Influence Selection for Active Learning: Supplementary Material

In this supplementary material, we provide additional details which we could not include in the main paper due to space constraints. The material is composed as follows:

1. The derivation of the influence of untrained samples.
2. The implementation details of s_{test} calculation.
3. The time complexity analysis.
4. The implementation details of comparing methods.
5. Additional experiments on CIFAR10 dataset and COCO dataset.
6. The tables in which we report the average performance for each plot.

1. The Derivation of the Influence of Untrained Samples

Newton Step and Quadratic Approximation. Assuming that we have labeled dataset L_i and loss function $\mathcal{L}(\theta) = \frac{1}{n}\sum_{z\in L_i} l(z,\theta)$. After training a model on L_i, we have the model parameters $\hat{\theta} \in \Theta$, where $\hat{\theta} = \arg\min_{\theta\in\Theta} \frac{1}{n}\sum_{z\in L_i} l(z,\theta)$. Our purpose is to estimate the parameters of model which is trained on L_i and the new added sample $z^{'}$. The new loss function is $\mathcal{L}_{z'}(\theta) = \frac{1}{n+1}\sum_{z'\cup L_i} l(z,\theta)$, giving new trained model parameter $\hat{\theta}_{z'} = \arg\min_{\theta\in\Theta} \frac{1}{n+1}\sum_{z'\cup L_i} l(z,\theta)$.

Considering the quadratic approximation of the $\mathcal{L}_{z'}(\hat{\theta}_{z'})$

$$\begin{aligned}\mathcal{L}_{z'}(\hat{\theta}_{z'}) = \mathcal{L}_{z'}(\hat{\theta}) + (\hat{\theta}_{z'} - \hat{\theta})^T\nabla_\theta\mathcal{L}_{z'}(\hat{\theta}) + \\ \frac{1}{2}(\hat{\theta}_{z'} - \hat{\theta})^T\nabla^2_\theta\mathcal{L}_{z'}(\hat{\theta})(\hat{\theta}_{z'} - \hat{\theta})\end{aligned} \tag{1}$$

If the $H^{-1}_{\hat{\theta}}$ is positive definite, the quadratic approximation is minimized at

$$\begin{aligned}\hat{\theta}_{z'} - \hat{\theta} &= -\frac{\nabla_\theta\mathcal{L}_{z'}(\hat{\theta})}{\nabla^2_\theta\mathcal{L}_{z'}(\hat{\theta})} \\ &= -[\nabla^2_\theta\mathcal{L}_{z'}(\hat{\theta})]^{-1}[\nabla_\theta\mathcal{L}_{z'}(\hat{\theta})]\end{aligned} \tag{2}$$

Thus, the quadratic approximation of $\hat{\theta}_{z'}$ is equal to $\hat{\theta} - [\nabla^2_\theta\mathcal{L}_{z'}(\hat{\theta})]^{-1}[\nabla_\theta\mathcal{L}_{z'}(\hat{\theta})]$, and $-[\nabla^2_\theta\mathcal{L}_{z'}(\hat{\theta})]^{-1}[\nabla_\theta\mathcal{L}_{z'}(\hat{\theta})]$ is the newton step.

Evaluate the Influence of an Untrained Sample. First we add a small influence from $z^{'}$ to the loss function $\mathcal{L}(\theta)$, the new loss function is

$$\begin{aligned}\mathcal{L}_{\varepsilon,z'}(\theta) &= \arg\min_{\theta\in\Theta} \frac{1}{n}\sum_{z\in L_i} l(z,\theta) + \varepsilon l(z^{'},\theta) \\ &= \mathcal{L}(\theta) + \varepsilon l(z^{'},\theta)\end{aligned} \tag{3}$$

With new loss function, the model new parameters is obtained $\hat{\theta}_{\varepsilon,z'} = \arg\min_{\theta\in\Theta} \mathcal{L}_{\varepsilon,z'}(\theta)$. We evaluate a sample $z^{'}$ importance by calculating the $\left.\frac{\mathrm{d}\,\hat{\theta}_{\varepsilon,z'}}{\mathrm{d}\,\varepsilon}\right|_{\varepsilon=0}$

