathcal{U}\{h_{io}/2, H_i - h_{io}/2 + 1\}]$
14             $bbox_{io} \leftarrow [ctr_{io}^{x}, ctr_{io}^{y}, w_{io}, h_{io}]$
15             if $(IoU(\ bbox_{io},\ bbox) < IoU_{thresh}\ \forall bbox \in y_i)\ or\ (cur\_iter == max\_iters)$ then
16                 $y_i \leftarrow y_i \cup \{(bbox_{io}, c_{io})\}$;  break
17             else
18                 $cur\_iter \leftarrow cur\_iter + 1$
19             end
20         end
21     end
22 end
```

area) and location of each object in the image. We now discuss in detail how our proposed algorithm handles each of these factors.

The number of objects is sampled uniformly i.e. $N_i \sim \mathcal{U}\{1, M\}$. The class label c_{io} is assigned to the o^{th} object using uniform distribution i.e. $c_{io} \sim \mathcal{U}\{1, C\}$ where C is the number of foreground classes. The aspect ratio of each object (r_{io}) is uniformly sampled from the range of anchor ratios defined by the teacher model. Based on the dimensions of x_i and value of N_i, we constrain the maximum possible object area and denote it by A_i^{max}(Line 6). We utilize Power Law distribution P to sample from the range [A^{min}, A_i^{max}], denoted by A_{io}. We define the parameter a_i as a function of N_i (Refer to supplementary for details). Using A_{io} and r_{io}, width (w_{io}) and height (h_{io}) of the target bounding box are obtained for the o^{th} object in x_i (Lines 10- 12). Next (Lines 13- 19), we place the o^{th} object in x_i such that object overlap between each pair is less than the IoU threshold. The threshold condition often gets satisfied within a few iterations. Otherwise we save the pseudo-targets after maximum iterations.

3.2 Crafting MOIs for Pseudo-Targets

Let the pretrained Faster RCNN model be denoted by T and its learnt model parameters by θ_T which is trained on a detection training dataset (D). In the absence of the dataset D, we aim to synthesize pseudo dataset $\hat{D}$. After obtaining the pseudo-target y_i using proposed algorithm 1, we need to synthesize x_i using pretrained weights of T such that:

$$T(x_i) = y_i, \forall i \in [1..K] \tag{1}$$

In order to satisfy this condition, every data sample x_i requires to have impressions of objects of variable sizes at different locations belonging to same or different classes. Therefore, we call such x_i's as Multi-Object Impressions (MOIs). In other words, the i^{th} sample of pseudo-dataset $\hat{D}$ corresponding to i^{th} pseudo-target (y_i) is the i^{th} MOI (denoted by x_i).

The RCNN based detectors classify the diverse backgrounds in images into a single background class. To handle this background variability, we initialize our MOIs with samples from a texture dataset [7]. In comparison to random noise, feature maps extracted by the layers of base network of pretrained detector have higher activation values and acts as better initialization. The texture images do not contain any objects. Also, the *Teacher* network predicts background class on such images with high confidence. Thus, the initialization of each x_i's with texture image act as background initialization for the MOIs.

Data augmentation [48] is a commonly used technique to improve the performance of the training model. But we cannot directly apply data augmentation on MOIs during KD, since MOIs are the samples on which the *Teacher* model is confident. We cannot predict the behaviour of the *Teacher* on the augmented MOIs. The *Teacher* may not predict the required pseudo-target on the augmented MOIs and can violate equation 1. So, we make the MOIs robust to augmentation operations via differentiable batch augmentation.

Batch augmentation [18] (BA) improves the generalization of the model and also leads to faster convergence. We use this approach[1] to make our MOIs invariant to certain augmentations. We perform two differential batch augmentations, namely Flip and Cutout [9]. In BA module in Fig. 1, x_i (i.e. i^{th} MOI) is passed through differentiable augmentation operations with probability p and q. We set $p = q = 0.5$ and x_i^{ba} is the output of BA module which is then fed to the *Teacher* network.

