Method	Small	Large	All	All F-1
Ours-I	0.3132	**0.9188**	0.8143	0.8295
(No sampling)	0.4551	**0.9379**	0.8452	
Ours-P-U	0.2702	0.8933	0.8111	0.8319
(1024-0-0)	0.6068	0.9124	0.8537	
Ours-P-US	0.3209	0.9055	0.8300	0.8460
(512-512-0)	0.5944	0.9264	0.8627	
Ours-P-UB	0.3112	0.9068	0.8293	0.8455
(512-0-512)	0.6037	0.9213	0.8603	
Ours-P-USB	**0.3607**	0.9025	**0.8310**	**0.8494**
(512-256-256)	**0.6161**	0.9286	**0.8686**	

Table 1. Ablation experiments for 5 different point sampling rules. In each cell of the table, the upper and lower values represent precision and recall, respectively. Point samples consist of uncertain points (U), false positive small object points (S) and memory bank points (B). The number of samples is shown in the order of U-S-B. Our new sampling rule shows the best result for small objects and all objects.

Supplementary Material

A. Implementation Details

We use U-net [13] based on ResNet-101 [5] as back-bone network. The edge detection network is trained in a similar way to ScribbleNet [14]. The point features are extracted from the D-dimension feature map of the last layer of the decoder, which has the same width and height as the input image. D is set to 32 for ResNet-101 U-net. The extracted point feature is input to the prediction head that consists of three 32-channel point-wise convolutional layers, which outputs C-class prediction.

The total number of sampled points is 1024. First, the number of uncertain points N_U is set to 512. We sampled uncertain points in a similar way to PointRend [8]. In PointRend, some points with the class probability value close to 0.5 are selected, and others are sampled at random. Instead, we select the points with the highest entropy of the class probability. The number of false positive small object point samples, N_S is set to 256. The number of memory bank point samples, N_B is set to 256. And the number of replaced bank items, N_K is set to 32. There is no significant change according to the value of N_K.

We use the batch size of 20. The training is performed for 100K iterations. The initial learning rate is set to 10^{-4} for 50K iterations, and 10^{-5} for next 50K iterations. We use He [4] initialization for weight initialization and Adam [7] for optimization. We use L-2 regularization for the weights.

For evaluation, the MS COCO's evaluation method [9] is used to evaluate the performance of small objects separately. Average precision and recall are used, and the performance for small objects, large objects, and all objects are shown separately. The F-1 scores are also presented.

Method	Small	Large	All	All F-1
No memory	0.3209	**0.9055**	0.8300	0.8460
	0.5944	0.9264	0.8627	
MNAD [12]	0.3378	0.9021	0.8225	0.8384
	0.5573	0.9257	0.8549	
Pure bank [3]	0.3248	0.9039	0.8291	0.8453
	0.5975	0.9249	0.8621	
Ours-P-USB	**0.3607**	0.9025	**0.8310**	**0.8494**
	0.6161	**0.9286**	**0.8686**	

Table 2. Ablation experiments according to memory bank architecture. In each cell of the table, the upper and lower values represent precision and recall, respectively. MNAD [12] shows bad results because it only reads memory items similar to input query features. Pure bank [3] does not show improvement, because less-consistent features can disturb training. The proposed memory bank shows improvement, due to the update of stored features.

B. Ablation Studies

We conduct ablation studies to analyze each element of our proposed method. The effects of the point sampling rule, the memory bank architecture, the network architecture, hyper-parameters (N_K and r) and the ways of separating small object prediction will be analyzed in this supplementary material.

Point sampling rule Table 1 shows the performance comparison according to the 5 different point sampling rules since performance is more affected by the sampling rule rather than the total number of samples as mentioned also in PointRend [8]. Compared to Ours-I which does not use the point sampling, the uncertain point sampling (Ours-P-U) increased detection of small objects, improving recall, but lowering precision due to the false positive. The performance for the large object itself is reduced due to the side effect of the increased small object detection. The false positive small object point samples (Ours-P-US) decreases the recall slightly, but increases the precision moderately compared to Ours-P-U. The detection performance of large objects also improves slightly. We use the memory bank point samples for additional labeled small object data (Ours-P-UB), but the experiment shows that the effect of reducing false positive is strong, like Ours-P-US which uses false positive small object point samples. Since these two types of sample points are differently extracted, they could be applied together (Ours-P-USB) for additional performance improvement. Compared to Ours-P-US and Ours-P-UB, there is a moderate improvement in the precision of small objects, and slight improvement in the recall of small objects.