From equation 2 we know that

$$\begin{aligned}\hat{\theta}_{\varepsilon,z'} - \hat{\theta} &= -[\nabla^2_\theta\mathcal{L}_{\varepsilon,z'}(\hat{\theta})]^{-1}[\nabla_\theta\mathcal{L}_{\varepsilon,z'}(\hat{\theta})] \\ &= -[\nabla^2_\theta\mathcal{L}(\hat{\theta}) + \varepsilon\nabla^2_\theta l(z^{'},\hat{\theta})]^{-1} \\ &\quad [\nabla_\theta\mathcal{L}(\hat{\theta}) + \varepsilon\nabla_\theta l(z^{'},\hat{\theta})]\end{aligned} \tag{4}$$

Since $\hat{\theta}$ minimizes $\mathcal{L}(\theta)$, $\nabla_\theta\mathcal{L}(\hat{\theta})$ is equal to 0. Dropping the $O(\varepsilon^2)$ terms, we have

$$\hat{\theta}_{\varepsilon,z'} - \hat{\theta} \approx -[\nabla^2_\theta\mathcal{L}(\hat{\theta})]^{-1}\varepsilon\nabla_\theta l(z^{'},\hat{\theta}) \tag{5}$$

We define $H^{-1}_{\hat{\theta}} \overset{def}{=} [\nabla^2_\theta\mathcal{L}(\hat{\theta})]^{-1}$, and we have

$$\hat{\theta}_{\varepsilon,z'} - \hat{\theta} \approx -H^{-1}_{\hat{\theta}}\varepsilon\nabla_\theta l(z^{'},\hat{\theta}) \tag{6}$$

Thus, we can evaluate a untrained sample by:

$$\begin{aligned}\left.\frac{\mathrm{d}\,\hat{\theta}_{\varepsilon,z'}}{\mathrm{d}\,\varepsilon}\right|_{\varepsilon=0} &= \left.\frac{\hat{\theta}_{\varepsilon,z'} - \hat{\theta}}{\varepsilon}\right|_{\varepsilon=0} \\ &= \left.\frac{-\varepsilon H^{-1}_{\hat{\theta}}\nabla_\theta l(z^{'},\hat{\theta})}{\varepsilon}\right|_{\varepsilon=0} \\ &= -H^{-1}_{\hat{\theta}}\nabla_\theta l(z^{'},\hat{\theta})\end{aligned} \tag{7}$$

2. The Implementation Details of s_{test} Calculation

To evaluate an untrained unlabeled sample, $I(z^{'},R) = -\nabla_\theta l(R,\hat{\theta})^T H^{-1}_{\hat{\theta}} G_{z'}$ needs to be calculated. However, it's impossible to calculate the inverse matrix of the Hessian matrix due to the memory constrain of GPU and the

time complexity, especially for the deep neural network. We use the method proposed by Agarwal [1] to effectively approximate the $s_{test} \overset{def}{=} H_{\hat{\theta}}^{-1} \nabla_\theta l(R, \hat{\theta})$ and then calculate $I(z^{'}, R) = -s_{test} \cdot G_{z'}$ for each samples.

Dropping the $\hat{\theta}$ subscript for clarity, we define

$$H_j^{-1} \overset{def}{=} \sum_{i=0}^{j} (I - H)^i \tag{8}$$

as the first j terms in the Taylor expansion of H^{-1}. When $j \to \infty$, we have $H_j^{-1} \to H^{-1}$.

From equation 8, we have

$$H_j^{-1} = I + (I - H) H_{j-1}^{-1} \tag{9}$$

The key idea of stochastic estimation is that we can substitute the full H in equation 9 with the any unbiased estimator of H to form $\tilde{H}_j$. Since $\mathbb{E}[\tilde{H}_j^{-1}] = H_j^{-1}$, we still have $\mathbb{E}[\tilde{H}_j^{-1}] = H^{-1}$, when $j \to \infty$. In practice, we can randomly sample z_i and use $\nabla_\theta^2 l(z_i, \hat{\theta})$ as the unbiased estimator of H. Algorithm 1 shows how we approximate the s_{test}.

Algorithm 1 The calculation of s_{test}

1: **Input:** $v = \nabla_\theta l(R, \hat{\theta})$
2: Random sample k images $\{z_1, z_2, \cdots, z_k\}$ from labeled dataset
3: initial the $s_{test_0} = v$
4: **for** i in range(1, $k+1$) **do**
5: $\quad s_{test_i} = v + (I - \nabla_\theta^2 l(z_i, \hat{\theta})) s_{test_{i-1}}$
6: **end for**
7: take the s_{test_k} as the unbiased estimator of s_{test}
8: **Return** s_{test}

In practice, we calculate the Hessian-vector products of $\nabla_\theta^2 l(z_i, \hat{\theta}) s_{test_{i-1}}$ instead of calculating the Hessian matrix $\nabla_\theta^2 l(z_i, \hat{\theta})$. We will repeat the algorithm 1 p times, and use the averaged result as the final estimation of s_{test}.