After the initialization and augmentation, we optimize each x_i keeping θ_T fixed with target labels $y_i, \forall i \in [1..K]$ using the loss L as defined:

$$L_{MOI} = \lambda_{gt} L_{gt} + \lambda_{div} L_{div} \tag{2}$$

where λ_{gt} and λ_{div} are hyperparameters which are used to balance the losses. The description of the losses are mentioned below:

Detection Loss (L_{gt}): The usual classification and bounding boxes losses [31] applied at RPN and RCNN layers in training the network T. In absence of original training dataset (D), we use our pseudo-dataset ($\hat{D}$).

Diversity Loss (L_{div}): To ensure intraclass variability across the foreground objects, we define diversity loss as:

$$L_{div} = -\frac{1}{C} \sum_{c=1}^{C} \frac{1}{|N_c|} \sum_{(i,j) \in N_c} d(f_i, f_j) \tag{3}$$

where C denotes the number of foreground classes, N_c denotes the collection of pairs of foreground objects belonging to class c in the current batch. f_i and f_j denote the pooled feature vectors of foreground objects i and j that also belongs to same class c. d is distance metric (euclidean, cosine). We use cosine similarity for the experiments.

Training Setup: MOIs initialized with texture images are trainable. The gradients are back-propagated with respect to each x_i's through the frozen *Teacher*. Each x_i is optimized for N iterations to minimize the loss between the *Teacher*'s prediction and pseudo-targets y_i's. The RPN loss from L_{gt} forces the region proposals to be near to the target bbox's in y_i's. Further RCNN loss corrects the proposals and predicts each target class in y_i's with high confidence. This would eventually lead to having foreground objects' impressions such that the *Teacher*

[1] performs augmentation on MOIs in a batch

Algorithm 2: Generation of pseudo-dataset ($\hat{D}$)

```
Input: $T$: Pretrained Faster RCNN model, $K$: Number of samples, $N$ : Number of iterations
Output: $\hat{D}$ : pseudo-dataset
$\hat{D} \leftarrow \varnothing$
Select batch size $b$, s.t. $K$ mod $b = 0, b > 1$
for $K/b$ batches do
    Sample a minibatch of $b$ background images, $\{x_1, x_2, \ldots, x_b\}$ from DTD texture data [7]
    Obtain a minibatch of $b$ pseudo-targets, $\{y_1, y_2, \ldots, y_b\}$ using Algo. 1
    Associate the pseudo-targets with sampled images $(x, y) = \{(x_1, y_1), (x_2, y_2), \ldots, (x_b, y_b)\}$
    for $N$ iterations do
        Perform Batch augmentation $(x^{ba}, y^{ba}) \leftarrow BA(x, y)$
        Update $x$ by descending its gradient $\nabla_x L_{MOI}(T(x^{ba}), y^{ba})$
    end
    $\hat{D} \leftarrow \hat{D} \cup (x, y)$
end
```

network when fed with optimized MOIs, predicts the desired pseudo-targets. Diversity loss encourages objects of a target class to have diverse features which helps it to improve KD performance. Refer to Figure 5 for visualization of synthesized MOIs and supplementary for more visualizations. Algo. 2 contains overall steps involved in the generation of $\hat{D}$.

3.3 Distillation using pseudo dataset ($\hat{D}$)

Let the student model be denoted by S and its trainable parameters by θ_S. After obtaining the pseudo-targets and corresponding MOIs (using Algorithm 2), we use our synthesized pseudo dataset ($\hat{D}$) as transfer set to perform knowledge transfer from T to S using the detection loss (L_{gt}) and feature imitation loss ($L_{imitation}$) used in [41]. We evaluate the generalization ability of the student S trained with transfer set $\hat{D}$ through mean average precision (mAP) on actual test samples and compare its performance with data dependent approaches.

4 Experiments

We first use Resnet-18 [15] Faster R-CNN model as *Teacher* network trained on KITTI [11] benchmark dataset. The performance of the trained *Teacher* model is 73.3% mAP. The models are evaluated based on the split followed by [1, 27, 41] using official evaluation tool.

