Coreset Selection for Object Detection

Hojun Lee[1] Suyoung Kim[1] Junhoo Lee[1] Jaeyoung Yoo[2] Nojun Kwak[1*]
[1]Seoul National University [2]NAVER WEBTOON AI
{hojun815,ksyo96,mrjunoo,nojunk}@snu.ac.kr, yoojy31@webtoonscorp.com

Abstract

*Coreset selection is a method for selecting a small, representative subset of an entire dataset. It has been primarily researched in image classification, assuming there is only one object per image. However, coreset selection for object detection is more challenging as an image can contain multiple objects. As a result, much research has yet to be done on this topic. Therefore, we introduce a new approach, **C**oreset **S**election for **O**bject **D**etection (CSOD). CSOD generates imagewise and classwise representative feature vectors for multiple objects of the same class within each image. Subsequently, we adopt submodular optimization for considering both representativeness and diversity and utilize the representative vectors in the submodular optimization process to select a subset. When we evaluated CSOD on the Pascal VOC dataset, CSOD outperformed random selection by +6.4%p in AP_{50} when selecting 200 images.*

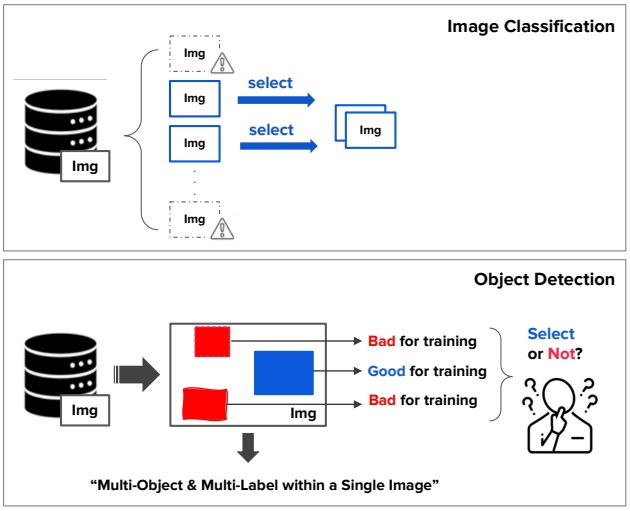

Figure 1. The difference in coreset selection between image classification and object detection.

1. Introduction

In today's data-driven era, managing the sheer volume and variety of data presents a crucial challenge, particularly in areas like computer vision and deep learning, which deal with a tremendous amount of data [4, 21, 35]. With the advent of technologies such as autonomous vehicles and smart surveillance systems, accurate and efficient recognition of such image data has become paramount. One key strategy in managing these massive datasets involves 'coreset selection,' a method aimed at identifying a smaller, representative subset of the original dataset. This subset is then used to streamline complex computations and enhance processing efficiency.

However, as illustrated in Figure 1, traditional coreset selection methods are unrealistic because they were developed under the naïve assumption of a single object per image, a condition that real-world images do not often meet [1, 5, 10]. Real-world images typically contain the natural variability, such as multiple objects of various categories, sizes, and locations.

This implies that we should develop methods that consider the natural variability. As Figure 1 illustrates, if we evaluate the suitability of an image on an object-by-object basis, considering one object as suitable for training does not necessarily imply that the others are also suitable. In other words, the decision of a core image should be based on all objects in an image. However, traditional methods, designed to solve only the single-image-single-object pairing, fail to consider this, and therefore struggle in realistic conditions.

In this paper, we address a significant limitation in the field of coreset selection which has traditionally operated under the assumption of a single object per image. We introduce a realistic approach tailored to the more complex, yet common, scenario where images inherently contain multiple objects. This shift from single-object to multi-object consideration is a central advancement of our work. CSOD not only recognizes the presence of numerous objects within each image but also tackles the compounded

*Corresponding author

uncertainties by considering objects' spatial information such as size and location. To validate our method, we implemented and conducted experiments in object detection, a representative task of situations where multiple objects may reside within a single image.

Our method, CSOD, is built upon a unique concept: the 'imagewise-classwise vector.' To select the most representative images among compounded uncertainties, we need a way to summarize the information of each image effectively. The imagewise-classwise vector serves this purpose by averaging the features of objects of the same class within an image. This comprehensive representation allows for informed decision-making when addressing the complexities of multi-object images.

We employ a greedy approach to select individual data points sequentially by class order, thereby constructing the coreset step by step. Although this method considers only one class at each selection step, it guarantees that the most pertinent selections for each class are made, enhancing the representativeness and diversity of the coreset. Furthermore, to ensure that the selected subset informatively represents the entire dataset, we introduce a mathematical tool known as a 'submodular function,' as delineated by Krause and Golovin [17]. The function aids in selecting the most informative subset based on the imagewise-classwise average features for each category.

Our empirical evaluations, particularly in scenarios involving the detection of multiple objects, demonstrate the effectiveness of CSOD. For instance, when selecting 200 images from the Pascal VOC dataset [8], our method achieved an impressive improvement of +6.4% point in AP_{50} compared to random selection. Moreover, we also evaluated it on the BDD100k [32] and MS COCO2017 [18] datasets and confirmed that our method outperforms random selection. These significant achievements emphasize the efficacy and innovativeness of CSOD in addressing the challenges of coreset selection in multi-object image data.