3. The Time Complexity Analysis

As demonstrated in Section 2, our method can be divided into two sections. First, instead of directly calculate the $H_{\hat{\theta}}^{-1}$, we sample images from the labeled dataset to calculate the s_{test}, which is the stochastic estimation of $\nabla_\theta l(R, \hat{\theta})^T H_{\hat{\theta}}^{-1}$. Since the number of sampled images is fixed, the time complexity is a constant C. Then, we calculate the influence for each unlabeled sample with s_{test}. Noted that $|U| = n$, the time complexity is $O(n)$.

4. The Implementation Details of Comparing Methods

4.1. Image Classification

For coreset sampling [7], we follow [9] and implement the K-Ceter-Greedy algorithm, which is just slightly worse than the mixed-integer program but much less time-consuming. We run the algorithm by using the feature before the classification layer as [7] reported. For the learning loss sampling, we connect the learning loss module to each block of ResNet-18, stopping the loss prediction module gradient from back-propagating to the model after 120 epochs, and set the λ to 1 as [9] do. We first randomly select a subset with 10000 images from unlabeled samples before predicting the loss and selecting the image with the largest predicted loss.

4.2. Object Detection

For coreset sampling, we implement the K-Ceter-Greedy algorithm. We apply global average pooling on the feature after the regression branch and the classification branch of FCOS [8], then we concatenate the features from both branches and use this to run the algorithm. We also tried using the feature from the Feature Pyramid Network(FPN) of FCOS to run the algorithm, but it does not perform better.

For the learning loss sampling, we use the 5 feature maps from the FPN of FCOS. We stopping the loss of the loss prediction module from back-propagating to the backbone, otherwise, the detector performance would deteriorate significantly. We set the λ to 1.

For localization stability sampling [4], we implement the Localization Stability method in the paper, since its performance is evaluated on both VOC2012 [3] and COCO [5] datasets.

4.3. Large Scale Experiment in Object Detection

All the implementation details of the comparing methods are exactly the same as 4.2

5. Additional Experiments

5.1. Image Classification

In this section, we provide additional experiments on image classification with CIFAR10 in large scale active learning setting.

Active Learning Settings. For the experiments on CIFAR10, we randomly select 5000 images from the unlabeled set as the initial labeled dataset, and in each of the following steps, we add 5000 images to the labeled dataset. The simulate 10 active learning steps and stop the active learning iteration. All other implementation details are exactly the same as we described in the main paper.

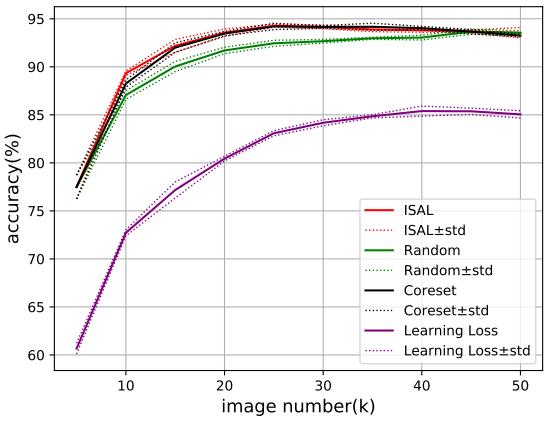

Figure 1: Result for CIFAR10 in large-scale active learning setting.

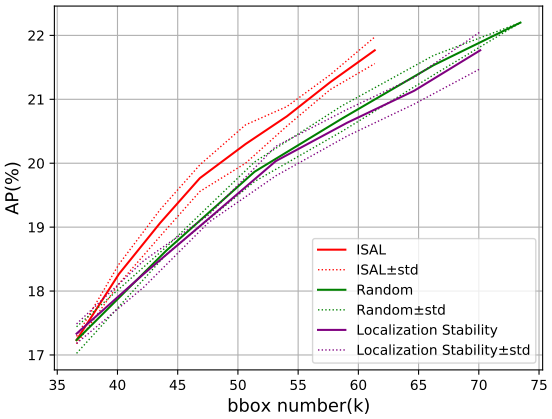

Figure 2: Result for COCO with Faster R-CNN.

