

Master's Thesis Presentation for the 146th Dissertation Evaluation

A Study on Image Forgery Localization based on Distant Point Prompting for Multiple Segments in Non-Forgery Region

다중 비위조 영역을 위한 복수 포인트 프롬프팅
기반 이미지 위조 지역화에 관한 연구

Hun Song

thdgns8@gmail.com



중앙대학교
CHUNG-ANG UNIVERSITY



Contents

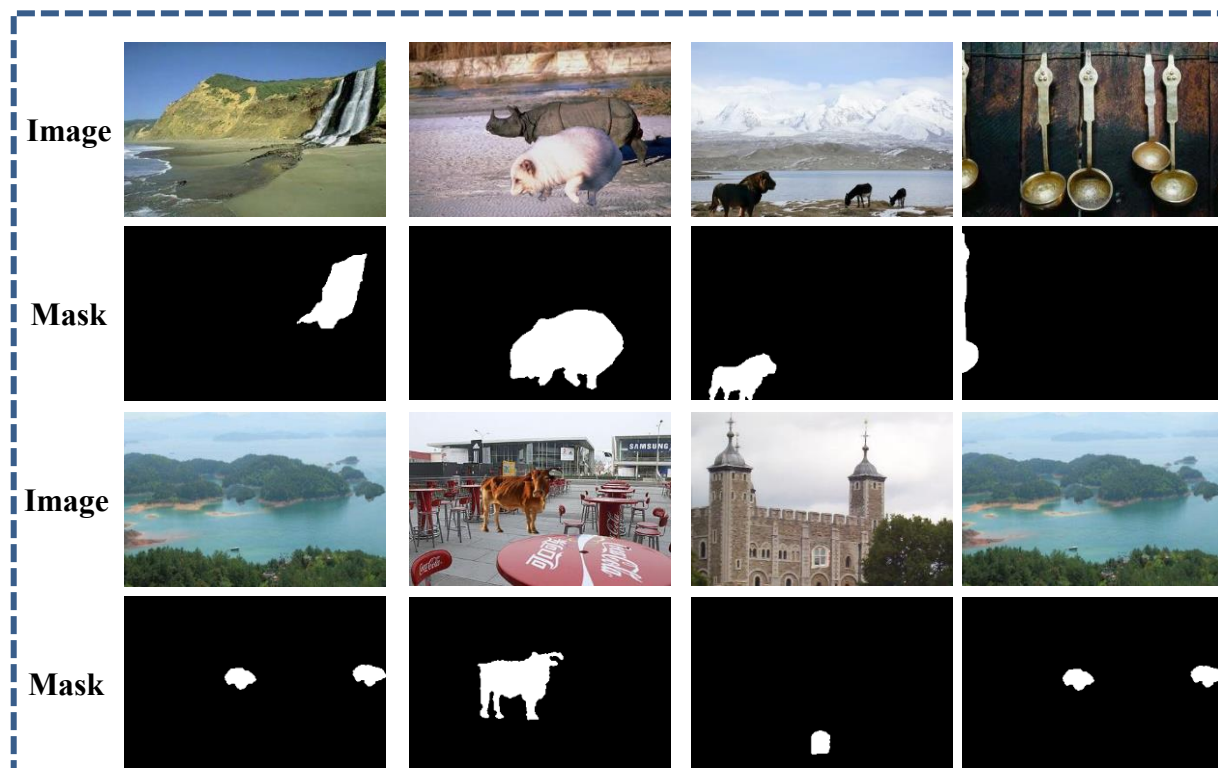
- 1. Introduction**
- 2. Related Work**
- 3. Proposed Method**
- 4. Experimental Settings**
- 5. Overall Experimental Results**
- 6. Conclusion**

Introduction

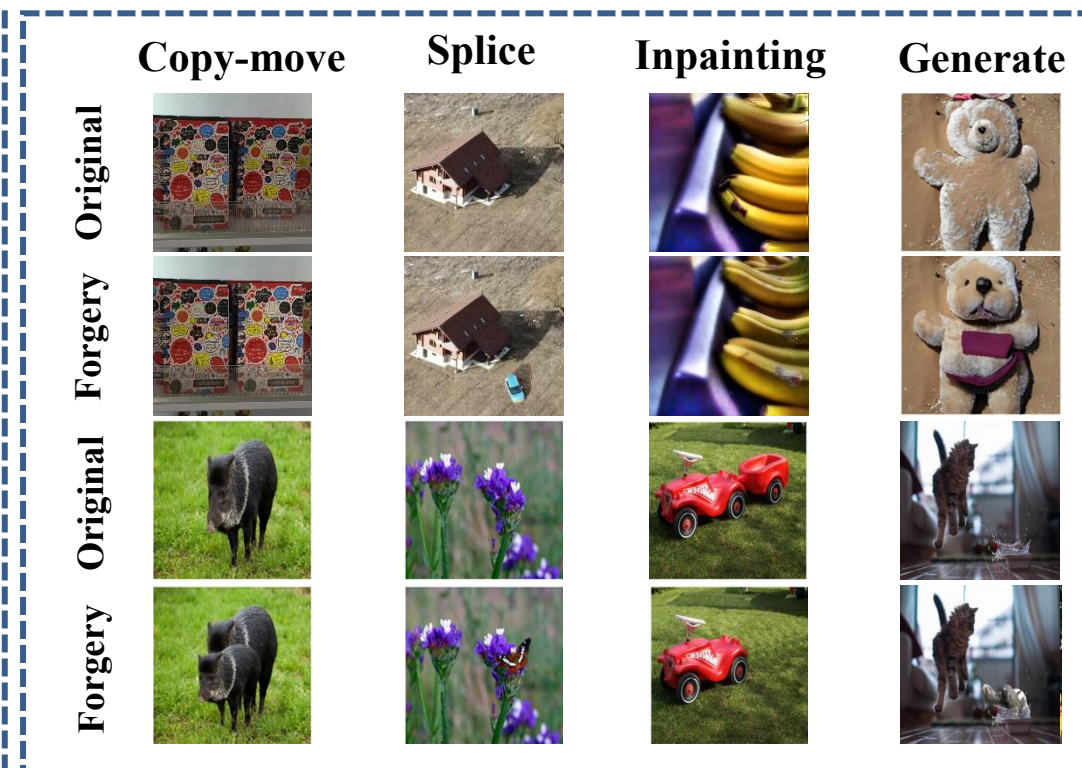
What is Image Forgery Localization & Why it is Important

- The goal of Image Forgery Localization (IFL) is to produce a pixel-level mask that precisely identifies all manipulated regions within an image, distinguishing them from the non-forgery regions under diverse editing operations [1,2].
- The importance of IFL is accelerating as powerful generative tools produce convincing forgeries, requiring robust localization to preserve generalization across domains despite challenges like varying sources, compressions, and artifacts [3,4].

- [1] Verdoliva, Luisa. "Media forensics and deepfakes: an overview." **IEEE J-STSP**, 2020.
- [2] Hu, Xuefeng, et al. "SPAN: Spatial pyramid attention network for image manipulation localization." **ECCV**, 2020.
- [3] Tolosana, Ruben, et al. "Deepfakes and beyond: A survey of face manipulation and fake detection." **INFORMATION FUSION**, 2020.
- [4] Wu, Yue, Wael AbdAlmageed, and Premkumar Natarajan. "Mantra-net: Manipulation tracing network for detection and localization of image forgeries with anomalous features." **CVPR**, 2019.