Several design choices are possible in crafting MOIs like maximum number of objects allowed per sample (M) and the size of pseudo-data (K). We discuss the effect of the aforementioned major factors on distillation performance in subsequent sections. We fix the size of the MOIs as 600×600 dimensions. We take t_1, t_2 and IOU_{thresh} as 1.2, 0.8 and 0.1 respectively. We use Adam optimizer and Pytorch framework for all the experiments. Refer supplementary for hyperparameter details used in experiments.

4.1 Maximum objects per sample

We vary the maximum number of objects (M) possible in any sample of pseudo-dataset $\hat{D}$ and check its impact on distillation performance of Resnet-18-half student. It is evident from Figure 2 that mAP improves with an increase in the value of M. Large value of M offers more variation in object sizes. However, we cannot have an arbitrary large M due to the constraint specified in line no. 2 of Algo. 1. Thus, we choose the value of M as 20 for subsequent experiments. These experiments are performed using transfer set size of 2500 MOIs. We perform an ablation on the number of generated samples in the next section.

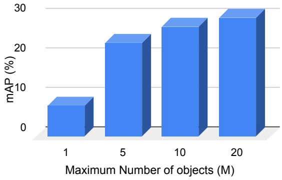


Figure 2: Comparison of distillation performance when maximum number of objects is varied.

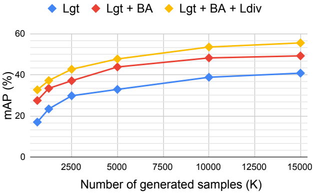


Figure 3: Detection accuracy (w/ KD) by varying number of samples, MOIs crafted via different losses.

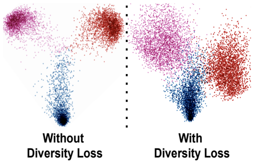


Figure 4: Visualization of pooled features of objects on the generated samples.

4.2 Size of pseudo-dataset

We generate different sized pseudo-datasets by varying K and analyze its effect while performing distillation on Resnet-18-half student. For a particular value of K, we generate MOIs using three different ways : a) using L_{gt} loss, b) using L_{gt} loss and batch augmentation (BA), c) using L_{div} loss along with L_{gt} loss and BA. The results are shown in Figure 3.

a) $\boldsymbol{L_{gt}}$: The performance of the *Student* model with L_{gt} (shown in blue curve) improves by increasing K and achieves 40.8% mAP with 15000 pseudo-samples.

b) $\boldsymbol{L_{gt}}$ **+ BA**: In order to make our MOIs robust to augmentations, we do simple differential batch augmentation like flipping and cutout during generation. This enforces the MOIs to be invariant to the augmentation operations. Robust MOIs indeed helps in improving the performance of the *Student* model as evident from Figure 3 (shown in red curve). Though BA encourages robustness, but do not explicitly force the object impressions belonging to the same class to be diverse. So, in order to enforce high intraclass variation, we additionally add our proposed diversity loss (L_{div}).

c) $\boldsymbol{L_{gt}}$ **+ BA +** $\boldsymbol{L_{div}}$: After adding L_{div} loss on top of L_{gt} and BA, we observe further gain in the performance of the *Student* model (shown in yellow curve in Figure 3).

Proposed Diversity Loss ($\boldsymbol{L_{div}}$**)**: We visualize the pooled features of foreground object impressions for each class. From Fig. 4 (left), we observe that without the diversity loss, the objects belonging to each class in the feature space lie close to their class means, and objects are well separated across classes. Through our proposed loss, we improve the intraclass variation as shown in Fig. 4 (right). Also, sample density near the boundary region has increased which helps the *Student* to learn the decision boundary resulting in improved mAP.

4.3 Results on KITTI

In Table 1, we report our overall results while distilling from Resnet-18 (73.3% mAP) as *Teacher* to Resnet-18 and Resnet-18-half as *Student*'s. We compare our data-free kd approach (no original training samples) with data kd approach (all original training samples) which serves as an upper bound. We also compare against the baseline that uses in-domain [8] and out-domain [33] data and report their average performance. We get a significant improvement over the baseline. Our MOIs generated through a combination of L_{gt} and L_{div} losses using BA obtains distillation performance of 64.2% and 55.5% mAP on student with same and half capacity. Please note that even without KD, we observe decent mAP of 57.8% by training Resnet-18 with our MOIs which shows its utility to be used beyond transfer set.