Results. The results on CIFAR10 with large-scale active learning setting are shown in Figure 1. Our proposed method outperforms all comparing methods before step 6. Our implementation shows that both our method and coreset sampling achieve the best performance at step 5, and the performance of the trained model deteriorates when we keep enlarging the labeled dataset. In practice, it is not necessary to continue the active learning iteration after step 5. This phenomenon indicates that, when using the ResNet-18 as the classifier and using the test set of CIFAR10 as the benchmark to evaluate the model performance, some images in the training set of CIFAR10 provide a negative influence on the model's performance. Active learning algorithm does help trained model to achieve better performance with fewer annotations.

5.2. Object Detection

In this section, we provide additional experiments on object detection with the COCO dataset.

Active Learning Settings. We randomly select 5000 images from the unlabeled set first and add 1000 images in the following steps. Since the number of bounding boxes selected by different methods has huge differences, for clearer comparison, we continue the active learning iteration until the trained model achieves $22 \pm 0.3\%$ in AP.

Target Model. We use Faster R-CNN [6] detector with backbone ResNet-50 implemented in mmdetection [2] to verify our method. We train the model for 12 epochs with the mini-batch size of 8 and the initial learning rate of 0.01. After training 8 and 11 epochs, we decrease the learning rate by 0.1 respectively. The momentum and the weight decay are 0.9 and 0.0001 respectively.

Implementation Details. When calculating the influence of the unlabeled data, we backpropagate the loss to the parameters in the last convolution layer for regression and classification in Region Proposal Network(RPN), and to fully connected layer for regression result and classification result in Region of Interest Network(RoI) of Faster R-CNN. We use the validation set as reference set. When calculating the s_{test}, we random sample at most 500 images from the labeled set. We repeatedly calculate the s_{test} 4 times and use the value after averaging. We compare our method with random sampling and localization stability sampling [4], which can be implemented in Faster R-CNN easily. For localization stability sampling [4], we implement the Localization Stability method in the paper.

Results. The results on Faster R-CNN are shown in Figure 2. When achieving 21.8 in AP, our method cost 7.1k fewer bounding boxes than random sampling, saving 10.4% annotations. This result shows that our method can be effective in both one-stage and two-stage detectors. It further substantiates that our method is task-agnostic and model-agnostic.

6. The Experiment Results

Table 1 and table 2 show the experiment results on the image classification of the main paper. Table 3 shows the experiment result of Section 5.1 in supplementary material.

Table 4, table 5 and table 6 show the experiment results on the object detection of the main paper. Table 7 shows the experiment result of Section 5.2 in supplementary material.

References

[1] Naman Agarwal, Brian Bullins, and Elad Hazan. Second-order stochastic optimization in linear time. *stat*, 1050:15, 2016. 2

[2] Kai Chen, Jiaqi Wang, Jiangmiao Pang, Yuhang Cao, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jiarui Xu, et al. Mmdetection: Open mmlab detection toolbox and benchmark. *arXiv preprint arXiv:1906.07155*, 2019. 3

[3] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes

Challenge 2012 (VOC2012) Results. http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html. 2

[4] Chieh-Chi Kao, Teng-Yok Lee, Pradeep Sen, and Ming-Yu Liu. Localization-aware active learning for object detection. In *Asian Conference on Computer Vision*, pages 506–522. Springer, 2018. 2, 3

[5] 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

[6] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: towards real-time object detection with region proposal networks. *IEEE transactions on pattern analysis and machine intelligence*, 39(6):1137–1149, 2016. 3

[7] Ozan Sener and Silvio Savarese. Active learning for convolutional neural networks: A core-set approach. *International Conference on Learning Representations*, 2018. 2

[8] 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. 2

[9] Donggeun Yoo and In So Kweon. Learning loss for active learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 93–102, 2019. 2

Methods	5 times average of Accuracy(%) in each step				
	1	2	3	4	5
ISAL	45.51799931	54.86599902	67.72399840	**76.69599825**	**81.23799817**
coreset	45.51799931	58.29599852	67.65599847	75.99799808	79.93399816
random	45.51799931	58.40199852	67.44199847	72.30399829	77.86199819
learningloss	**45.91599973**	**58.88999852**	**69.14799823**	75.80599807	80.11599825
Methods	5 times average of Accuracy(%) in each step				
	6	7	8	9	10
ISAL	**83.61199812**	**85.95799826**	**88.05599827**	**89.26399810**	**89.95799797**
coreset	81.53599801	85.36399841	87.19999817	88.61399807	89.05199809
random	81.93599806	83.05799810	84.75199825	86.45999833	87.28999833
learningloss	82.25999810	84.46999836	85.10199790	86.66799833	87.07999842

Table 1: The experiment result on CIFAR10 with ResNet-18.