[Figure 1] Forgery Image with Ground Truth

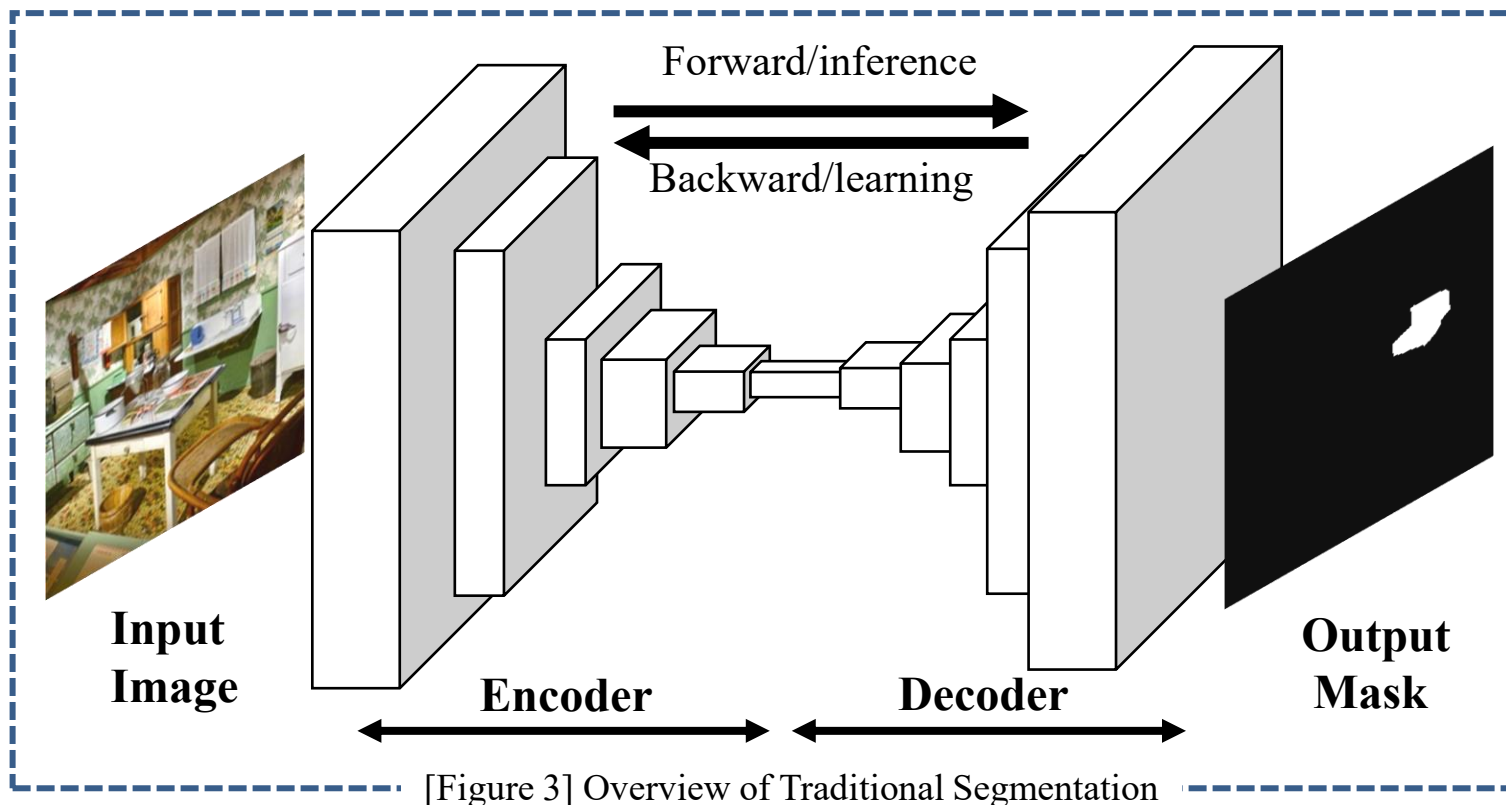


[Figure 2] Example of various forgery images

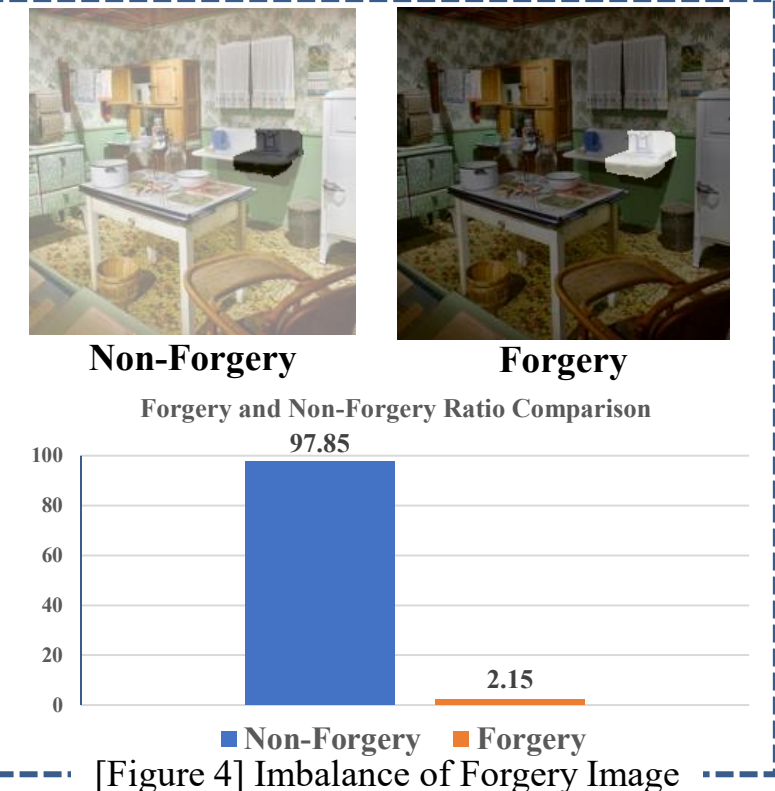
Introduction

Pixel Membership Region Localization

- While traditional segmentation model can be applied, the large number of non-forgery pixels overwhelms the sparse forgery pixels during training, including an unintended bias toward non-forgery predictions, which results in degraded accuracy [5].
- Recent studies indicate that this drawback can be overcome by reformulating the task as Pixel Membership Region (PMR) Localization, which identifies the membership region of selected prompt pixels in both forgery and non-forgery areas [6].



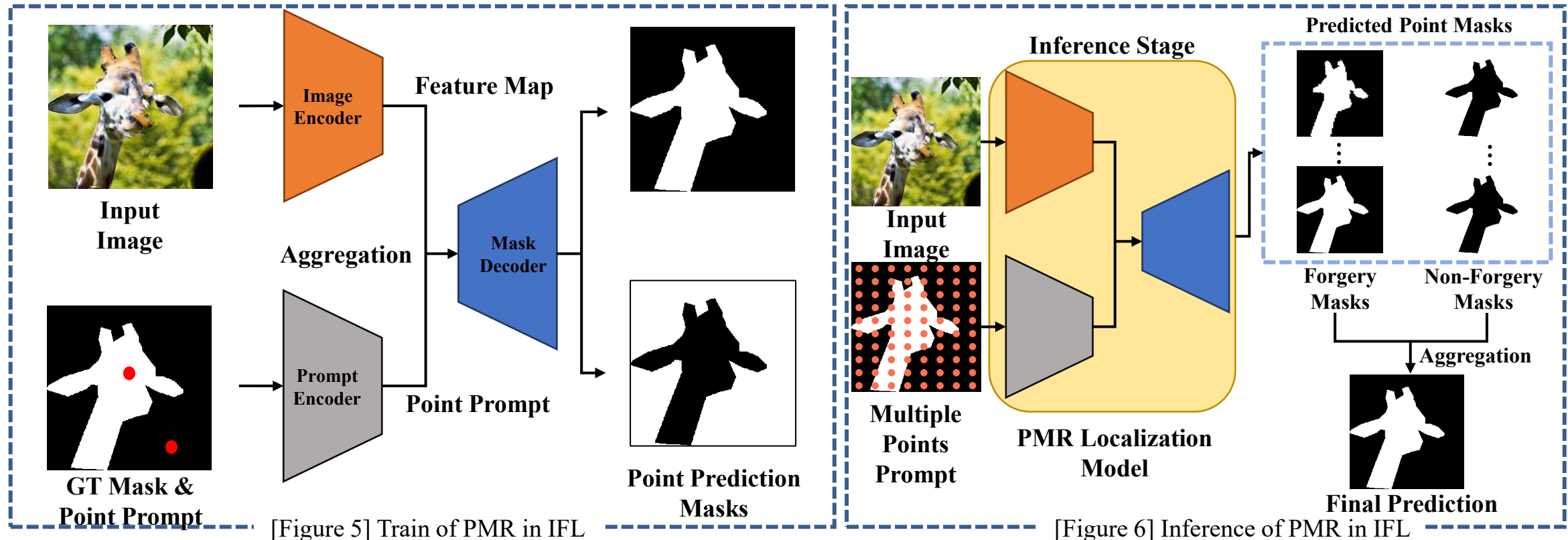
[5] Gu, A-Rom, Ju-Hyeon Nam, and Sang-Chul Lee. "FBI-Net: Frequency-based image forgery localization via multitask learning with self-attention." *IEEE ACCESS*, 2022.
[6] Kirillov, Alexander, et al. "Segment anything." *ICCV*, 2023.



Introduction

Pixel Membership Region Localization in Image Forgery Localization

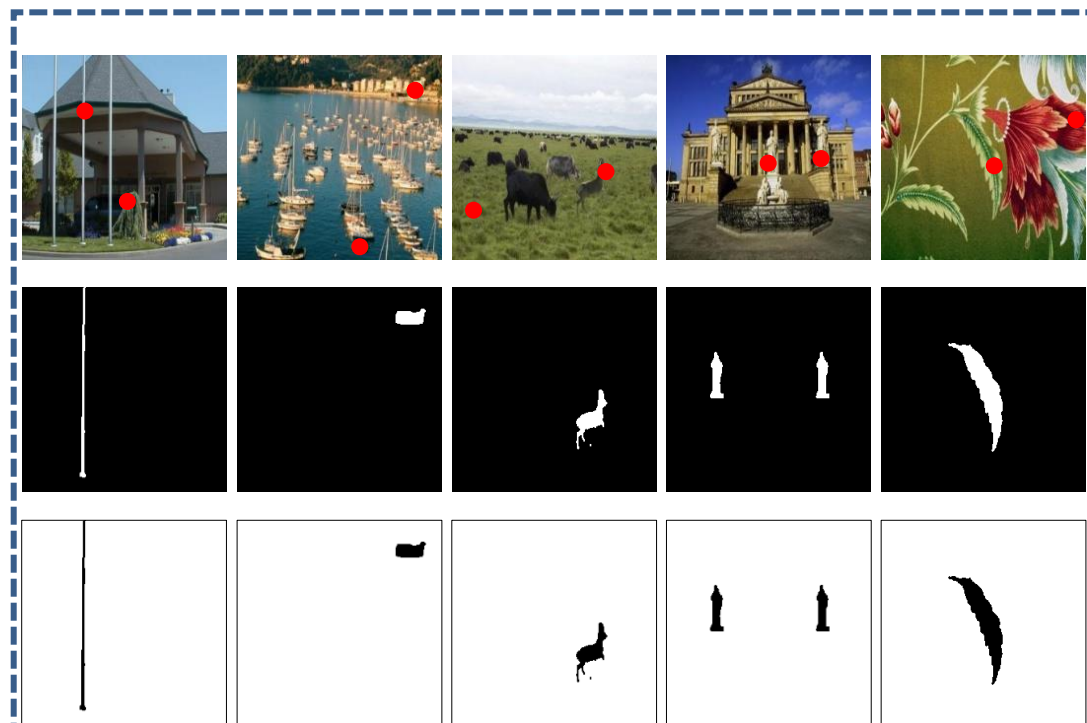
- During IFL training, PMR samples forgery and non-forgery prompt pixels, predicts a membership map for each prompt using the fused image features and prompt embedding, and supervises the map to ensure alignment with the prompt's true region [6].
- At inference, PMR samples multiple point prompts on the input image, feeds the prompts with the image into the PRM model, predicts a membership mask for each point, and aggregates these per-prompt maps to produce the final forgery mask [7].



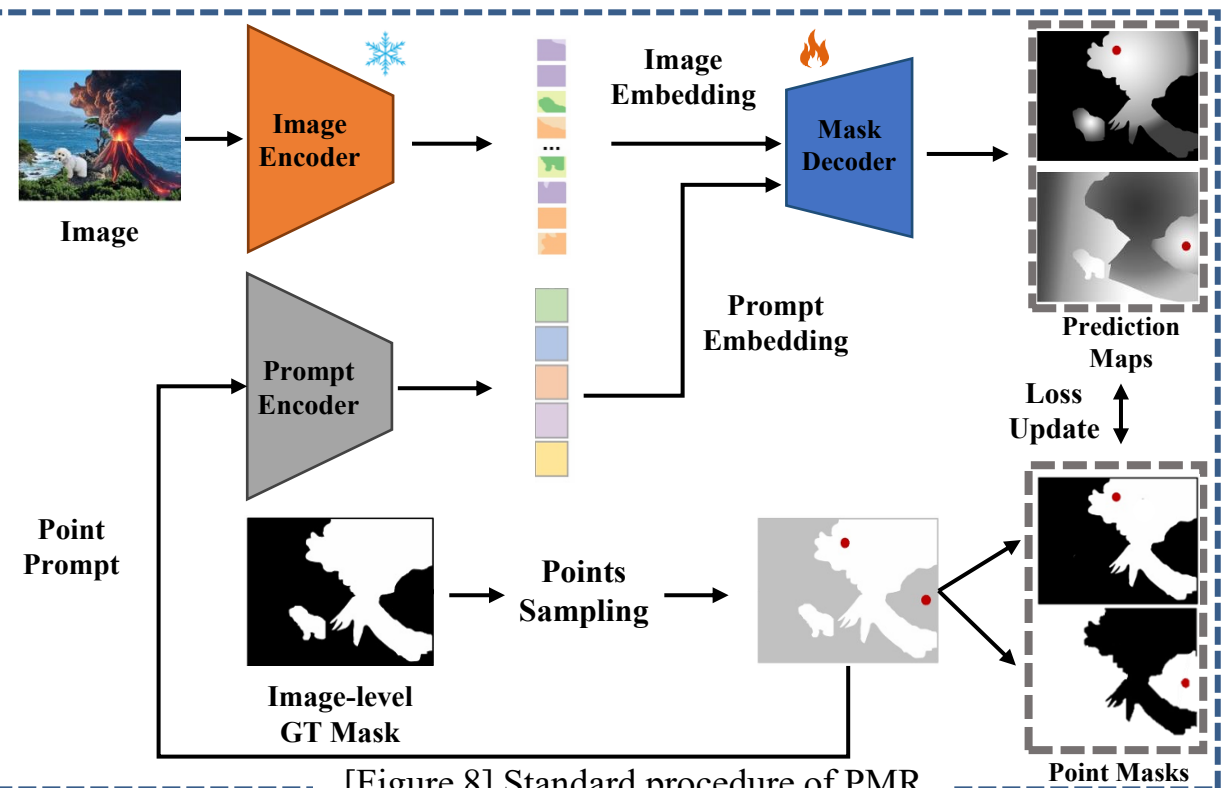
Introduction

Objective and Standard Procedure

- The objective of PMR localization is to accurately identify and segment the entire membership region to which a selected prompt pixel belongs, whether it is forgery or non-forgery, and to yield a reliable forgery/non-forgery prediction map [7].
- PMR localization takes an input image and points sampled from the mask, extracts image features using an image encoder, processes points using a prompt encoder, and finally fuses the outputs within a mask decoder to generate the pixel-level mask [8].