Memory bank Table 2 compares 4 memory bank architectures. When we use MNAD [12] or pure queue-based memory bank proposed in MoCo [3] instead of the pro-

Method	Small	Large	All	All F-1
S [14]	0.0020	0.5650	0.4758	0.4895
	0.0074	0.6221	0.5041	
RU+L_e+L_b	0.1321	0.7436	0.6328	0.6564
	0.1610	0.8057	0.6819	
RU	0.1527	0.9028	0.7652	0.7737
	0.1920	0.9227	0.7824	
RU+w_{size} [1]	0.2593	0.9145	0.7941	0.8053
	0.3251	0.9338	0.8169	
Ours-I	**0.3132**	**0.9188**	**0.8143**	**0.8295**
	0.4551	**0.9379**	**0.8452**	

Table 3. Ablation experiments for 5 different network architectures. In each cell of the table, the upper and lower values represent precision and recall, respectively. ScribbleNet [14] (S) shows poor result, and the replacement of back-bone network with ResNet-101 U-net (RU+L_e+L_b) shows better results. But the performance of pure ResNet-101 U-net (RU) for large objects is better than RU+L_e+L_b. The addition of size weight [1] (RU+w_{size}) improves the performance. Our method using whole pixels in an image without sampling (Ours-I) shows the best result.

posed memory bank, we observe similar or slightly lower performance than no memory.

This is because MNAD adopts soft reading for queries and it ignores the memory item not similar to input query features. And there occurs a possibility of using the average value of all memory items with different clusters, which can interfere with the training. MNAD updates its network parameters using the loss related to the distribution of memory samples. It can be helpful for anomaly detection, but maybe not for segmentation. On the other hand, our memory bank updates memory items for consistency with current mini-batch, increasing performance.

When we use the pure bank, small object features of various previous iterations can be retrieved, but making some of the features less-consistent. MoCo uses a pure queue-based memory bank to discard old mini-batch, but its slowly updated sub-network can be a problem. On the other hand, the memory bank we proposed has moderately improved performance for small objects, because it uses a writing policy of memory network to update memory samples, with the modification of entropy-based gating value.

Network architecture We compare 5 different network structures as shown in Table 3. The original ScribbleNet [14] (S) shows poor performance.

We replace the shallow encoder and the bilinear-interpolation-based decoder with ResNet-101 U-Net, which is called as RU+L_e+L_b. As a result, the output image size becomes equal to that of an input image and the performance is improved. The pure ResNet-101 U-net (RU) improves the performance further compared to RU+L_e+L_b because the use of the loss based on edge or boundary could be problematic for fully labeled large objects.

Method	Small	Large	All	All F-1
No separation	0.2914	0.9171	0.8148	**0.8307**
	0.4737	0.9360	**0.8472**	
Multi-branch	0.2987	**0.9231**	**0.8204**	0.8281
	0.4118	0.9366	0.8359	
Multi-class	**0.3132**	0.9188	0.8143	0.8295
	0.4551	**0.9379**	0.8452	

Table 4. Ablation experiments for 3 different ways to process small object predictions. The no separation method uses 2 classes of *object* and *background* without distinguishing small objects and large objects. There are two ways to separately predict small objects. The multi-branch method separates the branches for the prediction according to object size as in [2]. The multi-class approach employs the classes of *background*, *small object*, and *large object*. Performance is similar, but we use the multi-class approach due to easier sampling of small objects.

N_K	Small	Large	All
8	0.4329	0.9066	0.8402
16	0.4437	0.9151	0.8482
<u>32</u>	**0.4550**	**0.9154**	**0.8494**
64	0.4459	0.9146	0.8485

Table 5. Ablation experiments for hyper-parameter N_K. F-1 scores are similar, but the results with N_K=32 are slightly better.