In summary, CSOD represents a pivotal extension of existing coreset selection frameworks to encompass scenarios with multiple objects per image. This approach addresses a gap that traditional methods, which assumed only one object per image, did not cover. While our focus is on multi-objects of the same class, we acknowledge the potential for future expansions of our method to accommodate images featuring various categories of objects. With our unique problem recognition and solution, we aim to shift the paradigm of coreset selection towards more realistic scenarios, specifically image datasets containing multiple objects. This research transcends mere technical advancement, marking a pivotal shift in processing complex real-world datasets and opening new horizons for coreset selection, thereby addressing a significant challenge of the big data era.

2. Background and Prior works

2.1. Coreset selection

Welling [30] introduced the concept of herding for iterative data point selection near class centers. Wei et al. [29] applied the submodular function to the Naïve Bayes and Nearest Neighbor classifier. We also adopt this function, so we provide further explanation in Section 2.3. Braverman et al. [1], Huang et al. [12] modified statistical clustering algorithms like k-median and k-means to identify data points that effectively represent the dataset. Coleman et al. [5] utilized uncertainties measured by entropy or confidence. Huang et al. [13] theoretically explained the upper and lower bounds on the coreset size for k-median clustering in low-dimensional spaces. However, most previous researches focused on image classification, and to the best of our knowledge, our work is the first research to design coreset selection specifically for object detection.

2.2. Dataset Distillation

Coreset Selection and Dataset Distillation are crucial in enhancing model training efficiency, with the former selecting informative data points and the latter synthesizing data to distill the dataset's information. Despite their different approaches—selection versus synthesis—both methods aim to encapsulate data. Current Dataset Distillation research [3, 7, 20, 27, 34], primarily focused on image classification, presents unexplored potential in object detection. Advancements in Coreset Selection for object detection may have a significant influence on Dataset Distillation strategies for object detection.

2.3. Submodular function

A set function $f : 2^{\mathcal{V}} \rightarrow \mathbb{R}$ is considered submodular if, for any subsets $\mathcal{A}$ and $\mathcal{B}$ of $\mathcal{V}$ where $\mathcal{A} \subseteq \mathcal{B}$ and x is an element not in $\mathcal{B}$, the following inequality holds:

$$f(\mathcal{A} \cup \{x\}) - f(\mathcal{A}) \geq f(\mathcal{B} \cup \{x\}) - f(\mathcal{B}) \qquad (1)$$

Here, $\Delta(x|\mathcal{A}) := f(\mathcal{A} \cup \{x\}) - f(\mathcal{A})$ represents the benefit of adding x to the set $\mathcal{A}$. In simple terms, this inequality means that adding x to $\mathcal{B}$ provides less additional benefit than adding x to $\mathcal{A}$. This is because $\mathcal{B}$ already contains some of the information that x can offer to $\mathcal{A}$. Therefore, we can use submodularity to find a subset that maximizes the benefit of adding each element.

However, in general, selecting a finite subset $\mathcal{S}$ with the maximum benefit is a computationally challenging problem (NP-hard) [17]. To address this, we employ a greedy algorithm that starts with an empty set and adds one element at a time. Specifically, $\mathcal{S}_i$ is updated as $\mathcal{S}_{i-1} \cup \operatorname{argmax}_x \Delta(x|\mathcal{S}_{i-1})$. For more information, please refer to Krause and Golovin [17].

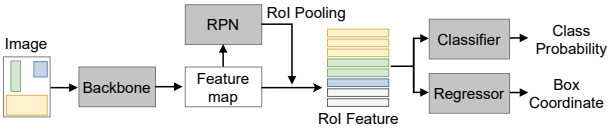

Figure 2. The forward process during the training phase of Faster R-CNN. The RoI features include both foreground and background regions at the forward process.

2.4. Faster R-CNN

Various object detectors exist, including Faster R-CNN [23], SSD [9], YOLO [22], and DETR [2]. We chose Faster R-CNN as our base model. This choice was motivated by its widespread adoption not only in supervised detection but also in various research areas such as few-shot detection [28], continual learning [26], and semi-supervised object detection [15].

Faster R-CNN operates as a two-stage detector. As illustrated in Figure 2, the first stage employs Region Proposal Network (RPN) to generate class-agnostic object candidate regions in the image, followed by pooling these regions to obtain Region of Interest (RoI) feature vectors. In the second stage, the model utilizes these RoI feature vectors for final class prediction and bounding box regression. Our research uses these RoI feature vectors for coreset selection.

2.5. Active Learning for Object Detection

Active learning is concerned with selecting which unlabeled data to annotate and is thus related to coreset selection. In the context of active learning for object detection, Yuan et al. [33] proposed a method based on uncertainty that utilizes confidence scores on unlabeled data. Kothawade et al. [16] aimed to address the low performance issue in rare classes when conducting active learning. The method extracted features of rare classes from labeled data and aimed to maximize the information of rare classes by submodular function and computing the cosine similarity between these labeled features and the features of unlabeled data.

3. Method

3.1. Problem Setup

We have an entire training dataset $\mathcal{T} = \{x_i, y_i\}_{i=1}^{D}$. Here, $x_i \in \mathcal{X}$ is an input image, and $y_i \in \mathcal{Y}$ is a ground truth. Because these data are for object detection, $y_i = \{c_{i,j}, b_{i,j}\}_{j=1}^{G_i}$ contains variable numbers of annotations depending on the image. In the G_i annotations, $c_{i,j}$ is a class index, and $b_{i,j} = \{b_{i,j}^{left}, b_{i,j}^{top}, b_{i,j}^{right}, b_{i,j}^{bottom}\}$ denotes the coordinates of the j-th bounding box. Coreset selection aims to choose a labeled subset $\mathcal{S} \subset \mathcal{T}$ that best approximates the performance of a model trained on the entire labeled dataset, $\mathcal{T}$.