[Figure 7] Examples of PMR (red dot indicates selected prompt pixel)

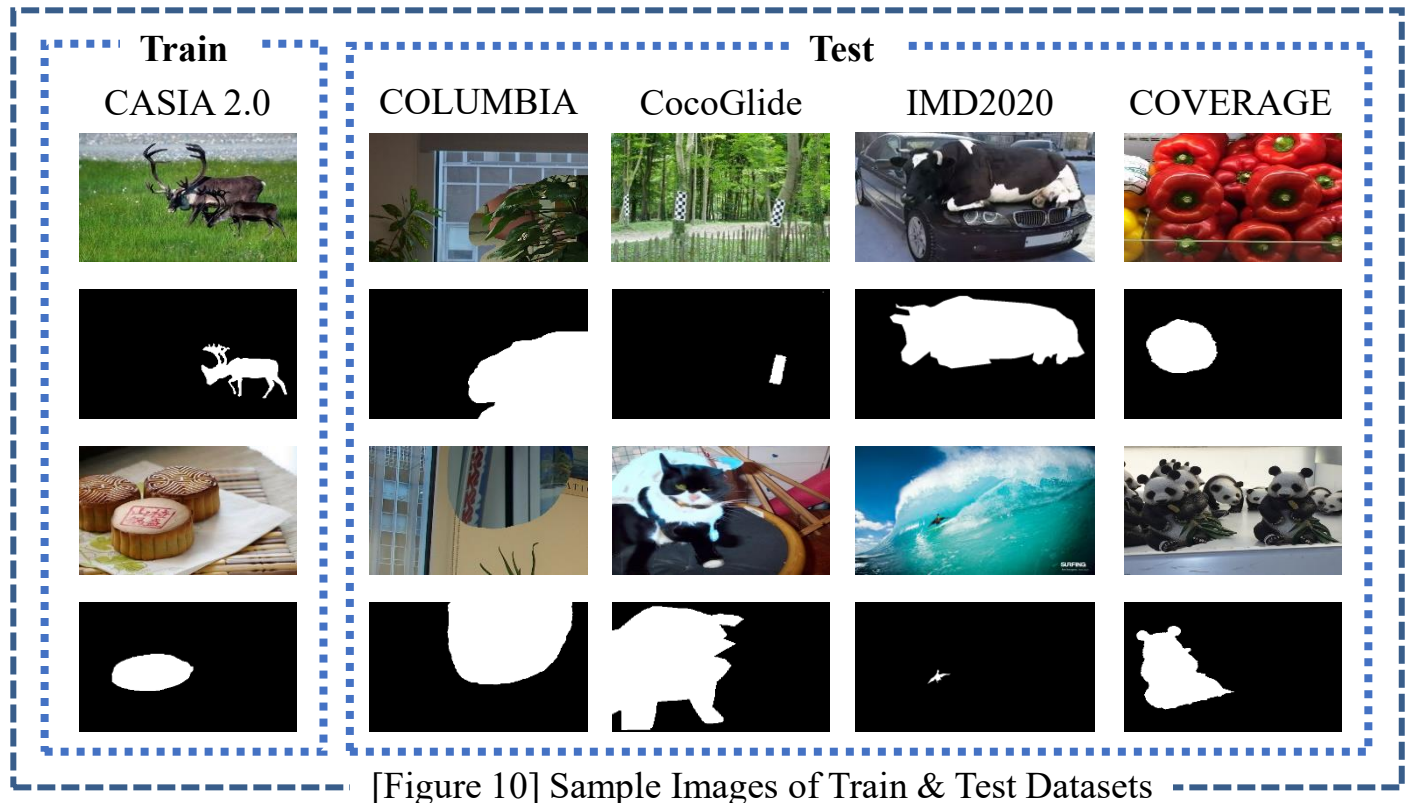
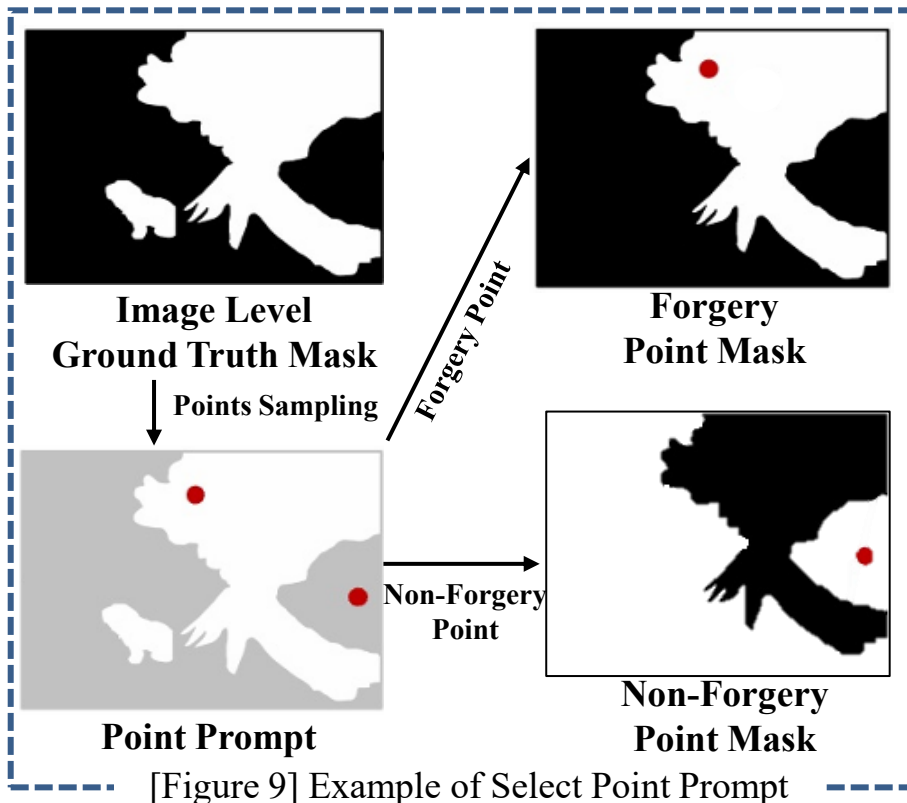


[Figure 8] Standard procedure of PMR

Introduction

Ideal Goal and Challenge

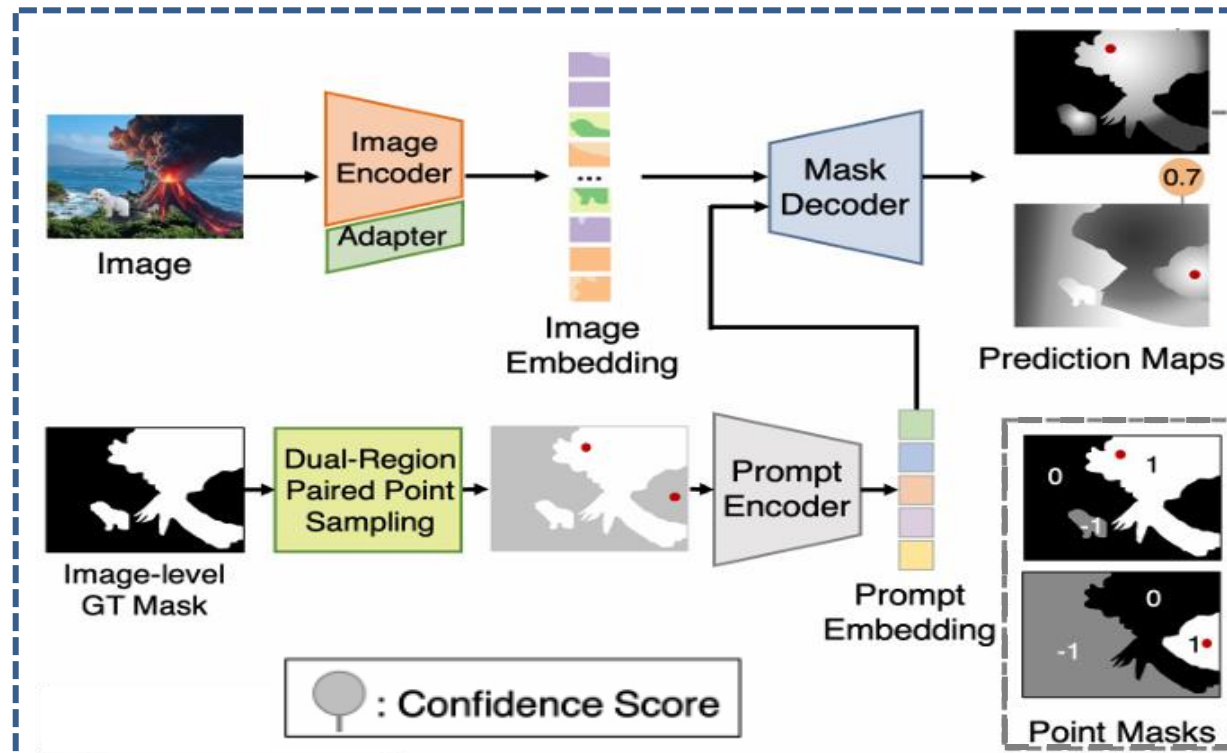
- In the viewpoint of IFL, the ideal goal of PMR localization is to identify all pixels that belong to the region by using a single prompt pixel, even though the region is often composed of multiple spatially distinct regions.
- The main challenge is that complex non-forgery regions often contain multiple objects of distinct semantics, making it difficult for a single pixel to sufficiently cover and represent all the distinct features within non-forgery regions.



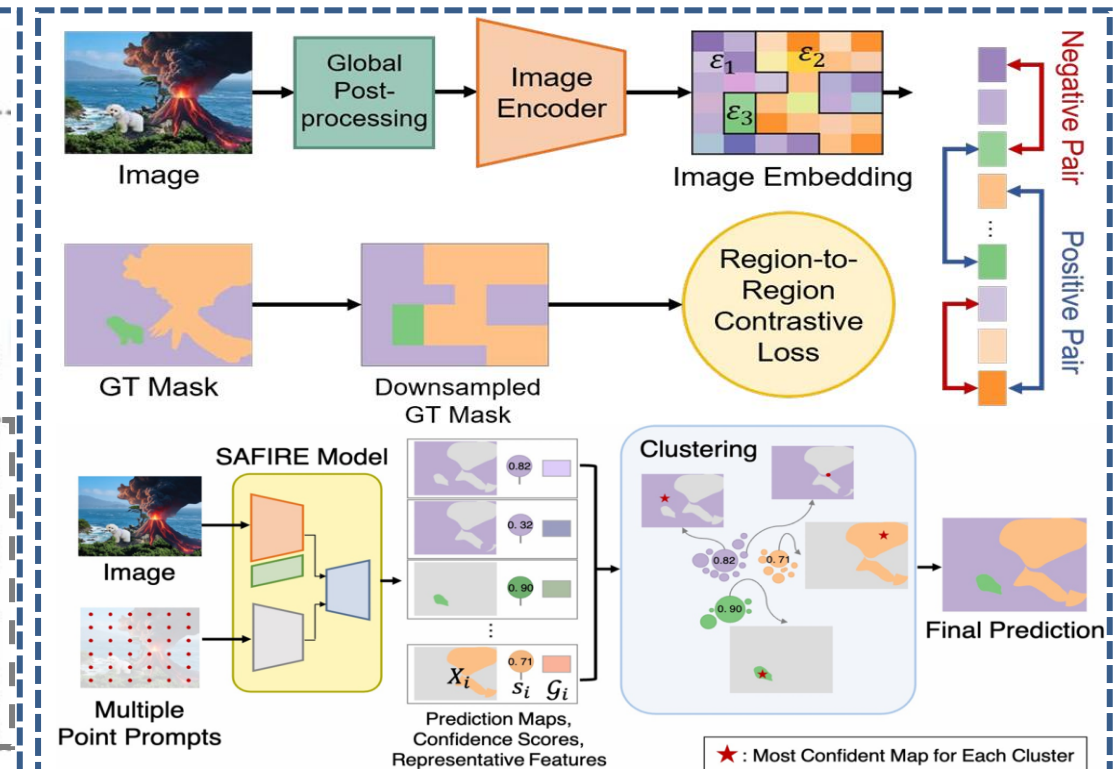
Related Work

SAFIRE: Segment Any Forged Image Region [7]

- SAFIRE uses SAM-based point prompting internally to partition an image into distinct source regions, thereby reframing image forgery localization from a binary task to a more fundamental source-based partitioning problem.
- The single prompt pixel used by SAFIRE is often insufficient to represent the diverse features of complex non-forgery regions, a limitation that results in inadequate segmentation for those regions.