Methods	5 times average of Accuracy(%) in each step				
	1	2	3	4	5
ISAL	**36.97799921**	42.69599870	45.82199922	**50.78799950**	**53.40799696**
coreset	**36.97799921**	**43.06599873**	**46.78799956**	50.53399960	53.17999910
random	**36.97799921**	41.70599863	46.59999991	49.17400009	52.11799930
learningloss	34.06799937	38.06399904	44.43599916	45.98999956	48.60199980
Methods	5 times average of Accuracy(%) in each step				
	6	7	8	9	10
ISAL	**56.45799872**	**58.26799854**	**59.87199865**	**61.7459986**	**63.37799866**
coreset	56.02399869	58.10799861	59.32599477	61.24399836	62.70399857
random	53.43599904	56.11799880	58.47799854	60.21399841	61.22999834
learningloss	52.50199925	53.82599889	55.67399864	57.63399866	59.55999863

Table 2: The experiment result on CIFAR100 with ResNet-18.

Methods	5 times average of Accuracy(%) in each step				
	1	2	3	4	5
ISAL	**77.48999786**	**89.33399824**	**92.19199778**	**93.55399844**	**94.27999855**
coreset	**77.48999786**	88.25799831	92.00199783	93.46399852	94.20599862
random	**77.48999786**	87.05999821	90.03599803	91.70399761	92.44999793
learningloss	60.66199856	72.72399856	77.17199846	80.43999806	83.08399811
Methods	5 times average of Accuracy(%) in each step				
	6	7	8	9	10
ISAL	**94.15999845**	93.89199833	93.80199861	**93.68999855**	**93.54599838**
coreset	94.15399850	**94.16999856**	**94.02599862**	93.66199836	93.24799830
random	92.66799801	92.97199826	93.04599803	93.65399836	93.53199820
learningloss	84.17599796	84.85199794	85.39199788	85.36999783	85.04999776

Table 3: The experiment result on CIFAR10 in large-scale setting with ResNet-18.

Method		3 times average of results in each step				
		1	2	3	4	5
ISAL	mAP	0.02366667	0.12766667	0.24633333	0.32366667	0.42
	bbox num	1338	2777.66667	3606.66667	4446	5616
	10k × mAP / bbox num	0.17688092	0.45961839	0.68299445	0.72799520	0.74786325
Coreset	mAP	0.02366667	0.13	0.25733333	0.38466667	0.459
	bbox num	1338	2624.66667	4149.33333	5736.66667	7292.66667
	10k × mAP / bbox num	0.17688092	0.49530099	0.62017995	0.67054038	0.62939940
Random	mAP	0.02366667	0.11666667	0.24133333	0.342	0.43666667
	bbox num	1338	2683.66667	4097.66667	5450.66667	6833.66667
	10k × mAP / bbox num	0.17688092	0.43472861	0.58895306	0.62744618	0.63899322
Learningloss	mAP	0.023	0.13	0.25233333	0.35933333	0.42433333
	bbox num	1338	2780.66667	4161.66667	5627	7236
	10k × mAP / bbox num	0.17189836	0.46751379	0.60632759	0.63858776	0.58641975
Localization stability	mAP	0.02366667	0.136	0.243	0.33233333	0.4245
	bbox num	1338	2713.33333	3940.66667	5653	7601.66667
	10k × mAP / bbox num	0.17688092	0.50122850	0.61664693	0.58788844	0.55843017