4.4 Results on Pascal and COCO

Pascal VOC07 dataset [10] is another popular benchmark dataset for object detection tasks and is relatively more complex than KITTI dataset. It contains foreground objects that can belong to one of the 20 classes. We perform distillation from two different *Teacher* networks i.e. Resnet-34 and VGG-16 trained on Pascal. Similar to KITTI, we synthesize samples with M as 20, K as 15000 with batch augmentation, and optimize using $L_{gt} + L_{div}$ losses. The models are evaluated using Pascal VOC convention i.e. mAP at 0.5 IoU. The results are shown in Table 2. Our proposed zero-shot kd method obtains a distillation performance of 55.2% on VGG-16 and 58.6% mAP on Resnet-34. In absence of teacher assistance, we still get a decent performance of 49.3% on VGG-16 and 50.9% mAP on Resnet-34 when our pseudo-dataset is used to train the networks from scratch.

Setting	**Training Method**	**Loss on MOIs**	**Teacher**	**Student**	**mAP**
With training data	w/o KD [31]	N/A	Resnet-18	———	73.3
	w/ KD [41]			Resnet-18-half	65.8
Without training data (Ours)	Baseline	N/A	Resnet-18	Resnet-18-half	42.3
	w/ KD	Lgt			40.8
	w/ KD	Lgt+BA			49.2
	w/ KD	Lgt +BA +Ldiv			55.5
	w/o KD		———	Resnet-18	57.8
	w/ KD		Resnet-18		**64.2**

Table 1: KD results using our proposed approach on KITTI dataset.

Setting	**Training Method**	**Teacher**	**Student**	**mAP**
With training data	w/o KD [31]	VGG-16	———	70.4
	w/o KD [31]	Resnet-34	———	70.1
	w/ KD [41]		Resnet-18	67.8
Without training data (Ours)	w/o KD	———	VGG-16	49.3
	w/ KD	VGG-16		55.2
	w/ KD	Resnet-34	Resnet-18	46.3
	w/o KD	———	Resnet-34	50.9
	w/ KD	Resnet-34		**58.6**

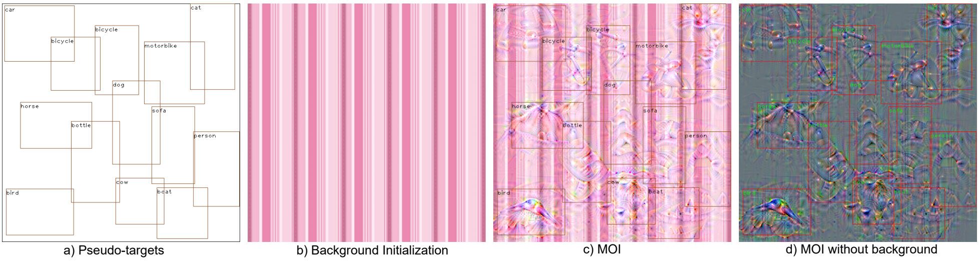
Table 2: Results of distillation on Pascal dataset using our proposed data-free approach.

Setting	**Training Method**	**COCO@0.5**	**COCO@[0.5,0.95]**
With training data	without KD [31]	53.8	33.9
Without training data	without KD (**Ours**)	30.9	15.6
	with KD (**Ours**)	**41.3**	**24.0**

Table 3: Performance of our proposed method on COCO dataset (with and without KD) on Resnet-101

We also perform experiments on COCO [24] which is large scale object detection dataset with 80 object categories. The models are evaluated as : a) average precision with IoU at 0.5 and b) mean of the average precisions calculated with IoU starting with 0.5 to 0.95 having a step size of 0.05. We denote the former one as COCO@0.5 and later one as COCO@[.5,.95]. The results are shown in Table 3. We use Resnet-101 pretrained model [45] on COCO as the *Teacher* network which obtains 53.8% and 33.9% mAP on evaluation using COCO@0.5 and COCO@[.5,.95]. Despite of the large *Teacher* architecture and complex training dataset, our zero-shot kd method using our pseudo-dataset as transfer set obtains decent performance of 41.3% and 24.0% while 30.9% and 15.6% without kd on COCO@0.5 and COCO@[.5,.95] respectively. Hence, our proposed method is scalable even on large scale detection datasets.

