Appendix

A.1 Boundary Region Calculation

As mentioned in Section 3.3, we design a convolutional operator to approximate the calculation of boundary regions defined in Equation 1 for efficient implementation. In this section, we give details of the convolutional operator.

Implementation of the approximation approach. To calculate the boundary region B^k of instance mask M^k, we use a specific convolutional operator to calculate the foreground boundary region and the background boundary region respectively, where k denotes index of refinement stages in the mask head. When calculating the background boundary region, we first reverse binary values of M^k. Taking boundary width of 1 as an example, kernel weights of the operator are defined as a 3×3 matrix:

$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

Let D^k denotes the intermediate output of above convolutional layer, and it has the same size as M^k. Values of B^k are determined as follows:

$$B^k(i,j) = \begin{cases} 1, & \text{if } D^k(i,j) > 0 \\ 0, & \text{otherwise.} \end{cases}$$

Specifically, the custom operator defined above executes on a binary instance mask M^k, and it generates the binary boundary mask B^k of M^k. If the operator calculates on a pure foreground region (all ones) or background region (all zeros), the output $D^k(i,j)$ is zero. Only when the operator calculates on a boundary region, the output $D^k(i,j)$ can be larger than zero. To calculate boundary regions with width of 2, the operator can be defined as a 5×5 matrix:

$$\begin{bmatrix} -1 & -1 & -1 & -1 & -1 \\ -1 & -1 & -1 & -1 & -1 \\ -1 & -1 & 24 & -1 & -1 \\ -1 & -1 & -1 & -1 & -1 \\ -1 & -1 & -1 & -1 & -1 \end{bmatrix}$$

To calculate boundary regions with larger width, the convolutional operator can be defined similarly, making sure sum of all weights is zero.

Comparisons between two implementations. We counted the average IoU between boundary regions generated by the definition and the approximation approach (Table 12). The high IoU indicates our faster implementation generates similar results with the definition. We also give examples of boundary regions calculated by these two methods in Figure 7 respectively. As we can see, boundary regions generated by these two methods are visually near the same.

Output size	IoU
28×28	0.76
56×56	0.75
112×112	0.80

Table 12: IoU between boundary regions from two implementations.

Boundary width	AP	AP*	AP$^\star_S$	AP$^\star_M$	AP$^\star_L$
1	37.2	40.7	24.1	47.7	57.7
2	37.3	40.9	24.1	48.8	58.0
3	37.3	40.8	23.6	48.4	58.4

Table 13: Different boundary widths for boundary-aware refinement.

Model	Backbone	AP	AP*	F$_{1px}$	F$_{3px}$
Mask R-CNN	X101-FPN	**37.8**	40.1	64.1	82.6
Mask R-CNN	R50-FPN	34.7	36.8	62.0	80.6
RefineMask	R50-FPN	37.3	**40.9**	**69.6**	**84.9**
		+2.6	**+4.1**	**+7.6**	**+4.3**

Table 14: Comparison between Mask R-CNN and RefineMask on COCO $val2017$ under different evaluation metrics.

In addition, experiments show that the effectiveness of the boundary-aware refinement is not sensitive to the boundary width (Table 13), which further indicates the subtle differences between these two implementations are not essential to the final performance (the differences between boundary regions with different boundary widths are obviously much larger than the differences between results generated by these two implementations).

A.2 Direct Measurement of Boundary Quality

In order to directly measure the boundary quality, we also evaluated our method by the metric F$_{npx}$, which is designed by adapting the boundary F1 score proposed in [21] from semantic segmentation to instance segmentation. For each instance, we computed the boundary F1 score within n pixels from object contour between the ground truth mask and its positive prediction mask with maximum IoU. It was ignored if there was no positive prediction for a given instance. Results in Table 14 show that more gains are observed on COCO, where refinemask even outperforms the Mask R-CNN model with heavier backbone X101-FPN by a large margin, further indicating that RefineMask improves the boundary quality significantly.

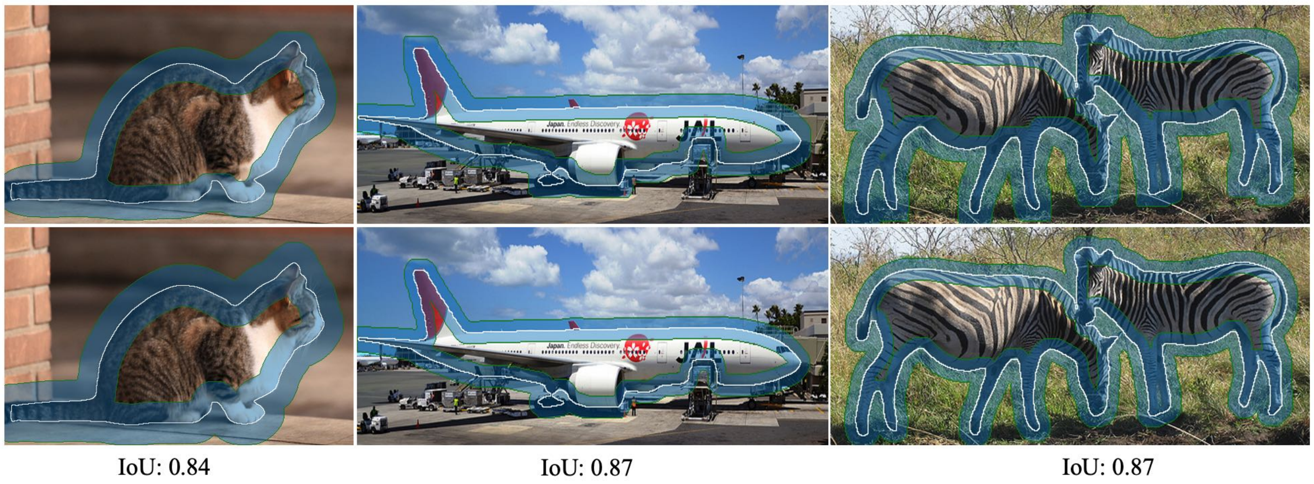

Figure 7: Visualization results of the approximated boundary regions by convolutional kernel (the top row) and the defined boundary regions by Equation 1 in Section 3.3 (the bottom row).