The size weight [1] (RU + w_{size}) improves the small object detection performance, whereas the resulting shapes are suffered by circular artifacts as shown in Figure 1. Our proposed network considering whole pixels with the small object mask in an image without sampling (Ours-I) yields the best performance compared with the baseline methods.

Small object prediction In Table 4, we compare 3 different methods to process small object predictions with Ours-I. The no separation method uses 2 classes of *object* and *background*. The multi-branch [2] method uses a network consisting of two branches for small objects and large objects, respectively. The proposed multi-class method distinguishes small and large objects, resulting in 3 prediction classes of *background*, *small object*, and *large object*.

All methods show similar performance. In other words, the multi-branch and multi-class methods do not help to improve performance if small objects are labeled by points as we propose. The sampling of small objects of the multi-branch method is time-consuming. But, our multi-class method allows the easier sampling of the false positive points of small objects based on the false positive score.

Overview We show the abstraction of our method in Figure 3. Inference is done without the memory network and the point sampling. During training, we virtually increase the number of small object labels using the memory.

r	S→B	S→U	B→U
7	12.5185	80.3164	0.4803
14	2.7547	90.1139	1.7606
21	0.8959	91.9586	3.6034
28	0.3603	92.4868	5.5768
35	0.2312	92.6546	7.9156

Table 6. The relative percentage of area of label change according to the value of radius r. S→B (magenta in Figure 4(d)) shows the erroneous label change from small objects to background with respect to the area of small objects, S→U (yellow in Figure 4(d)) is the label change from small objects to unknown with respect to the area of small objects, and B→U (cyan in Figure 4(d)) shows the label change from background to unknown with respect to the area of background.

r	Small	Large	All
7	0.4389	0.8850	0.8179
14	0.4381	0.9151	0.8478
21	0.**4550**	**0.9154**	**0.8494**
28	0.4459	0.9137	0.8476
35	0.4497	0.9153	0.8482

Table 7. Ablation experiments for hyper-parameter r. r = 7 shows poor performance because of large mislabeling (S→B) as shown in Table 6. But the label missing (S→U, B→U) doesn't affect the performance significantly. F-1 scores of r=14∼35 are similar, meaning that we have some degree of freedom on selection of r. But we select r=21 because the results with r=21 are slightly better.

Hyper-parameters Table 5 shows the performance comparison according to the number of bank items replaced, N_K. It controls feature diversity and old data replacement. Although the performance does not change much, the value of N_K is set to 32 because it shows slightly better performance.

Change of the value of radius r could cause label change as shown in Table 6 and Figure 4. We have high mislabeling(S→B) with r=7, and it is changed to label missing(S→U) when r increased to 14. And there is not much change from r=14 to 35. This tendency is also reflected in performance with different values of r as shown in Table 7. Performance is lowest when we use r=7, but it remains similar from r=14∼35. Thus, it seems that we have a some freedom of choosing the value of r in some range, implying the background point can be easily chosen to some degree during human labeling. In other words, less time-consuming labeling is possible by setting r appropriately. In our experiments, the value of r is set to 21 because it shows slightly better performance.

References

[1] Jakub Czakon, Kamil A. Kaczmarek, Andrzej Pyskir, and Piotr Tarasiewicz. Best practices for elegant experimentation in data science projects (case study). *EuroPython*, 2018.

[2] Ryuhei Hamaguchi and Shuhei Hikosaka. Building detection from satellite imagery using ensemble of size-specific detectors. *IEEE Conf. Comput. Vis. Pattern Recog. Worksh.*, pages 187–191, 2018.

[3] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 9729–9738, 2020.

[4] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. *Int. Conf. Comput. Vis.*, pages 1026–1034, 2015.

[5] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 770–778, 2016.

[6] Shunping Ji, Shiqing Wei, and Meng Lu. Fully convolutional networks for multisource building extraction from an open aerial and satellite imagery data set. *IEEE Transactions on Geoscience and Remote Sensing*, 57(1):574–586, 2018.