Method		3 times average of results in each step				
		6	7	8	9	10
ISAL	mAP	0.47166667	0.515	0.55233333	0.57666667	0.596
	bbox num	7190.33333	8703.66667	10160	11503	12967.6667
	10k × mAP / bbox num	0.65597330	0.59170465	0.54363517	0.50131850	0.45960466
Coreset	mAP	0.51033333	0.552	0.57666667	0.59666667	0.604
	bbox num	8812	10286.3333	11635	12888.3333	14194.3333
	10k × mAP / bbox num	0.57913451	0.53663437	0.49563100	0.46295099	0.42552192
Random	mAP	0.4845	0.53533333	0.56133333	0.57866667	0.595
	bbox num	8188.33333	9575.66667	11011.6667	12426	13813
	10k × mAP / bbox num	0.59169550	0.55905594	0.50976237	0.46569022	0.43075364
Learningloss	mAP	0.49566667	0.54866667	0.566	0.58266667	0.59866667
	bbox num	8752.33333	10368	11750	12981.6667	14119.3333
	10k × mAP / bbox num	0.56632517	0.52919239	0.48170213	0.44883811	0.42400491
Localization stability	mAP	0.46533333	0.52766667	0.55933333	0.59	0.60466667
	bbox num	9169.66667	11056.3333	12580	13853.6667	14843.6667
	10k × mAP / bbox num	0.50747028	0.47725286	0.44462109	0.42588003	0.40735667

Table 4: The experiment result on VOC2012 with FCOS.

Method		3 times average of results in each step				
		1	2	3	4	5
ISAL	AP	0.12833333	0.14433333	0.153	0.16233333	0.166
	bbox num	36603.6667	37934	40028.3333	42021	43947.3333
	10k × AP / bbox num	0.03506024	0.03804854	0.03822293	0.03863148	0.03777249
Coreset	AP	0.12833333	0.15266667	0.17333333	0.18766667	0.19833333
	bbox num	36603.6667	47194	57032.3333	66384.3333	75564.3333
	10k × AP / bbox num	0.03506024	0.03234875	0.03039212	0.02826972	0.02624695
Random	AP	0.12833333	0.15033333	0.16566667	0.17733333	0.188
	bbox num	36603.6667	44055.6667	51381.3333	58653.6667	66240
	10k × AP / bbox num	0.03506024	0.03412350	0.03224258	0.03023397	0.02838164
Learningloss	AP	0.127	0.147	0.163	0.17766667	0.185
	bbox num	36603.6667	62127.6667	84566.3333	105900.333	126722.333
	10k × AP / bbox num	0.03469598	0.02366096	0.01927481	0.01677678	0.01459885
Localization stability	AP	0.12833333	0.149	0.16566667	0.179	0.191
	bbox num	36603.6667	47503.3333	58252.6667	69085.6667	79574.6667
	10k × AP / bbox num	0.03506024	0.03136622	0.02843933	0.02590986	0.02400261

Method		3 times average of results in each step				
		6	7	8	9	10
ISAL	AP	0.172	0.18666667	0.18333333	0.189	0.19133333
	bbox num	45810	47864.6667	49865	51930.6667	54414.3333
	10k × AP / bbox num	0.03754639	0.03899884	0.03676594	0.03639468	0.03516230
Coreset	AP	0.20933333	0.21766667	N/A	N/A	N/A
	bbox num	84924	94075.6667	N/A	N/A	N/A
	10k × AP / bbox num	0.02464949	0.02313740	N/A	N/A	N/A
Random	AP	0.199	0.20533333	0.21433333	0.22	N/A
	bbox num	73457	80720.6667	88030.6667	95464.3333	N/A
	10k × AP / bbox num	0.02709068	0.02543752	0.02434758	0.02304526	N/A
Learningloss	AP	0.19433333	0.20233333	0.21166667	0.21766667	N/A
	bbox num	145197	163798	181040.333	197804	N/A
	10k × AP / bbox num	0.01338412	0.01235261	0.01169169	0.01100416	N/A
Localization stability	AP	0.2	0.20866667	0.217	N/A	N/A
	bbox num	89557.3333	99480.6667	109179.333	N/A	N/A
	10k × AP / bbox num	0.02233206	0.0209756	0.01987556	N/A	N/A

Method		3 times average of results in each step				
		11	12	13	14	15
ISAL	AP	0.19466667	0.197	0.20233333	0.207	0.20933333
	bbox num	56457	59059	61677	64093.3333	66729
	10k × AP / bbox num	0.03448052	0.03335647	0.03280531	0.03229665	0.03137067

Method		3 times average of results in each step				
		16	17	18	19	20
ISAL	AP	0.21	0.21133333	0.216	0.21633333	0.218
	bbox num	69402	72282.3333	74690	77545.6667	80139
	10k × AP / bbox num	0.03025849	0.02923720	0.02891953	0.0278975	0.02720274

Table 5: The experiment result on COCO with FCOS.