We also tried to use other well-known losses like TV loss [34] and L2 regularizer to generate natural-looking MOIs but we observed that such losses did not yield improvement in mAP. We also empirically demonstrated that our prepared pseudo-targets reasonably capture the label distribution of training data. Refer supplementary for more details and analysis.



a) Pseudo-targets

b) Background Initialization

c) MOI

d) MOI without background

Figure 5: Visualization of a synthesized sample: a) Pseudo-Targets using Algo.1, b) Background Init. using texture image, c) MOI obtained using Algo.2, d) MOI without background for better visibility.

4.5 Efficacy of our pseudo-dataset beyond KD

Architecture	Original Dataset (D)	Our Pseudo-dataset ($\hat{D}$)	
	mAP (Upper Bound)	mAP (w/ KD)	mAP (training from scratch)
Resnet-18 (KITTI)	73.3	64.2	**57.8**
Resnet-34 (Pascal)	70.1	58.6	**50.9**
Resnet-101 (COCO)	53.8	41.3	**30.9**

Table 4: Performance (in %) on our pseudo-dataset when used to train the network from scratch across architectures and datasets.

Setting	Dataset	mAP
Proxy Data	Cityscapes	53.1
	Cityscapes + **Ours**	62.0 (+ 8.9)
	Synthia	55.6
	Synthia + **Ours**	62.8 (+ 7.2)
Few original training samples	5% KITTI	58.5
	5% KITTI + **Ours**	64.8 (+ 6.3)
	10% KITTI	60.1
	10% KITTI + **Ours**	66.2 (+ 6.1)

Table 5: Performance (in %) on our pseudo-dataset when used as augmentation on Resnet-18 without KD

As shown in Table 4, we obtain respectable performances across datasets and architectures when the network is trained from scratch using our pseudo-dataset ($\hat{D}$) without any Teacher assistance. We observe a similar performance gap between the network trained on $\hat{D}$ and D, even on a large scale challenging dataset like COCO. This consistent behaviour highlights the scalability of our proposed approach across datasets and architectures. Our pseudo-dataset $\hat{D}$ has a similar behaviour as D. For instance; our generated pseudo-samples are robust to augmentations like original data. Also, when KD is applied on $\hat{D}$, we obtain a significant performance improvement similar to the data-KD setup.

We further analyse the efficacy of our pseudo-dataset by investigating its performance under different scenarios: a) proxy data + pseudo-data and b) few samples of training data (few-shot) + pseudo-data. We take an equal number of samples from proxy data and pseudo-data for a fair comparison. Also, total number of samples is similar to the size of the KITTI training dataset. The models are evaluated using the official KITTI evaluation tool. As shown in Table 5, we obtain a noticeable performance improvement of $\approx 6-9\%$ when our generated data ($\hat{D}$) is used in conjunction with either proxy data or few samples of original training data. However, we do not observe any significant performance improvement when our pseudo-data is used along with all training samples. *The performance on other state-of-the-art object detection approaches such as Yolo [30] and FCOS [39], are put in the supplementary (sec.* 1*) where our dataset ($\hat{D}$) is used as training set.* Overall, the observations from Tables 4 and 5 and the results from supplementary indicate that our pseudo-dataset ($\hat{D}$) can be treated as reliable representatives of the original training data and can be used in applications beyond KD where such data is required but is present in small amounts or not present at all.

5 Conclusion

Recent approaches have focused much on data-oriented knowledge distillation (KD) for object detection. However, there are limitations on the availability of the training data due to data privacy and sensitivity concerns. To handle them, we proposed a novel zero-shot KD method on two-stage Faster RCNN object detection models. Through extensive experiments on several architectures and datasets, we showed the utility of our generated data as a transfer set in KD. We observe decent mAP even when a network is trained on our generated data without *Teacher* assistance. Our pseudo-dataset further leads to significant improvement in mAP when augmented with proxy data or few training samples, proving them as reliable estimates of original training data. However, our current method depends on the anchor information from the pretrained model to get an estimate about the range of object sizes in the original training dataset. As a future direction, we will reduce this dependency further to make them suitable for anchor-free detectors. Also, extending our work to black box setting, assuming no access to the pretrained model weights would be another interesting direction.