[7] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv:1412.6980*, 2014.

[8] Alexander Kirillov, Yuxin Wu, Kaiming He, and Ross Girshick. Pointrend: Image segmentation as rendering. *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 9799–9808, 2020.

[9] 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. *Eur. Conf. Comput. Vis.*, pages 740–755, 2014.

[10] Volodymyr Mnih. Machine learning for aerial image labeling. 2013.

[11] Sharada Prasanna Mohanty. Crowdai mapping challenge 2018: Baseline with mask rcnn, 2018. https://github.com/crowdai/crowdai-mapping-challenge-mask-rcnn.

[12] Hyunjong Park, Jongyoun Noh, and Bumsub Ham. Learning memory-guided normality for anomaly detection. *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 14372–14381, 2020.

[13] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241, 2015.

[14] Jing Zhang, Xin Yu, Aixuan Li, Peipei Song, Bowen Liu, and Yuchao Dai. Weakly-supervised salient object detection via scribble annotations. *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 12546–12555, 2020.

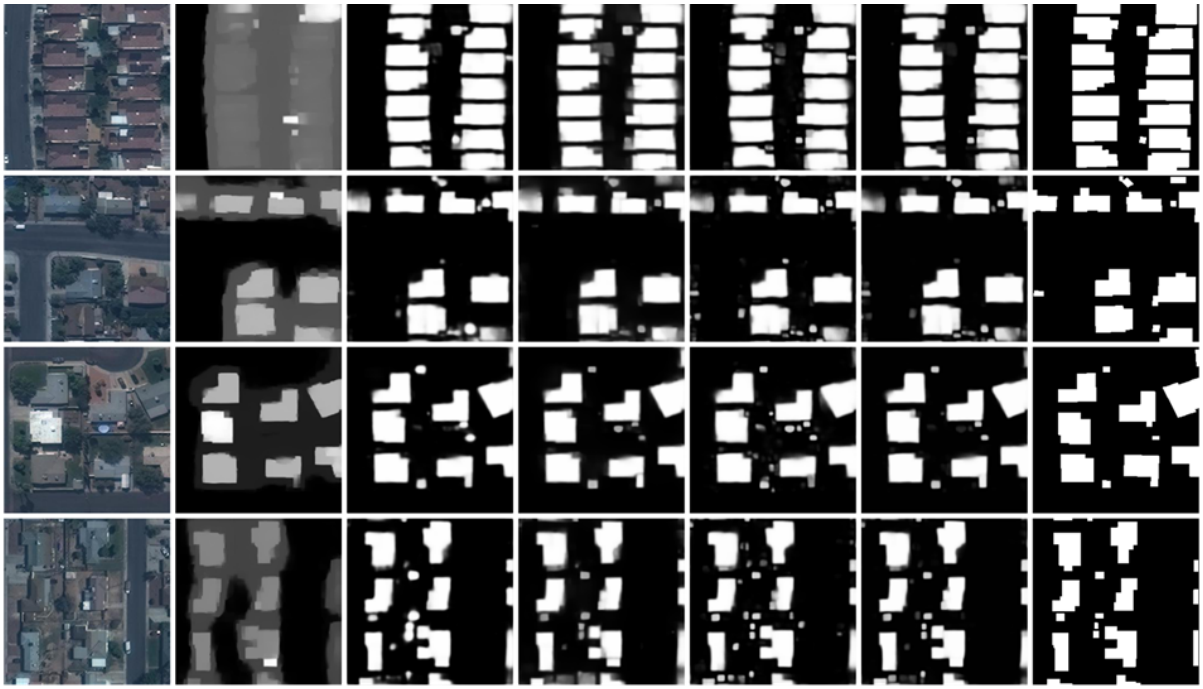

Figure 1. Experimental results on CrowdAI dataset [11]. (a) Input image. (b) ScribbleNet [14]. (c) RU+w_{size} [1]. (d) Ours-I. (e) Ours-P-U. (f) Ours-P-USB. (g) Ground truth.