In our approach, we prioritize the number of images over the number of annotations. This is because annotations typically consist of relatively few strings, and what primarily affects training time and data storage is the number of images rather than the number of annotations.

Algorithm 1 CSOD Pseudocode

Require: Training Data $\mathcal{T} = \{(x_i, y_i)\}_{i=1}^{D}$ with C classes, where $y_i = \{(c_{i,j}, b_{i,j})\}_{j=1}^{G_i}$ and G_i is the number of ground truth objects in the i-th image. Trained backbone f_θ. RoI pooler g, Global Average Pooling function h.
Ensure: Selected subset $\mathcal{S}$ with size N

Initialize $\mathcal{S}_c = \emptyset$, $\mathcal{P}_c = \emptyset$, $\mathcal{Q}_c = \emptyset$ for all $c \in \{1, \dots, C\}$
Stage 1: Preparing Imagewise-Classwise Features
for $i = 1$ to D **do**
  RoI features $\mathcal{R}_i = \{\boldsymbol{r}_{i,j}\}_{j=1}^{G_i} = h(g(f_\theta(x_i), y_i))$ ▷ Sec. 3.3
  for all classes c present in y_i **do**
    $\boldsymbol{p}_{i,c} = \frac{1}{|\{j|c_{i,j}=c\}|} \sum_{\{j|c_{i,j}=c\}} \boldsymbol{r}_{i,j}$ ▷ Sec. 3.4
    Update $\mathcal{P}_c = \mathcal{P}_c \cup \{\boldsymbol{p}_{i,c}\}$
  end for
end for
Stage 2: Subset Selection
while $|\mathcal{S}| < N$ **do**
  for $c = 1$ to C **do** ▷ Sec. 3.5
    Compute scores $s_{i,c} = \text{score}(\boldsymbol{p}_{i,c}, \mathcal{Q}_c), \forall i$ ▷ Eq.4
    Select the image $i^* = \arg\max_i s_{i,c}$
    Update $\mathcal{S}_c = \mathcal{S}_c \cup \{(x_{i^*}, y_{i^*})\}$
    Update $\mathcal{Q}_{c'} = \mathcal{Q}_{c'} \cup \{\boldsymbol{p}_{i^*,c'}\}, \forall c' \in y_{i^*}$
    Remove $\boldsymbol{p}_{i^*,c'}$ from $\mathcal{P}_{c'}, \forall c' \in y_{i^*}$
  end for
  Update $\mathcal{S} = \bigcup_{c \in \{1,\dots,C\}} \mathcal{S}_c$
end while

3.2. Overview

CSOD picks out the most useful images by looking at one object category at a time. Below are the steps of our CSOD:
Preparing Object Features: We extract RoI feature vectors from the ground truth of the entire training set (Sec. 3.3). Then, we average the RoI features of the same class within one image (Sec. 3.4).
Choosing the Best Images: We utilize the averaged RoI feature vectors to greedily select images one by one for each class in a rotating manner (Sec. 3.5). In doing so, the submodular optimization technique is introduced to ensure that the selection process considers both representativeness and diversity (Eq. 4). When we pick an image, we do not just use one object in it for training; we use all the objects it contains.

Algorithm 1 provides the pseudocode, while Figure S1 in the supplementary material aids understanding.

3.3. Ground Truth RoI Feature Extraction

With Faster R-CNN, we extract RoI feature vectors from training images by the ground truth (not from the RPN output). If the i-th training image contains G_i ground truth objects, then we have G_i RoI feature vectors, $\mathcal{R}_i$, as follows:

$$\mathcal{R}_i = \{\boldsymbol{r}_{i,j}\}_{j=1}^{G_i} = h(g(f_\theta(x_i), y_i)) \quad (2)$$

where x_i is an input image, y_i is a ground truth, f_θ is the backbone trained by the entire data, g is the RoI pooler, h is global average pooling and $\boldsymbol{r}_{i,j}$ is the j-th RoI feature vector of the i-th image.

3.4. Imagewise and Classwise Average

Once we have extracted all the RoI feature vectors for each image, we have a choice to make: For coreset selection, should we average the RoI feature vectors of the same class within a single image to create a single prototype vector representing that class for the image, or should we use these RoI feature vectors directly?

As mentioned in Section 1, we chose the averaging approach. If $\mathcal{R}_i = \{\boldsymbol{r}_{i,j}\}_{j=1}^{G_i}$ represents the RoI feature vectors for the i-th data with G_i ground truth objects, then the average RoI feature vector for class c in the i-th data, denoted as $\boldsymbol{p}_{i,c}$, is calculated as follows:

$$\boldsymbol{p}_{i,c} = \frac{1}{|\{j|c_{i,j}=c\}|} \sum_{\{j|c_{i,j}=c\}} \boldsymbol{r}_{i,j} \tag{3}$$

3.5. Greedy Selection

After obtaining averaged RoI feature vectors, our selection process follows a greedy approach, iteratively choosing one data point from each class at a time. To facilitate this, we compute a similarity-based score for each RoI feature vector. This scoring mechanism based on the submodular function assigns higher scores to RoI feature vectors that are similar to others within the same class and lower scores to those similar to RoI feature vectors that have already been selected. This strategy enables us to take into account previously selected data points when making new selections.