6 Acknowledgements

This work is supported by a Start-up Research Grant (SRG) from SERB, DST, India (Project file number: SRG/2019/001938). We would like to extend our gratitude to all the reviewers for their valuable suggestions. We would also like to thank Inder Khatri and Vignesh Nagarajan who helped in running few experiments.

References

[1] Zhaowei Cai, Quanfu Fan, Rogerio S Feris, and Nuno Vasconcelos. A unified multi-scale deep convolutional neural network for fast object detection. In *European conference on computer vision*, pages 354–370. Springer, 2016.

[2] Akshay Chawla, Hongxu Yin, Pavlo Molchanov, and Jose Alvarez. Data-free knowledge distillation for object detection. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 3289–3298, 2021.

[3] Guobin Chen, Wongun Choi, Xiang Yu, Tony Han, and Manmohan Chandraker. Learning efficient object detection models with knowledge distillation. In *Advances in Neural Information Processing Systems*, pages 742–751, 2017.

[4] Hanting Chen, Yunhe Wang, Chang Xu, Zhaohui Yang, Chuanjian Liu, Boxin Shi, Chunjing Xu, Chao Xu, and Qi Tian. Data-free learning of student networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3514–3522, 2019.

[5] Long Chen, H. Ai, Zijie Zhuang, and Chong Shang. Real-time multiple people tracking with deeply learned candidate selection and person re-identification. *2018 IEEE International Conference on Multimedia and Expo (ICME)*, pages 1–6, 2018.

[6] Xiaozhi Chen, Kaustav Kundu, Ziyu Zhang, Huimin Ma, Sanja Fidler, and Raquel Urtasun. Monocular 3d object detection for autonomous driving. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2147–2156, 2016.

[7] Mircea Cimpoi, Subhransu Maji, Iasonas Kokkinos, Sammy Mohamed, and Andrea Vedaldi. Describing textures in the wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3606–3613, 2014.

[8] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[9] Terrance DeVries and Graham W Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017.

[10] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results. http://www.pascal-network.org/challenges/VOC/voc2007/workshop/index.html.

[11] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[12] Georgios Georgakis, Arsalan Mousavian, Alexander C Berg, and Jana Kosecka. Synthesizing training data for object detection in indoor scenes. *arXiv preprint arXiv:1702.07836*, 2017.

[13] Song Han, Jeff Pool, John Tran, and William Dally. Learning both weights and connections for efficient neural network. In *Advances in neural information processing systems*, pages 1135–1143, 2015.

[14] Matan Haroush, Itay Hubara, Elad Hoffer, and Daniel Soudry. The knowledge within: Methods for data-free model compression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8494–8502, 2020.

[15] 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.

[16] Stefan Hinterstoisser, Vincent Lepetit, Paul Wohlhart, and Kurt Konolige. On pre-trained image features and synthetic images for deep learning. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 0–0, 2018.

[17] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

[18] E. Hoffer, Tal Ben-Nun, Itay Hubara, Niv Giladi, Torsten Hoefler, and Daniel Soudry. Augment your batch: better training with larger batches. *ArXiv*, abs/1901.09335, 2019.

[19] Hyeonseong Jeon, Siho Han, Sangwon Lee, and Simon S Woo. Compensating for the lack of extra training data by learning extra representation. In *Proceedings of the Asian Conference on Computer Vision*, 2020.

[20] Akisato Kimura, Zoubin Ghahramani, Koh Takeuchi, Tomoharu Iwata, and Naonori Ueda. Few-shot learning of neural networks from scratch by pseudo example optimization. In *British Machine Vision Conference (BMVC)*, 2018.