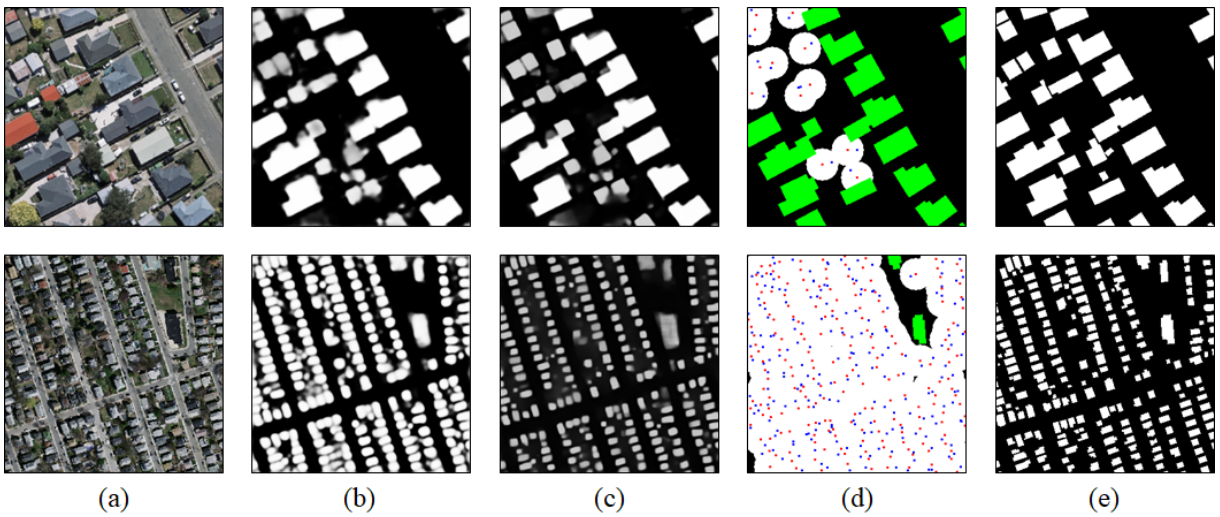

Figure 2. Experimental results on WHU building dataset [6] (1st row) and Massachusetts buildings dataset [10] (2nd row). (b) RU+w_{size}. (c) Ours-P-USB. (d) Point label. (e) Full label.