[Figure 11] Training Pipeline of SAFIRE

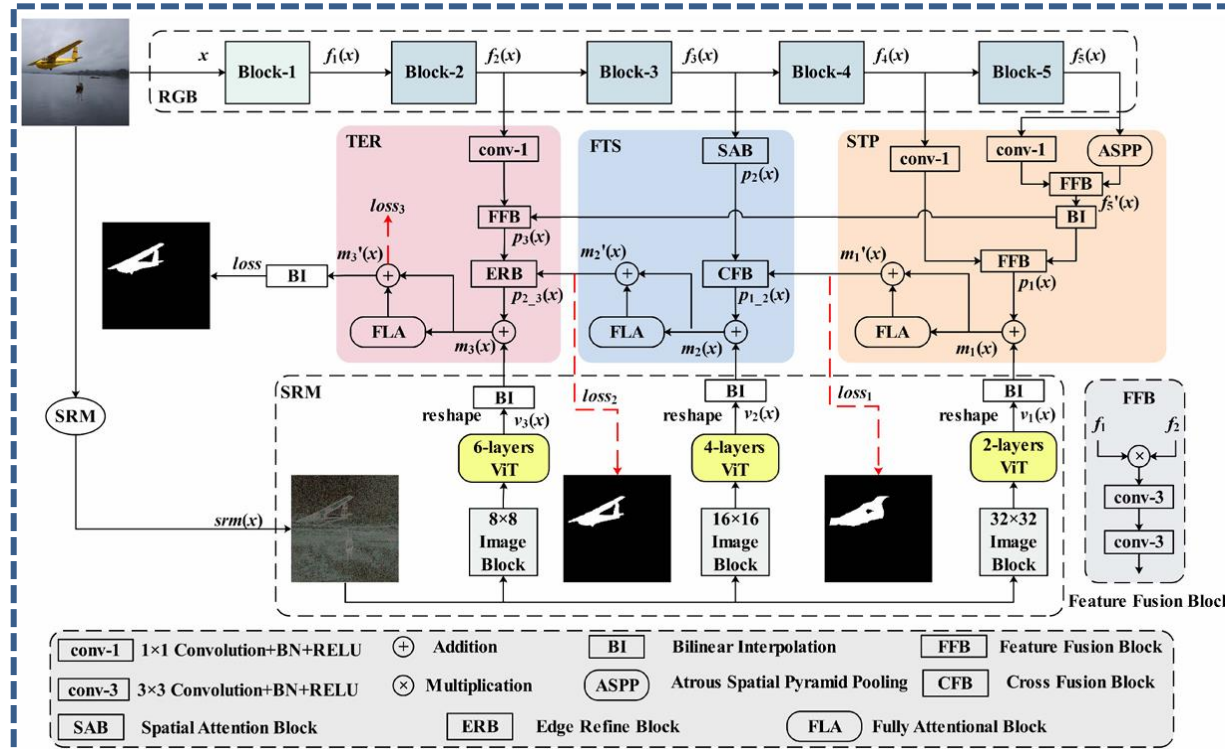


[Figure 12] Pretraining and Inference Pipeline of SAFIRE

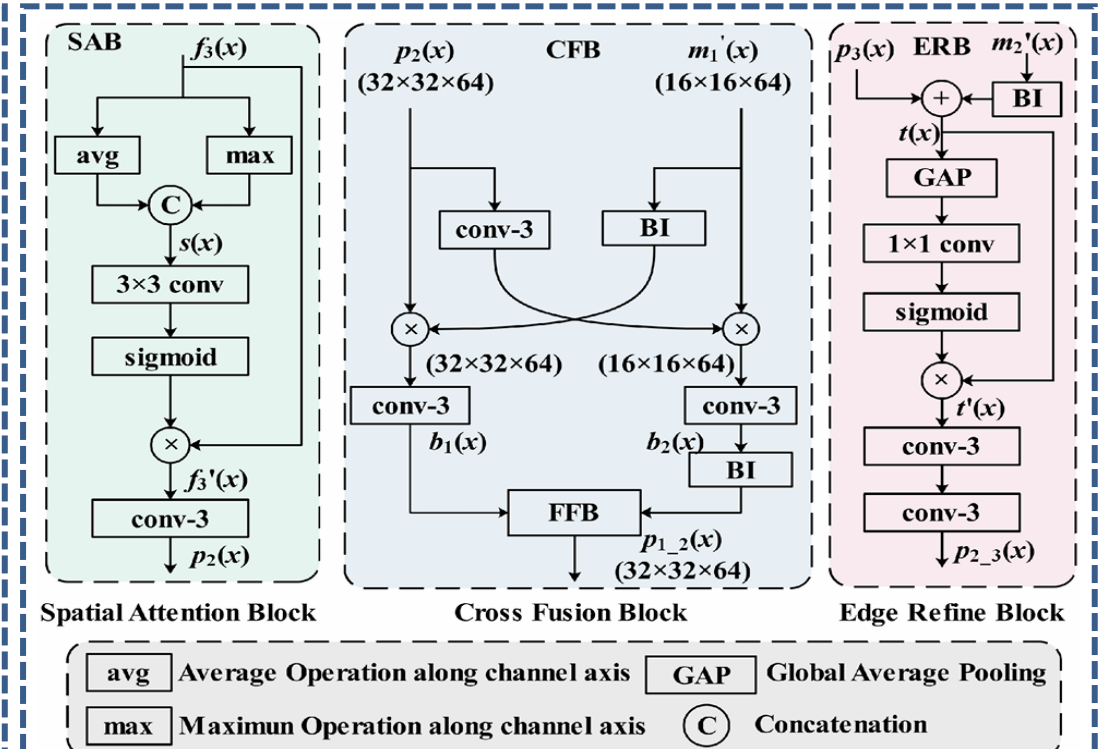
Related Work

HDF-Net: Capturing Homogeny Difference Features to Localize the Tampered Image [9]

- HDF-Net utilizes an RGB and SRM dual-stream network, featuring a multi-scale ViT for noise analysis, to capture "homogeny difference features" (inconsistencies in metadata structure) using a "coarse-fine-finer" module strategy.
- As a whole-image segmentation model, the non-forgery majority tends to dominate training, yielding conservative masks that miss minority forgery pixels—so recall falls below precision, especially in cross-domain evaluations.



[Figure 13] Architecture of HDF-Net

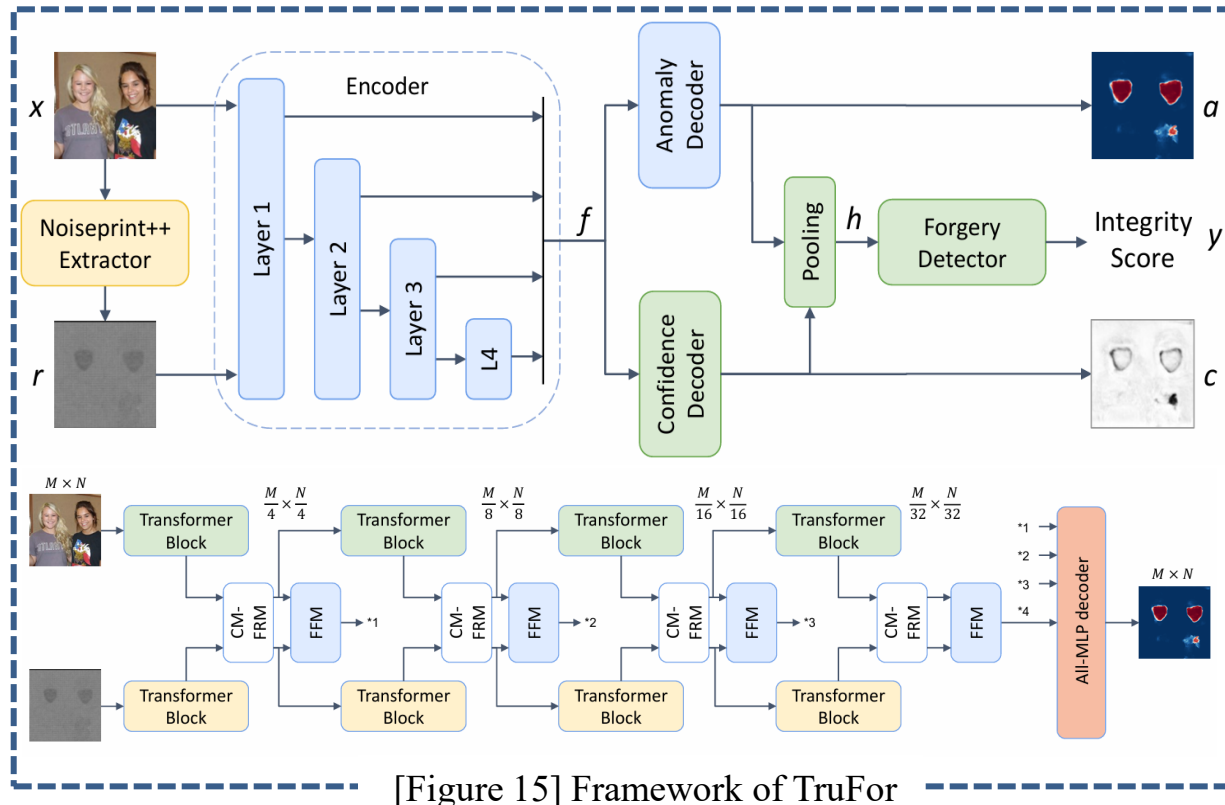


[Figure 14] The concrete structure of SAB, CFB, and ERB

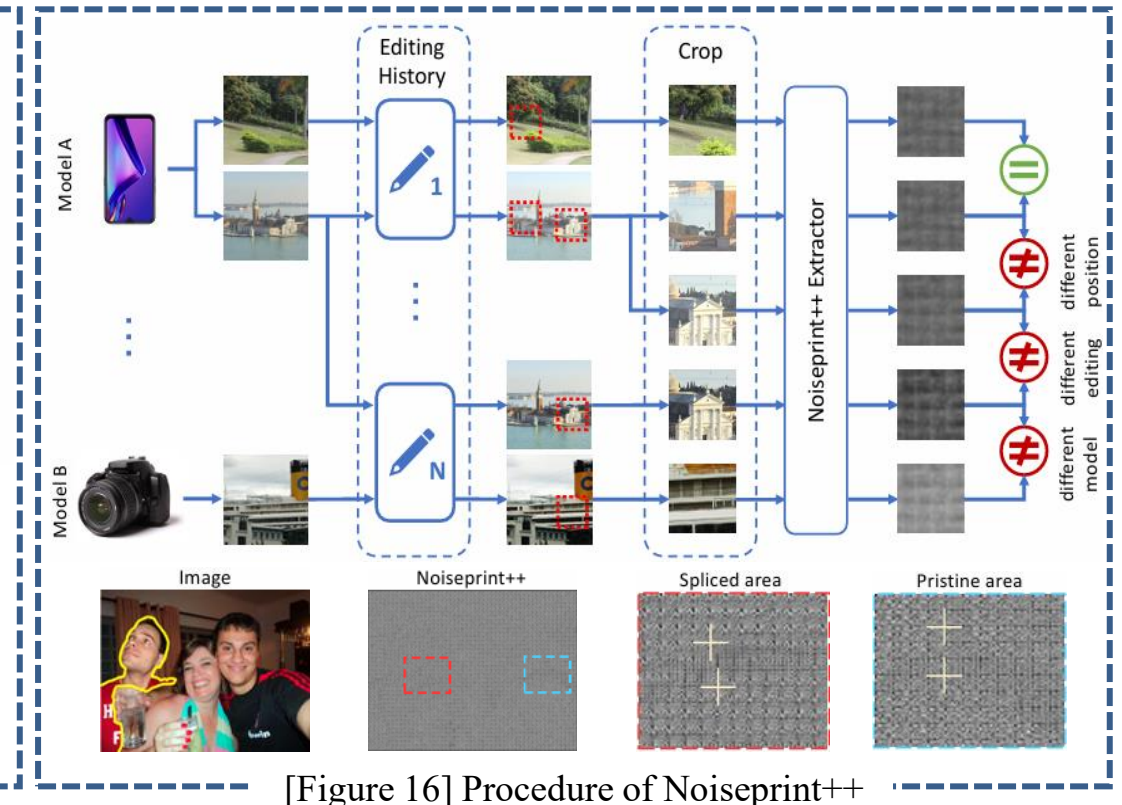
Related Work

TruFor: Leveraging all-round clues for trustworthy image forgery detection and localization [10]