The score function computes the score s for the i-th data point within class c as follows:

$$s_{i,c} = \lambda \cdot \sum_j cos(\boldsymbol{p}_{i,c}, \boldsymbol{p}_{j,c}) - \sum_j cos(\boldsymbol{p}_{i,c}, \boldsymbol{q}_{j,c}) \tag{4}$$

The term "cos" represents the cosine similarity, $\boldsymbol{p}_i$ represents the averaged RoI feature vectors that have not been selected yet, and $\boldsymbol{q}_i$ denotes the previously selected RoI feature vectors. The hyperparameter λ is introduced to balance the contributions within the scoring function, in which the former term aims to select the most representative one from among those that have not been selected, while the latter term aims to select something different from what has already been selected before. The experiment related to λ can be found in Section 4.4.2.

CSOD selects data corresponding to the maximum value in Eq. (4) for each class. If a chosen data point includes multiple classes, the features of these classes are considered part of the previously selected q_i. This method systematically cycles through each class, ensuring unique selections, until it reaches the targeted number of choices.

4. Experiments

In this section, we empirically validate the effectiveness of CSOD through experiments. First, we will show that CSOD outperforms various random selections and other coreset selection methods originally designed for image classification. We will then investigate the tendency associated with the number of selected images and the hyperparameter λ of Eq. (4). Additionally, we will compare performance when averaging RoI features of a class within an image versus using individual features as they are. Furthermore, we will extend our analysis to evaluate the performance of different datasets and various network architectures.

4.1. Implementation details

We conducted experiments on Pascal VOC 2007+2012 [8], using the trainval set for selection and training, and the VOC07 test set for evaluation. The metric is Average Precision at IoU 0.5 (AP_{50}). For ablation and analysis, we chose 200 images from 20 classes, training for 1000 iterations. We averaged performance over 20 runs due to the limited number of images. We used Faster R-CNN-C4 [23] with ResNet50 [11]. For the selection phase, we used the model weight trained on VOC07+12, provided by detectron2 [31]. After selection, a new model was trained on the chosen subset, with a backbone pre-trained on ImageNet [6]. For further details, see Section S1 in the supplementary material.

4.2. Comparision with Other Selections

Comparison Targets. Table 1 shows the comparison list. Random refers to the method where one image per class is randomly selected in turn. There can be multiple classes in a single image, and when selecting in turn by class, the images were chosen without duplication to make the target number of images. Additional experiments regarding Random Selection are conducted in Section 4.3.

Coreset selection methods for image classification, such as Herding [30], k-Center Greedy [24], and Submodular function [14], were also compared. The CSOD's network weight was employed, and the backbone feature was globally average pooled to utilize that feature vector for selection. Similar to random selection, images were evenly chosen from each class.

Result. As seen in Table 1, our method consistently shows the highest results. Random selection shows higher performance than some existing methods, which implies that these methods are designed only for the image classification task and yield lower results in object detection. The submodular function showed some effectiveness when the number of data was low. However, as mentioned in Section 1 and Figure 1, it is a modeling that fundamentally cannot consider multiple objects of various sizes and locations. Therefore, it not only showed lower results than random when selecting over 500 images but also significantly lower results than

Selection Method	20	100	200	500	1000
Random (Uniform)	9.8 ± 2.2	27.9 ± 1.6	37.9 ± 1.1	50.7 ± 1.0	58.4 ± 0.6
Herding [30]	4.1 ± 0.7	17.7 ± 1.2	26.0 ± 0.8	37.8 ± 0.7	46.4 ± 0.7
k-Center Greedy [24]	10.0 ± 1.3	21.8 ± 2.1	32.3 ± 0.9	47.4 ± 1.0	55.9 ± 0.3
Submodular Function [14]	12.9 ± 0.9	30.5 ± 1.3	38.6 ± 0.9	48.8 ± 0.7	55.8 ± 0.3
CSOD (Ours)	$\mathbf{14.5 \pm 1.6}$	$\mathbf{34.4 \pm 1.0}$	$\mathbf{44.3 \pm 0.7}$	$\mathbf{54.1 \pm 0.7}$	$\mathbf{60.6 \pm 0.4}$

Table 1. Comparison with random and coreset selection for image classification, reporting AP_{50} on the VOC07 test data. We ran all experiments 20 times, with $\pm$ indicating standard deviation. Note that the standard deviation is to show that the performance gap is clear. Herding, Submodular, and CSOD select subsets deterministically, with only network weight initialization affected by random seeds.

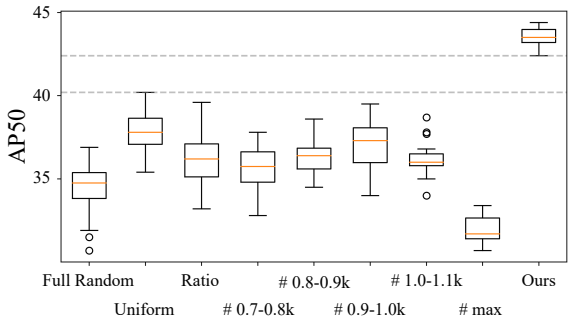

Figure 3. Comparison with various selection methods. '#' denotes the number of objects in the selected data.