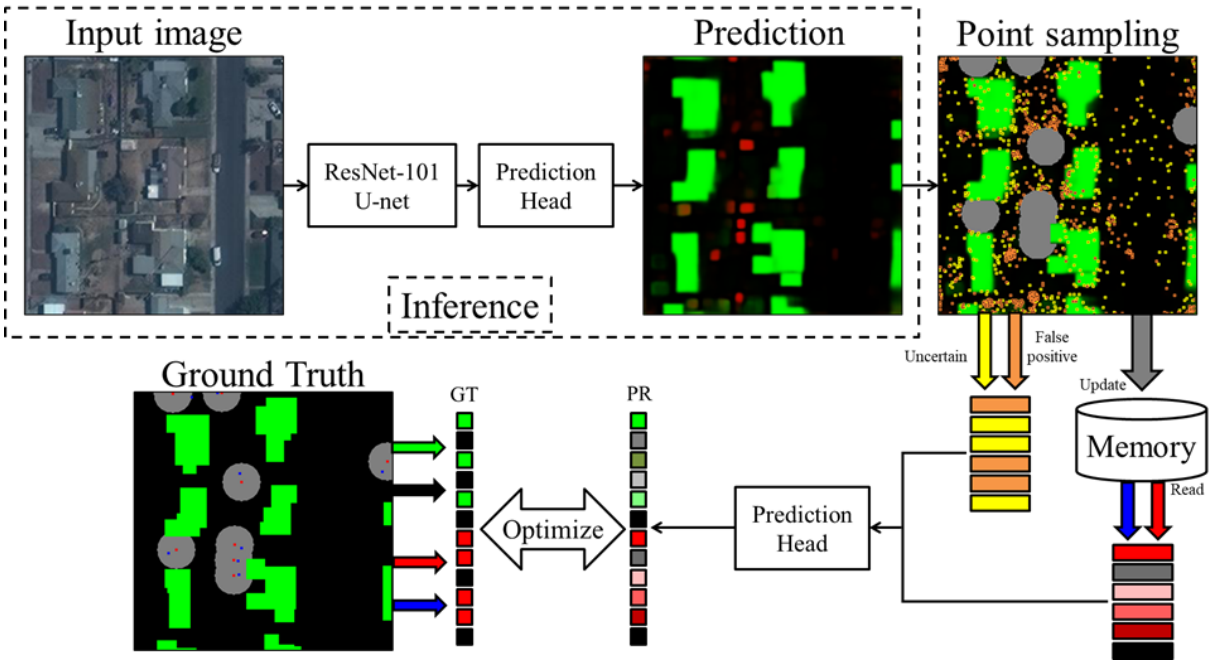

Figure 3. Overall process of the proposed method. Sampling points that are *uncertain* (high entropy of class prediction probability), *false positive* (predicted as small object although their ground truth are either background or large object), and *updated* (labeled as background or small object inside the circles).

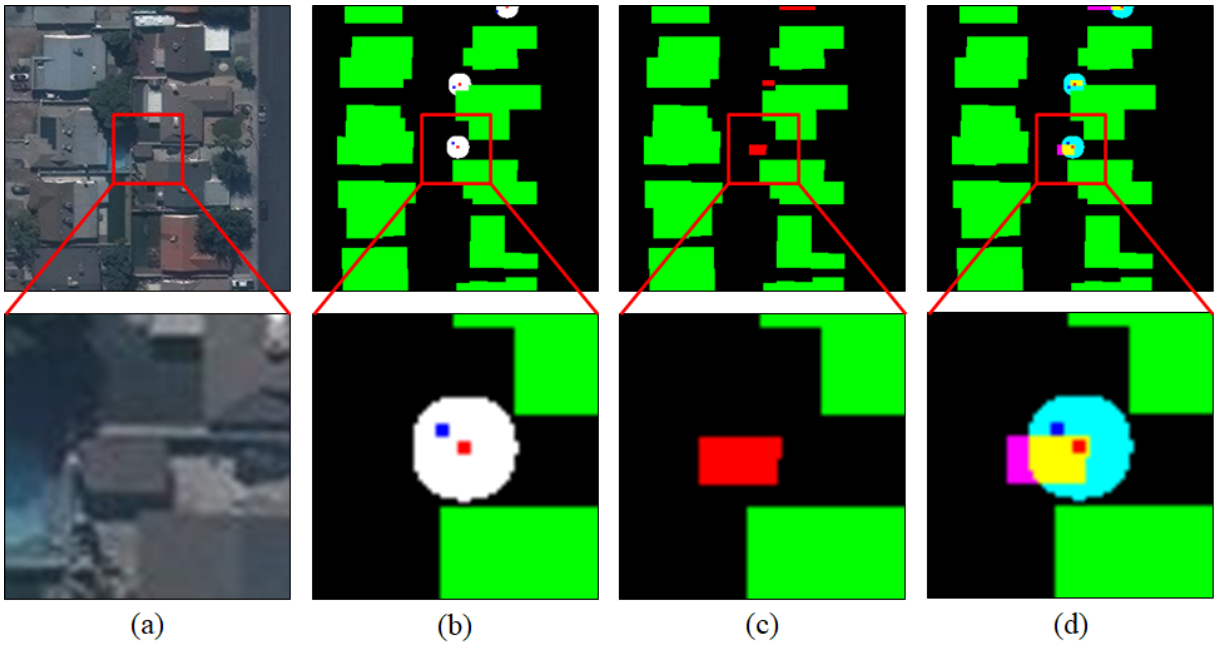

Figure 4. Imperfect point-labeled image with small radius ($r = 7$). (a) Input image. (b) Point label. (c) Full label. (d) Overlap of point label and full label. There occur 3 types of label change. The first one (yellow) and the second one (cyan) is changed from small object to unknown region and from background to unknown region, respectively. The third one (magenta) is erroneously changed from small object to background.