- TruFor employs a transformer-based fusion network to combine RGB features with its learned "Noiseprint++" fingerprint, identifying forgeries as anomalies while simultaneously generating a confidence map to reduce false alarms.
- Despite multi-clue fusion, global prediction remains biased toward non-forgery under severe class imbalance, often missing small or disconnected edits and producing lower recall than precision across varied manipulations.



[Figure 15] Framework of TruFor

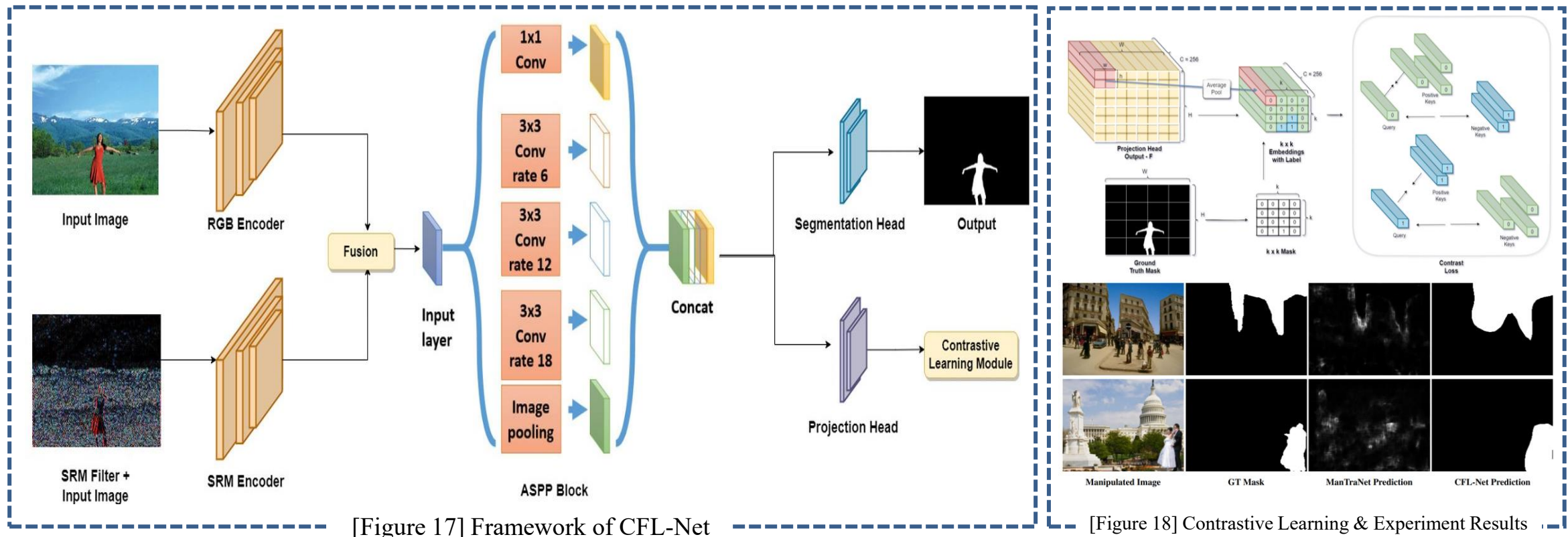


[Figure 16] Procedure of Noiseprint++

Related Work

CFL-Net: Image Forgery Localization Using Contrastive Learning [11]

- CFL-Net has Two-branch encoder with ASPP and a contrastive projection head (SupCon) to separate forgery/non-forgery features, coupled with a segmentation head that sharpens boundaries and reduces artifact overfitting for stronger cross-domain masks.
- Even with contrastive supervision, full-image training favors the non-forgery majority, leaving forgery regions under-represented and recall consistently below precision, particularly for tiny or fragmented forgeries.



Common Drawback and Hypothesis

- Existing methods lack the ability to capture the complete membership region for areas and rely on a limited number of prompt points, fail to effectively model their diverse, multi-object characteristics, leading to incomplete segmentation.
- By sampling prompt points, with a single point for forgery regions and multiple (k) points for non-forgery areas, the diverse characteristics of complex backgrounds across domains can be comprehensively and robustly modeled.

[Table 1] Summary of recent Pixel Membership Region Localization methods, outlining their key contributions and a common drawback

Method	Venue	Summary	Drawback
SAFIRE [7]	AAAI 2025	Partitions an image into distinct source regions using a contrastive-trained SAM and clustering of internal point prompts.	Existing methods lack the ability to capture the complete membership region for areas and rely on a limited number of prompt points, fail to effectively model their diverse, multi-object characteristics, leading to incomplete segmentation.
HDF-Net [9]	T-PAMI 2024	Captures "homogeny difference features" using a dual-stream (RGB+SRM) network with a coarse-to-fine module strategy.	
TurFor [10]	CVPR 2023	Identifies forgery anomalies by fusing RGB images and a Noiseprint++ fingerprint using a transformer network.	
CFL-Net [11]	WACV 2023	RGB+SRM + ASPP + SupCon head; separates forgery/non-forgery features for artifact-agnostic localization and strong cross-domain results	

[7] Kwon, Myung-Joon, et al. "Safire: Segment any forged image region." **AAAI**, 2025.
[9] Han, Ruidong, et al. "HDF-Net: Capturing homogeny difference features to localize the tampered image." **IEEE T-PAMI**, 2024.
[10] Guillaro, Fabrizio, et al. "Trufor: Leveraging all-round clues for trustworthy image forgery detection and localization." **CVPR**, 2023.
[11] Niloy, Fahim Faisal, Kishor Kumar Bhaumik, and Simon S. Woo. "CFL-Net: Image forgery localization using contrastive learning." **WACV**, 2023.

Proposed Method

Notation and Definition

- Image forgery localization dataset is defined as $\mathcal{D} = \{(I_i, M_i)\}_{i=1}^N$, where $I_i \in \mathbb{R}^{H \times W \times 3}$ is an input image and $M_i \in \{0,1\}^{H \times W}$ is its corresponding ground-truth mask for supervised training; here, N denotes the dataset size.
- PMR localization aims to learn a function f that maps the input image I_i and a set of prompt points P_i to a predicted mask $\hat{M}_i = f(I_i, P_i)$, which accurately segments the membership region corresponding to the given prompts.

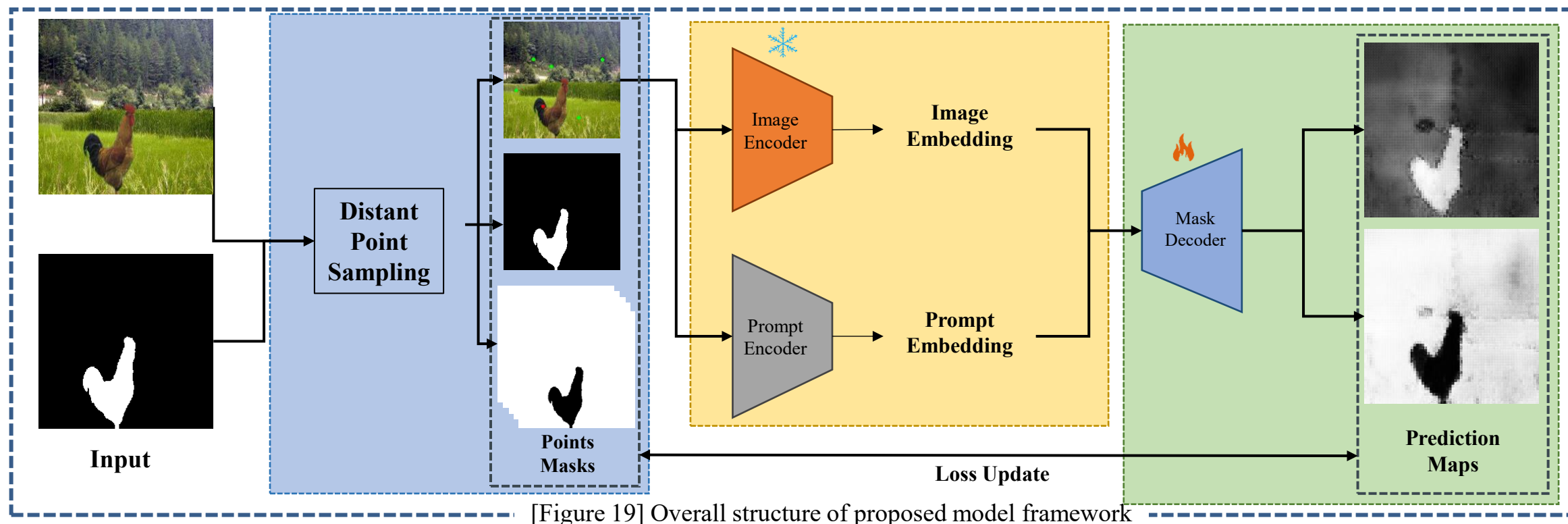
[Table 2] Notations and definitions used in the formulation of the Pixel Membership Region Localization frameworks

Symbol	Name	Description
I	Input Image	RGB image that may contain forgery regions
M	Ground-Truth Mask	Binary mask where 1 indicates forgery and 0 indicates non-forgery
P	Prompt Points	Set of k sampled pixel coordinates used as prompts
E_{img}	Image Encoder	Network that extracts a high-level embedding from the input image
E_{prm}	Prompt Encoder	Network that converts prompt points into an embedding
D_{mask}	Mask Decoder	Network that fuses embeddings to predict the final mask
$\hat{M}$	Predicted Mask	Final binary mask output by the model