[21] Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Joan Puigcerver, Jessica Yung, Sylvain Gelly, and Neil Houlsby. Big transfer (bit): General visual representation learning. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part V 16*, pages 491–507. Springer, 2020.

[22] Raghuraman Krishnamoorthi. Quantizing deep convolutional networks for efficient inference: A whitepaper. *arXiv preprint arXiv:1806.08342*, 2018.

[23] Yuhang Li, Feng Zhu, Ruihao Gong, Mingzhu Shen, Xin Dong, Fengwei Yu, Shaoqing Lu, and Shi Gu. Mixmix: All you need for data-free compression are feature and data mixing, 2021.

[24] 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.

[25] Raphael Gontijo Lopes, Stefano Fenu, and Thad Starner. Data-free knowledge distillation for deep neural networks. In *LLD Workshop at Neural Information Processing Systems (NIPS)*, 2017.

[26] Jian-Hao Luo, Jianxin Wu, and Weiyao Lin. Thinet: A filter level pruning method for deep neural network compression. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[27] Jiayuan Mao, Tete Xiao, Yuning Jiang, and Zhimin Cao. What can help pedestrian detection? In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3127–3136, 2017.

[28] Paul Micaelli and Amos J Storkey. Zero-shot knowledge transfer via adversarial belief matching. In *Advances in Neural Information Processing Systems*, pages 9551–9561, 2019.

[29] 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.

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

[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*, pages 91–99, 2015.

[32] Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. Fitnets: Hints for thin deep nets. *arXiv preprint arXiv:1412.6550*, 2014.

[33] German Ros, Laura Sellart, Joanna Materzynska, David Vazquez, and Antonio M. Lopez. The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[34] 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.

[35] Tara N Sainath, Brian Kingsbury, Vikas Sindhwani, Ebru Arisoy, and Bhuvana Ramabhadran. Low-rank matrix factorization for deep neural network training with high-dimensional output targets. In *2013 IEEE international conference on acoustics, speech and signal processing*, pages 6655–6659. IEEE, 2013.

[36] Igor Ševo and Aleksej Avramović. Convolutional neural network based automatic object detection on aerial images. *IEEE geoscience and remote sensing letters*, 13(5): 740–744, 2016.

[37] Suraj Srinivas and R Venkatesh Babu. Data-free parameter pruning for deep neural networks. *arXiv preprint arXiv:1507.06149*, 2015.

[38] Shitao Tang, Litong Feng, Wenqi Shao, Zhanghui Kuang, Wei Zhang, and Yimin Chen. Learning efficient detector with semi-supervised adaptive distillation. *arXiv preprint arXiv:1901.00366*, 2019.

[39] 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.

[40] Josh Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 23–30. IEEE, 2017.

[41] Tao Wang, Li Yuan, Xiaopeng Zhang, and Jiashi Feng. Distilling object detectors with fine-grained feature imitation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4933–4942, 2019.

[42] Yan Wang, Wei-Lun Chao, Divyansh Garg, Bharath Hariharan, Mark Campbell, and Kilian Q Weinberger. Pseudo-lidar from visual depth estimation: Bridging the gap in 3d object detection for autonomous driving. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8445–8453, 2019.

[43] N. Wojke, A. Bewley, and Dietrich Paulus. Simple online and realtime tracking with a deep association metric. *2017 IEEE International Conference on Image Processing (ICIP)*, pages 3645–3649, 2017.

[44] Fan Yang, Heng Fan, Peng Chu, Erik Blasch, and Haibin Ling. Clustered object detection in aerial images. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 8311–8320, 2019.

[45] Jianwei Yang, Jiasen Lu, Dhruv Batra, and Devi Parikh. A faster pytorch implementation of faster r-cnn. *https://github.com/jwyang/faster-rcnn.pytorch*, 2017.

[46] 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.

[47] 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*, pages 8715–8724, 2020.

[48] Barret Zoph, Ekin D Cubuk, Golnaz Ghiasi, Tsung-Yi Lin, Jonathon Shlens, and Quoc V Le. Learning data augmentation strategies for object detection. *arXiv preprint arXiv:1906.11172*, 2019.