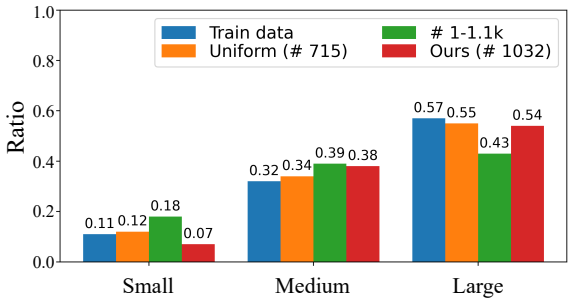

Figure 4. Ratio of box sizes. We followed the size criteria provided by VOC. '#' denotes the number of objects in the selected data.

CSOD. Based on these results, future experiments for comparison will be conducted with Random selection.

4.3. Comparison with Random Selections

Figure 3 shows that our approach consistently outperforms other selection methods when selecting 200 images. Notably, in this comparison, "# max" and "Ours" are the only methods without randomness, while the rest incorporate some degree of randomness. Therefore, we did not specifically address the performance variance of each selection method. Our method was implemented with a fixed set and without randomness, leading to reduced performance variance only comes from the training process. This experiment's significance lies in the clearly higher performance of ours compared with those of other methods.

We categorized random selection into several methods. "Full random" selects 200 images randomly, but repeats the process if any classes have no objects in those 200 images. "Uniform" and "Ratio" involve sampling images one by one for each class until 200 images are selected (sampling without replacement). In these cases, images selected from one class are excluded from selection in other classes, as an image can contain objects in multiple classes. "Uniform" distributes images evenly with 10 per class, while "Ratio" selects images based on the proportion of images per class.

The CSOD result consists of 1,032 annotations. Therefore, we also experimented with random selection while controlling for the number of annotations. "# 700-1100" limits annotations to this range using the Uniform method. "# max" also follows Uniform but selects images based on the annotation count in descending order rather than selecting them randomly.

4.3.1 The performance and object size ratio.

Figure 4 shows the relationship between box size, object count, and performance. We conducted two comparisons. First, we compared Uniform and CSOD. We observed that the ratio of Uniform was closer to that of the entire dataset in terms of KL divergence, while Ours had more annotations. Second, we compared CSOD and '# 1-1.1k'. Both methods had similar object counts, but CSOD's box size ratio was closer to that of the training data. While we cannot definitively assert causality, it appears that a well-represented subset with an equal number of images has a correlation with both box size and object count.

4.4. Analysis of the number of images and the hyperparameter λ

4.4.1 The number of selected images

Figure 5 shows performance with the number of selected images. Since selecting 20 images indicates only one image per class is selected, λ is meaningless. For other cases (100, 200, 500, and 1000), we set λ as (0.0125, 0.04375, 0.0625,

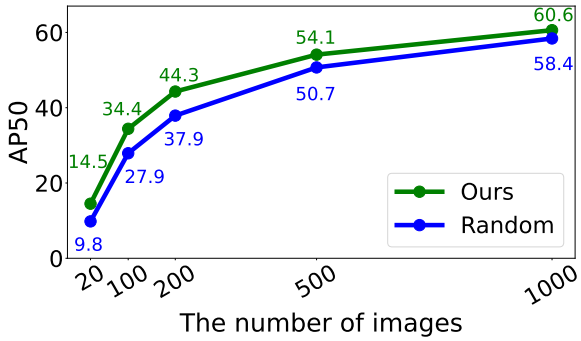

Figure 5. Performance according to the selected image counts.

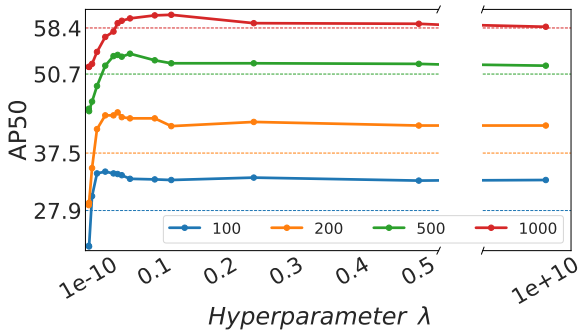

Figure 6. AP_{50} and λ. Dashed lines represent random selection performance.

and 0.025), respectively. Compared to the random selection, we observe that as the number of selected images increases, the performance gap naturally decreases, but it consistently remains at a high level.

4.4.2 Balance hyperparameter λ

Figure 6 illustrates the relationship between performance and λ in Eq. (4). A high λ value (1e+10) means selecting images based solely on cosine similarity, prioritizing representative images. In contrast, a small λ value (1e-10) means selecting an image per class with the highest cosine similarity first and then selecting images that are as dissimilar as possible from those already selected. In other words, it emphasizes diversity from a cosine similarity viewpoint.

The observations can be made: Firstly, our approach outperforms random selection when λ is above a certain threshold. Secondly, it is better to consider both representativeness and diversity by appropriately tuning λ rather than simply selecting images purely based on the order of cosine similarity (1e+10). Lastly, the optimal λ value varies depending on the number of images to be selected, as the greedy selection process (Section 3.5) progressively increases the number of selected images. Please refer to Table S1 in the supplementary material for the AP_{50} values corresponding to the λ values.