Proposed Method

Procedural Overview

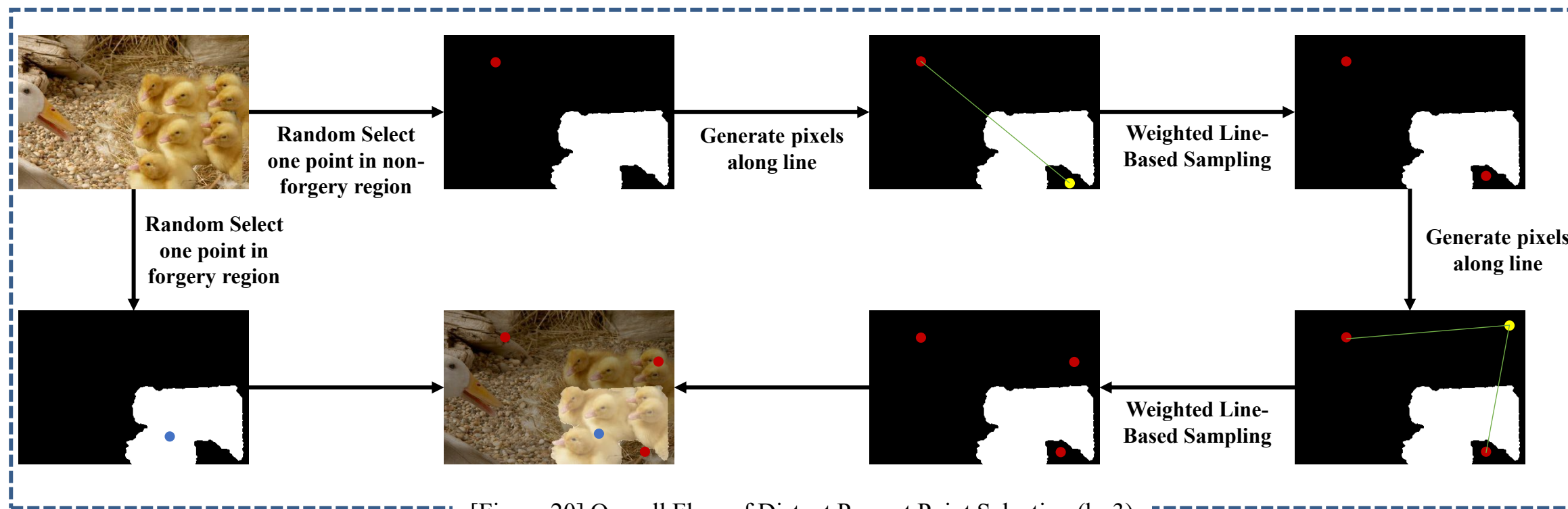
- The proposed method first extracts a global image embedding from the input image via an image encoder, while simultaneously generating prompt embeddings by inputting distant sampled forgery and non-forgery points into a prompt encoder.
- The extracted image and prompt embeddings are fed into a mask decoder where they are fused to predict the final forgery mask, and the model is updated during training by calculating the loss between the prediction maps and points masks.



Proposed Method

Distant Prompt Point Selection

- Forgery regions are typically compact and homogeneous, while non-forgery areas are broad and heterogeneous; Distant Prompt Point Selection assigns one forgery prompt and k non-forgery prompts to reduce class imbalance and cover non-forgery variability.
- Start from a random non-forgery seed, iteratively choose the farthest anchor, gather non-forgery candidates along lines to existing points, weight by distance, and sample one; repeat to k for spatially diverse coverage (the forgery prompt is a single random sample).



[Figure 20] Overall Flow of Distant Prompt Point Selection ($k=3$)

Proposed Method

Pseudo Algorithm

- Extract Ω_{non} ; sample the first non-forgery seed $p^{(1)}$ and set $S = \{p^{(1)}\}$. Until $|S| = k$, choose the farthest anchor $A \leftarrow \operatorname{argmax}_{c \in \Omega_{\text{non}}} (\sum_{q \in S} \|c - q\|_2)$; for each $s \in S$ trace *BresenhamLine*(s, A) to collect non-forgery pixels L with $w = \max(\|p' - s\|_2, \varepsilon)$ assigning distant points.
- Normalize W to probabilities P , sample one $p^{(1)} \sim \text{Categorical}(L, P)$, and update $S \leftarrow S \cup \{p^{(\text{new})}\}$ until $|S| = k$; coupling deterministic line tracing with diversity-driven sampling to yield a well-spread non-forgery prompt set.

Algorithm 1 : Select k non-forgery points via weighted line-based sampling

Algorithm Probabilistic Line – Skew Sampling:

Input: Image mask $I \in \{0, 255\}^{H \times W}$ (0=non-forgery, 255=forgery), integer k
Output: non-forgery points set S with $|S| = k$

```
1   $\Omega_{\text{non}} \leftarrow \{(y, x) \mid I(y, x) = 0\}$ 
2  pick  $p^{(1)} \sim \text{Uniform}(\Omega_{\text{non}})$ ,  $S \leftarrow \{p^{(1)}\}$ 
3  for  $t \leftarrow 1$  to  $k - 1$  do
4     $A \leftarrow \operatorname{argmax}_{c \in \Omega_{\text{non}}} (\sum_{q \in S} \|c - q\|_2)$ 
5     $L \leftarrow []$ ,  $W \leftarrow []$ 
6    foreach  $s \in S$  do
7       $L_s \leftarrow \text{BresenhamLine}(s, A)$ 
8      foreach  $p' \in L_s$  do
9        if  $p' \in \Omega_{\text{non}} \wedge p' \neq s$  then
10          append  $p'$  to  $L$ 
11          append  $\max(\|p' - s\|_2, \varepsilon)$  to  $W$ 
12   $W_{\text{total}} \leftarrow \sum_{w_j \in W} w_j$ 
13   $P \leftarrow [w_i / W_{\text{total}} \text{ for each } w_i \in W]$ 
14  sampling  $p^{(\text{new})} \sim \text{Categorical}(L, P)$ 
15   $S \leftarrow S \cup \{p^{(\text{new})}\}$ 
16  return  $S$ 
```

Line 1 → Extract non-forgery pixels

Line 2 → Randomly select the first non-forgery point and initialize point set

Line 3 → Loop until t reached $k-1$

Line 4 → Find the farthest point from all points in S

Line 5 → Initialize list of candidate points and weights

Line 6 → For each selected point $s \in S$, iterate

Line 7 → Generate pixels along line between s and A

Line 8 → For each pixel p' on the Bresenham path L_s between s and A

Line 9 → If p' is in the non-forgery set Ω_{non} and $p' \neq s$

Line 10 → Add valid non-forgery pixel

Line 11 → Add distance-based weight

Line 12 → Compute total weight

Line 13 → Normalize weights to probabilities

Line 14 → Sample one point using weighted probabilities

Line 15 → Add the new point to set

Line 16 → Return the selected non-forgery points

Proposed Method

Pseudo Algorithm

- Initialize integer deltas dx, dy , step signs sx, sy , and the error term $err = dx - dy$; $set(x, y) = (x_1, y_1)$ and handle all directions by sign/axis swaps, enabling integer, one-pixel stepping as a deterministic path generator between two points.
- At each step output (x, y) , stop at (x_2, y_2) compute $e2 = err$, then take an x-step $e2 > dy$ and a y-step if $e2 > dx$, updating err accordingly, thereby generating a deterministic sequence of $O(|dx| + |dy|)$ grid-aligned pixels in all directions.

Algorithm 2 : BresenhamLine: two points $\rightarrow$ a sequence of discrete pixels on the segment

Function *BresenhamLine* $((y_1, x_1), (y_2, x_2))$:

```
1  coords  $\leftarrow []$ 
2   $dy \leftarrow |y_2 - y_1|, \quad dx \leftarrow |x_2 - x_1|$ 
3   $sy \leftarrow 1[y_1 < y_2]? 1: -1,$ 
4   $sx \leftarrow 1[x_1 < x_2]? 1: -1$ 
5   $err \leftarrow dx - dy;$ 
6   $y \leftarrow y_1; \quad x \leftarrow x_1$ 
7  while true do
8    append  $(y, x)$  to coords
9    if  $(y = y_2) \wedge (x = x_2)$  then
10     break
11     $e2 \leftarrow 2 \cdot err$ 
12    if  $e2 > dy$  then
13       $err \leftarrow err - dy;$ 
14       $x \leftarrow x + sx$ 
15    if  $e2 < dx$  then
16       $err \leftarrow err + dx;$ 
17       $y \leftarrow y + sy$ 
18  return coords
```

Line 1 $\rightarrow$ Initialize coordinate list

Line 2 $\rightarrow$ Get total change in y and x

Line 3-4 $\rightarrow$ Set x & y direction step (1 if x_1 (or y_1) $<$ x_2 (or y_2), else -1)