Method		Results in each step				
		1	2	3	4	5
ISAL	AP	0.212	0.25	0.275	0.291	0.305
	bbox num	86838	118762	169688	232923	286589
	10k × AP / bbox num	0.02441328	0.02105050	0.01620621	0.01249340	0.01064242
Coreset	AP	0.212	0.273	0.305	0.321	0.334
	bbox num	86838	201072	307552	413095	510927
	10k × AP / bbox num	0.02441328	0.01357723	0.00991702	0.00777061	0.00653714
Random	AP	0.212	0.264	0.294	0.309	0.322
	bbox num	86838	173507	259539	345013	430922
	10k × AP / bbox num	0.02441328	0.01521552	0.01132778	0.00895618	0.00747235
Learningloss	AP	0.212	0.271	0.3	0.319	0.33
	bbox num	86838	291934	426039	532231	609475
	10k × AP / bbox num	0.02441328	0.00928292	0.00704161	0.00599364	0.00541450
Localization stability	AP	0.212	0.271	0.296	0.314	0.327
	bbox num	86838	194677	289580	385590	485663
	10k × AP / bbox num	0.02441328	0.01392049	0.0102217	0.00814337	0.00673306

Method		Results in each step				
		6	7	8	9	10
ISAL	AP	0.322	0.331	0.347	0.354	0.363
	bbox num	351747	449780	579039	719234	860001
	10k × AP / bbox num	0.00915431	0.00735915	0.00599269	0.00492190	0.00422093
Coreset	AP	0.344	0.351	0.355	0.36	0.364
	bbox num	601362	680853	748513	806205	860001
	10k × AP / bbox num	0.00572035	0.00515530	0.00474274	0.00446537	0.00423255
Random	AP	0.332	0.343	0.349	0.356	0.362
	bbox num	516689	602084	688451	774142	860001
	10k × AP / bbox num	0.00642553	0.0056969	0.00506935	0.0045986	0.00420930
Learningloss	AP	0.338	0.35	0.35	0.358	0.361
	bbox num	668558	713657	751218	805063	860001
	10k × AP / bbox num	0.00505566	0.00490432	0.0046591	0.00444686	0.00419767
Localization stability	AP	0.339	0.344	0.348	0.358	0.36
	bbox num	583831	674087	744716	801876	860001
	10k × AP / bbox num	0.00580648	0.00510320	0.00467292	0.00446453	0.00418604

Table 6: The experiment result on COCO in large-scale setting with FCOS.

Method		3 times average of results in each step				
		1	2	3	4	5
ISAL	AP	0.17233333	0.18266667	0.19066667	0.19766667	0.203
	bbox num	36603.6667	40120.3333	43547.3333	46840.6667	50633.3333
	10k × AP / bbox num	0.04708089	0.04552970	0.04378378	0.04219980	0.04009217
Random	AP	0.17233333	0.18633333	0.19866667	0.207	0.21533333
	bbox num	36603.6667	44055.6667	51381.3333	58653.6667	66240
	10k × AP / bbox num	0.04708089	0.04229498	0.03866514	0.03529191	0.03250805
Localization stability	AP	0.17333333	0.18266667	0.19133333	0.20033333	0.20633333
	bbox num	36603.6667	42171	47576.3333	53144.6667	59066.3333
	10k × AP / bbox num	0.04735410	0.04331571	0.04021607	0.03769585	0.03493248

Method		3 times average of results in each step				
		1	2	3	4	5
ISAL	AP	0.20733333	0.21266667	0.21766667	N/A	N/A
	bbox num	54060.3333	57655	61360.3333	N/A	N/A
	10k × AP / bbox num	0.03835221	0.03688608	0.03547351	N/A	N/A
Random	AP	0.222	N/A	N/A	N/A	N/A
	bbox num	73457	N/A	N/A	N/A	N/A
	10k × AP / bbox num	0.03022176	N/A	N/A	N/A	N/A
Localization stability	AP	0.21133333	0.21766667	N/A	N/A	N/A
	bbox num	64631.3333	70119.3333	N/A	N/A	N/A
	10k × AP / bbox num	0.03269828	0.03104232	N/A	N/A	N/A

Table 7: The experiment result on COCO with Faster R-CNN.