		The number of changes in the image list				
		0	18	32	45	113
Objectwise	λ	1e+10	0.125	0.075	0.051	0.015
	AP_{50}	40.4	40.3	40.7	41.4	39.0
Imagewise (Ours)	λ	1e+10	0.100	0.050	0.038	0.013
	AP_{50}	42.1	43.3	43.5	43.8	41.5

Table 2. Comparison with two methods, "Imagewise" (averaging RoI vectors) and "Objectwise" (not averaging), for selecting 200 images. The number of changes in the image list is based on $\lambda = 1e + 10$ as the reference point (The smaller λ, the severer the change). λ is rounded to the fourth decimal place.

	20	40	60	80	100	200
Objectwise	13.0	20.6	25.8	29.1	31.5	40.4
Imagewise (Ours)	14.2	23.1	27.3	30.1	32.9	42.1

Table 3. λ is 1e+10 in all cases, meaning that we selected based solely on cosine similarity ranking.

4.5. Effectiveness of Averaging RoI feature vectors

4.5.1 Performance comparison

Table 2 compares performance between averaging the RoI feature vectors of the same class (Imagewise) or not (Objectwise). These two cases have different balance strengths for λ, as Imagewise averages within the same class, resulting in significantly fewer RoI feature vectors. Therefore, we compared the extent to which the image list changes, using 1e+10 as the reference point. Remarkably, we observed consistent high performance regardless of λ values. However, even Objectwise outperformed random selection, achieving a result higher than 37.5 of random selection.

4.5.2 Representativeness: Objectwise vs. Imagewise feature vector

Table 3 compares Objectwise and Imagewise based on the number of selected images when λ=1e+10. This experiment highlights that even if the cosine similarity of a single object within an image is exceptionally high, that image may not effectively represent the overall distribution of the data. For example, the case where only one image per class is selected (20 in the table) indicates how well a single image represents the corresponding class. The table shows the superiority of our imagewise selection over objectwise selection.

4.5.3 Visulization of objectwise selection

Figure 7 illustrates a limitation of the objectwise approach, where the selected image may not effectively represent the

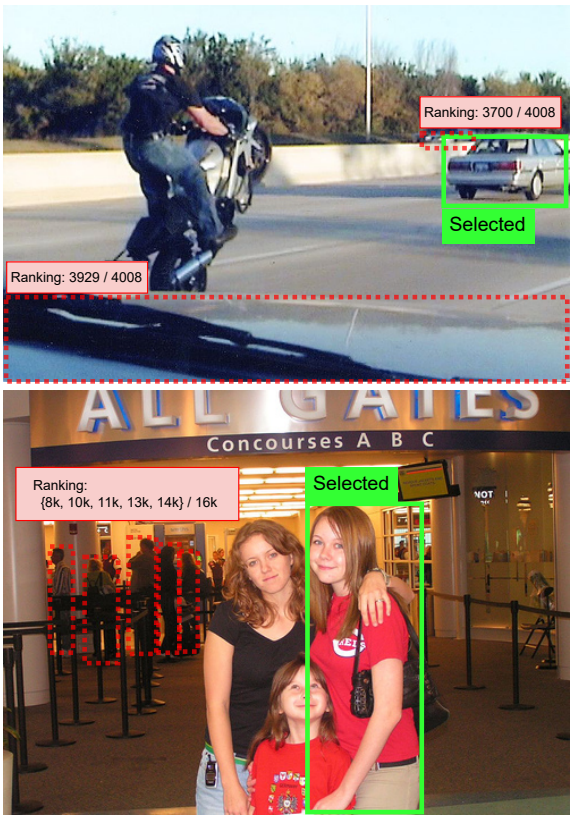

Figure 7. Examples of the Objectwise selection. Top: 'car' class. Bottom: 'person' class. The red dotted boxes indicate objects with low rankings in Eq. (4).

entire dataset. Even if an image is selected because it contains an object with high cosine similarity, it does not guarantee that other objects within the same image will have similarly high cosine similarities. In other words, the cosine similarity of one object in an image with all the other objects in the entire dataset may not accurately represent the cosine similarities of all objects in that image.

4.5.4 Why imagewise (averaging) selection over objectwise selection?

Let us compare object counts and size ratios in Section 4.3.1. In our Imagewise approach, there are 1,032 objects in our 200 selected images, which is higher than 806 in the Objectwise approach. Additionally, when considering the size ratios (small, medium, large), the Imagewise approach results in (7.3%, 38.2%, 54.5%), which is closer to the large object ratio in the entire train data, (10.5%, 32.0%, 57.5%), compared to the Objectwise, (10.2%, 37.2%, 52.6%). When we calculated the KL divergence between the distributions of the selected images and the entire training data, we found that Objectwise had a KL divergence of 0.006, lower than Imagewise's 0.013.

Object count	1	2-4	5 or more
Large	0.866	0.931	0.939
Medium	0.862	0.915	0.931
Small	0.798	0.818	0.831

Table 4. Cosine similarity between the entire average features and the average feature of each image by size in the 'person' class.

This suggests that the number of annotations played a more significant role in the performance than the size ratio in the case of Imagewise and Objectwise. Despite the higher KL divergence for Ours, there were substantial differences in the number of annotations for each size. Ours had counts of (75, 395, 562) for each size, whereas Objectwise had counts of (82, 300, 424).

We formulated a hypothesis that "As the number of objects within an image increases and their sizes are larger, the cosine similarity between the class's entire average RoI vector (class prototype) and the image's average RoI vector (image prototype) for that class will be higher."