Line 5 $\rightarrow$ Initialize the error term

Line 6 $\rightarrow$ set current position to start point

Line 7 $\rightarrow$ Loop until the end point is reached

Line 8 $\rightarrow$ Add current pixel to list

Line 9 $\rightarrow$ If current pixel equals the end point

Line 10 $\rightarrow$ Exit loop if end point is reached

Line 11 $\rightarrow$ Calculate double error for comparison

Line 12 $\rightarrow$ If the double error is below the dy

Line 13 $\rightarrow$ Update the error accordingly

Line 14 $\rightarrow$ Take an x-step

Line 15 $\rightarrow$ If the double error is below the dx

Line 16 $\rightarrow$ Update the error accordingly

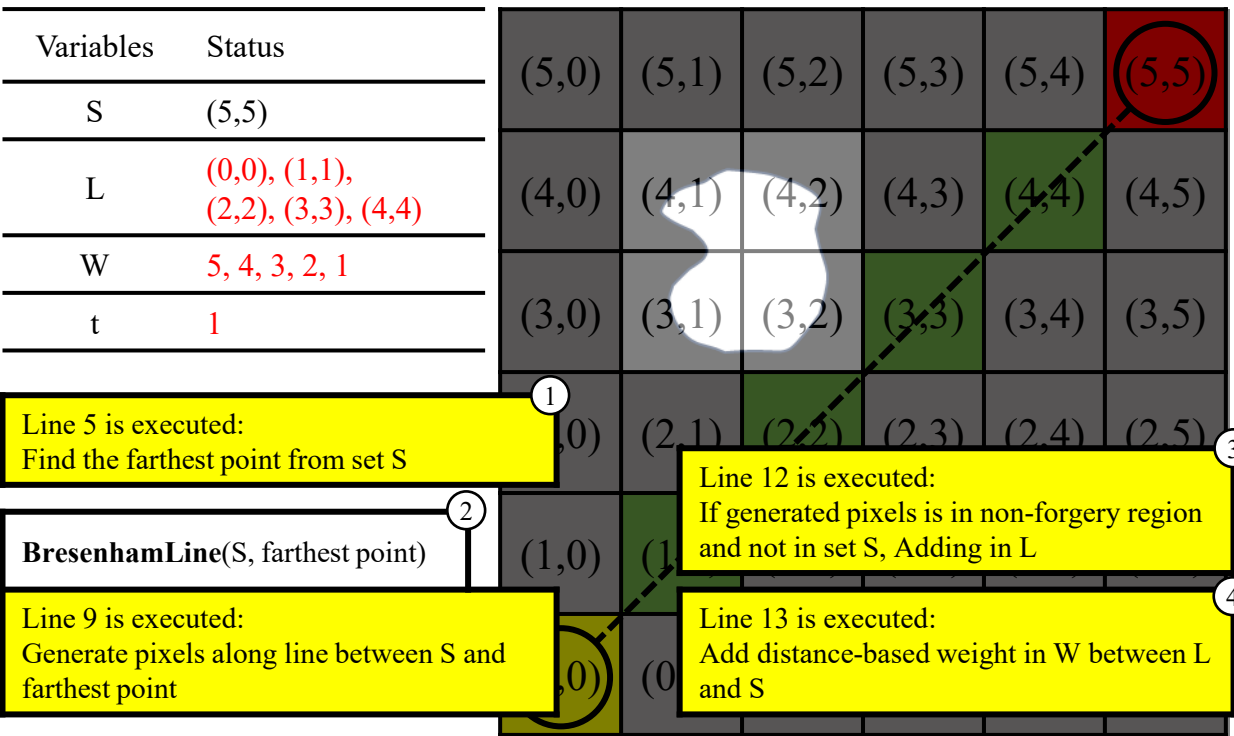
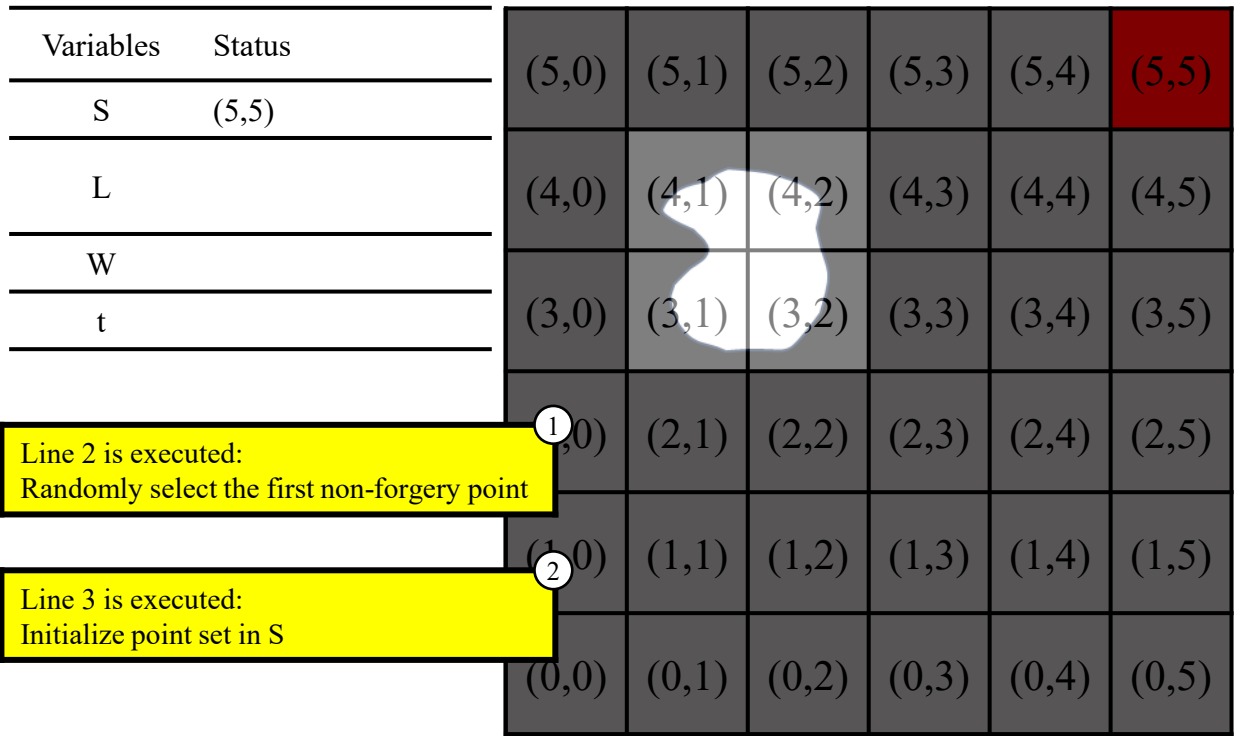
Line 17 $\rightarrow$ Take a y-step

Line 18 $\rightarrow$ Return the list of pixel

Proposed Method

Pseudo Algorithm - Details

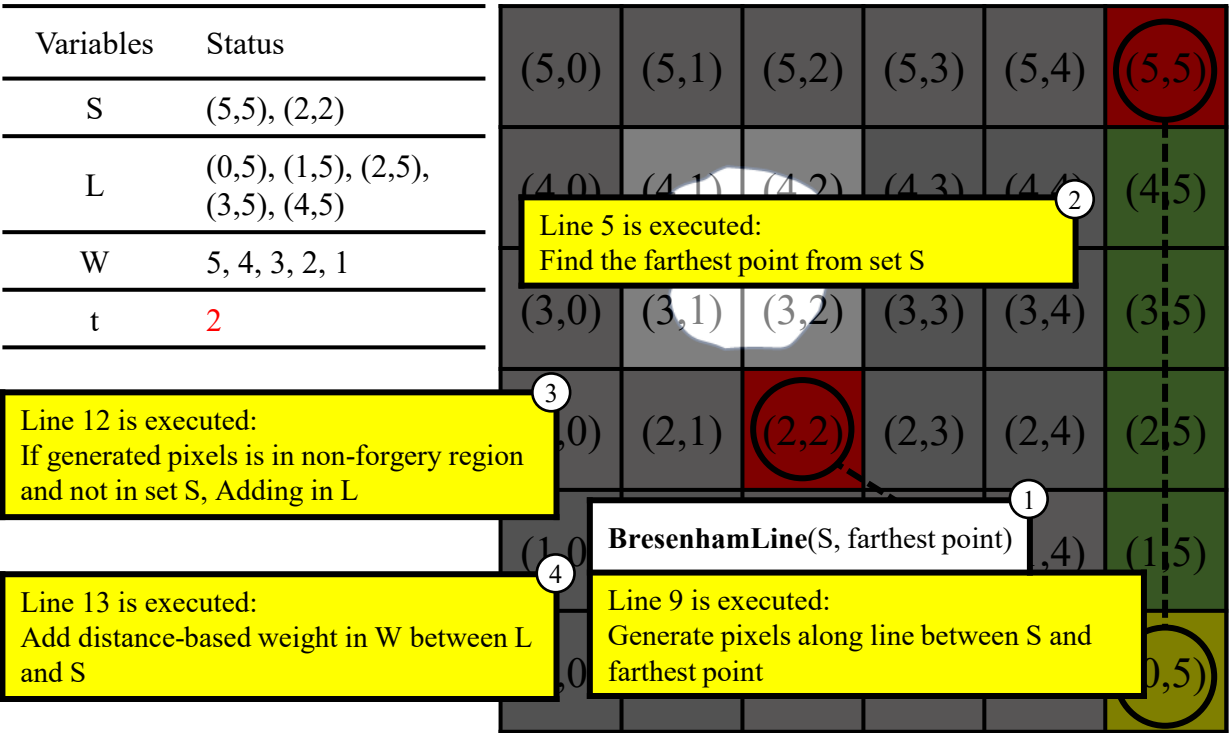
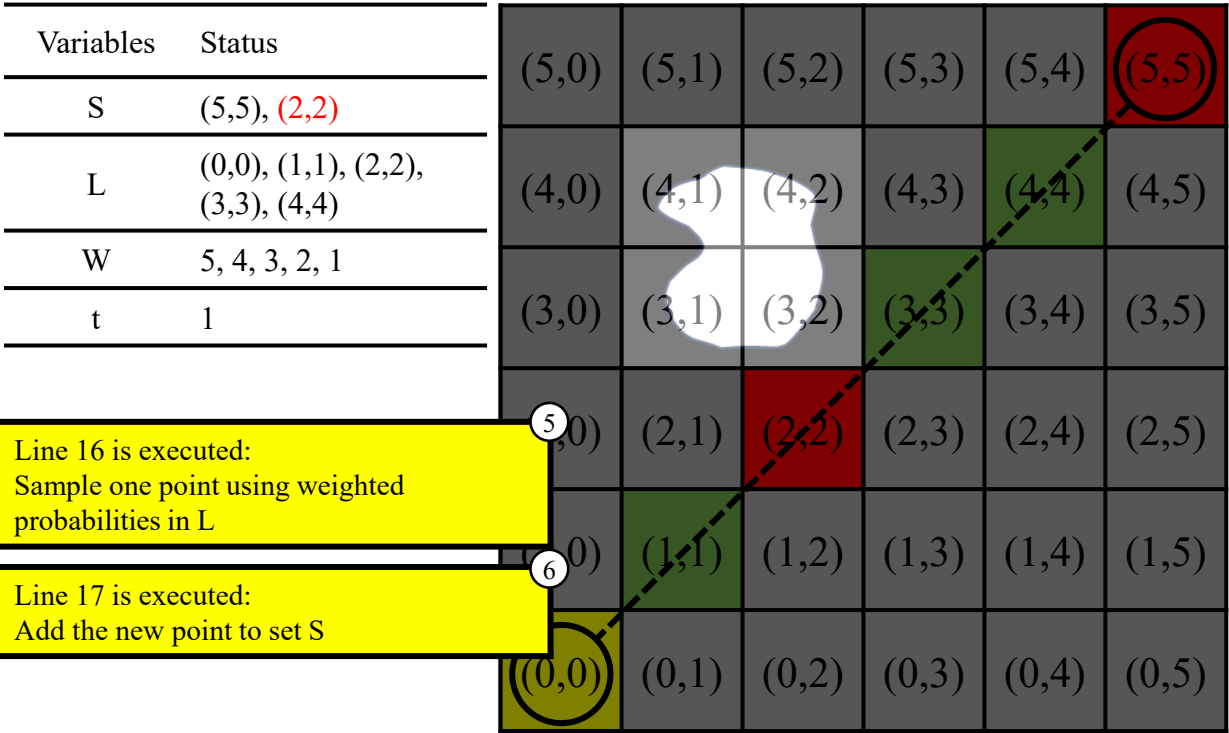
- The left figure illustrates the initialization phase, where an anchor point (0,0) is identified as the farthest pixel from the initial random point (5,5) to maximize the spatial exploration range within the non-forgery region.
- The right figure demonstrates the candidate generation process, where pixels along the Bresenham Line connecting the source and anchor are collected into list L , with distance-based weights (W) assigned to prioritize selecting spatially diverse points.



Proposed Method

Pseudo Algorithm - Details

- The left figure illustrates the sampling phase, where a new non-forgery point (e.g., (2,2)) is probabilistically selected from the weighted candidate list L and added to the prompt set S , completing the first iteration.
- The right figure demonstrates the subsequent iteration ($t = 2$), where a new anchor point is identified based on the maximum cumulative distance, and candidate lines are generated from all existing points to this anchor to explore diverse regions.

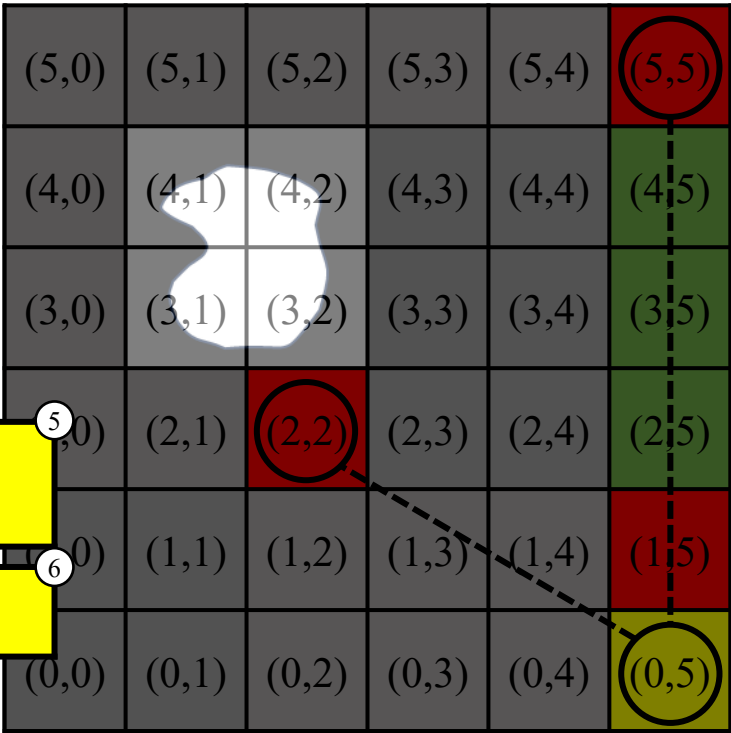


Proposed Method

Pseudo Algorithm - Details

- The algorithm iterates this weighted sampling process $k - 1$ times, dynamically updating the anchor point to select new candidates from unexplored areas, thereby incrementally building a spatially diverse point set.
- Finally, when the target count k is reached (e.g., adding (1,5) as the last point), the algorithm terminates and returns the set S , which provides a comprehensive representation of the heterogeneous non-forgery region.

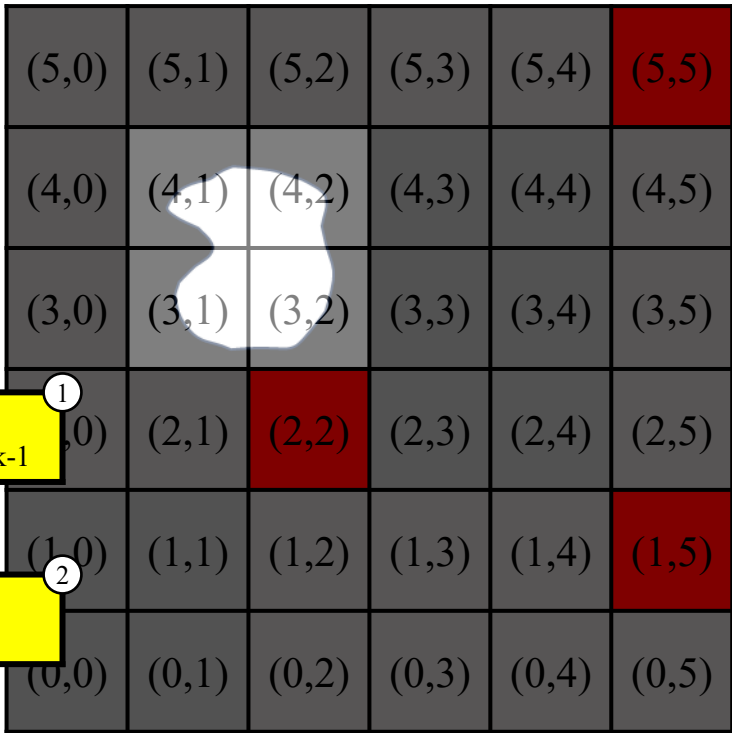
Variables	Status
S	(5,5), (2,2), (1,5)
L	(0,5), (1,5), (2,5), (3,5), (4,5)
W	5, 4, 3, 2, 1
t	2



Line 16 is executed:
Sample one point using weighted probabilities in L

Line 17 is executed:
Add the new point to set S

Variables	Status
S	(5,5), (2,2), (1,5)
L	
W	
t	$k-1$



Line 4 is executed:
Do this sampling point algorithm to $t = k-1$

Line 18 is executed:
Return the selected non-forgery points

Experimental Settings

Dataset

- CASIA 2.0 dataset is utilized as the sole training dataset, which is a large-scale collection of high-resolution images containing realistic Splicing and Copy-Move forgeries, providing a comprehensive base for learning generalizable manipulation features.
- To robustly evaluate generalization, all models are tested on four completely unseen datasets, which feature diverse manipulation types (including Inpainting), different post-processing, and varying image sources.

[Table 3] Summary of Image Forgery Localization datasets

Dataset	Year	Image Size	Train/Test	# of Images	Manipulate Methods
CASIA 2.0	2013	Various	Train	5,123	Copy-move, Splicing
COLUMBIA	2006	768 × 512	Test	180	Splicing
Coverage	2016	400 × 486	Test	100	Copy-move
IMD2020	2020	Various	Test	2,010	Copy-move, Splicing, Inpainting
CocoGlide	2023	512 × 512	Test	512	Splicing

Experimental Settings

Evaluation Measures

- *F1-score*

$$F1 = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

- *Area Under the ROC Curve*

$$AUC = \int_0^1 TPR(TPR(FPR^{-1}(x)))dx$$

- *Precision*

$$\text{Precision} = \frac{TP}{TP + FP}$$

- *Recall*

$$\text{Recall} = \frac{TP}{TP + FN}$$

TPR : True Positive Rate

$$TPR = \frac{TP}{TP + FN}$$

FPR : False Positive Rate

$$FPR = \frac{FP}{FP + TN}$$

Cross Validation

- Number of Iteration: 1
- Repeated Shuffle Split
- Data Split Ratio (Train:Validation) = 8:2

Hyperparameter Settings

- Batch size: 4×2
- Epochs : 50
- Non-forgery points : 5

Experimental Results

Performance Comparison

- The proposed model achieves the highest F1-score and AUC on various unseen datasets under cross-domain shifts, outperforming all comparison methods including the recent state-of-the-arts models such as SAFIRE.
- Compared to SAFIRE, largest gains on COVERAGE (+0.2143 F1, +0.3131 AUC), with consistent improvements on CocoGlide (+0.1595 F1, +0.2308 AUC), Columbia (+0.581 F1, +0.636 AUC), and IMD2020 (+0.550 F1, +0.1302 AUC).

[Table 4] Experimental results of five IFL methods regarding **F1-score** performance in cross-domain setting

Training		Test		
CASIA_2.0	CocoGlide	Columbia	COVERAGE	IMD2020
Proposed	0.5104	0.9122	0.5338	0.3578
SAFIRE	0.3509	0.8541	0.3195	0.3028
HDF-Net	0.1193	0.1469	0.1143	0.0986
CFL-Net	0.1893	0.2675	0.1552	0.1979
TruFor	0.2268	0.3287	0.1506	0.2459

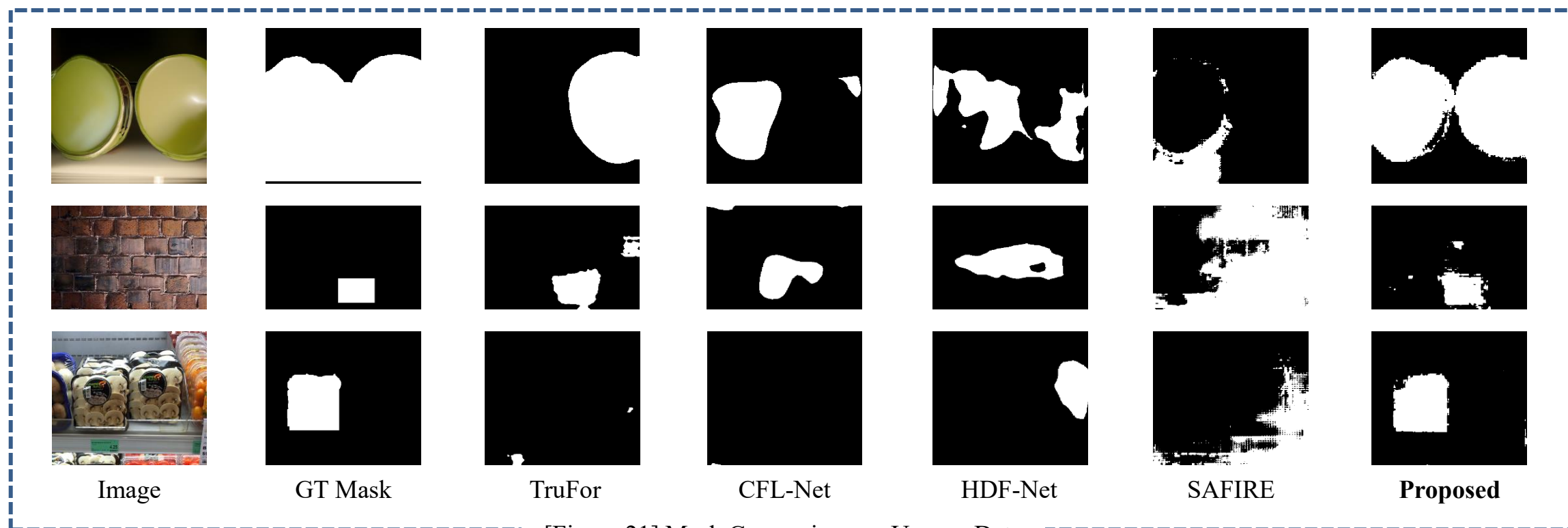
[Table 5] Experimental results of five IFL methods regarding **AUC** performance in cross-domain setting

Training		Test		
CASIA_2.0	CocoGlide	Columbia	COVERAGE	IMD2020
Proposed	0.7708	0.9497	0.8393	0.7899
SAFIRE	0.5400	0.8861	0.5262	0.6597
HDF-Net	0.5222	0.5176	0.5379	0.5397
CFL-Net	0.5620	0.5644	0.5582	0.6169
TruFor	0.7106	0.7458	0.6686	0.7736

Experimental Results

Qualitative Analysis

- Reformulating IFL as PMR largely suppresses background-biased errors seen in traditional segmentation (HDF-Net, CFL-Net, TruFor), yielding cleaner masks, stronger cross-dataset performance and improved robustness.
- Spatially diverse non-forgery prompts (farthest-anchor + line-skew sampling) expand coverage on heterogeneous backgrounds and across scales, boosting recall and recovering fragmented forgery regions under domain shift.



[Figure 21] Mask Comparison on Unseen Data

Experimental Results

Impact on the number of prompting pixel k

- Increasing the number of non-forgery points (k) generally leads to significant performance gains over the single-point baseline ($k=1$), with performance tending to saturate or show slight fluctuations beyond a certain threshold.
- Proposed method adopted $k = 5$ as the optimal configuration because it most frequently secures the best (red) or second-best (blue) rankings across diverse datasets, demonstrating the most robust and balanced generalization capability among the candidates.

[Table 6] Results of **F1-score** performance with varying number of non-forgery points (k) in cross-domain setting. The **red** and **blue** colors indicate the best and second performance

Training		Test		
CASIA_2.0	CocoGlide	Columbia	COVERAGE	IMD2020
K=1	0.3509	0.8541	0.3195	0.3028
K=3	0.4299	0.9195	0.5218	0.3344
K=5	0.5215	0.9122	0.5338	0.3578
K=7	0.5104	0.8931	0.5761	0.3487

[Table 7] Results of **AUC** performance with varying number of non-forgery points (k) in cross-domain setting. The **red** and **blue** colors indicate the best and second performance

Training		Test		
CASIA_2.0	CocoGlide	Columbia	COVERAGE	IMD2020
K=1	0.5400	0.8861	0.5262	0.6597
K=3	0.7504	0.9661	0.8616	0.7972
K=5	0.7552	0.9497	0.8393	0.7899
K=7	0.7708	0.9388	0.8706	0.7818

Conclusion

- This study reframes IFL as Pixel Membership Region Localization and builds an end-to-end pipeline that predicts per-prompt membership maps and aggregates them into a coherent mask, incorporating DPPS for cross-domain robustness.
- This study selects one forgery seed and multiple spatially distant non-forgery prompts to cover heterogeneous backgrounds, improving boundary fidelity and disentangling context for complete membership recovery.
- This study surpasses HDF-Net, CFL-Net, and TruFor in F1/AUC on unseen datasets; relative to SAFIRE, it achieves the largest gains on COVERAGE (+0.214 F1, +0.313 AUC) with consistent improvements on CocoGlide, Columbia, and IMD2020.

Achievement

- Effective Image Forgery Localization based on Distant Point Prompting for Multiple Segments in Non-Forgery Regions, *Information*, 2025
- Teaching Assistant, *Hyundai Bootcampus*, 2025
- Detecting Fake Images with Surrounding Pixel Relationships, *IEIE*, 2025

Thank you