To validate this hypothesis, we conducted the experiment presented in Table 4. Initially, we averaged all RoI vectors for the 'person' class (class prototype). Then, we made an averaged RoI vector by size within each image (imagewise-sizewise prototype). We subsequently computed the cosine similarity between the class prototype and the imagewise-sizewise prototypes. The results confirmed that the Imagewise approach leads to a higher selection of larger objects, resembling the entire dataset.

4.6. Evaluation on the BDD100k dataset

BDD100k [32] is a significant dataset for autonomous driving, consisting of 100k images, 1.8M annotations, and 10 different classes. Following the official practice, we split the dataset into 70k for training with 1.3M annotations. Table 5 shows the results on the validation data, illustrating that our method consistently achieves higher AP_{50}, AP_{75}, and AP compared to random selection. Notably, similar performance improvements were observed in our experiments with the VOC dataset. However, BDD100k, explicitly designed for real-world autonomous driving applications, offers a more challenging and realistic benchmark. The fact that our method excels even in the challenging environment of BDD100k further demonstrates its effectiveness and practicality. For implementation details, kindly refer to Section S2 in the supplementary material.

4.7. Evaluation on the COCO dataset

The COCO2017 dataset [18] encompasses 80 classes, partitioned into 118K images for the training set and 5K for the validation set. Experiments were conducted with subsets of 400 and 800 images selected from the training data. Table 6

	num img	AP_{50}	AP_{75}	AP
200	Random	25.8	9.7	12.0
	Ours	29.0	10.8	13.5
	Δ	+3.2	+1.1	+1.5
500	Random	32.2	13.4	15.8
	Ours	35.1	14.9	17.5
	Δ	+2.9	+1.5	+1.7
1000	Random	37.1	16.2	18.6
	Ours	39.4	17.8	20.1
	Δ	+2.3	+1.6	+1.5
2000	Random	42.1	19.7	22.0
	Ours	43.7	21.0	23.2
	Δ	+1.6	+1.3	+1.2

Table 5. BDD100k result

	num img	AP_{50}	AP_{75}	AP
400	Random	15.1	4.9	6.6
	Ours	16.7	5.8	7.5
	Δ	+1.6	+0.9	+0.9
800	Random	19.4	7.5	9.1
	Ours	20.1	8.2	9.6
	Δ	+0.7	+0.7	+0.5

Table 6. COCO2017 result

shows the outcomes for the validation set. Despite the substantially larger size of the dataset compared to VOC, it was observed that our method was effective. For reproducibility please refer to Section S3 in the supplementary material.

The BDD dataset has more annotations despite having fewer images compared to COCO. Interestingly, the performance improvement margin on COCO was smaller than on BDD. This observation raises several points for consideration. The BDD dataset, focused on autonomous driving, predominantly encompasses outdoor scenes with less diversity, such as perspective, compared to COCO. Conversely, COCO spans a wider spectrum, including both indoor and outdoor scenes and a larger variety of classes. This diversity potentially renders the accurate representation of the entire data distribution with a subset of images more challenging. This observation not only clarifies our current results but also highlights this as a key area for future study.

4.8. Cross-architecture evaluation

Table 7 presents an experiment in which we assessed whether the 500 images selected using Faster R-CNN remained effective for different networks, namely RetinaNet [19] and FCOS [25]. We were able to confirm the effectiveness of images selected with Faster R-CNN for other networks as well. Unlike Faster R-CNN, these two networks often encountered training issues due to loss explosion when following their respective default hyperparameters. Therefore, we adjusted the hyperparameters, such as the learning rate and gradient clipping, but it is important to note that the hyperparameters for random selection and our method remained consistent. Please refer to Section S4 in the supplementary material for reproducibility.

	RetinaNet	FCOS
random	54.5	47.9
ours	58.3	53.1
Δ	+3.8	+5.2

Table 7. Cross architecture experiment. We trained the models on 500 VOC images and reported AP_{50}.

5. Discussion

Conclusion. We have proposed a Coreset selection method for Object Detection tasks, addressing the unique challenges presented by multi-object and multi-label scenarios. This stands in contrast to traditional image classification approaches. Our approach considers both representativeness and diversity while taking into account the difficulties we have outlined in Section 1 and illustrated in Figure 1. Through experiments, we have demonstrated the effectiveness of our method, and its applicability to various architectures. We hope this research will further develop and find applications in diverse areas, such as dataset distillation.

Limitation. While our research leveraged RoI features from ground truth boxes and achieved promising results, it is important to note certain limitations. Firstly, we did not explicitly incorporate background features, which could provide additional context and potentially enhance coreset selection in object detection. Future research could explore the explicit utilization of background features. Our approach, which selects greedily on a class-by-class basis, can take into account the RoI features of the current class even when they were selected during the turn of other classes. However, our method does not simultaneously incorporate the features of other classes within the same image. Further research could explore ways to capture interactions between different classes more effectively within a single image.

Future work. Since CSOD considers localization, there may be aspects that can be applied to other tasks related to localization, such as 3D object detection. Furthermore, while dataset distillation has predominantly been studied in the context of image classification, it could also become a subject of research in the field of object detection datasets.

Acknowledgements. This work was supported by NRF grant (2021R1A2C3006659) and IITP grants (2022-0-00953, 2021-0-01343), all funded by MSIT of the Korean Government.

References

[1] Vladimir Braverman, Shaofeng H-C Jiang, Robert Krauthgamer, and Xuan Wu. Coresets for ordered weighted clustering. In *International Conference on Machine Learning*, pages 744–753. PMLR, 2019. 1, 2

[2] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European conference on computer vision*, pages 213–229. Springer, 2020. 3

[3] George Cazenavette, Tongzhou Wang, Antonio Torralba, Alexei A Efros, and Jun-Yan Zhu. Dataset distillation by matching training trajectories. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4750–4759, 2022. 2

[4] Bowen Cheng, Ishan Misra, Alexander G. Schwing, Alexander Kirillov, and Rohit Girdhar. Masked-attention mask transformer for universal image segmentation. 2022. 1

[5] Cody Coleman, Christopher Yeh, Stephen Mussmann, Baharan Mirzasoleiman, Peter Bailis, Percy Liang, Jure Leskovec, and Matei Zaharia. Selection via proxy: Efficient data selection for deep learning. *arXiv preprint arXiv:1906.11829*, 2019. 1, 2

[6] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR*, 2009. 4

[7] Tian Dong, Bo Zhao, and Lingjuan Lyu. Privacy for free: How does dataset condensation help privacy? In *International Conference on Machine Learning*, pages 5378–5396. PMLR, 2022. 2

[8] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes challenge 2007 (voc2007) results. 2007. 2, 4

[9] Cheng-Yang Fu, Wei Liu, Ananth Ranga, Ambrish Tyagi, and Alexander C Berg. Dssd: Deconvolutional single shot detector. *arXiv preprint arXiv:1701.06659*, 2017. 3

[10] Chengcheng Guo, Bo Zhao, and Yanbing Bai. Deepcore: A comprehensive library for coreset selection in deep learning. In *International Conference on Database and Expert Systems Applications*, pages 181–195. Springer, 2022. 1

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016. 4

[12] Lingxiao Huang, Shaofeng Jiang, and Nisheeth Vishnoi. Coresets for clustering with fairness constraints. *Advances in Neural Information Processing Systems*, 32, 2019. 2

[13] Lingxiao Huang, Ruiyuan Huang, Zengfeng Huang, and Xuan Wu. On coresets for clustering in small dimensional euclidean spaces. *arXiv preprint arXiv:2302.13737*, 2023. 2

[14] Rishabh Iyer, Ninad Khargoankar, Jeff Bilmes, and Himanshu Asanani. Submodular combinatorial information measures with applications in machine learning. In *Algorithmic Learning Theory*, pages 722–754. PMLR, 2021. 4, 5

[15] Jisoo Jeong, Seungeui Lee, Jeesoo Kim, and Nojun Kwak. Consistency-based semi-supervised learning for object detection. *Advances in neural information processing systems*, 32, 2019. 3

[16] Suraj Kothawade, Saikat Ghosh, Sumit Shekhar, Yu Xiang, and Rishabh Iyer. Talisman: targeted active learning for object detection with rare classes and slices using submodular mutual information. In *European Conference on Computer Vision*, pages 1–16. Springer, 2022. 3

[17] Andreas Krause and Daniel Golovin. Submodular function maximization. *Tractability*, 3(71-104):3, 2014. 2

[18] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014. 2, 7

[19] 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. 8

[20] Timothy Nguyen, Roman Novak, Lechao Xiao, and Jaehoon Lee. Dataset distillation with infinitely wide convolutional networks. *Advances in Neural Information Processing Systems*, 34:5186–5198, 2021. 2

[21] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021. 1

[22] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018. 3

[23] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NIPS*, pages 91–99, 2015. 3, 4

[24] Ozan Sener and Silvio Savarese. Active learning for convolutional neural networks: A core-set approach. *arXiv preprint arXiv:1708.00489*, 2017. 4, 5

[25] 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

[26] Jianren Wang, Xin Wang, Yue Shang-Guan, and Abhinav Gupta. Wanderlust: Online continual object detection in the real world. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10829–10838, 2021. 3

[27] Tongzhou Wang, Jun-Yan Zhu, Antonio Torralba, and Alexei A Efros. Dataset distillation. *arXiv preprint arXiv:1811.10959*, 2018. 2

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

[29] Kai Wei, Rishabh Iyer, and Jeff Bilmes. Submodularity in data subset selection and active learning. In *International conference on machine learning*, pages 1954–1963. PMLR, 2015. 2

[30] Max Welling. Herding dynamical weights to learn. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 1121–1128, 2009. 2, 4, 5

[31] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2. https://github.com/facebookresearch/detectron2, 2019. 4

[32] Fisher Yu, Haofeng Chen, Xin Wang, Wenqi Xian, Yingying Chen, Fangchen Liu, Vashisht Madhavan, and Trevor Darrell. Bdd100k: A diverse driving dataset for heterogeneous multitask learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2636–2645, 2020. 2, 7

[33] Tianning Yuan, Fang Wan, Mengying Fu, Jianzhuang Liu, Songcen Xu, Xiangyang Ji, and Qixiang Ye. Multiple instance active learning for object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5330–5339, 2021. 3

[34] Yongchao Zhou, Ehsan Nezhadarya, and Jimmy Ba. Dataset distillation using neural feature regression. *Advances in Neural Information Processing Systems*, 35:9813–9827, 2022. 2

[35] Zhuofan Zong, Guanglu Song, and Yu Liu. Detrs with collaborative hybrid assignments training, 2022. 1